UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

PROGRAMA DE PÓS-GRADUAÇÃO EM ENSINO DE CIÊNCIA E TECNOLOGIA

MESTRADO PROFISSIONAL EM ENSINO DE CIÊNCIA E TECNOLOGIA

JOÃO HENRIQUE BERSSANETTE

ENSINO DE PROGRAMAÇÃO DE COMPUTADORES: UMA PROPOSTA DE ABORDAGEM PRÁTICA BASEADA EM AUSUBEL

DISSERTAÇÃO

PONTA GROSSA

2016

JOÃO HENRIQUE BERSSANETTE

ENSINO DE PROGRAMAÇÃO DE COMPUTADORES: UMA PROPOSTA DE ABORDAGEM PRÁTICA BASEADA EM AUSUBEL

Dissertação apresentada como requisito parcial à obtenção do título de Mestre em Ensino de Ciência e Tecnologia, do Programa de Pós-Graduação em Ensino de Ciência e Tecnologia, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. André Koscianski

PONTA GROSSA

2016

Ficha catalográfica elaborada pelo Departamento de Biblioteca
da Universidade Tecnológica Federal do Paraná, Campus Ponta Grossa
n.13/16

B535 Berssanette, João Henrique

Ensino de programação de computadores: uma proposta de abordagem prática baseada em Ausubel. / João Henrique Berssanette. -- Ponta Grossa, 2016.
144 f : il. ; 30 cm.

Orientador: Prof. Dr. André Koscianski

Dissertação (Mestrado em Ensino de Ciência e Tecnologia) - Pós-Graduação em Ensino de Ciência e Tecnologia. Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2016.

1. Programação de computadores. 2. Aprendizagem. 3. Ausubel, David Paul, 1918-. I. Koscianski, André. II. Universidade Tecnológica Federal do Paraná. III. Título.

CDD 507

Universidade Tecnológica Federal do Paraná
Campus de Ponta Grossa
Diretoria de Pesquisa e Pós-Graduação
PROGRAMA DE PÓS-GRADUAÇÃO EM ENSINO DE CIÊNCIA E TECNOLOGIA

FOLHA DE APROVAÇÃO

Título de Dissertação **Nº 104/2016**

ENSINO DE PROGRAMAÇÃO DE COMPUTADORES: UMA PROPOSTA DE ABORDAGEM PRÁTICA BASEADA EM AUSUBEL

por

João Henrique Berssanette

Esta dissertação foi apresentada às **14 horas** do dia **31 de março de 2016** como requisito parcial para a obtenção do título de MESTRE EM ENSINO DE CIÊNCIA E TECNOLOGIA, com área de concentração em Ciência, Tecnologia e Ensino, linha de pesquisa em Educação Tecnológica, Programa de Pós-Graduação em Ensino de Ciência e Tecnologia. O candidato foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. Dr. Esteban Walter Gonzalez Clua (UFF)

Prof. Dr. Guataçara dos Santos Junior (UTFPR)

Prof. Dr. Antonio Carlos de Francisco (UTFPR)

Prof. Dr. André Koscianski (UTFPR) ***- Orientador***

Visto do Coordenador:

Prof. Dr. Luis Mauricio Martins de Resende
Coordenador do PPGECT

- A FOLHA DE APROVAÇÃO ASSINADA ENCONTRA-SE ARQUIVADA NO DEPARTAMENTO DE REGISTROS ACADÊMICOS DA UTFPR –

Dedico este trabalho à minha família,
Fabiane, Guilherme e João Vitor
pelos momentos de ausência.

AGRADECIMENTOS

Certamente estes parágrafos não irão atender a todas as pessoas que fizeram parte dessa importante fase de minha vida. Portanto, desde já peço desculpas àquelas que não estão presentes entre essas palavras, mas elas podem estar certas que fazem parte do meu pensamento e de minha gratidão.

Agradeço ao meu orientador Prof. Dr. André Koscianski, pela sabedoria com que me guiou nesta trajetória.

Aos meus colegas de sala.

A Secretaria do Curso, pela cooperação.

A todo o corpo docente do PPGECT, que brilhantemente compartilharam seus conhecimentos.

Gostaria de deixar registrado também, o meu reconhecimento à minha família, pois acredito que sem o apoio deles seria muito difícil vencer esse desafio.

Enfim, a todos os que por algum motivo contribuíram para a realização desta pesquisa.

“Depois de muito tempo acordado e
cansado de tanto sofrer, esta noite eu
dormi um pouquinho sonhei com você!"
(Milionário e José Rico)

RESUMO

BERSSANETTE, João Henrique. **Ensino de programação de computadores:** uma proposta de abordagem prática baseada em Ausubel. 2016. 144f. Dissertação (Mestrado em Ensino de Ciências e Tecnologia) - Universidade Tecnológica Federal do Paraná. Ponta Grossa, 2016.

A aprendizagem de programação de computadores é essencial para todas as carreiras ligadas a computação e informática, entretanto percebe-se que parcela significativa dos alunos apresenta dificuldades em assimilar e aplicar certos conceitos. Esta pesquisa buscou elaborar uma proposta de abordagem prática, baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação. Com base na revisão de literatura referente as dificuldades no processo de ensino/aprendizagem de programação de computadores, e também nas propostas existentes, elaborou-se uma proposta de abordagem para as disciplinas introdutórias de programação. Esta proposta foi submetida a um teste piloto, seguido de uma pesquisa experimental, onde foram conduzidas aplicações da proposta pelo professor pesquisador. A partir da coleta de dados de documentos oficiais como diários de classes e avaliações, foi realizada uma comparação qualitativa entre a proposta de abordagem e a abordagem tradicional da disciplina. A análise dos dados, indicou resultados positivos a exposição dos estudantes mais cedo ao uso prático do computador, e a assuntos que normalmente são vistos primeiramente de maneira conceitual. Além disso, a proposta de abordagem desenvolvida possibilitou aos alunos verem os conteúdos mais vezes, o que pode contribuir para a aquisição de experiência em programação.

Palavras-chave: Ensino. Programação de computadores. Proposta de abordagem. Aprendizagem significativa.

ABSTRACT

BERSSANETTE, João Henrique. **Teaching of Computer programming:** a proposal for a practical approach based on Ausubel. 2016. 144f. Dissertation (Masters in Science and Technology Education) - Federal Technological University of Paraná. Ponta Grossa, in 2016.

Learning computer programming is essential for all careers in computing and information technology. However, a significant part of student presents difficulties in learning and applying certain concepts. This research aimed to develop a practical approach, based on the theory of meaningful learning, emphasizing / valuing the interaction with machine and exposing students earlier to a practical use of computers for teaching programming. Based on the literature review concerning the difficulties in teaching / learning computer programming, and also on existing proposals, it was drawn up a approach to introductory courses in programming. This proposal was submitted to a pilot test, followed by an experimental research conducted by research professor. From the collection of official documents such as daily classes and evaluations, a qualitative comparison was made between the proposal and the traditional approach to discipline. Data analysis indicated that an earlier exposure to a practical use of the computer and to matters usually first seen conceptually are positive for students. Moreover, this proposal enabled the students to see content more often. It may contribute to the acquisition of programming experience.

Keywords: Teaching. Computer programming. Proposed approach. Meaningful learning.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os cursos da área de computação e informática de nível superior ou técnico tem como uma de suas metas capacitar estudantes a apresentar soluções computadorizadas para diversos problemas do mundo real. Para produzir estas soluções os alunos devem utilizar comandos definidos a partir de uma linguagem de programação. Uma linguagem de programação tem um conjunto de símbolos e regras de sintaxe e semântica que permite descrever um conjunto de processos de forma precisa e que possam ser executadas por um computador.

Portanto, aprendizagem de programação é essencial para todas as carreiras ligadas à computação e a informática. Esta aprendizagem ocorre em uma série de disciplinas como algoritmos, lógica de programação, linguagem de programação, técnicas de programação, estrutura de dados, entre outras.

Estas disciplinas podem ser consideradas fundamentais para formação de alunos que terão no desenvolvimento de softwares o produto final de seu trabalho.

No entanto, o processo de ensino/aprendizagem de programação tem se demonstrado difícil para estudantes e professores, acarretando grandes índices de reprovação, desistência e abandono de cursos em instituições de ensino.

Isto ocorre devido a diversos problemas, dentre esses, a literatura aponta causas como a falta de competências na resolução de problemas, poucas habilidades matemáticas, baixo nível de abstração, dificuldades de interpretação do problema e compreensão de texto por parte dos estudantes.

Visando minimizar o impacto das dificuldades, diversas alternativas são propostas na literatura, existindo três caminhos gerais: (i) ferramentas, (ii) metodologias ou estratégias, (iii) ferramentas e metodologias associadas.

Este trabalho se enquadra na segunda categoria (metodologias ou estratégias), e se apoia na teoria da aprendizagem significativa de David Ausubel, para propor uma abordagem prática baseada em Ausubel para a apresentação dos conteúdos de programação aos estudantes.

Ausubel considera que o fator mais importante para que a aprendizagem ocorra, é determinar aquilo que o estudante já sabe ou conhece, para que estes conhecimentos prévios sirvam de ponto de ancoragem para os novos conhecimentos a serem adquiridos. Se considerarmos que para a maioria dos estudantes iniciantes

em informática e computação a programação é um assunto completamente novo, a ausência de conhecimentos prévios pode tornar o processo de ensino/aprendizagem mais delicado.

Assim, ao se trabalhar com um assunto desconhecido é importante resgatar alguma referência, para que sirvam de ponto de ancoragem para os novos conhecimentos, nesta pesquisa propomos a utilização do computador como referência, visando tornar o conteúdo menos estranho aos estudantes.

1.1 PROBLEMA DE PESQUISA

Este trabalho estudou as dificuldades relacionadas ao ensino/aprendizagem de programação, para o desenvolvimento de uma proposta de abordagem prática baseada em Ausubel que possa colaborar com a formação e também o quadro de índices de reprovação e evasão nestas disciplinas.

Esta dissertação investigou as possíveis contribuições de uma proposta de abordagem prática baseada em Ausubel para o ensino de programação de computadores que vise conduzir os alunos através das dificuldades da disciplina.

A hipótese que orientou o trabalho, é a de que uma proposta de abordagem prática baseada em Ausubel, que priorize e exponha os alunos mais cedo ao uso prático do computador traga resultados positivos comparativamente à abordagem tradicional.

Desta forma elaborou-se a seguinte questão norteadora para esta pesquisa:

Quais as contribuições de uma proposta de abordagem prática baseada em Ausubel, que priorize e exponha os alunos mais cedo ao uso prático do computador comparativamente à abordagem tradicional?

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Partindo do princípio que a utilização de uma proposta de abordagem prática que priorize e exponha os alunos mais cedo ao uso prático do computador, baseada na teoria da aprendizagem significativa possa contribuir para o ensino de programação, delineou-se o seguinte objetivo geral:

Elaborar uma proposta de abordagem prática, baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação.

1.2.2 Objetivos Específicos

Visando nortear a aplicação desse estudo, foram elaborados os seguintes objetivos específicos:

- Identificar habilidades e competências exercidas na programação de computadores presentes na literatura;
- Identificar dificuldades na aprendizagem de programação de computadores presentes na literatura;
- Implementar o uso da abordagem proposta;
- Analisar as contribuições da abordagem proposta.

1.3 ESTRUTURA DO TRABALHO

Este texto está organizado da seguinte forma:

O capítulo 1 apresenta o tema com a declaração do problema, a justificativa, a hipótese e os objetivos deste estudo.

O capítulo 2 apresenta as bases teóricas que subsidiam e fundamentam a pesquisa.

O capítulo 3 descreve a metodologia utilizada para o desenvolvimento deste estudo, bem como os procedimentos metodológicos, referenciando os instrumentos de coleta de dados e as atividades desenvolvidas.

O capítulo 4 apresenta a proposta de abordagem prática baseada em Ausubel desenvolvida para o ensino de programação de computadores.

O capítulo 5 apresenta o desenvolvimento da pesquisa, com suas etapas e aplicação, apresentando a análise e interpretação dos dados coletados, as discussões acerca dos efeitos da utilização da proposta de abordagem.

No capítulo 6, são expostas as considerações finais, as principais conclusões e limitações da pesquisa, assim como, sugestões para trabalhos futuros

2 REFERENCIAL TEÓRICO

Este capítulo apresenta os pressupostos teóricos que norteiam este trabalho.

Inicialmente apresentam-se alguns conceitos de programação de computadores, visando esclarecer ao leitor o contexto deste trabalho. Em seguida, é apresentada uma visão geral das habilidades e competências envolvidas na programação de computadores e posteriormente o processo de ensino de programação.

A partir disso, discutem-se as dificuldades apresentadas no processo de ensino/aprendizagem de programação de computadores, onde destacam-se as propostas encontradas na literatura para atenuar estas dificuldades.

Por fim, discute-se a teoria da aprendizagem significativa de David Ausubel e como ela poderia contribuir no ensino/aprendizagem de programação de computadores.

2.1 PROGRAMAÇÃO DE COMPUTADORES

Os computadores podem ser considerados como uma ferramenta indispensável a vida moderna, estando presente nos mais variados segmentos. Esta máquina vem alterando significativamente nossas vidas nos mais variados contextos.

Hoje, para a maioria dos usuários o computador é algo simples, isso é possível devido aos avanços que o tornou cada vez mais intuitivo e de fácil utilização.

Um *software* ou programa de computador, é o resultado da compilação ou interpretação de um conjunto de instruções elaboradas para realização de uma tarefa ou resolução de um problema. Estas instruções devem seguir padrões de sintaxe para que possam ser compreendidas e executadas pelo computador.

O desenvolvimento da solução e implementação por meio da codificação engloba criar ou deduzir o raciocínio necessário para resolução do problema pelo computador.

Há diversas linguagens de programação com características específicas. A escolha da linguagem a ser utilizada está associada a diversos fatores, dentre os mais importantes pode-se citar o paradigma de desenvolvimento do *software.*

Um paradigma pode ser definido como um modelo de programação, ou seja, é um padrão utilizado pelo programador para orientar a maneira como o *software* será desenvolvido.

Geralmente o paradigma procedural é o mais comum para o ensino dos fundamentos de programação de computadores e será adotado nesta dissertação (BARANAUSKAS, 1993).

Assim, programar nos diferentes paradigmas significa representar segundo padrões diferentes a solução do problema a ser resolvido. Pode-se dizer que um mesmo problema pode ser abordado empregando qualquer um dos paradigmas existentes, resultando em maior ou menor esforço do programador.

Portanto, programar computadores é uma atividade que requer o conhecimento do conteúdo que está sendo tratado, o domínio de uma linguagem de programação onde a solução do problema deve ser representada, e criatividade, uma vez que há sempre inúmeras maneiras de se chegar a uma solução por meio da programação. A programação de computadores é uma atividade exigente, que requer do programador certas habilidades a fim de que possa implementar soluções para um determinado problema e representa-las no ambiente computacional. A programação está relacionada com outras atividades como especificação, projeto e modelagem (BRASIL, 2001).

2.1.1 Visão geral de fatores envolvidos na programação de computadores

A programação de computadores requer diversas habilidades e competências para que a solução de um determinado problema possa ser representada no ambiente computacional. Para a execução de todas estas atividades, a literatura menciona diversas características, visando sintetizar as principais, foi elaborado o Quadro 1.

Síntese dos principais fatores envolvidos na programação mencionadas na literatura		
Habilidade/Competência	Definição/relação com a programação	Autores
Raciocínio lógico	Processo de estruturação do pensamento de acordo com as normas da lógica que permite chegar a uma determinada conclusão ou resolver um problema.	(DEREMER, 1993) (WILSON; SHROCK, 2001) (RAABE; SILVA, 2005) (PEREIRA JÚNIOR *et al.*, 2005) (SANTOS; COSTA, 2006) (GOMES; MENDES, 2007)
Resolução de problemas	Habilidade que envolve processos cognitivos como criatividade e racionalidade, a partir de um conjunto de meta-habilidades mentais tais como a abstração, inferência, dedução entre outros, além disso, esta habilidade está diretamente ligada a leitura e compreensão.	(FALKEMBACH; AMORETTI; *et al.*, 2003) (GOMES; MENDES, 2007) (MARTINS; MENDES; FIGUEIREDO, 2010)
Abstração	Processo de generalização ao reduzir o conteúdo de informação de um conceito, a fim de reter apenas uma informação que seja relevante para um propósito particular.	(PEREIRA JÚNIOR; RAPKIEWICZ, 2004) (RAABE; SILVA, 2005) (HABERMAN; MULLER, 2008) (PIVA JR; FREITAS, 2010) (MARTINS; MENDES; FIGUEIREDO, 2010)
Habilidade matemática	Existe a crença de que os conceitos a fim de dominar problemas matemáticos são semelhantes aos de programação	(HENDERSON, 1987) (DEREMER, 1993) (WILSON; SHROCK, 2001) (KOLIVER; DORNELES; CASA, 2004) (RAABE; SILVA, 2005) (GOMES, A.; MENDES, 2007) (MOTA *et al.*, 2009) (BYRNE; LYONS, 2001)

Quadro 1: Síntese dos principais fatores envolvidos na programação mencionadas na literatura

Fonte: Autoria própria – com base na revisão da literatura

Além destes fatores presentes no Quadro 1, Ambrósio et al. (2011), sintetiza um outro conjunto de características que também são valorizados na programação, dentre as quais vale destacar: leitura e compreensão, o raciocínio crítico, o pensamento sistêmico, identificação, planejamento, a criatividade e curiosidade intelectual, raciocínio condicional, o pensamento procedimental e o raciocínio temporal, o raciocínio analítico e o raciocino analógico, silogístico e combinatório.

Desta forma, programadores experientes possuem um conjunto de recursos (habilidades, competências, experiências) que podem utilizar em qualquer parte do processo da solução do problema para computador.

O aprendizado de programação está associado ao desenvolvimento deste conjunto de recursos.

2.2 ENSINO DE PROGRAMAÇÃO

No Brasil, o ensino de programação ocorre principalmente por cursos superiores nas áreas de Computação e Informática. Esse conteúdo também é trabalhado em certos cursos técnicos profissionalizantes subsequentes ou na modalidade integrada ao ensino médio.

Segundo as diretrizes curriculares para os cursos de informática e computação do MEC, a matéria de Programação faz parte da área de formação básica em Computação e Informática, juntamente com as matérias de Computação, Algoritmos e Arquitetura de computadores (BRASIL, 2001).

Geralmente o processo inicial de ensino/aprendizagem de programação ocorre por meio de um conjunto de disciplinas introdutórias que são identificadas por nomes como: Algoritmos, Lógica de programação, Linguagem de programação, Técnicas de programação entre outras.

Estas disciplinas têm como objetivo fornecer aos alunos os conceitos básicos de programação, isto representa um pequeno conjunto de comandos e conceitos, dos quais os alunos devem utilizar para implementar soluções para um determinado problema e representa-las num ambiente computacional.

A programação é possivelmente uma das únicas matérias que tenta ensinar métodos de resolução de problemas gerais, visto que em cursos como filosofia, matemática e física, o estudante lê e aprende conceitos para se resolver problemas específicos não sendo capaz de generalizá-los (GRIES, 1974).

Alguns aspectos fundamentais em um curso introdutório seriam: 1º Como resolver problemas; 2º Como descrever uma solução algorítmica; 3º Como verificar se um algoritmo está correto (GRIES, 1974; GOMES; HENRIQUES; MENDES, 2008).

De modo geral, nestas disciplinas introdutórias, o conteúdo tratado inclui principalmente a descrição dos passos necessários para se solucionar um problema, entradas e saídas, constantes e variáveis, tipos primitivos de dados, instrução de atribuição, operadores aritméticos, relacionais e lógicos, estruturas simples de controle de fluxo, decisão e repetição. Em algumas salas de aula evita-se o contato com uma linguagem de programação nesse instante.

Neste contexto, atividades práticas podem ter uma importante função, pois conforme apontam (HABERMAN; MULLER, 2008; BENNEDSSEN; CASPERSEN, 2008), um dos entraves no ensino de programação e a forte carga de conceitos abstratos presentes no processo da elaboração e implementação da solução do problema no ambiente computacional.

Sendo assim, e importante oportunizar ao estudante que observe a execução de um programa e seus resultados, o que lhe oferece mais um caminho para tratar dificuldades encontradas em aspectos teóricos, uma vez que outra limitação encontrada em algumas salas de aula e justamente deixar de enfatizar a solução de problemas para se concentrar em conceitos teóricos (NOBRE; MENEZES, 2002; KOLIVER; DORNELES; CASA, 2004; GOMES; MENDES, 2007).

Outra característica interessante relacionada aos alunos, é que percebe-se que num determinado momento alguns deles apresentam um ganho significativo de aprendizagem e avançam sem problemas durante os conteúdos, e no entanto outros alunos mesmo se esforçando não conseguem atingir os objetivos propostos pela disciplina (AMBRÓSIO et al., 2011).

Dunican (2002), identifica três categorias de estudantes iniciantes em programação:

I. Alunos que não têm a aptidão para compreender os conceitos básicos, este é muitas vezes resultado de uma escolha equivocada do curso;
II. Alunos que podem captar os conceitos essenciais se expostos a abordagens de ensino eficazes; e
III. Alunos que são totalmente confortáveis com a natureza abstrata de conceitos de programação.

Seguindo essa classificação e assumindo que esteja correta o professor deve ficar atento, e se possível identificar os alunos pertencentes à segunda categoria, e assegurar condições adequadas de ensino/aprendizagem para que a maioria destes possam progredir. Entretanto, esperar que o processo de ensino/aprendizagem de

programação, possa ser considerado apenas uma receita didática é seguramente um erro.

Conforme apresentado durante esta seção, é possível observar que o aprendizado de programação de computadores é uma das bases na formação de estudantes dos cursos de informática e computação. Entretanto aprender a programar computadores não é uma tarefa simples, tampouco trivial (JENKINS, 2002; ROBINS; ROUNTREE; ROUNTREE, 2003).

Ao longo dos anos, o processo de ensino/aprendizagem dos fundamentos de programação de computadores, tem se mostrado difícil para estudantes e professores, estas dificuldades levaram muitos professores e pesquisadores a estudar suas causas e propor soluções variadas que visam de alguma maneira atenuar estes problemas. Grande parte destas pesquisas são voltadas especialmente para estudantes novatos em programação, como em Brusilovsky et al. (1994), Soloway et al. (1983), Delgado et al. (2004), Pereira Júnior; Rapkiewicz (2004), apenas para citar alguns trabalhos.

O grande volume de literatura referente à programação introdutória é reflexo das dificuldades relacionadas ao tema (SHEARD et al., 2009) que faz com que o ensino de programação de computadores seja considerado um dos sete grandes desafios na educação em informática (SLEEMAN, 1986).

A literatura apresenta uma série de fatores, que podem de alguma maneira contribuir para as dificuldades apresentadas no processo de ensino/aprendizagem de programação de computadores, a subseção a seguir busca apresentar parte destas dificuldades destacadas na literatura.

2.2.1 Dificuldades no ensino/aprendizagem de programação

A importância e destaque das pesquisas relacionadas as dificuldades apresentadas por alunos e professores durante o processo de ensino/aprendizagem de programação originam-se em dois fatores frequentemente encontrados na literatura, sendo:

I. Os elevados níveis de insucesso; e
II. As elevadas taxas de desistência e até mesmo abandono do curso;

Essas disciplinas podem ser consideradas um dos gargalos existentes nos cursos, dificultando ou até mesmo impedindo a continuidade dos estudantes (HINTERHOLZ JUNIOR, 2006; PEREIRA JÚNIOR et al., 2005; RAPKIEWICZ et al., 2006).

As dificuldades abrangem os quatro integrantes envolvidos no processo instituição, professor, o aluno, e os conteúdos.

O Quadro 2 apresenta uma síntese das dificuldades inerentes as instituições de ensino mencionadas na literatura.

Síntese das dificuldades inerentes às instituições de ensino mencionadas na literatura	
Dificuldades	Autores
Concepção errada da grade curricular	(DELGADO et al., 2004) (KOLIVER; DORNELES; CASA, 2004) (PEARS et al., 2007)
Turmas demasiadamente grandes	(TOBAR et al., 2001) (JENKINS, 2001) (PARREIRA; FORSTER, 2002) (BIGGS, 2003) (PIMENTEL; FRANÇA; OMAR, 2003) (PEREIRA JÚNIOR et al., 2005) (GOME; HENRIQUES; MENDES, 2008)
Suporte institucional, ausência de ambientes adequados (laboratórios de informática) e ferramentas	(DENNING et al., 1981) (ROBERTS, 1999) (CATTERALL, 2008)

Quadro 2: Síntese das dificuldades inerentes às instituições de ensino mencionadas na literatura
Fonte: Autoria própria, a partir da revisão da literatura

Conforme o Quadro 2, entre as variáveis referentes às instituições de ensino vale destacar:

A metodologia empregada pelas instituições e a grande quantidade de alunos presentes na disciplina e ausência de ambientes apropriados.

O Quadro 3 a seguir contém uma síntese das dificuldades inerentes aos conteúdos mencionadas na literatura.

Síntese das dificuldades inerentes aos conteúdos mencionadas na literatura	
Dificuldades	Autores
Forte carga de conceitos abstratos	(HABERMAN; MULLER, 2008) (BENNEDSSEN; CASPERSEN, 2008)
Restrição de tempo	(KOLIVER; DORNELES; CASA, 2004) (ROCHA *et al.*, 2010)
Métodos de avaliação inadequados	(ROCHA *et al.*, 2010) (RODRIGUES JÚNIOR, 2004)

Quadro 3: Síntese das dificuldades inerentes aos conteúdos mencionadas na literatura
Fonte: Autoria própria, a partir da revisão da literatura

Tradicionalmente, os conteúdos das disciplinas introdutórias a programação são apresentado da forma mais comum, como em todas as áreas de conhecimento, ou seja, apresentação da teoria, apresentação de exemplos e proposição de exercícios práticos inicialmente simples e paulatinamente complexos (RODRIGUES JÚNIOR, 2004).

O modelo tradicional de ensino de programação não consegue facilmente motivar os alunos a se interessar pela disciplina, entre outras razões, pois não é clara para os mesmos a importância destes conteúdos para sua formação (BORGES, 2000).

Essa falta de motivação dos alunos possivelmente pode ser agravada especialmente quando os conteúdos são apresentados sem o auxílio de uma linguagem de programação, podendo dificultar o entendimento e utilidade dos conteúdos apresentados (AUSUBEL; NOVAK; HANESIAN, 1980; HABERMAN; MULLER, 2008).

A partir da síntese dos estudos das principais dificuldades referentes aos conteúdos, passamos a apresentar a seguir o Quadro 4 contendo uma síntese das dificuldades inerentes aos professores.

Síntese das dificuldades inerentes aos professores mencionadas na literatura	
Dificuldades	Autores
Prática docente	(BIGGS, 1999) (JENKINS, 2001) (GIRAFFA; MARCZAK; ALMEIDA, 2003) (PIMENTEL; FRANÇA; OMAR, 2003) (CASPERSEN; BENNEDSEN, 2007) (RODRIGUES JÚNIOR, 2004)
Ausência de metodologias e/ou práticas de ensino adequadas	(JENKINS, 2001) (GIRAFFA; MARCZAK; ALMEIDA, 2003) (PIMENTEL; FRANÇA; OMAR, 2003) (RODRIGUES JÚNIOR, 2004) (SANTOS; COSTA, 2006) (CASPERSEN; BENNEDSEN, 2007) (GOMES; HENRIQUES; MENDES, 2008)
Restrição de tempo	(KOLIVER; DORNELES; CASA, 2004) (ROCHA et al., 2010)
Impossibilidade de acompanhamento individual da aprendizagem dos alunos	(JENKINS, 2001) (TOBAR et al., 2001) (BIGGS, 2003) (RAABE; SILVA, 2005) (ROCHA et al., 2010)
Ensino sem enfoque na resolução de problemas	(NOBRE; MENEZES, 2002) (KOLIVER; DORNELES; CASA, 2004) (GOMES; MENDES, 2007)
Abordagem pouco motivadora	(BORGES, 2000) (KOLIVER; DORNELES; CASA, 2004) (RODRIGUES JÚNIOR, 2004)
Heterogeneidade dos alunos	(ROBERTS, 1999) (ROBERTS, 2001) (KOLIVER; DORNELES; CASA, 2004)

Quadro 4: Síntese das dificuldades inerentes aos professores mencionadas na literatura
Fonte: Autoria própria, a partir da revisão da literatura

O uso de metodologias e/ou práticas de ensino adequadas pode contribuir para atenuar algumas destas dificuldades, para isto é importante que o docente vise um ensino contextualizado com enfoque na resolução de problemas, e sempre que possível utilizando-se de abordagens motivadoras.

Outros fatores como heterogeneidade dos alunos, impossibilidade de acompanhamento individual da aprendizagem e restrição de tempo se mostram mais difíceis de contornar.

Complementando este quadro, Nobre e Menezes (2002) apontam as seguintes dificuldades vivenciadas pelos professores:

I. Reconhecer as habilidades inatas de seus alunos;
II. Apresentar técnicas para soluções de problemas;
III. Trabalhar a capacidade de abstração do aluno, tanto na busca das possíveis soluções como na escolha da estrutura de dados a ser utilizada; e
IV. Promover a cooperação e a colaboração entre os alunos.

O Quadro 5 apresenta uma síntese das dificuldades inerentes aos alunos mencionadas na literatura.

Síntese das dificuldades discutidas inerentes aos alunos	
Dificuldades	Autores
Dificuldade em leitura e interpretação os enunciados	(DEREMER, 1993) (WILSON; SHROCK, 2001) (FALKEMBACH; AMORETTI; et al., 2003) (DELGADO et al., 2005) (GOMES; MENDES, 2007)
Dificuldade no desenvolvimento do raciocínio lógico	(DEREMER, 1993) (WILSON; SHROCK, 2001) (RAABE; SILVA, 2005) (PEREIRA JÚNIOR et al., 2005) (SANTOS; COSTA, 2006) (GOMES; MENDES, 2007)
Poucas habilidades na resolução de problemas	(FALKEMBACH; AMORETTI; et al., 2003) (GOMES, A.; MENDES, 2007) (MARTINS; MENDES; FIGUEIREDO, 2010)
Baixa capacidade de abstração	(PEREIRA JÚNIOR; RAPKIEWICZ, 2004) (RAABE; SILVA, 2005) (HABERMAN; MULLER, 2008) (PIVA JÚNIOR; FREITAS, 2010) (MARTINS; MENDES; FIGUEIREDO, 2010)
Baixo nível de conhecimento em matemática	(HENDERSON, 1987) (DEREMER, 1993) (WILSON; SHROCK, 2001) (KOLIVER; DORNELES; CASA, 2004) (RAABE; SILVA, 2005) (GOMES; MENDES, 2007) (MOTA et al., 2009)
Hábitos de estudo equivocados	(BIGGS, 1999) (JENKINS, 2001) (RODRIGUES, 2002) (PEREIRA JÚNIOR; RAPKIEWICZ, 2004) (KOLIVER; DORNELES; CASA, 2004)
Estilos de aprendizagem diferentes	(BIGGS, 1999) (PEREIRA JÚNIOR et al., 2005) (CÂNDIDA; MARCELINO; MENDES, 2007) (CASPERSEN; BENNEDSEN, 2007)
Pouca motivação	(JENKINS, 2001) (RODRIGUES, 2002) (RODRIGUES JÚNIOR, 2004) (GOMES; MENDES, 2007) (MARTINS; MENDES; FIGUEIREDO, 2010)
Problemas extraclasses (vida pessoal), adaptação a nova vida acadêmica.	(RODRIGUES, 2002) (PEREIRA JÚNIOR et al., 2005)

Síntese das dificuldades discutidas inerentes aos alunos - continuação	
Dificuldades	Autores
Pré-conceito atribuído as disciplinas de programação.	(PEREIRA JÚNIOR; RAPKIEWICZ, 2004)
Falta de persistência, ou pouco empenho.	(JENKINS, 2001) (FORTE; GUZDIAL, 2005) (GOMES; MENDES, 2007)

Quadro 5: Síntese das dificuldades inerentes aos alunos mencionadas na literatura
Fonte: Autoria própria, a partir da revisão da literatura

A extensa lista de fatores relacionados com a disciplina de programação pode ser organizada em três grandes grupos de natureza de problemas, sendo eles: didática, cognitiva e afetiva (RAABE; SILVA, 2005). Essa classificação é mostrada no Quadro 6.

Síntese dos tipos de problemas que podem contribuir para as dificuldades de ensino/aprendizagem de programação	
Problemas de natureza didática	
Problema	Descrição
Grande número de alunos	Turmas grandes dificultam o acompanhamento individualizado do aluno e avaliações
Dificuldade do professor compreender a lógica do aluno	O uso de um raciocínio lógico pronto para solução de problemas, dificulta que o professor compreenda a lógica individual do aluno, assim geralmente o professor tende a direcionar a solução do aluno
Diferenças de experiências e ritmo de aprendizagem entre os alunos	Turmas heterogêneas, tornam mais complexo administrar a disciplina.
Ambiente de realização de avaliações	Há fatores que não favorecem a concentração e o raciocínio do estudante, como tempo limitado, pressão e stress
Pouco uso dos monitores da disciplina	Alunos com dificuldades, muitas vezes não aproveitam a ajuda de monitores
Ausência de bons materiais	Alguns professores enfatizam livros e estes nem sempre apresentam o conteúdo de forma apropriada a cada estudante
Alunos desorientados na escolha do curso	Alunos novatos podem não ter uma visão adequada sobre o perfil do curso, levando a um ambiente de incompreensão e descaso frente aos desafios impostos pela disciplina
Problemas de natureza cognitiva	
Alunos sem o perfil para resolução de problemas	Muitos alunos não desenvolveram adequadamente as estratégias necessárias para a resolução de problemas durante os estudos anteriores, e por isso apresentam maior dificuldade em solucionar os algoritmos propostos
Alunos sem base operatória formal	Aparentemente o raciocínio operatório formal, base para compreensão do raciocínio lógico, não foi adequadamente desenvolvido nos estudos anteriores
Conteúdo sem proximidade com o conteúdo escolar	A lógica algorítmica é algo totalmente novo para a maioria dos alunos, e com isso eles não conseguem estabelecer relações com os conteúdos já apreendidos anteriormente, principalmente com a matemática
Problemas de natureza afetiva	
Ocasionais	Problemas esporádicos de ordem pessoal que afetam o aluno impedindo que este consiga se concentrar nas explicações e/ou influenciando em seu desempenho nas avaliações
Constantes	Problemas de ordem afetiva que se manifestam durante todo o decorrer da disciplina em maior ou menor grau. Baixa autoestima, pouca motivação, aversão ao conteúdo ou ao professor, insegurança são exemplos de emoções que podem afetar negativamente a aprendizagem do aluno

Quadro 6: Síntese dos tipos de problemas que podem contribuir para as dificuldades de ensino/aprendizagem de programação

Fonte: Adaptado de (RAABE; SILVA, 2005)

A origem dos problemas associados ao processo ensino/aprendizagem programação é muito ampla, uma vez que envolve diversas variáveis. Analisando estas questões sob a ótica de uma pedagogia que encara a aprendizagem como um processo de construção de conhecimento, é possível identificar e intervir nas diferentes etapas deste processo, no entanto, essas intervenções não garante sucesso no resultado final.

Esta subseção abordou as dificuldades apresentadas no processo de ensino/aprendizagem de programação, a partir da subseção seguinte apresentaremos algumas das propostas presentes na literatura que visam atenuar estas dificuldades.

2.3 PROPOSTAS NA LITERATURA PARA TRATAR O PROBLEMA

A literatura disponibiliza uma série de propostas que visam contribuir com o processo de ensino/aprendizagem de programação, porém nenhuma destas se mostrou completa ou mesmo genérica a ponto de sanar os problemas de aprendizado de programação que ainda persistem.

Geralmente as propostas presentes para tratar este problema na literatura se encaixam em três principais vertentes, sendo: Ferramentas, Metodologias e a união de ambas (PEREIRA JÚNIOR; RAPKIEWICZ, 2004). Essas três possibilidades serão analisadas nesta seção.

Um outro ponto, que pode ser acrescentado em qualquer uma das três linhas mencionadas, é tratar aspectos motivacionais e afetivos, que guardam uma relação direta do envolvimento e aprendizado (SHELL *et al.*, 2013). Diversos estudos relatam que o uso de ferramentas e/ou metodologias podem melhorar o engajamento dos estudantes em aprender (SCAICO *et al.*, 2013; WOLBER, 2011).

A questão de motivação é bastante ampla, mas a princípio não tratá-la explicitamente não invalida um trabalho com as outras possibilidades. Dada a amplitude do assunto, decidiu-se não explora-lo neste trabalho, deixando-se sua exploração para um projeto futuro.

Portanto, a seguir é apresentado algumas propostas na literatura para tratar o problema.

2.3.1 Ferramentas

A atividade de programação envolve a escrita de código, que pode acontecer dentro de ferramentas desenvolvidas para esse fim. Infelizmente grande parte destes ambientes são elaborados mais para fins profissionais do que pedagógicos. Assim, um dos caminhos adotados por professores é buscar ferramentas desenvolvidas com o intuito de ajudar estudantes.

O Quadro 7 apresenta algumas ferramentas direcionadas ao ensino/aprendizagem programação mencionadas na literatura.

Síntese das ferramentas mencionadas na literatura		
Ferramenta	Autor e/ou referências bibliográficas	Descrição
LOGO	(PAPERT, 1983)	Desenvolvida por Seymour Papert como resultado da influência do construtivismo de Piaget. O LOGO ganhou fama como a primeira linguagem de programação para crianças, devido ao seu apelo lúdico, esta ferramenta pode ser definida como um micromundo programável onde estudantes podem exercitar as suas competências.
ASTRAL	(GARCIA; REZENDE; CALHEIROS, 1997)	É um ambiente de programação para produção de animações de algoritmos e de estruturas de dados com propósito instrucional. O ambiente foi desenvolvido no Instituto de Computação da UNICAMP.
SICAS	(MENDES; GOMES, 2000)	Ferramentas de simulação da lógica de algoritmos, para implementar a dinâmica dos testes de mesa e fluxogramas a partir da utilização de recursos visuais.
ALICE	(COOPER; DANN; PAUSCH, 2000) (COOPER; DANN; PAUSCH, 2003)	Plataforma desenvolvida na Universidade Carnegie Mellon, projetada para ensinar alunos de programação orientada a objetos, envolvendo-os em algo divertido, tais como fazer filmes de animação e jogos. Esta ferramenta fornece animação 3D e manipulação direta dos elementos de uma linguagem de programação. A ferramenta remove a necessidade dos alunos em escrever código e lidar com a sintaxe.
AMBAP	(ALMEIDA et al., 2002)	Ferramenta que inclui uma linguagem simples e o seu processador, que permitem ao aluno construir seu programa, executando-o, depurando-o e tendo a oportunidade de entender conceitos, tais como: variáveis, comandos, recursão. Segundo os autores usa um processo de simulação, sem se preocupar, com os detalhes inerentes à implementação comprometida com características específicas da máquina, comumente encontrados nas linguagens de programação convencionais
GREENFOOT	(HENRIKSEN; KÖLLING, 2004)	Ferramenta desenvolvida pelo grupo de educação em computação da Kent University, nos Estados Unidos, é um ambiente de desenvolvimento Java interativo projetado principalmente para fins educacionais, permitindo facilmente o desenvolvimento de aplicativos gráficos bidimensionais, como simulações e jogos interativos.
TBC-AED e TBC-AED/Web	(SANTOS; COSTA, 2005)	Ferramenta de ensino; inclui links explicativos, evitando a necessidade de tutorial; interface gráfica para professor apresentar conceitos como faria com transparências; apresenta processos gráficos passo a passo, com elementos numéricos e legendas explicativas, que ilustram etapas do processo de apresentação de algoritmos.
jGRASP	(JAIN et al., 2006)	Ferramenta que gera efeitos visuais dinâmicos baseados no estado de objetos e variáveis primitivas da linguagem de programação Java

Síntese das ferramentas mencionadas na literatura – continuação I		
Ferramenta	Autor e/ou referências bibliográficas	Descrição
VisuAlg	(SOUZA, 2009)	É um aplicativo para digitar, executar e depurar pseudocódigo, fornecendo também recursos como execução passo a passo, visualização do conteúdo das variáveis, exame da pilha de ativação no caso de subprogramas, contador de execuções de cada linha do programa, etc.
SCRATCH	(RESNICK et al., 2009)	Ferramenta desenvolvida dentro do mesmo contexto lúdico proposto pelo LOGO, nesta utiliza-se a metáfora na qual as instruções de um programa são elaboradas a partir de blocos de comandos que se encaixam uns nos outros. Pode ser usada para elaboração de jogos de computador, histórias interativas, obras de arte gráfica e animação por computador, e outros projetos de multimídia.
Portugol IDE	(MANSO; OLIVEIRA; MARQUES, 2009)	A ferramenta possui duas linguagens portugol e fluxograma. Permite realizar as operações necessárias para a codificação de algoritmos simples, são compatíveis entre si e é possível alternar entre as duas nas fases de edição, execução e depuração.
AIIP	(GOMES et al., 2011)	Um ambiente de ensino com conceitos e exemplos sobre programação, com possibilidade de resolver problemas armazenados em uma base de dados. Inclui exibição de dicas durante a resolução dos problemas, apresentação de feedback, sistema de estatísticas apresentadas ao usuário a cada unidade de conhecimento finalizada, assistente inteligente, que acompanha o aluno em toda sua interação com o ambiente, monitorando-o e quando necessário interrompendo-o para informá-lo sobre algum aspecto relevante em seu aprendizado.
APIN	(DIM; ROCHA, 2011)	A ferramenta combina o esquema de software tutorial-exercício, contendo material didático para ensino de diferentes lógicas e exercícios relativos aos tópicos estudados, com o esquema de jogos, fornecendo em um ambiente lúdico e didático diversas possibilidades de aplicação de conhecimentos de lógica para estímulo do raciocínio e solução de problemas.
iVprog	(BRANDÃO; BRANDÃO; RIBEIRO, 2012)	A ferramenta visa diminuir a sobrecarga cognitiva para os alunos das disciplinas de Introdução a Programação, a fim de permitir que eles se concentrem em organizar o pensamento e criar boas estratégias para resolver problemas.
Code Academy	(LAUBER, 2012)	Site que possui uma plataforma interativa oferece e aulas gratuitas de codificação em linguagens de programação como jQuery, Javascript, Python, Ruby, PHP, bem como as linguagens de marcação, incluindo HTML e CSS. O serviço funciona de maneira que os usuários progridam nas lições, com códigos de dificuldade crescente.
App Inventor	(FINIZOLA *et al.*, 2014; WOLBER, 2011)	A plataforma fornece a possibilidade dos estudantes praticarem conceitos de algoritmos para elaborarem aplicativos que serão utilizados em seus dispositivos móveis.

Quadro 7: Síntese das ferramentas mencionadas na literatura
Fonte: Autoria própria com base na literatura

Além das ferramentas observadas conforme o Quadro 7, a literatura nos remete também a diversos outros tipos de materiais desenvolvidos com a mesma finalidade.

Pode-se destacar elaboração de instrumentos e testes cognitivos para a avaliação do nível de conhecimentos (BERGERSEN; GUSTAFSSON, 2011; CHATZOPOULOU; ECONOMIDES, 2010).

Outros estudos destacam-se pela investigação dos aspectos cognitivos principalmente relacionados a motivação, visando a construção de instrumentos de mediação, para que sejam propostas ações de intervenção positivas em sala de aula (IEPSEN; BERCHT; REATEGUI, 2010; LIMA; LEAL, 2013; ZAMBON; SOUZA; ROSE, 2012).

Observa-se também a elaboração de instrumentos que visam avaliar aspectos que possam auxiliar o professor, na identificação dos níveis de confiança dos estudantes (COMPEAU; HIGGINS, 1995; EACHUS; CASSIDY, 2002).

2.3.2 Estratégias/metodologias

Esta subseção apresenta alguns trabalhos que se concentram em estratégias e/ou metodologias de ensino.

A literatura tem relatado esforços de muitos professores e pesquisadores na busca de estratégias e metodologias para apoiar a aprendizagem da programação (PEARS et al., 2007).

Uma abordagem muito comum consiste em ministrar a disciplina com grande carga de definições e conceitos, além disso, há grande destaque para a formalização por meio de exercícios que em muitos casos possuem pouco ou nenhuma relação com a programação (BORGES, 2000; PEREIRA JÚNIOR; RAPKIEWICZ, 2004). Posteriormente o professor apresenta uma ferramenta para formalização da solução, normalmente em forma de pseudocódigo ou fluxograma.

Um fator encontrado com certa frequência é a utilização exagerada da repetição de problemas que torna o processo de ensino e aprendizagem monótono e cansativo (DETERS et al., 2008; RAPKIEWICZ et al., 2006). Esta metodologia ou

estratégia de ensino de algoritmos e programação não contribui para a motivação dos alunos (LIMA; LEAL, 2013).

Percebendo este cenário, alguns professores e pesquisadores propõem novas estratégias e metodologias, que visam atenuar as dificuldades encontradas no processo e criar melhores condições de aprendizagem aos estudantes (SANTOS; COSTA, 2006).

O Quadro 8 contém uma síntese extraída da literatura.

Síntese das propostas baseadas em estratégias ou metodologias de ensino mencionadas na literatura		
Estratégia/Metodologia	Autores e referências bibliográficas	Descrição
Ensino como uma disciplina matemática	(DIJKSTRA, 1989)	O ponto defendido por Dijkstra é que programas são em sua essência funções que produzem um dado de saída a partir de um dado de entrada. Ele sugere que programação deve ser ensinada como uma disciplina matemática, com forte foco em provas e correção.
Estudos de caso	(LINN; CLANCY, 1992)	O método propõe ênfase no processo de programação, priorizando aspectos como a declaração do problema, a descrição dos processos utilizados para resolvê-lo e o perfil de codificação. O ponto defendido pelos autores é que muitos livros de programação tendem a enfatizar o produto, mas não o próprio processo de programação.
Programação em pares	(WILLIAMS et al., 2000) (MCDOWELL et al., 2002)	Estabelece pares de estudantes, para que possam discutir e colaborar entre si na resolução dos problemas.
Técnica ABP	(KOZAK; EBERSPÄCHER, 2001)	Baseia-se na solução de um problema real por pequenos grupos de discussão conduzido por um "tutor" no papel de "aprendiz sênior". Esse arranjo promove o processamento da informação pelo grupo.
Construtivismo	(BEN-ARI, 2001) (FERNANDES, 2002) (GOMES; SCHIMIGUEL, 2010)	Preconiza que o aluno passe pelos seguintes processos: Perturbação do equilíbrio dos seus conceitos; Conservação que é a compensação da modificação simultâneas do objeto; Assimilação x acomodação do mesmo conceito. A metodologia propõe: A utilização de situações problema que envolvam a formulação de hipóteses investigação e/ou a comparação; A apresentação de outros caminhos para solucionar um determinado problema; A permissão para que o aprendiz construa; Devendo ser adaptável ao nível do aprendiz
Separação por fases	(RODRIGUES JÚNIOR, 2004)	O autor propõe a separação da tarefa de construção de programas em duas fases, sendo: 1º fase de análise do problema e 2º fase de implementação
Hierarquia de cinco níveis	(ECKERDAL; THUNÉ; BERGLUND, 2005)	O método propõe uma hierarquia de cinco níveis: 1) aprender a ler e escrever uma linguagem de programação, 2) a aprendizagem de uma forma de pensamento alinhado com uma linguagem de programação, 3) entender o que são programas de computador e como eles aparecem na vida cotidiana, 4) a aprendizagem de uma maneira de pensar que permite a resolução de problemas, 5) aprender uma habilidade que pode ser usada fora das aulas de programação.
Aprendizagem baseada em projetos hands-on	(LEWANDOWSKI; JOHNSON; GOLDWEBER, 2005)	A abordagem prioriza atividades práticas. Nesta estratégia, a disciplina de Ciência da Computação I, da ênfase em atividades práticas enquanto em Estruturas de Dados e Algoritmos, é oportunizado aos alunos escolher um dos vários projetos.

Síntese das propostas baseadas em estratégias ou metodologias de ensino mencionadas na literatura – continuação I		
Estratégia/Metodologia	Autores e referências bibliográficas	Descrição
Iniciação em lógica de programação em nível médio	(PEREIRA JÚNIOR et al., 2005)	Propõe a iniciação do aluno em lógica de programação aconteça no nível médio, visando estimular o aluno a optar por cursos na área de computação e criar condições para que desenvolva soluções lógicas e matemáticas.
Interdisciplinaridade	(YAMAMOTO et al., 2005)	Visa a abordagem do ensino de programação, utilizando de projetos integrados entre disciplinas.
Uso de jogos	(KÖLLING; HENRIKSEN, 2005) (RAPKIEWICZ et al., 2006)	Visa promover de forma lúdica, a formação de novas atitudes a fim de diminuir as dificuldades encontradas pelos alunos e facilitar o processo de aprendizagem. Os autores argumentam que a utilização de jogos de forma lúdica propicia flexibilidade e criatividade fazendo o aluno explorar, pesquisar, encorajando o pensamento criativo.
PBL- Problem Based Learnig	(NUUTILA et al., 2008)	O PBL é definido como um processo de ensino que tem foco em atividades centradas nos alunos e usa problemas concretos para motivá-los.
ERM2C	(CAMPOS, 2010)	A metodologia possui 5 etapas (Entender, Revisar, Melhorar, Complementar e Construir) e cada uma delas possui 3 níveis distintos (básico, intermediário e avançado). Ao final do desenvolvimento da etapa. Entender, o discente estará apto a ler/entender um algoritmo ao Revisar, identifica o conjunto de ações que produzem o resultado final. Na etapa Melhorar, o aluno proporá modificações para garantir ou melhorar o resultado final. Por fim a complementação visa evoluir ou adaptar a solução para atender a novas necessidades.
Sistema personalizado de ensino	(ROCHA et al., 2010)	Proposta denominada Personalized System of Instruction – PSI, embasada em princípios da análise comportamental de Keller (1968), dentre os procedimentos desta proposta, destacam-se: Disponibilização fácil de materiais e exercícios; Organização em Níveis; Mobilização de professores e monitores; Disponibilização permanente a orientação e monitoramento; Encontros periódicos de estudo assistido; Avaliação em cada nível. Além disso, esta proposta se baseia na flexibilização do tempo e de conteúdos, onde o aluno só pode avançar quando alcança 100% de aproveitamento no nível anterior

Quadro 8: Síntese das propostas baseadas em estratégias e/ou metodologias de ensino mencionadas na literatura
Fonte: Autoria própria com base na revisão da literatura

Além destas propostas apresentadas no Quadro 8, merecem destaque em nossa pesquisa os estudos de Rodrigues Júnior (2004), que propõe uma mudança na metodologia, onde temas como motivação, mudança na forma de avaliação, relacionamento professor-aluno, material utilizado e preparação das aulas devem ser revistos, e Dunican (2002), que sugere uma técnica baseada na utilização de exemplos que sejam familiares aos estudantes para explicar com mais facilidade os conceitos abstratos da programação.

2.3.3 Ferramentas e estratégias

Ferramentas aliadas a estratégias com propósitos pedagógicos são de origem essencialmente de trabalhos acadêmicos. São ambientes menos complexos que os ambientes profissionais, tentando simplificar a aprendizagem inicial dos conceitos de programação. Por vezes incorporam ainda funcionalidades importantes para programadores pouco experientes.

A seguir o Quadro 9 apresenta uma síntese de trabalhos nessa linha.

Síntese de algumas ferramentas associadas a estratégias e/ou metodologias de ensino mencionadas na literatura		
Ferramenta	Estratégia/Metodologia	Autores e/ou referências bibliográficas
SAAP	Trabalho colaborativo	(CASTRO, 2002)
A4	Ascendente, utilizada no ambiente, para trabalhar de forma mais eficiente os processos cognitivos referentes à Resolução de Problemas, em especial a abstração e a formalização dos procedimentos necessários ao desenvolvimento de um algoritmo	(FALKEMBACH; ARAUJO; *et al.*, 2003)
VDSP	Abordagem colaborativa que parte do princípio de que o conflito sócio cognitivo entre os aprendizes pode melhorar o seu aprendizado	(FARIA; COELLO, 2004)
SICAS-COL	Trabalho colaborativo	(REBELO, 2006)
AVEP	Ensino e aprendizagem baseada na resolução de problemas	(PEREIRA JÚNIOR *et al.*, 2006)

Quadro 9: Síntese de algumas ferramentas associadas a estratégias e/ou metodologias de ensino mencionadas na literatura
Fonte: Autoria própria – baseado na revisão de literatura

Há evidências de que a união de ferramentas computacionais e estratégias pedagógicas adequadas culminam em melhores resultados, se comparados a utilização de ferramentas de maneira isoladas. Entretanto, o que se observa é uma tendência em tratá-las separadamente (PEREIRA JÚNIOR; RAPKIEWICZ, 2004).

2.4 APRENDIZAGEM SIGNIFICATIVA

Aprendizagem significativa é o conceito central da teoria da aprendizagem de David Ausubel, proposta na década de 60, (AUSUBEL; NOVAK; HANESIAN, 1968; AUSUBEL, 1963), nesta teoria o autor destaca que a aprendizagem significativa é o mecanismo humano para adquirir e armazenar a vasta quantidade de ideias e informações representadas em qualquer campo de conhecimento.

Ausubel (2003, p.vi), destaca que:

> O conhecimento é significativo por definição. É o produto significativo de um processo psicológico cognitivo (“saber”) que envolve a interação entre ideias “logicamente” (culturalmente) significativas, ideias anteriores (“ancoradas”) relevantes da estrutura cognitiva particular do aprendiz (ou estrutura dos conhecimentos deste) e o “mecanismo” mental do mesmo para aprender de forma significativa ou para adquirir e reter conhecimentos.

Essa teoria preconiza que o conhecimento se organiza em estruturas cognitivas, que são conjuntos de conhecimentos que o indivíduo possui sobre um determinado assunto. A aprendizagem torna-se significativa quando os conhecimentos anteriores são inter-relacionados ao novo conteúdo a ser estudado o qual passa a ser incorporado às estruturas de conhecimento, adquirindo significado especial.

Este conceito não consiste numa simples associação e sim uma interação relevante entre os conhecimentos que se possui e os que será conhecido (MOREIRA, 1999).

Sendo assim, a aprendizagem significativa é um processo de mudança do conhecimento alterando a estrutura cognitiva do aprendiz, modificando os conceitos pré-existentes e criando novas conexões. É um processo onde a configuração da estrutura cognitiva passa de um estado a outro.

Nessa perspectiva, novos conhecimentos são construídos à medida que o aprendiz se movimenta no sentido de articular novos saberes aos que já possui,

assim, aprendizagem precisa ser ancorada a uma outra já existente na estrutura cognitiva do sujeito para que possa ser então assimilada.

Portanto, um dos pontos fundamentais desta teoria consiste em determinar aquilo que o aprendiz já sabe ou conhece (MASINI; MOREIRA, 2001), ou seja, o estado atual da sua estrutura, para que a proposta de ensino seja baseada nestes conhecimentos.

No caso do aprendiz não possuir conhecimentos prévios sobre o novo conceito a aprendizagem não ocorrerá de maneira significativa, pois não resulta na aquisição de significados para o sujeito. Quando isso ocorre dá-se o nome de aprendizagem mecânica ou automática.

A aprendizagem mecânica, ou automática é o contrário da aprendizagem significativa: a aprendizagem de novas informações ocorre com pouca ou nenhuma associação com conceitos relevantes existentes na estrutura cognitiva, o que fará com que estes conhecimentos sejam esquecidos com maior facilidade (MASINI; MOREIRA, 2001) .

Para evitar isso, Ausubel sugere deliberadamente manipular a estrutura cognitiva usando organizadores prévios (MOREIRA; SOUSA, 1996). Organizadores prévios tem a função de ponte entre o que o aprendiz sabe e o que deve saber. Eles visam estabelecer relações entre ideias, proposições e conceitos já existentes na estrutura cognitiva, a fim de que a aprendizagem possa ser significativa. Além disso, também podem ser usados para “reativar” significados obliterados, para “buscar” na estrutura cognitiva do aluno significados que já existiam, mas que não eram usados há algum tempo.

Para que a aprendizagem seja significativa há três condições: predisposição do indivíduo; material potencialmente significativo e estrutura cognitiva capaz de assimilar a nova informação (AUSUBEL; NOVAK; HANESIAN, 1980).

A predisposição para aprender está intimamente relacionada com a experiência afetiva que o aprendiz tem no evento educativo (NOVAK, 2000). Além disso, a aprendizagem significativa propõe a participação ativa do aluno na aquisição de conhecimento, de maneira a evitar-se uma mera reprodução de conceitos formulados pelo professor ou pelo livro-texto, mas uma reelaboração do aluno (PELIZZARI; KRIEGL, 2002).

Para Ausubel, cada disciplina tem uma estrutura articulada e hierarquicamente organizada de conceitos (MASINI; MOREIRA, 2001). No entanto, a

ordem em que os principais conceitos e ideias da matéria de ensino são apresentadas muitas vezes não é a mais adequada para facilitar a interação com o conhecimento prévio do aluno.

Por isso, é essencial uma análise crítica da matéria de ensino a ser apresentada ao estudante; o conteúdo precisa ter boa organização lógica, cronológica e epistemológica, mas além disso é indispensável uma análise com foco no estudante e em particular seu conhecimento prévio.

Além disso, é importante também não sobrecarregar o aluno de informações desnecessárias, dificultando a organização cognitiva. É preciso buscar a melhor maneira de relacionar, explicitamente, os aspectos mais importantes do conteúdo da matéria de ensino aos aspectos especificamente relevantes de estrutura cognitiva do aprendiz. Este relacionamento é imprescindível para a aprendizagem significativa.

O adiamento da experiência de aprendizagem para além da maturidade do estudante desperdiça oportunidades de aprendizagens valiosas e, muitas vezes, reduzindo de forma desnecessária, a quantidade e complexidade dos conteúdos que se pode dominar num determinado período da aprendizagem escolar. Por outro lado, quando um aluno é exposto, prematuramente, a uma tarefa de aprendizagem, antes de estar preparado de forma adequada para a mesma, não só não aprende a tarefa em questão (ou aprende-a com muitas dificuldades), como também aprende com esta experiência a temer, desgostar e evitar a tarefa (AUSUBEL, 2003).

Neste sentido, se faz necessário determinar continuamente o que o aprendiz conhece para ensiná-lo de acordo. Para tal, faz-se necessário mecanismos que visem identificar o estado mental de cada aprendiz, relativo ao domínio de conhecimento em questão, para que possam auxiliar os professores nesta tarefa.

Ausubel enfoca a linguagem como um facilitador importante para a ocorrência da aprendizagem significativa (MOREIRA, 1983). Desta forma, é possível observar que os conceitos abordados serão realmente assimilados, se eles forem apresentados numa linguagem coerente e que faça sentido para o aprendiz.

Destaca-se que embora tenha contribuído para o avanço da teoria da construção do conhecimento, Ausubel não proporcionou aos educadores instrumentos simples e funcionais para ajudá-los a averiguar “o que o aluno já sabe”.

De maneira resumida, nesta subseção apresentamos os significados atribuídos por Ausubel ao conceito de aprendizagem significativa.

3 PROCEDIMENTOS METODOLÓGICOS

Neste capítulo é apresentada a fundamentação metodológica do trabalho, iniciando pelo delineamento da pesquisa, o local e população, as etapas da pesquisa, a coleta e análise de dados.

3.1 DELINEAMENTO DA PESQUISA

Esta pesquisa visa explorar diferentes contextos de ensino de programação de computadores, tendo por objetivo elaborar uma proposta de abordagem prática, baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador.

A hipótese que orienta este trabalho, é a de que uma proposta de abordagem prática baseada em Ausubel, que priorize e exponha os alunos mais cedo ao uso do computador traga resultados positivos comparativamente à abordagem tradicional.

Com relação à natureza, esta pesquisa classifica-se como aplicada, pois objetiva gerar novos conhecimentos relacionados ao ensino e aprendizagem de programação de computadores.

Do ponto de vista dos objetivos, trata-se de uma pesquisa exploratória, pois visa proporcionar maior familiaridade com o problema com vistas a torná-lo explícito, construindo assim novas hipóteses.

De acordo com (SELLTIZ et al., 1974, p. 63)

> As pesquisas exploratórias têm como propósito proporcionar maior familiaridade com o problema, com vistas a torná-lo mais explícito ou a constituir hipóteses. Seu planejamento é, portanto, bastante flexível, de modo que possibilite a consideração dos mais variados aspectos relativos ao fato estudado. Na maioria dos casos, essas pesquisas envolvem: 1. Levantamento bibliográfico; 2. Entrevistas com pessoas que tiveram experiências práticas com o problema pesquisado; e 3. Análise de exemplos que "estimulem a compreensão".

O procedimento metodológico de abordagem dados é predominantemente qualitativo, porque considerou haver uma dinâmica que não pode ser transformada em números. O foco da análise são os conteúdos apresentados aos estudantes e seus desempenhos de modo geral na disciplina, comparando-se a proposta de abordagem

elaborada, com uma abordagem que se poderia chamar de tradicional de ensino de programação.

> A análise qualitativa depende de muitos fatores, tais como a natureza dos dados coletados, a extensão da amostra, os instrumentos de pesquisa e os pressupostos teóricos que nortearam a investigação. Pode-se, no entanto, definir esse processo como uma sequência de atividades, que envolve a redução dos dados, a categorização desses dados, sua interpretação e a redação do relatório. (GIL, 2002, p.133)

3.2 LÓCUS DA PESQUISA E POPULAÇÃO

A pesquisa foi executada no Instituto Federal do Paraná – IFPR, campus Telêmaco Borba.

O Instituto Federal do Paraná (IFPR) é uma instituição pública federal de ensino vinculada ao Ministério da Educação (MEC) por meio da Secretaria de Educação Profissional e Tecnológica (SETEC). É voltada a educação superior, básica e profissional, especializada na oferta gratuita de educação profissional e tecnológica nas diferentes modalidades e níveis de ensino.

A cidade de Telêmaco Borba tem aproximadamente 75.000 habitantes (IBGE/2015), o campus de Telêmaco Borba iniciou suas atividades no dia 29 de março de 2010, oferecendo cursos que buscam atender às demandas produtivas relacionadas ao papel e à madeira, setor de grande expressão na economia do município, e também favorecer a transformação da realidade social da região por meio de outros cursos. O campus Telêmaco Borda do IFPR, atualmente oferece cursos de nível técnico integrados ao ensino médio, cursos de superiores e de qualificação profissional.

A amostragem para o teste piloto desenvolvido na Universidade Tecnológica Federal do Paraná – UTFPR, campos Ponta Grossa compreende 7 (sete) alunos voluntários do primeiro semestre do curso Bacharelado em Ciência da Computação.

A população deste estudo compreende 29 (vinte e nove) alunos do segundo ano do curso Técnico em Informática para Internet integrado ao ensino médio e 16 (dezesseis) alunos do curso de Qualificação Profissional Programador Web do Programa Nacional de Acesso ao Ensino Técnico e Emprego.

3.3 ETAPAS DA PESQUISA

A pesquisa foi desenvolvida conforme as etapas da Figura 1.

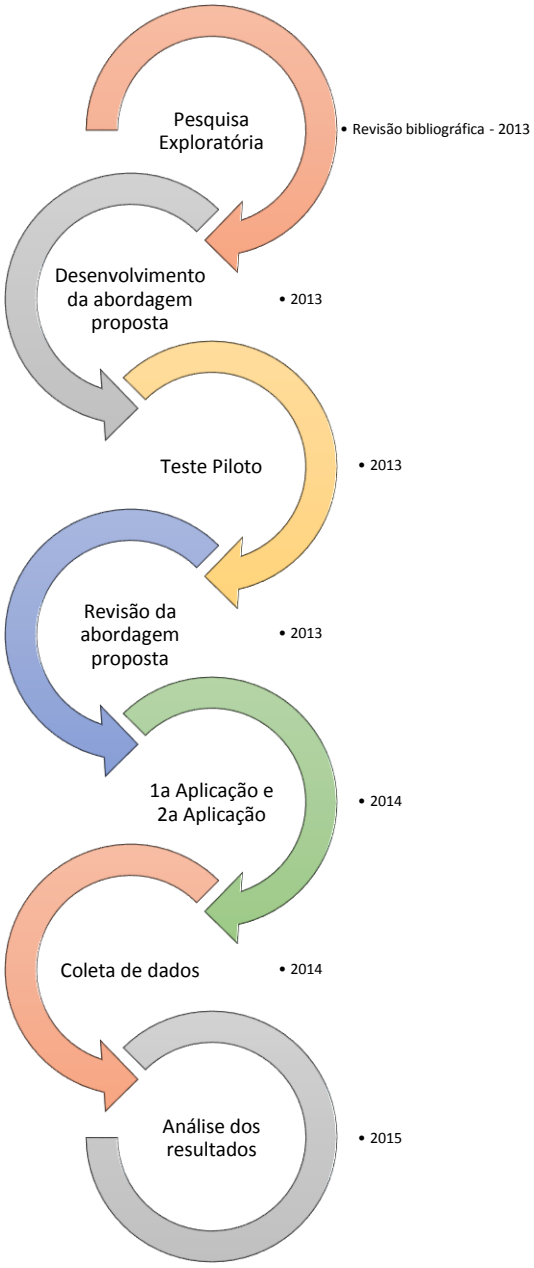


Figura 1: Etapas da pesquisa
Fonte: Autoria própria

Inicialmente com intuito de conhecer melhor como se dá o processo de ensino e aprendizagem dos conceitos inicias de programação de computadores, realizou-se pesquisa exploratória com a revisão bibliográfica do tema.

Por meio desta pesquisa pode-se identificar:

I. As habilidades e competências exercidas na programação de computadores;
II. As dificuldades presentes no processo de ensino/aprendizagem de programação; e
III. As propostas presentes na literatura que visam atenuar estas dificuldades.

Com base nestes conhecimentos adquiridos referentes ao processo de ensino/aprendizagem, formulou-se os objetivos e a hipótese deste estudo.

A partir disso, buscou-se elaborar uma proposta de abordagem prática, baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação.

Ao término do processo de elaboração da proposta de abordagem, optou-se por realizar um teste piloto para verificar a aplicabilidade da sequência e a resposta dos alunos acerca dos conteúdos trabalhados. Essa etapa aconteceu durante os meses de setembro e outubro de 2013, com alunos do primeiro semestre do curso de Ciência da Computação da Universidade Tecnológica Federal do Paraná – UTFPR, campus Ponta Grossa.

A etapa seguinte da pesquisa constituiu na revisão da abordagem proposta e correções necessárias acerca das observações do teste piloto a fim de cumprir com o objetivo desta pesquisa.

3.3.1 Aplicações em sala

A partir das revisões efetuadas após o teste piloto, a pesquisa seguiu para a sua aplicação prática. Ocorreram duas edições da aplicação desta proposta de abordagem pelo professor pesquisador e mais a aplicação paralela por outro docente em nível de ensino distinto conforme a Figura 2.

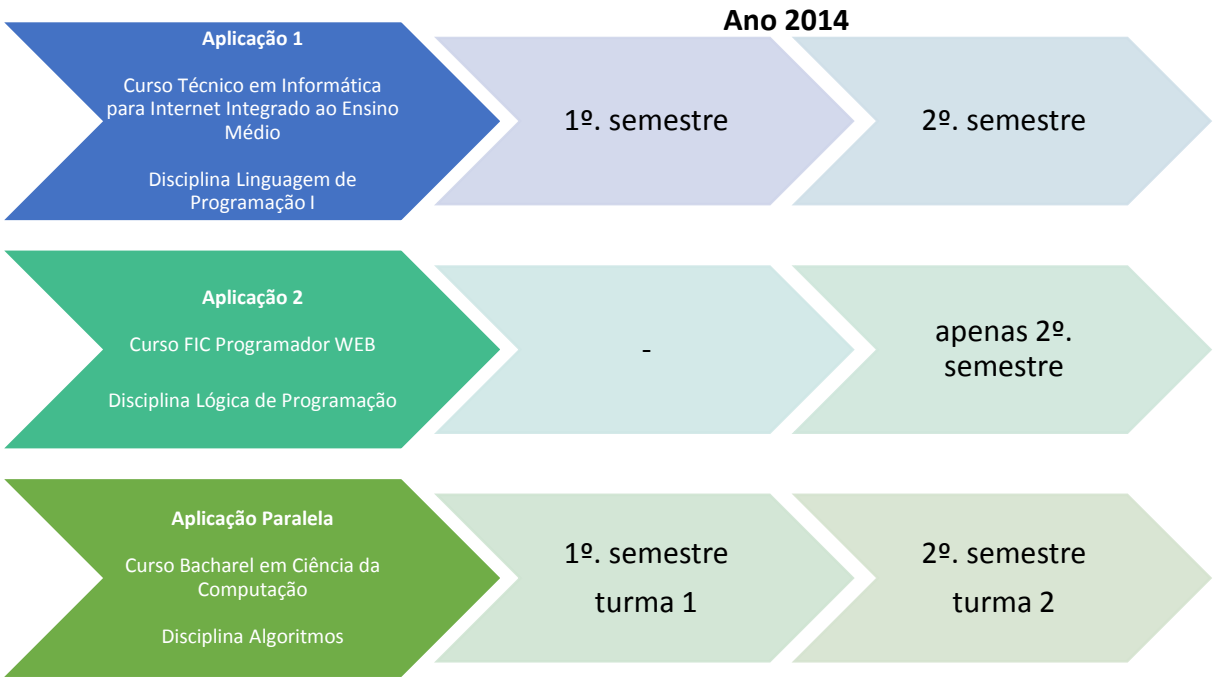


Figura 2: Aplicações em sala de aula
Fonte: Autoria própria

A primeira edição ocorreu no nível técnico, no segundo ano do curso Técnico em Informática para Internet Integrado ao Ensino Médio do Instituto Federal do Paraná – IFPR, campus Telêmaco Borba, na disciplina de Linguagem de Programação I – LPI durante o ano de 2014.

A disciplina LPI está presente no 2º ano do curso, e possui carga horária de 120 horas aula que normalmente é ministrada por um único professor.

Para a realização desta pesquisa, esta disciplina foi dividida entre o professor titular e o pesquisador, ficando a cargo do professor titular 2/3 da carga horária e 1/3 a cargo do professor pesquisador, conforme a Figura 3.

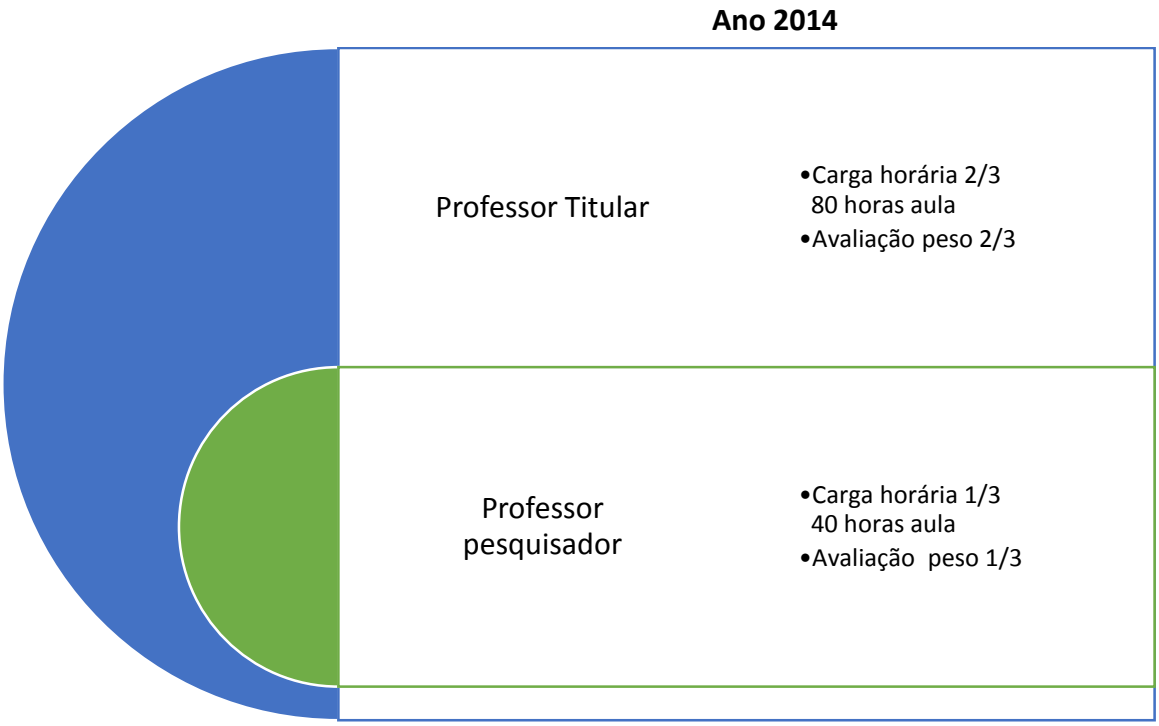


Figura 3: Divisão da disciplina LPI para primeira aplicação prática
Fonte: Autoria própria

Dessa forma, a disciplina foi ministrada em 3 horas-aula por semana; desse total, uma hora aula (50 minutos relógio) ficou a cargo do pesquisador e as outras duas aulas eram ministradas pelo titular. Isso totalizou 40 horas aula de conteúdo trabalhado pelo pesquisador. Definiu-se também que a avaliação de desempenho dos alunos seria composta com 2/3 de avaliações do titular e 1/3 de avaliações do pesquisador conforme a Figura 3.

Ficou estabelecido que os dois professores iriam trabalhar de maneira independente, porém foi considerado que esta primeira aplicação haveria interferência mútua pelos professores pelo fato de dividirem entre si a mesma turma. E apesar deste procedimento não ser o recomendado devido a possíveis interferências, este processo foi utilizado devido a limitação tempo da pesquisa.

A segunda edição ocorreu no nível de qualificação, no curso de Programador Web, do Programa Nacional de Acesso ao Ensino Técnico e Emprego – PRONATEC, no Instituto Federal do Paraná – IFPR, campus Telêmaco Borba, na disciplina de Lógica de Programação durante os meses de junho a dezembro de 2014.

A aplicação paralela por outro docente ocorreu no nível superior, no primeiro semestre do curso Bacharel em Ciência da Computação da Universidade Tecnológica Federal do Paraná – UTFPR, campus Ponta Grossa, durante dois semestres de 2014.

É importante ressaltar que essa aplicação paralela aconteceu fora do âmbito deste trabalho, ou seja, sem estar sob controle do pesquisador. Ela servirá para ilustrar o uso da proposta de abordagem por outro docente e em outro nível de ensino.

A partir das aplicações foram coletados os dados que subsidiam a análise e os resultados desta pesquisa.

3.4 COLETA E ANÁLISE DE DADOS

A coleta de dados para esta pesquisa foi realizada por meio de documentos oficiais como os diários de classe dos professores da disciplina, suas avaliações e o desempenho dos estudantes na disciplina, com foco na análise qualitativa deste material, além disso, foram registradas pelo pesquisador observações da sala de aula e diálogos sem estruturas ocasionais com o professor titular da disciplina durante a primeira aplicação.

Com a posse destes dados buscou-se comparar a aplicação da abordagem proposta e a abordagem tradicional para o ensino dos fundamentos de programação de computadores. Para isso elencou-se os seguintes critérios:

1. Cumprimento da ementa e objetivos da disciplina, observando se os tópicos previstos foram exercidos de forma:
    a. Parcial;
    b. Parcial com inserção de tópicos não contemplados na ementa.
    c. Plena;
    d. Plena com inserção de tópicos não contemplados na ementa.

2. Comparação da época em que os assuntos foram introduzidos, observando se foram:
    a. Mais cedo;
    b. Igual;
    c. Mais tarde.

3. Comparação da época de realização de avaliações e provas, observando se foram:
    a. Mais cedo;
    b. Igual;
    c. Mais tarde.

4. Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina, observando se estas contemplam:
    a. Mais tópicos;
    b. Igual;
    c. Menos tópicos.

5. Comparação do desempenho dos alunos, observando:
    a. Conceitos bimestrais;
    b. Índice de aprovação da disciplina.

6. Observações qualitativas do pesquisador a respeito do engajamento dos alunos e sua participação;

7. Comentários do professor titular da disciplina a respeito do desempenho dos estudantes.

4 PROPOSTA DE ABORDAGEM

Neste capítulo é apresentada a proposta de abordagem, iniciando pela exposição dos cursos e disciplinas onde esta abordagem pode ser aplicada, sua descrição, e por fim roteiros de conteúdos com sua estrutura de aplicação.

4.1 CURSOS E DISCIPLINAS

Conforme as Diretrizes Curriculares de Cursos da área de Computação e Informática:

> Os cursos da área de computação e informática tem como objetivos a formação de recursos humanos para o desenvolvimento científico e tecnológico da computação (hardware e software), para atuação na área de educação em computação em geral e para o desenvolvimento de ferramentas de informática que atendam a determinadas necessidades humanas. (BRASIL, 2001, p.4)

Sendo assim, visando atender ao objetivo de desenvolvimento de ferramentas de informática, a Programação de Computadores faz parte da área de formação básica dos cursos de Informática e Computação.

Os conteúdos presentes nas disciplinas de Programação, devem conter o ensino de linguagens de programação, seus conceitos, os princípios e os modelos de programação e o estudo de estruturas de dados e de métodos de classificação e pesquisa de dados (SANTOS; COSTA, 2005).

Dessa forma, as disciplinas de programação de computadores, são caracterizadas como uma atividade voltada à solução de problemas, sendo relacionada com uma variada gama de outras atividades como especificação, projeto, validação, modelagem e estruturação de programas e dados, utilizando linguagens de programação como ferramentas. (BRASIL, 2001)

Conforme o Quadro 10, todos os cursos de informática e computação devem cobrir os conceitos relacionados a programação de computadores, sendo que alguns cursos priorizam certos conteúdos ao detrimento de outros.

Matéria Programação	
Curso	**Conteúdos**
Bacharelado em Ciência da Computação Engenharia de Computação Bacharelado	As disciplinas devem cobrir, com abrangência e profundidade, pelo menos uma linguagem de programação desta matéria (primeira linguagem de programação). Devem cobrir também com abrangência e profundidade paradigmas de linguagens de programação, estrutura de dados e pesquisa e ordenação de dados.
Bacharelado em Sistemas de Informação Licenciatura em Computação	As disciplinas devem cobrir todas as principais linguagens de programação com abrangência e profundidade. Devem cobrir também com abrangência e profundidade estrutura de dados e pesquisa e ordenação de dados.

Quadro 10: Matéria Programação Cursos X Conteúdos
Fonte: Adaptado de Diretrizes Curriculares de Cursos da área de Computação e Informática (BRASIL, 2001)

Com relação aos cursos técnicos na área de informática e Computação o Ministério da Educação (MEC), classifica os cursos de acordo com Eixos Tecnológicos, sendo que os cursos relacionados a esta pesquisa estão presentes no Eixo de Informação e Comunicação.

De acordo com o Catálogo Nacional de Cursos Técnicos (CNCT) o eixo de Informação e Comunicação conta com nove cursos, dentre estes cursos são especialmente relevantes para este estudo os cursos presentes no Quadro 11.

Matéria Programação	
Curso	**Atividades**
Técnico em Informática:	Desenvolve programas de computador, seguindo as especificações e paradigmas da lógica de programação e das linguagens de programação. Utiliza ambientes de desenvolvimento de sistemas, sistemas operacionais e banco de dados. Realiza testes de programas de computador, mantendo registros que possibilitem análises e refinamento dos resultados. Executa manutenção de programas de computadores implantados
Técnico em Informática para Internet:	Desenvolve programas de computador para internet, seguindo as especificações e paradigmas da lógica de programação e das linguagens de programação. Utiliza ferramentas de desenvolvimento de sistemas, para construir soluções que auxiliam o processo de criação de interfaces e aplicativos empregados no comércio e marketing eletrônicos. Desenvolve e realiza a manutenção de sites e portais na internet e na intranet.
Técnico em Programação de Jogos Digitais:	Compõe equipes multidisciplinares na construção dos jogos digitais. Utiliza técnicas e programas de computadores especializados de tratamento de imagens e sons. Desenvolve recursos, ambientes, objetos e modelos a ser utilizados nos jogos digitais. Implementa recursos que possibilitem a interatividade dos jogadores com os programas de computador. Integra os diversos recursos na construção do jogo.

Quadro 11: Cursos técnicos e atividades
Fonte: Adaptado de BRASIL. Mec, Setec. Catálogo Nacional de Cursos Técnicos, 2012.

Assim, os conteúdos como Lógica e Linguagens de Programação devem estar presentes nos currículos destes cursos, visto que os alunos terão na atividade de desenvolvimento de programas de computador as bases para sua formação.

A próxima seção trata da descrição da proposta de abordagem.

4.2 DESCRIÇÃO DA PROPOSTA DE ABORDAGEM

Antes da apresentação da descrição da proposta de abordagem elaborada, cabe compará-la à abordagem tradicional e pontuar alguns aspectos, para isso é apresentado a seguir o Quadro 12.

Elementos	Abordagem Tradicional	Abordagem Proposta
Exposição conteúdos	Frequentemente teórica-conceitual	Prática; informação teórica surge para explicar mecanismos.
Diferentes assuntos	Exposição de comandos isolados.	Integração (FOR usando IF, etc..)
Sequenciamento	Fortemente linear, conforme atestam diários de classe	Cíclico: mesmo comando é discutido várias vezes em diferentes contextos.
Atividades práticas em laboratório	Poucas	Muitas
Representação de soluções	Fluxogramas e pseudocódigo	Linguagens de programação
Resolução de problemas	Apresentação de soluções prontas	Estimulo a proposição de soluções
Atividades para o desenvolvimento de experiências em programação	Listas de exercícios	Desafios semanais
Perspectiva do aluno em relação ao conteúdo apresentado	Tendência a vê-lo de forma mais Abstrata	Atividades práticas para introduzir ou confirmar conceitos teóricos.

Quadro 12: Abordagem Tradicional X Abordagem Proposta
Fonte: Autoria própria

Na abordagem tradicional geralmente ocorre a teorização dos conteúdos introdutórios de programação de computadores, aula após aula o professor apresenta conceitos aos estudantes. No entanto, a teorização dos conteúdos introdutórios de programação de computadores num primeiro momento é pouco relevante para os alunos, visto que para grande maioria dos alunos a programação de computadores é um assunto novo, portanto os mesmos não possuem os subsunçores necessários.

Deste modo, nesta etapa do processo, a exposição insistente a conceitos e teorias pode levar a um aprendizado mecânico. Um forte indício disso é a dificuldade dos alunos de construírem programas, embora em sala declarem ter entendido e demonstrem saber ler códigos.

Em nossa proposta de abordagem, durante as primeiras semanas, em nenhum momento esses conceitos são expostos de forma teórica, os estudantes são expostos a situações práticas que oportunizam a eles desenvolver seus próprios conceitos.

Outro aspecto diz respeito a como os conteúdos geralmente são trabalhados na abordagem tradicional. Nesta abordagem geralmente o professor separa os conteúdos em caixas, e apresenta aos estudantes de maneira isolada, normalmente

a estrutura básica, a sequência, os comandos de entrada e saída, variáveis, estrutura de decisão, repetição e etc.

Nossa proposta visa apresentar os mesmos conteúdos os mesmos conteúdos descritos anteriormente mas de maneira integrada. Pois é assim que a programação acontece, ou seja, na hora de se solucionar um problema se faz necessário que os estudantes relacionem esses elementos e isto acaba por se tornar um problema.

Além disso, na abordagem tradicional os conteúdos possuem uma sequência, ou seja, tem uma estrutura articulada e hierarquicamente organizada de conceitos, no entanto, a ordem em que os principais conceitos e ideias da matéria de ensino são apresentadas muitas vezes não é a mais adequada para facilitar a interação com o conhecimento prévio do aluno.

Assim, a apresentação dos conteúdos presente na proposta de abordagem, se dá de forma cíclica, isto se faz necessário uma vez que cada aluno irá formar sua própria estrutura cognitiva e progredir em uma velocidade diferente. Muitos alunos não possuem os subsunçores, maturidade ou as competências necessárias para se aprimorar estes conteúdos na primeira vez que estes são apresentados, de forma cíclica os conteúdos são apresentados por diversas vezes, em momentos e contexto diferentes o que pode favorecer o processo.

É possível observar que as pessoas constroem e reconstroem o seu conhecimento ao longo da vida, e os estudantes que não aprendem satisfatoriamente o que tentamos ensinar, muito provavelmente não o fazem por não terem o conhecimento prévio necessário para uma aprendizagem significativa (BRAATHEN, 2003).

Com relação as atividades práticas, normalmente na abordagem tradicional o professor faz pouco uso dos laboratórios, seja porque a instituição não oferece um suporte adequado ou pela grande quantidade de alunos presentes em sala de aula no início da disciplina.

A abordagem proposta preza por grande quantidade de atividades práticas em laboratório (se possível todas as aulas), visto que ao se utilizar o computador não se exige do estudante imaginar o funcionamento de algum comando, a execução de um comando pela máquina adquire um caráter quase concreto, possibilitando o estudante a experimentar hipóteses e sedimentar seus conceitos. Comparando com abordagem tradicional o estudante só irá confirmar o que viu mais tarde ou outro dia em laboratório.

Na abordagem tradicional inicialmente é solicitado aos estudantes que descrevam processos que em muitos casos não tem nenhuma relação com a programação de computadores, (exemplo: algoritmo para trocar lâmpada, vir de casa até a escola, etc.), na sequência geralmente o professor passa a utilizar pseudocódigo ou fluxograma para representar pequenos programas.

Deve-se tomar cuidado também com relação a escolha do ambiente e linguagem programação, tendo em vista que o foco principal é a aprendizagem dos conteúdos e não a sintaxe de uma determinada linguagem específica.

Em nossa proposta inicialmente utilizamos linguagens de sintaxe simples como o *BASIC* e *PYTHON*, sempre enfatizando os conteúdos da programação de computadores, recomendando que a partir do momento em que seja detectado que as aprendizagens destes conteúdos sejam satisfatórias, seja apresentado aos alunos ambientes e linguagens de sintaxe mais complexa, como C.

É comum também na abordagem tradicional a apresentação de soluções prontas, como por exemplo sequência Fibonacci, calculo fatorial e outros, nestes exemplos o professor descreve os processos da solução, ficando o estudante limitado a apresentar um único tipo de solução a correta e na maioria das vezes seguindo apenas o raciocínio do professor, a abordagem proposta estimula a proposição de soluções, desta forma o aluno fica livre para criar sua própria proposta de solução e até mesmo formular outros problemas derivados do problema inicial.

Desta forma deve-se tomar cuidado em algumas etapas do processo, visto que alguns alunos podem ter dificuldades na hora de propor soluções, uma atenção especial neste contexto deve ser dada a motivação e ao relacionamento professor-aluno, pois caso contrário, o aluno pode se sentir desestimulado e ficar alheio ao processo.

Portanto, o professor deve ser muito flexível e tentar estimular ao máximo a proposição de soluções mesmo que estas não sejam as corretas, o aluno deve ser instigado a realizar operações e atividades de alteração de códigos, e partir destas o professor deve questiona-los sobre o que o programa produz.

Dessa forma, espera-se que os estudantes possam desenvolver seu próprio repertório de conhecimentos e habilidades, podendo assim resolver as atividades propostas.

Na abordagem tradicional é comum professores se utilizarem de listas de exercícios visando incentivar os alunos a adquirirem uma maior experiência em

programação, entretanto, vivemos em uma época em que tudo é compartilhado e que é difícil não se encontrar a solução para um problema de uma lista de exercícios pela internet, além disso, observa-se que em alguns caso um único estudante faz a lista de exercícios e compartilha com os demais colegas, isto acontece devido a vários motivos. A proposta de abordagem estimula os alunos a adquirirem essa experiência em programação por meio de desafios semanais, que na grande maioria das vezes é a solução para um único problema desenvolvido em laboratório, cujo não é dada a solução, e isso estimula a curiosidades dos alunos em relação a resolução do problema e a competividade fazendo com que eles não compartilhem suas soluções.

Por fim, sintetizando a proposta de abordagem, esta consiste na apresentação de pequenos trechos de códigos simples, que devem ser examinados e modificados pelos alunos, usando uma linguagem de programação.

Dessa forma, primeiramente o professor realiza a exposição de um comando ou trecho de um programa no projetor. Em seguida questiona os alunos sobre como obter um resultado ligeiramente diferente na tela por meio de modificação do comando ou trecho. Neste momento o professor deve ter atenção para não podar as soluções propostas pelos alunos mesmo que estas sejam erradas, após breve discussão, os alunos deveriam sugerir modificações nestes comandos ou programas, os alunos eram instigados a modificar os programas e perceberem o que acontecia, visando chegar a uma solução satisfatória, neste momento os alunos também eram incentivados a compartilhar suas soluções e explicar o que eles fizeram.

Salvo estas comparações e indicações passamos a apresentar os Planos de Aula.

4.3 Planos de aula

Os planos de aula que serão apresentados no decorrer desta seção, foram desenvolvidos a partir da proposta abordagem objeto deste estudo, para cada aula sugere-se uma carga horária de duas horas aula em laboratório de informática.

A estrutura dos roteiros foi inspirada no trabalho de (FELDER; BRENT, 2003). Os roteiros contém os objetivos da aula, conteúdos e atividades a serem realizadas. Cada aula não trata de um assunto específico devido a integração dos conteúdos,

também não se estabelece rigidamente o tempo que o professor irá usar para sua aplicação, pois depende de uma série de fatores como por exemplo a turma, laboratório, entre outros.

Os conteúdos abordados nestes roteiros com a proposta de abordagem são: Comandos de entrada e saída; Variáveis; Estrutura de decisão; Estrutura de repetição; Implementação de problemas numa linguagem de programação.

A seguir é apresentado os roteiros de cinco aulas com a proposta de abordagem.

<table>
<tr><th colspan="2">Aula 01</th></tr>
<tr><td colspan="2">Objetivos:
Identificar os conhecimentos prévios dos alunos referente a programação de computadores;
Apresentar a área de trabalho da ferramenta Decimal Basic
Apresentar por meio de atividades práticas os comandos de entrada e saída; Variáveis; Estrutura de repetição;
Implementar problemas numa linguagem de programação.</td></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Conhecimentos prévios</td><td>Fazer levantamentos dos conhecimentos em programação de cada aluno.</td></tr>
<tr><td>Download da ferramenta</td><td>Fazer download da ferramenta Decimal Basic por meio do link http://www.geocities.jp/thinking_math_education/EnglishWindows.htm</td></tr>
<tr><td>Ferramenta</td><td>Apresentar a área de trabalho da ferramenta.</td></tr>
<tr><td rowspan="5">PRINT</td><td>Mostrar comando PRINT “nome”</td></tr>
<tr><td>Solicitar aos alunos que modifiquem o comando PRINT para imprimir outra coisa PRINT “outra coisa”</td></tr>
<tr><td>Apresentar o seguinte código:<pre>PRINT 123
PRINT 100 + 200
PRINT “100 + 200”
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Explicar diferença entre caracteres e números ‘de verdade’</td></tr>
<tr><td rowspan="2">Variáveis, PRINT</td><td>Apresentar o seguinte código:<pre>N = 10
PRINT N
PRINT N * 2
END</pre></td></tr>
<tr><td>Explicar aos alunos o que é uma variável.</td></tr>
<tr><td rowspan="3">Laço FOR, Variáveis, PRINT</td><td>Apresentar o seguinte código:<pre>FOR i=0 to 10
   PRINT “NOME”
NEXT i
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem o código para imprimir 15 vezes, 5 vezes e 100 vezes</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 01 – continuação</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="5">Laço FOR, Variáveis, PRINT</td><td>Apresentar o seguinte código:
<pre>FOR J = 1 to 10
   PRINT J
NEXT J
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem o código para imprimir J de 5 até 10 e para imprimir de 10 até 50</td></tr>
<tr><td>Solicitar aos alunos que modifiquem o código para imprimir de 10 até 1, grande parte dos alunos irá propor a solução FOR J = 10 to 1 (o que não irá funcionar), apresentar o código para solução do problema proposto:
<pre>FOR I = 1 TO 10
   PRINT 10 - I
NEXT I
END</pre></td></tr>
<tr><td>Explicar aos alunos que em programação, muitas coisas têm que ser ‘ajustadas’.</td></tr>
<tr><td rowspan="4">Laço FOR, Variáveis, PRINT, Variante STEP</td><td>Explicar que isso se aprende com o tempo e experiência em programação.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem o código para imprimir todos os números impares de 1 a 33, e na sequência todos os números pares de 0 a 44.</td></tr>
<tr><td>Apresentar o seguinte código:
<pre>FOR J = 39 to 0 step -2
   PRINT J
NEXT J
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos feitos anteriormente para imprimir todos os números impares de 1 a 33, e na sequência todos os números pares de 0 a 44, incluindo a variante STEP</td></tr>
<tr><td>Desafio Semanal I</td><td>Solicitar os alunos que desenvolvam o código que mostre na tela o seguinte resultado:
<pre>11
22
33
44
55
66
77
88
99</pre></td></tr>
</table>

Quadro 13: Aula 01 com a abordagem proposta

Fonte: Autoria própria.

<table>
<tr><th colspan="2">Aula 02</th></tr>
<tr><td colspan="2">Objetivos:
Avaliar os conhecimentos dos alunos referente aos conteúdos apresentados na aula anterior;
Reforçar os conceitos por meio de atividades práticas os comandos de entrada e saída; Variáveis; Estrutura de repetição;
Apresentar por meio de atividades práticas a Estrutura de decisão;
Implementar problemas numa linguagem de programação.</td></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Conhecimentos prévios</td><td>Questionar aos alunos quais comandos foram apresentados até o momento e anotar em local visível para os alunos.</td></tr>
<tr><td>Desafio Semanal I</td><td>Verificar quais alunos conseguiram resolver o Desafio Semanal I, com base nas diferentes propostas de solução para problema desenvolvidas pelos alunos, apresentar aos demais estudantes e pedir que eles comentem as soluções propostas</td></tr>
<tr><td rowspan="5">PRINT</td><td>Apresentar o seguinte código:<pre>PRINT 1
PRINT 2,
PRINT 3;
PRINT 4;
PRINT 5,
PRINT 6
PRINT 7
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.<pre>1
 2   3 4 5   6
 7</pre></td></tr>
<tr><td>Explicar aos estudantes que muitas coisas em computação acabam sendo deduzidas, observando os códigos e resultados apresentados pela máquina.</td></tr>
<tr><td>Explicar que é interessante fazer pequenos testes para aprender coisas novas.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos feitos anteriormente para que o código mostre na tela o seguinte resultado:<pre>1 2 3 4 5 6 7 8 9</pre></td></tr>
<tr><td>Laço FOR, Variáveis, PRINT</td><td>Solicitar aos alunos que modifiquem os códigos feitos anteriormente para que o código mostre na tela o mesmo resultado, mas com a seguinte restrição: Usar apenas um comando PRINT N</td></tr>
<tr><td rowspan="2">Variáveis, INPUT, PRINT</td><td>Apresentar o seguinte código:<pre>INPUT nome$
PRINT nome$
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 02 – continuação 1</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="4">Variáveis, INPUT, PRINT</td><td>Apresentar o seguinte código:
<pre>INPUT x
INPUT y
PRINT X+Y
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Explicar aos estudantes as diferenças no armazenamento de dados de variáveis Números X Caracteres</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos apresentados para que seja feita a soma de 5 números</td></tr>
<tr><td rowspan="2">Variáveis, Laço For, PRINT, INPUT, Acumulador</td><td>Solicitar aos alunos que modifiquem os códigos apresentados para que seja feita a leitura e soma de 10 números, utilizando apenas um comando INPUT e 2 variáveis, os alunos têm 90 segundos para fazer isso, quem conseguir ganha um doce</td></tr>
<tr><td>Apresentar a solução para o problema proposto, explicar o que é um acumulador:
<pre>LET soma = 0

FOR i = 1 TO 5

   PRINT "diga um número"
   INPUT n

   LET soma = soma + n

NEXT i

PRINT "total = "; soma

END</pre></td></tr>
<tr><td rowspan="4">Variáveis, INPUT, IF, PRINT, Operadores relacionais</td><td>Apresentar o seguinte código:
<pre>INPUT N

IF (N>10) THEN
   PRINT "Maior que 10"
END IF

IF (N<10) THEN
   PRINT "Menor que 10"
END IF
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td>Explicar que o IF só executará o código que tem dentro dele caso a condição seja verdadeira.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos para que seja apresentada a mensagem Igual a 10.</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 02 – continuação 2</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Variáveis, INPUT, IF, ELSE, PRINT</td><td>Apresentar o seguinte código:
<pre>INPUT N

IF (N>10) THEN
   PRINT "Maior que 10"
ELSE
   PRINT "Menor que 10"
END IF

END</pre>Questionar aos alunos o resultado da execução do código.</td></tr>
<tr><td rowspan="2">Variáveis, INPUT, IF, ELSE, PRINT</td><td>Explicar que o ELSE só executará o código que tem dentro dele caso a condição a condição do IF seja falsa. E que um ELSE por receber outros IF dentro dele.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos para que seja apresentada também a mensagem Igual a 10.</td></tr>
<tr><td>Desafio Semanal II</td><td>Solicitar os alunos que desenvolvam o código que mostre na tela o seguinte resultado:
<pre>XXX
XXX
XXX</pre>Usando apenas um único comando PRINT “X”</td></tr>
</table>

Quadro 14: Aula 02 com a abordagem proposta

Fonte: Autoria própria

<table>
<tr><th colspan="2">Aula 03</th></tr>
<tr><td colspan="2">Objetivos:
Avaliar os conhecimentos dos alunos referente aos conteúdos apresentados até o momento;
Reforçar os conceitos por meio de atividades práticas os comandos de entrada e saída; Variáveis; Estrutura de repetição; Estrutura de decisão;
Apresentar por meio de atividades práticas o conceito de Identação do código e os comandos RANDOMIZE e EXIT FOR;
Implementar problemas numa linguagem de programação.</td></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Conhecimentos prévios</td><td>Questionar aos alunos quais comandos foram apresentados até o momento e anotar em local visível para os alunos.</td></tr>
<tr><td>Desafio Semanal II</td><td>Verificar quais alunos conseguiram resolver o Desafio Semanal II, com base nas diferentes propostas de solução para problema desenvolvidas pelos alunos, apresentar aos demais estudantes e pedir que eles comentem as soluções propostas</td></tr>
<tr><td>Estrutura do programa e Identação</td><td>A partir desta aula, enfatizar uso de espaços em branco entre linhas de comandos e TAB para comandos que estejam dentro das estruturas FOR, IF, ELSE.</td></tr>
<tr><td rowspan="4">RANDOMIZE, PRINT, RND</td><td>Apresentar o seguinte código:<pre>RANDOMIZE
PRINT RND
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código, solicitar que os alunos executem o código diversas vezes até eles perceberem que o RND gera um número aleatório.</td></tr>
<tr><td>Solicitar aos alunos que modifiquem os códigos para que mostre na tela um número entre 0 e 10. Lembra-los que em programação, muitas coisas têm que ser ‘ajustadas’.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:<pre>RANDOMIZE
PRINT 10*RND
END</pre></td></tr>
<tr><td rowspan="2">RANDOMIZE, PRINT, INT, RND</td><td>Solicitar aos alunos que modifiquem os códigos para que mostre na tela um número entre 7 e 10. Os alunos têm 90 segundos para fazer isso, quem conseguir ganha um doce</td></tr>
<tr><td>Apresentar a solução para o problema proposto:<pre>RANDOMIZE
PRINT INT (7+3*RND)
END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código, explicar que o comando INT transforma um valor real num valor inteiro</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 03 – continuação 1</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Variáveis, RANDOMIZE, INT, RND, Laço FOR, IF, ELSE</td><td>Solicitar aos alunos que desenvolvam um programa com as seguintes características:
Escrever um programa que sorteia um número inteiro aleatório X entre 1 e 10 e guarda na variável X
Depois o programa lê um número na variável N cinco vezes, em um laço
Cada vez que o programa lê um número, ele mostra a diferença entre N e X.
Se aparecer o número zero no vídeo, significa que o usuário descobriu o valor de X
Dar um bom tempo para que os alunos desenvolvam suas soluções.</td></tr>
<tr><td>Variáveis, RANDOMIZE, INT, RND, Laço FOR, PRINT, INPUT</td><td>Apresentar a solução para o problema proposto:
<pre>RANDOMIZE
LET X = INT (10*RND)

FOR i = 1 TO 5
   PRINT "digite um número";
   INPUT n
   PRINT x-n
NEXT i

END</pre></td></tr>
<tr><td rowspan="2">Variáveis, RANDOMIZE, INT, RND, Laço FOR, PRINT, INPUT, IF, ELSE</td><td>Solicitar aos alunos que reescrevam a solução para o problema anterior, acrescentando os comandos IF e ELSE, para informar ao usuário as seguintes mensagens: Em caso de acerto “Parabéns! Que lindo!”, em caso de erro “Errou! Tente novamente”.
Dar um bom tempo para que os alunos desenvolvam suas soluções.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:
<pre>RANDOMIZE
LET X = INT (10*RND)

FOR i = 1 TO 5
   PRINT "digite um número";
   INPUT n

   IF n = x THEN
      PRINT "Parabéns! Que lindo!"
   ELSE
      PRINT "Errou! Tente novamente."
   END IF
NEXT i

END</pre></td></tr>
</table>

<table>
<tr><th colspan="2">Aula 03 – continuação 2</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="2">Variáveis, RANDOMIZE, INT, RND, Laço FOR, PRINT, INPUT, IF, ELSE, EXIT FOR</td><td>Apresentar o comando EXIT FOR (talvez seja pedido pelos alunos no programa anterior)
<pre>RANDOMIZE
LET X = INT (10*RND)

FOR i = 1 TO 5
   PRINT "digite um número";
   INPUT n

   IF n = x THEN
      PRINT "Parabéns! Que lindo!"
      EXIT FOR
   ELSE
      PRINT "Errou! Tente novamente."
   END IF
NEXT i

END</pre></td></tr>
<tr><td rowspan="2">Variáveis, RANDOMIZE, INT, RND, Laço FOR, PRINT, INPUT, IF, ELSE</td><td>Solicitar aos alunos que modifiquem o programa anterior para a cada vez que o usuário tenta acertar o número, mas errar, o programa deverá ‘zoar’ com o jogador, cada vez apresentando uma frase diferente. Dar um bom tempo para que os alunos desenvolvam suas soluções.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:
<pre>RANDOMIZE
LET X = INT(10*RND)

FOR i = 1 TO 5

   PRINT "digite um número";
   INPUT n

   IF n = x THEN
      PRINT "Parabéns! Que lindo!"
      EXIT FOR
   ELSE
      IF i = 1 then
          PRINT "Errou!"
      end if
      IF i = 2 then
          PRINT "não, criatura!"
      end if
      IF i = 3 then
          PRINT "ai que tristeza"
      end if

   END IF
NEXT i
END</pre></td></tr>
</table>

Aula 03 – continuação 3	
Conteúdos	**Atividades**

Desafio Semanal III	Solicitar os alunos que desenvolvam o código que mostre na tela o seguinte resultado:
	`x.`
	`xx.`
	`xxx.`
	`xxxx.`
	`xxxxx.`
	Usando apenas um único comando PRINT “X” e PRINT “.”

Quadro 15: Aula 03 com a abordagem proposta

Fonte: Autoria própria

<table>
<tr><th colspan="2">Aula 04</th></tr>
<tr><td colspan="2">Objetivos:
Avaliar os conhecimentos dos alunos referente aos conteúdos apresentados até o momento;
Reforçar os conceitos por meio de atividades práticas os comandos de entrada e saída; Variáveis; Estrutura de repetição; Estrutura de decisão;
Apresentar por meio de atividades práticas os conceitos de contador e acumulador;
Implementar problemas numa linguagem de programação.</td></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Conhecimentos prévios</td><td>Questionar aos alunos quais comandos foram apresentados até o momento e anotar em local visível para os alunos.</td></tr>
<tr><td>Desafio Semanal III</td><td>Verificar quais alunos conseguiram resolver o Desafio Semanal III, com base nas diferentes propostas de solução para problema desenvolvidas pelos alunos, apresentar aos demais estudantes e pedir que eles comentem as soluções propostas</td></tr>
<tr><td rowspan="2">Variáveis, PRINT, INPUT, IF, ELSE</td><td>Solicitar aos alunos que desenvolvam um programa que leia dois valores e mostre na tela uma das três mensagens a seguir:
‘Números iguais’, caso os números sejam iguais;
‘Primeiro é maior’, caso o primeiro seja maior que o segundo;
‘Segundo maior’, caso o segundo seja maior que o primeiro.
Os alunos têm 5 minutos para desenvolver a solução para este problema, aquele que terminar primeiro ganha um doce.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:<pre>
LET n1=0
LET n2=0

PRINT "Digite um valor para n1..: "
INPUT n1

PRINT "Digite um valor para n1..: "
INPUT n2

IF n1=n2 THEN
   PRINT "Números iguais"
ELSE
   IF n1>n2 THEN
      PRINT "Primeiro é maior"
   ELSE
      PRINT "Segundo é maior"
   END IF
END IF

END
</pre></td></tr>
</table>

<table>
<tr><th colspan="2">Aula 04 – continuação 1</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="2">Variáveis, PRINT, INPUT</td><td>Solicitar aos alunos que desenvolvam um programa que leia dois valores e ao término do programa mostre a soma destes valores. Os alunos têm 5 minutos para desenvolver a solução para este problema, aquele que terminar primeiro ganha um doce.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:
<pre>LET soma=0
LET n1=0
LET n2=0

PRINT "Digite um valor..: "
INPUT n1
PRINT "Digite outro valor..: "
INPUT n2
LET soma = n1 + n2

PRINT "A soma dos valores é..: "; soma
END</pre></td></tr>
<tr><td rowspan="2">Variáveis, Laço FOR, PRINT, INPUT</td><td>Solicitar aos alunos que alterem o programa anterior para que leia uma quantidade de valores informada pelo usuário e ao término do programa mostre a soma destes valores e a média. Os alunos têm 5 minutos para desenvolver a solução para este problema, aquele que terminar primeiro ganha um doce.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:
<pre>LET soma=0
LET n=0
LET q=0

INPUT q

FOR i=1 TO q
   PRINT "Digite um valor..: "
   INPUT n

   LET soma = soma + n
NEXT i

PRINT "A soma dos valores é..: "; soma
PRINT "A média dos valores é..: "; soma/q
END</pre></td></tr>
</table>

<table>
<tr><th colspan="2">Aula 04 – continuação 2</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="2">Variáveis, DO ~ LOOP, PRINT</td><td>Apresentar o seguinte código:
<pre>LET n=0

DO
   LET n=n+1
   PRINT n
LOOP

END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código, explicar o que é um laço infinito</td></tr>
<tr><td rowspan="2">Variáveis, DO ~ WHILE LOOP, PRINT</td><td>Apresentar o seguinte código:
<pre>LET n=0

DO WHILE n<100
   LET n=n+1
   PRINT n
LOOP

END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código, explicar as diferenças entre DO ~ LOOP e DO ~ WHILE LOOP</td></tr>
<tr><td rowspan="2">Variáveis, DO ~ WHILE LOOP, PRINT, INPUT, IF, ELSE, EXIT DO</td><td>Apresentar o seguinte código:
<pre>LET n=0

DO
   PRINT "Informe um valor: "
   INPUT n

   IF n=0 THEN
      EXIT DO
   ELSE
      PRINT n
   END IF

LOOP

END</pre></td></tr>
<tr><td>Questionar aos alunos o resultado da execução do código, e o que acontece quando o usuário informa o valor 0 para n, associar ao EXIT FOR</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 04 – continuação 3</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td rowspan="3">Variáveis, DO, PRINT, INPUT, IF, ELSE
EXIT DO, CONTADOR, ACUMULADOR</td><td>Solicitar aos alunos que desenvolvam um programa que leia a idade de várias pessoas, a leitura deverá parar quando a idade digitada for zero. O programa deverá mostrar a soma das idades digitadas; o programa deverá contar quantas idades foram digitadas; o programa deverá mostrar a média das idades digitadas. Os alunos têm 10 minutos para desenvolver a solução para este problema, aquele que terminar primeiro ganha um doce.</td></tr>
<tr><td>Apresentar a solução para o problema proposto:
<pre>LET idade=0
LET soma=0
LET qtd=0

DO
   PRINT "Informe a Idade: "
   INPUT idade

   IF idade=0 THEN
      EXIT DO
   ELSE
      LET soma = soma + idade
      LET qtd = qtd + 1
   END IF

LOOP
PRINT "A média de Idade é...:"; soma/qtd

END</pre></td></tr>
<tr><td>Solicitar aos alunos que alterem o programa anterior para que para que o programa mostre além da média de idade, a menor e maior idade. Os alunos têm 3 minutos para desenvolver a solução para este problema, aquele que terminar primeiro ganha um doce.</td></tr>
</table>

<table>
<tr><th colspan="2">Aula 04 – continuação 4</th></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Variáveis, DO ~ LOOP, EXIT DO, PRINT, INPUT CONTADOR, ACUMULADOR</td><td>Apresentar a solução para o problema proposto:
<pre>
LET idade=0
LET soma=0
LET qtd=0
LET maior=0
LET menor=0

DO
   PRINT "Informe a Idade: "
   INPUT idade

   IF idade=0 THEN
      EXIT DO
   ELSE
      LET soma = soma + idade
      LET qtd = qtd + 1

      IF (menor>idade) OR (menor=0)THEN
         LET menor = idade
      END IF

      IF (maior<idade) THEN
         LET maior = idade
      END IF

   END IF

LOOP

PRINT "A média de Idade é...:"; soma/qtd
PRINT "A maior Idade é...:"; maior
PRINT "A menor Idade é...:"; menor
END
</pre></td></tr>
<tr><td>Desafio Semanal IV</td><td>Apresentar aos alunos os seguintes códigos:
<pre>
SET WINDOW -10,10, -10,10
DRAW GRID

PLOT LINES: 1,1;1,4

END
</pre>Usando o comando PLOT LINES e as coordenadas do plano cartesiano os alunos devem tentar mostrar na tela um quadrado, um retângulo e um triangulo.</td></tr>
</table>

Quadro 16: Aula 04 com a abordagem proposta

Fonte: Autoria própria

<table>
<tr><th colspan="2">Aula 05</th></tr>
<tr><td colspan="2">Objetivos:
Avaliar os conhecimentos dos alunos referente aos conteúdos apresentados até o momento;
Reforçar os conceitos por meio de atividades práticas os comandos de entrada e saída; Variáveis; Estrutura de repetição; Estrutura de decisão;
Apresentar por meio de atividades práticas o conceito de sub-rotinas e área de plotagem do ambiente;
Implementar problemas numa linguagem de programação.</td></tr>
<tr><th>Conteúdos</th><th>Atividades</th></tr>
<tr><td>Conhecimentos prévios</td><td>Questionar aos alunos quais comandos foram apresentados até o momento e anotar em local visível para os alunos.</td></tr>
<tr><td>Desafio Semanal IV</td><td>Verificar quais alunos conseguiram resolver o Desafio Semanal IV, com base nas diferentes propostas de solução para problema desenvolvidas pelos alunos, apresentar aos demais estudantes e pedir que eles comentem as soluções propostas</td></tr>
<tr><td>Variáveis, Laço FOR, CALL SUB, PLOT LINES</td><td>Apresentar aos alunos os seguintes códigos:<pre>
SET WINDOW -10,10, -10,10
DRAW GRID

CALL teste (0,0,1)
CALL teste (-3, -2, 0.5)
CALL teste (2,2,1)

SUB teste (x,y,raio)
   OPTION ANGLE DEGREES
   FOR t=0 TO 360
      PLOT LINES: x+raio*COS(t),y+raio*SIN(t);
   NEXT t
   PLOT LINES

END SUB
END
</pre></td></tr>
<tr><td rowspan="3">Variáveis, PRINT, INPUT, Laço FOR, IF e ELSE, CALL e SUB, PLOT LINES</td><td>Questionar aos alunos o resultado da execução do código, solicitar que os alunos modifiquem os valores das coordenadas da sub-rotina teste.</td></tr>
<tr><td>Solicitar aos alunos que reescrevam o desafio semanal IV utilizando os comandos CALL e SUB.</td></tr>
<tr><td>Solicitar aos alunos que utilizando os comandos CALL e SUB, tentem desenhar um boneco palito, e uma forca. Os alunos têm 10 minutos para desenvolverem a solução o primeiro que terminar ganha um doce.</td></tr>
</table>

Aula 05 – continuação 1	
Conteúdos	**Atividades**
Variáveis, PRINT, INPUT, Laço FOR, IF e ELSE, CALL e SUB, PLOT LINES	Solicitar aos alunos que desenvolvam um programa utilizando CALL e SUB com as seguintes características: Escrever um programa que sorteia um número inteiro aleatório X entre 1 e 10 e guarda na variável X; Depois o programa lê um número na variável N seis vezes, em um laço; Cada vez que o usuário tenta acertar o número, mas errar, o programa deverá mostrar na área de plotagem uma parte do boneco palito na forca. Caso o usuário acerte o número o programa, deve emitir a seguinte mensagem “Parabéns que Lindo! ” e encerrar o programa. Dar um bom tempo para que os alunos desenvolvam suas soluções. O primeiro que terminar ganha um doce.
Variáveis, PRINT, INPUT, IF ELSE, FOR, DO ~ WHILE, RANDOMIZE, RND, PLOT LINES, CALL SUB	Solicitar aos alunos que alterem o programa anterior para que seja incluído um menu e mais um jogo o de operações matemáticas, com as seguintes características: O menu deve exibir as seguintes opções 0 para sair, 1 para o jogo da forca, 2 para operações matemáticas e 3 para exibir os créditos do desenvolvedor do programa; O jogo de operações matemáticas deve ter as seguintes características: O programa deve sortear dois números aleatórios N1 e N2 entre 1 e 10 e guarda nas variáveis N1 e N2; O programa deve sortear um número entre 1 e 4 e guardar na variável OP; O programa deve mostrar na tela o valor sorteado para N1 uma das seguintes operações soma, subtração, divisão e multiplicação, e N2 =? Depois o programa deve ler um número a Resposta; Se a resposta estiver correta o programa da os parabéns e acumula 10 pontos; Se a resposta estiver incorreta, o programa informa que a resposta está errada e da nova chance ao usuário diminuindo 10 pontos; A pontuação nunca deverá ser inferior a 0.
	Verificar quais alunos conseguiram resolver o problema, e com base nas diferentes propostas de solução para problema desenvolvidas pelos alunos, apresentar aos demais estudantes e pedir que eles comentem as soluções propostas

Quadro 17: Aula 05 com a abordagem proposta

Fonte: Autoria própria

5 DESENVOLVIMENTO E ANÁLISE DE RESULTADOS

Neste capítulo é apresentado o desenvolvimento e análise de resultados, iniciando pela descrição do Teste Piloto, a descrição das aplicações práticas, e por fim a análise e discussão dos resultados.

5.1 TESTE PILOTO

Após a realização da pesquisa bibliográfica acerca das dificuldades apresentadas pelos alunos iniciantes em programação de computadores e das propostas presentes na literatura para atenuar estas dificuldades foi elaborado uma proposta de abordagem que priorize e exponha os alunos ao uso do computador baseada nos pressupostos teóricos da aprendizagem significativa de David Ausubel.

A proposta aborda os conteúdos introdutórios de programação de computadores para alunos iniciantes e estão presentes na maioria das disciplinas de algoritmos, programação e suas equivalentes como: lógica de programação, linguagem de programação I, técnicas de programação, entre outras.

Dentre os conteúdos abordados pela proposta estão constantes, variáveis, estrutura sequencial, estrutura de decisão, estrutura de repetição, sub-rotinas e o uso de uma linguagem de programação para representação de problemas. Para se abordar estes conteúdos neste teste piloto, foi definida a seguinte carga horária de trabalho, 12 (doze) horas aula, sendo divididas em 6 (seis) encontros de 2 (duas) horas aula cada, sendo que toda esta carga horária foi executada em laboratório em atividades práticas, seguindo a proposta desta pesquisa. A Figura 4 apresenta a aplicação do teste piloto em laboratório de informática.

Figura 4: Aplicação do teste piloto em laboratório
Fonte: Autoria própria

Para se trabalhar estes conteúdos, foi selecionada a linguagem de programação *BASIC - Beginner's All-purpose Symbolic Instruction Code*, em português Código de Instruções Simbólicas de Uso Geral para Principiantes, o *BASIC* foi desenvolvido por J.Kemeny e T. Kurtz em 1963 no Dartmouth College, esta linguagem foi elaborada com o intuito de tornar claro o ensino dos conceitos da programação.

Entre os motivos para a escolha do *BASIC* para este experimento, pode-se destacar, que esta linguagem é de alto nível, ou seja, próxima da linguagem humana e com uma sintaxe simples, favorecendo que os alunos possam se dedicar a aprender e aplicar os conceitos apresentados.

Para implementação foi utilizado o interpretador da linguagem *DECIMAL BASIC256* versão para o sistema operacional *Windows,* disponível em http://www.geocities.jp/thinking_math_education/EnglishWindows.htm.

A aplicação do teste piloto teve por finalidade verificar o comportamento dos alunos mediante a proposta de abordagem e testar a mesma, visando identificar possíveis correções para que os objetivos de aprendizagem fossem atingidos.

Para a execução deste teste piloto (experimento) definiu-se como adequado a implementação em um curso da área de informática de nível superior, visando estudar a viabilidade deste estudo e de sua aplicação em níveis de ensino diferentes como superior e técnico.

Assim, foi solicitado junto ao professor responsável pela disciplina de Algoritmos e ao Departamento de Informática da Universidade Tecnológica Federal do Paraná – UTFPR, a autorização para a execução deste experimento.

Após a autorização, foram convidados a participar os alunos do 1º semestre o curso de Ciência da Computação da UTFPR. Aos alunos foi explicado o intuito do experimento e que a participação seria voluntária, e que não haveria qualquer relação com a disciplina de Algoritmos, que é parte integrante do currículo do curso.

O experimento ocorreu durante os meses de setembro e outubro de 2013, em horário de aula vago, e contou com a participação de 7 alunos, neste experimento os alunos foram motivados a competir entre si e para uma maior motivação foi realizada a distribuição de doces aos alunos, conforme estes apresentassem soluções para os problemas propostos.

Além disso, visando ampliar o espaço temporal dos encontros e ampliar a pratica de programação para os alunos foi proposto um desafio semanal, que consistia na resolução de um problema proposto, utilizando os conceitos apresentados durante o encontro.

Durante a execução do teste piloto foram coletados dados por meio da observação direta e indireta, e avaliação final dos conteúdos aprendidos.

Por meio da observação, pode-se perceber que a maior parte dos estudantes participantes do experimento se sentiram confortáveis a linguagem de programação apresentada e também aos conteúdos, além disso, a metodologia empregada por meio da apresentação de exemplos aos alunos seguida da alteração dos códigos destes exemplos, forneceram aos alunos as bases necessárias para a solução de novos problemas relacionados aos exemplos apresentados.

Ainda com relação a utilização de uma linguagem de programação de alto nível e de sintaxe relativamente simples, pode ser observado que estudantes com pouca ou nenhuma experiência em programação, podem ter o processo de aprendizagem favorecido, pois os estudantes não se preocuparão tanto com a sintaxe que em algumas linguagens é extremamente rígido.

Pode-se observar também que o ensino dos conceitos iniciais de programação por meio da prática, pode favorecer o aprendizado, pois o *feedback* é imediato, proporcionando aos alunos um maior raciocínio lógico e um melhor planejamento para implementação e/ou correções dos códigos, podendo contribuir

principalmente para a motivação do aluno e também para a diminuição das abstrações presentes no processo.

Durante o experimento, observou-se também que não foi dada a atenção adequada a alguns conceitos simples como atribuição, contadores e acumuladores, e apesar destes estarem presente em todo o processo, alguns alunos possuíam algumas dificuldades no emprego destes conceitos quando necessário.

Ao final do experimento, os alunos realizaram uma avaliação prática que contemplava todos os conceitos apresentados, os alunos envolvidos no experimento demonstraram índices satisfatórios de conhecimentos acerca do conteúdo apresentado.

A avaliação revelou também que alguns conceitos como atribuição, contador e acumulador, conforme já percebido pela observação precisariam de maior atenção. Mediante a estas constatações algumas alterações foram realizadas, visando melhor apresentar estes conceitos.

A próxima seção trata da descrição da aplicação prática da proposta de abordagem elaborada.

5.2 APLICAÇÃO PRÁTICA

A aplicação prática da proposta de abordagem desenvolvida, ocorreu em duas edições sendo que a primeira edição ocorreu na disciplina de Linguagem de Programação I no segundo ano do curso Técnico em Informática para Internet integrado ao Ensino Médio durante o ano de 2014, e a segunda edição ocorreu na disciplina de Lógica de Programação no curso de Programador Web, do Programa Nacional de Acesso ao Ensino Técnico e Emprego – PRONATEC, durante o segundo semestre de 2014, além destas, ocorreu também uma aplicação paralela por outro docente na disciplina de algoritmos do curso Bacharel em Ciência da Computação da Universidade Tecnológica Federal do Paraná – UTFPR, durante o ano de 2014.

A seguir são descritas as aplicações práticas identificando primeiramente a aplicação e apresentando os resultados.

5.2.1 Primeira Edição

A aplicação ocorreu na disciplina de Linguagem de Programação I (LPI), do curso Técnico em Informática para Internet Integrado ao Ensino Médio do Instituto Federal do Paraná – IFPR – Campus Telêmaco Borba. A turma iniciou com trinta alunos e terminou com vinte e nove, pois houve uma transferência durante o ano letivo.

A disciplina LPI está presente no 2º ano do curso, e possui carga horária de 120 horas aula. A Figura 5 a seguir apresenta a ementa e os conteúdos que devem ser trabalhados.

Ementa:
Fundamentos de algoritmos e programação. Fluxogramas. Pseudocódigo (*Portugol*). Programação sequencial. Instruções de seleção. Instruções de repetição. Vetores e matrizes. Funções e procedimentos. Implementação de problemas em uma linguagem de programação.

Figura 5: Ementa disciplina Linguagem de Programação I (LPI)
Fonte: PPC Técnico em informática para internet integrado ao ensino médio (3 anos)

É valido ressaltar que apesar da disciplina estar situada no 2º ano, os alunos da disciplina não tiveram nenhum contato com programação. Embora no primeiro ano conste da grade uma disciplina denominada Programação Web, a mesma trabalha apenas com *HTML e CSS*, para o desenvolvimento de *Blogs* e Sites.

A carga horaria de 120 horas da disciplina LPI que normalmente é ministrada por um único professor titular, foi dividida entre o professor titular e o pesquisador, para a realização desta pesquisa. Dessa forma, a disciplina era ministrada em 3 horas-aula por semana; desse total, uma hora aula (50 minutos relógio) ficou a cargo do pesquisador e as outras duas aulas eram ministradas pelo titular. Isso totalizou 40 horas aula de conteúdo trabalhado pelo pesquisador.

Ficou estabelecido que os dois professores trabalharam de maneira independente. Definiu-se que a avaliação de desempenho dos alunos seria composta com 2/3 de avaliações do titular e 1/3 de avaliações do pesquisador.

Todas as aulas ministradas pelo professor pesquisador foram práticas em laboratório. Foram utilizadas, em sequência as seguintes ferramentas:

- Primeiro bimestre o interpretador *Decimal Basic*,
- Segundo bimestre, *PASCAL*, por meio do compilador *PascalZim*

- Terceiro e quarto bimestres a linguagem de programação C, compilador *DevC++*.

No último bimestre, utilizando a linguagem de programação C, foi possível aprofundar mais os conteúdos abordando superficialmente recursividade, ponteiros e estruturas de dados como Fila e Pilhas.

5.2.1.1Resultados

A seguir são apresentados os resultados coletados na primeira edição da aplicação prática desta pesquisa, conforme proposto nos procedimentos metodológicos no subitem de coleta e análise de dados.

Cumprimento da ementa e objetivos da disciplina	
Professor Titular	Professor Pesquisador
Plena com inserção de tópicos não contemplados na ementa.	Parcial com inserção de tópicos não contemplados na ementa.

Quadro 18: Cumprimento da ementa e objetivos da disciplina
Fonte: Autoria própria

Conforme os registros de conteúdos dos diários de classes do professor titular da disciplina (anexo A -Diário de Classe da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Titular – Registros de Conteúdos) observa-se que o professor titular cumpriu com todos os elementos da ementa com a inserção de tópicos não contemplados na ementa como Banco de dados no último bimestre.

O professor pesquisador conforme os registros de conteúdos dos diários de classes (anexo B -Diário de Classe da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Pesquisador – Registros de Conteúdos), não contemplou os seguintes elementos referentes a ementa da disciplina: Fundamentos de algoritmos e programação; Fluxogramas; Pseudocódigo (Portugol), é valido ressaltar que a não apresentação destes elementos foi por opção do professor pesquisador por destoar da estratégia didática proposta, desta forma assume-se que o cumprimento da ementa foi parcial.

Entretanto o professor pesquisador inseriu itens mais avançados não contemplados na disciplina como: Recursividade, ponteiros e estrutura de dados (fila e pilha), desta forma parcial com inserção de tópicos não contemplados na ementa.

Para a comparação da época em que os assuntos foram introduzidos e comparação dos conteúdos abordados em avaliações e provas, observou-se os seguintes tópicos referentes a ementa da disciplina: 1.Programação sequencial; 2.Instruções de seleção; 3.Instruções de repetição; 4.Vetores; 5.Matrizes; 6.Funções e procedimentos; 7.Implementação de problemas em uma linguagem de programação. A seguir é apresentado o Quadro 19, comparando a época em que os assuntos foram introduzidos.

Comparação da época em que os assuntos foram introduzidos		
Tópicos	Professor Titular	Professor Pesquisador
Programação sequencial	Mais tarde	Mais cedo
Instruções de seleção	Mais tarde	Mais cedo
Instruções de repetição	Mais tarde	Mais cedo
Vetores	Mais tarde	Mais cedo
Matrizes	Igual	Igual
Funções e procedimentos	Mais tarde	Mais cedo
Implementação de problemas em uma linguagem de programação	Mais tarde	Mais cedo

Quadro 19: Comparação da época em que os assuntos foram introduzidos
Fonte: Autoria própria

Conforme Quadro 19, observa-se que com a exceção do assunto matrizes que foi visto na mesma época (4º bimestre) todos os assuntos foram introduzidos mais cedo pelo professor pesquisador.

Os registros de conteúdos indicam também que a época de realização de avaliações, provas e trabalhos, foram semelhantes entre o professor titular e o professor pesquisador.

A seguir é apresentado a comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina pelo professor titular e pesquisador conforme o anexo C - Avaliações e Provas da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Titular e anexo D - Avaliações e Provas da disciplina

Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Pesquisador.

Com base nas avaliações do 1º bimestre do professor titular e professor pesquisador foi elaborado o Quadro 20, comparando os conteúdos abordados em avaliações e provas referentes a ementa da disciplina no 1º bimestre.

Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 1º Bimestre		
Tópicos	Professor Titular	Professor Pesquisador
Programação sequencial	Contempla	Contempla
Instruções de seleção	Não contempla	Contempla
Instruções de repetição	Não contempla	Contempla
Vetores	Não contempla	Não contempla
Matrizes	Não contempla	Não contempla
Funções e procedimentos	Não contempla	Não contempla
Implementação de problemas em uma linguagem de programação	Não contempla	Contempla
Total de tópicos contemplados	1	4

Quadro 20: Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 1º Bimestre

Fonte: Autoria própria

De acordo com o Quadro 20, observa-se que a disciplina ministrada pelo professor pesquisador no 1º bimestre contempla mais tópicos referentes a ementa da disciplina.

Com base nas avaliações do professor titular e professor pesquisador, foi elaborado o Quadro 20, visando comparar os conteúdos abordados em avaliações e provas referentes a ementa da disciplina no 2º bimestre.

Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 2º Bimestre		
Tópicos	Professor Titular	Professor Pesquisador
Programação sequencial	Contempla	Contempla
Instruções de seleção	Contempla	Contempla
Instruções de repetição	Não contempla	Contempla
Vetores	Não contempla	Não contempla
Matrizes	Não contempla	Não contempla
Funções e procedimentos	Não contempla	Não contempla
Implementação de problemas em uma linguagem de programação	Não contempla	Contempla
Total de tópicos contemplados	2	4

Quadro 21: Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 2º Bimestre

Fonte: Autoria própria

De acordo com o Quadro 21, observa-se que a disciplina ministrada pelo professor pesquisador no 1º bimestre contempla mais tópicos referentes a ementa da disciplina que a disciplina ministrada pelo professor titular.

Com base nas avaliações do professor titular e do professor pesquisador, foi elaborado o Quadro 22, visando comparar os conteúdos abordados em avaliações e provas referentes a ementa da disciplina no 3º bimestre.

Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 3º Bimestre		
Tópicos	Professor Titular	Professor Pesquisador
Programação sequencial	Contempla	Contempla
Instruções de seleção	Contempla	Contempla
Instruções de repetição	Contempla	Contempla
Vetores	Não contempla	Contempla
Matrizes	Não contempla	Não contempla
Funções e procedimentos	Não contempla	Contempla
Implementação de problemas em uma linguagem de programação	Contempla	Contempla
Total de tópicos contemplados	4	6

Quadro 22: Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 3º Bimestre

Fonte: Autoria própria

O Quadro 22, indica que a disciplina ministrada pelo professor pesquisador no 3º bimestre contempla mais tópicos referentes a ementa da disciplina que a disciplina ministrada pelo professor titular.

De acordo com as avaliações do professor titular e do professor pesquisador, foi elaborado o Quadro 23, comparando os conteúdos abordados em avaliações e provas referentes a ementa da disciplina no 4º bimestre.

Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 4º Bimestre		
Tópicos	Professor Titular	Professor Pesquisador
Programação sequencial	Contempla	Contempla
Instruções de seleção	Contempla	Contempla
Instruções de repetição	Contempla	Contempla
Vetores	Contempla	Não contempla
Matrizes	Não contempla	Contempla
Funções e procedimentos	Contempla	Contempla
Implementação de problemas em uma linguagem de programação	Contempla	Contempla
Total de tópicos contemplados	6	6

Quadro 23: Comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina – 4º Bimestre

Fonte: Autoria própria

O Quadro 23, indica que a disciplina ministrada pelo professor pesquisador e professor titular no 4º bimestre contemplam a mesma quantidade de tópicos referentes a ementa da disciplina.

As comparações dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina realizadas indicam que de um modo geral a disciplina ministrada pelo professor pesquisador contemplou mais tópicos nos três primeiros bimestres.

A seguir é realizado a comparação do desempenho dos alunos nas disciplinas do professor titular e do professor pesquisador, de acordo com os registros de avaliações do professor titular e do professor pesquisador, foram elaboradas tabelas, referente ao desempenho dos alunos. O processo de avaliação adotado no curso é na forma de conceitos onde: o conceito A significa “Aprendizagem Plena”; B - “Aprendizagem Parcialmente Plena”; C - “Aprendizagem Suficiente”; D - “Aprendizagem insuficiente”. Os estudantes são avaliados bimestralmente e ao

término da disciplina de seus conceitos são analisados gerando um conceito final da disciplina, onde somente ficará retido na disciplina o estudante que apresente conceito final D.

A seguir é apresentado na forma de tabelas o desempenho dos alunos nos quatro bimestres da disciplina, do professor titular e professor pesquisador.

Tabela 1: Desempenho dos alunos na disciplina no 1º Bimestre

Desempenho na disciplina 1º Bimestre	Professor Titular	Professor Pesquisador
Conceito A (aprendizagem plena)	1	6
Conceito B (aprendizagem parcialmente plena)	7	8
Conceito C (aprendizagem suficiente)	10	10
Conceito D (aprendizagem insuficiente)	11	5

Fonte: Autoria própria de acordo com o registro de avaliações dos diários de classe do professor titular da disciplina e professor pesquisador

Tabela 2: Desempenho dos alunos na disciplina no 2º Bimestre

Desempenho na disciplina 2º Bimestre	Professor Titular	Professor Pesquisador
Conceito A (aprendizagem plena)	0	3
Conceito B (aprendizagem parcialmente plena)	4	12
Conceito C (aprendizagem suficiente)	14	11
Conceito D (aprendizagem insuficiente)	11	3

Fonte: Autoria própria de acordo com o registro de avaliações dos diários de classe do professor titular da disciplina e professor pesquisador

Tabela 3: Desempenho dos alunos na disciplina no 3º Bimestre

Desempenho na disciplina 3º Bimestre	Professor Titular	Professor Pesquisador
Conceito A (aprendizagem plena)	3	3
Conceito B (aprendizagem parcialmente plena)	3	13
Conceito C (aprendizagem suficiente)	12	10
Conceito D (aprendizagem insuficiente)	11	3

Fonte: Autoria própria de acordo com o registro de avaliações dos diários de classe do professor titular da disciplina e professor pesquisador

Tabela 4: Desempenho dos alunos na disciplina no 4º Bimestre

Desempenho na disciplina 4º Bimestre	Professor Titular	Professor Pesquisador
Conceito A (aprendizagem plena)	6	8
Conceito B (aprendizagem parcialmente plena)	6	6
Conceito C (aprendizagem suficiente)	7	10
Conceito D (aprendizagem insuficiente)	10	5

Fonte: Autoria própria de acordo com o registro de avaliações dos diários de classe do professor titular da disciplina e professor pesquisador

Tabela 5: Desempenho final dos alunos na disciplina

Desempenho final na disciplina	Professor Titular	Professor Pesquisador
Conceito A (aprendizagem plena)	3	6
Conceito B (aprendizagem parcialmente plena)	6	8
Conceito C (aprendizagem suficiente)	11	11
Conceito D (aprendizagem insuficiente)	9	4

Fonte: Autoria própria de acordo com o registro de avaliações dos diários de classe do professor titular da disciplina e professor pesquisador

Com base no desempenho final dos alunos na disciplina (Tabela 5), foi elaborado o Gráfico 1 que exibe a comparação do desempenho final dos alunos na disciplina ministrada pelo Professor Titular X Professor Pesquisador.

É valido ressaltar que os resultados referentes a comparação do desempenho dos alunos na disciplina, expostos neste trabalho não possui caráter quantitativo, devido a subjetividade do processo, uma vez que as duas disciplinas foram ministradas concomitantemente, e também a particularidade do processo de avaliativo onde cada professor tem a sua maneira de avaliar os alunos.

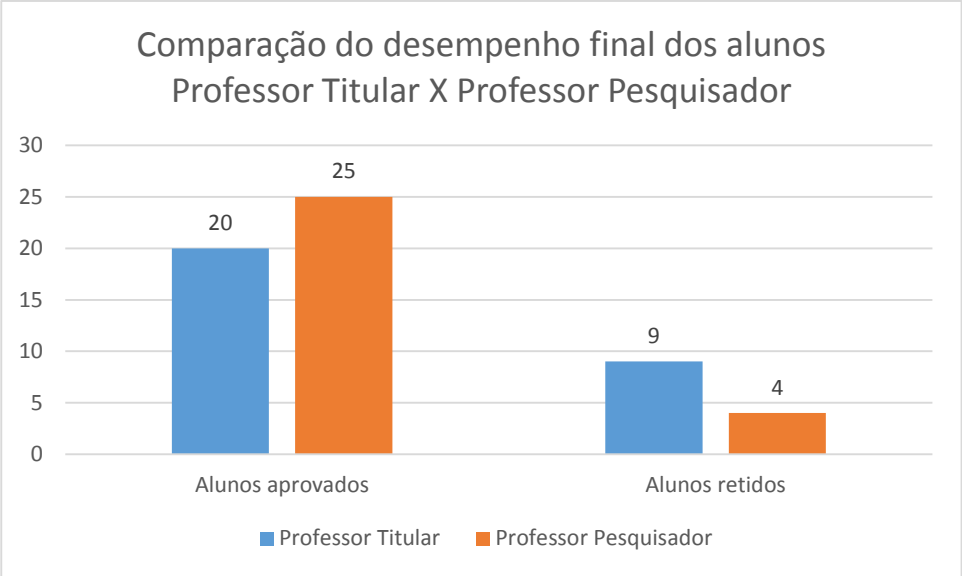


Gráfico 1: Comparação do desempenho final dos alunos Professor Titular X Professor Pesquisador
Fonte: Autoria própria

Além destes resultados apresentados ao o momento, cabe mencionar observações qualitativas do professor pesquisador sobre o engajamento dos alunos e sua participação durante esta primeira aplicação, percebeu-se indícios de que as aulas práticas dos conteúdos introdutórios de programação de computadores podem produzir um engajamento maior por parte dos alunos e, além disso, podem diminuir a lacuna no quesito abstração, um dos grandes problemas apontados por pesquisas sobre o ensino/aprendizagem de programação de computadores.

Vale também apresentar alguns comentários do professor titular da disciplina a respeito do desempenho dos estudantes, como:

Os alunos nas atividades práticas em laboratórios se sentiram confortáveis na utilização de ambientes de programação e tiveram um aproveitamento melhor nas atividades;

Muitos alunos ao se expor um determinado conceito como por exemplo Se Então, mencionavam: Este é o IF e ELSE do Basic que estamos vimos com o outro professor? Dessa forma, comparando os comandos.

Em alguns casos os conceitos apresentados a turma não era algo, fazendo com que em alguns momentos a disciplina fosse acelerada. (O professor titular não esperava que o conteúdo estivesse pronto na cabeça dos alunos, exemplo laço FOR, que foi trabalhado com os alunos nas primeiras aulas da aplicação e visto posteriormente na disciplina com o professor titular.

Do ponto de vista investigativo, conclui-se que esta aplicação obteve êxito e diversos fatores foram identificados ao longo de sua execução, apesar do fato da disciplina ministrada pelo professor pesquisador abordar em grande maioria os conteúdos mais cedo não foi possível mensurar o quanto ela influenciou e foi influenciada pela disciplina regular que foi apresentada concomitantemente pelo professor titular, apesar do relato do professor referente ao fato de que quando os conceitos foram empregados em sua disciplina de maneira prática os alunos realizaram comparações aos conceitos aprendidos anteriormente, segundo o professor os estudantes já aparentavam conhecimentos prévios, facilitando o entendimento de novos conceitos.

Dessa forma, podemos considerar que esta primeira edição da aplicação prática obteve diversos fatores positivos, conforme mencionados anteriormente, e estes fatores colaboram na confirmação da hipótese sugerida por este estudo.

5.2.2 Segunda Edição

Esta seção relata como foi realizada a segunda edição da aplicação da proposta de abordagem desenvolvida.

A aplicação ocorreu na disciplina de Lógica de Programação no curso de Programador Web, do Programa Nacional de Acesso ao Ensino Técnico e Emprego – PRONATEC, durante o segundo semestre de 2014, no Instituto Federal do Paraná – IFPR – Campus Telêmaco Borba.

A disciplina de Lógica de Programação, é uma das bases do curso de programador web, e possui carga horária de 30 horas aula. A Figura 6 a seguir apresenta a ementa e os conteúdos que devem ser trabalhados.

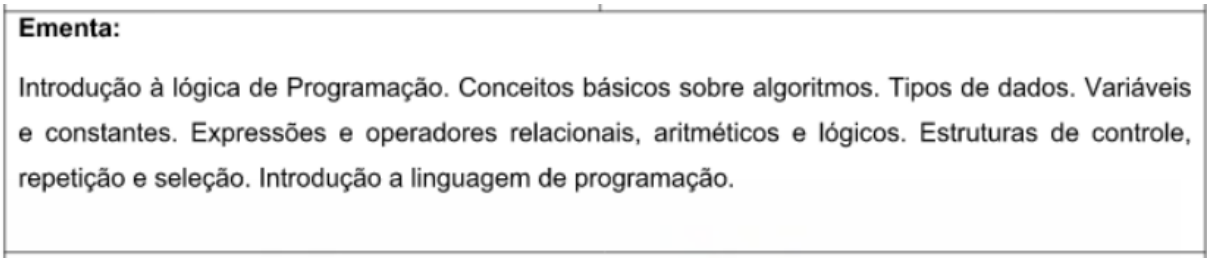


Figura 6: Ementa disciplina Lógica de Programação (LP)
Fonte: PPC FIC Programador Web

A disciplina ocorreu no período noturno com 10 (dez) encontros de 3 (três) horas cada, conforme a proposta deste trabalho todas as aulas foram práticas em laboratório de informática, foi utilizada a linguagem de programação *PYTHON*, por meio do interpretador *IDLE-PYTHON-3.3*, no sistema operacional *Linux Ubuntu.*

A linguagem de programação *Python*, foi escolhida porque assim como o *Basic* na primeira aplicação, possui uma sintaxe simples, fazendo com que desta forma os alunos possam se dedicar a aprender e aplicar os conhecimentos adquiridos, não gastando o tempo em aprender as minucias de sintaxe apresentadas por algumas linguagens.

Com relação a amostragem, a disciplina iniciou-se com 26 (vinte e seis) alunos, e terminou com 16 (quinze) alunos, sendo que, dos 10 (dez) alunos que não concluíram a disciplina, 7 (sete) foram considerados evadidos, e 3 (três) foram desistentes.

É válido ressaltar que neste caso específico, a disciplina de programação teve pouca relação com taxa de evasão e desistência. A matrícula propicia acesso a serviços sociais, como seguro desemprego e outros, dessa forma, os motivos que levaram muitos destes estudantes a se matricularem neste tipo de curso contribuem diretamente para a falta de identificação com o curso.

Os dados coletados por meio desta aplicação foram obtidos por meio da observação direta e indireta, e também pelo diário de classe.

5.2.2.1Resultados

Os resultados coletados na segunda edição da aplicação prática deste estudo, conforme proposto nos procedimentos metodológicos no subitem de coleta e análise de dados são apresentados a seguir, deve-se considerar que esta aplicação foi

executada inteiramente pelo professor pesquisador e além disso não existem registros anteriores desta disciplina, pois esta aplicação foi realizada na primeira edição do curso de Programador Web, na modalidade PRONATEC no campus, e por estes motivos não há dados para que sejam realizadas comparações.

Com relação ao cumprimento da ementa e objetivos da disciplina, observando se os tópicos previsto na ementa da disciplina podemos afirmar que conforme a descrição das aulas apresentadas anteriormente e anexo E - Diário de Classe da disciplina de Lógica de Programação do Curso Programador Web – PRONATEC 2014 Professor Pesquisador – Registros de Conteúdos, esta aplicação pode ser classificada como Plena com inserção de tópicos não contemplados na ementa, pois a mesma cobriu todos os tópicos presentes na ementa e ainda acrescentou tópico de funções.

A respeito da comparação da época em que os assuntos foram introduzidos, apesar da aplicação não possuir dados históricos para que possa ser comparada, presume-se que considerando a abordagem tradicional, onde inicialmente há grande ênfase em aspectos teóricos e comparando-a com descrição das aulas apresentadas anteriormente, observa-se que esta aplicação abordaria os assuntos presentes na ementa da disciplina mais cedo.

Com relação a época de realização de avaliações e provas, presume-se também que se comparado a abordagem tradicional seria igual.

Não foi possível realizar a comparação dos conteúdos abordados em avaliações e provas tampouco o desempenho dos alunos. Entretanto apresenta-se a seguir (Tabela 6) os resultados obtidos pelos estudantes na disciplina.

Tabela 6: Desempenho dos alunos na disciplina Lógica de Programação

Desempenho dos alunos	Alunos
Conceito A (aprendizagem plena)	4
Conceito B (aprendizagem parcialmente plena)	5
Conceito C (aprendizagem suficiente)	6
Conceito D (aprendizagem insuficiente)	1

Fonte: Autoria própria

Com base no desempenho final dos alunos na disciplina (Tabela 6), foi elaborado o Gráfico 2, onde é exposto a porcentagem de alunos aprovados e retidos na disciplina.

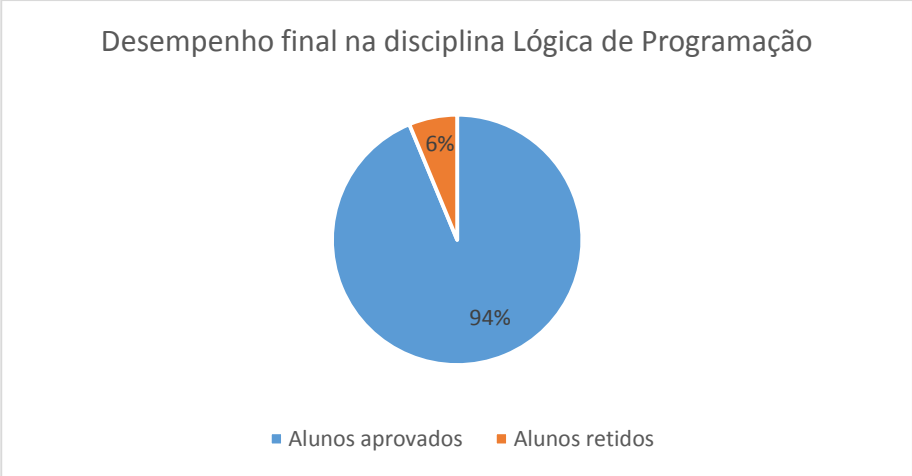


Gráfico 2: Desempenho final na disciplina Lógica de Programação
Fonte: Autoria própria

Com base no Gráfico 2, observar-se que dos 16 (dezesseis) estudantes, que concluíram a disciplina 15 (quinze) conseguiram atingir índices satisfatórios para aprovação.

O professor pesquisador por meio das observações qualitativas sobre o engajamento dos alunos na disciplina constatou que os resultados presentes na segunda edição da aplicação foram similares ao da primeira aplicação, ou seja, indícios de que a aplicação prática contribuiu para diminuir o grau de abstração presente na matéria de programação, favorecendo entre outros aspectos a motivação dos estudantes.

Assim, podemos concluir que esta aplicação, do ponto de vista investigativo, reforça alguns resultados obtidos durante a aplicação da primeira edição, além disso, o fato desta aplicação acontecer isolada de outra disciplina ao contrário do que

aconteceu durante a primeira aplicação, aponta que os resultados obtidos durante a primeira aplicação pouco foram influenciados pela disciplina que ocorreu concomitantemente.

Dessa forma, podemos considerar que esta segunda edição da aplicação prática obteve fatores colaboram na confirmação da hipótese sugerida por este estudo.

5.3 APLICAÇÃO PARALELA POR OUTRO DOCENTE

A sistemática descrita nessa dissertação foi também usada por um docente em outra instituição. O objetivo dessa aplicação foi obter mais observações sobre a metodologia em outra instituição e com outro nível de ensino.

É importante ressaltar que a experiência desse docente é incluída aqui como um complemento, uma vez que não foi realizada diretamente pelo pesquisador.

A aplicação ocorreu na disciplina de Algoritmos, do curso Bacharelado em Ciência da Computação da Universidade Tecnológica Federal do Paraná – UTFPR – Campus Ponta Grossa. O curso tem duração de 4 anos, visa uma formação profissional e o embasamento para uma possível pós-graduação dos egressos, com um perfil, portanto, diferente dos alunos que participaram dos experimentos realizados pelo pesquisador.

A disciplina de Algoritmos está no 1º semestre do curso e possui carga horária de 75 horas. O docente responsável trabalhou toda essa carga seguindo as diretrizes descritas neste trabalho, ou seja: priorizando o uso de laboratório e o contato prático dos alunos com programação de computadores.

Como no caso do IFPR, a disciplina ministrada na UTFPR visa apresentar a programação de computadores; os tópicos tratados são a princípio os mesmos, como laços, variáveis e sub-rotinas, entretanto a abordagem deve acontecer com mais amplitude e profundidade.

Para efeito de comparação, foram obtidos dados sobre aplicações realizadas por esse docente e uma edição anterior da disciplina, a cargo de outro professor. O Quadro 24 a seguir sumariza as primeiras 15 horas aula da disciplina.

Comparação entre outra abordagem e a abordagem proposta		
Número da aula	**Conteúdo (outra abordagem)**	**Conteúdo (abordagem proposta)**
1	Introdução ao curso.	Introdução ao curso. Exercícios de raciocínio lógico e pensamento crítico, envolvendo textos.
2	Exemplos do cotidiano de algoritmos. Funcionamento básico do computador.	BASIC: comandos PRINT e FOR
3	Pseudocódigo; entrada, saída, variáveis.	Exercícios para localizar oportunidades para usar laços. ASCII.
4	Pseudocódigo e fluxograma. O processo de criação de um programa.	Primeira avaliação curta. Comandos gráficos em BASIC e criação de desenhos na tela (por exemplo, série de quadrados).
5	A linguagem C. Corpo de um programa. main, printf, scanf, variáveis e operadores.	Conceito de variável; espaços de memória e endereços. Exercícios em BASIC envolvendo expressões, atribuições e laços.
6	Aula em laboratório; uso de linux. Compilação de programas.	Noção de acumulador; exemplos e exercícios. Implementação em Basic de Bhaskara. O problema de raízes negativas e solução usando comando IF-THEN.
7	Procedimentos: definição e exemplos.	Exercícios usando todos comandos vistos até aqui.
8	Funções: definição e exemplos. Passagem por valor e por referência.	Avaliação. Conteúdo = programação em BASIC 256; comandos PRINT, INPUT, IF sem cláusula ELSE; FOR-TO-STEP; variáveis acumuladoras; resto de divisão
9	Exemplos de uso de funções.	Solução da prova em sala. A cláusula ELSE do comando IF.
10	Desvio condicional.	Visão geral de laço WHILE. Comandos para leitura de estado de teclas. Esqueleto básico de um programa de animação gráfica. Aula em laboratório: criação de um jogo simples.

Quadro 24: Comparação entre outra abordagem e a abordagem proposta
Fonte: Autoria própria com base nos diários de classes dos professores

A comparação das 15 primeiras horas aulas entre outra abordagem e a abordagem proposta, presente no Quadro 24 indica que a abordagem proposta contemplou a apresentação de mais comandos, e além disso, a aplicação prática destes comandos o que pode favorecer a aquisição de experiência em programação por parte dos estudantes, podendo servir de ponto de ancoragem para a aprendizagem futura destes conceitos.

Em contrapartida a outra abordagem privilegiou uma grande carga de definição de conceitos e se considerarmos que os estudantes não possuíam os subsunçores adequados, visto que para muitos a programação de computadores é

algo completamente novo, a apresentação destes conceitos sem a devida aplicação, ou seja, sem o devido entendimento para que estes serviam por parte dos alunos, podendo assim contribuir para aumentar o grau de abstração presente no processo e tornando a aprendizagem mecânica.

5.4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Com base na revisão de literatura e nos resultados coletados na primeira e segunda aplicação prática deste estudo, apresenta-se a seguir a análise e discussão dos dados desta pesquisa.

O objetivo da pesquisa foi elaborar uma proposta de abordagem prática, baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação.

A hipótese que orientou o trabalho, é a de que uma proposta de abordagem prática baseada em Ausubel, que priorize e exponha os alunos mais cedo ao uso prático do computador traga resultados positivos comparativamente à abordagem tradicional.

Para isso, primeiramente foi realizado um teste piloto onde foi possível verificar a aplicabilidade da proposta de abordagem e seus conteúdos, e também analisar a repercussão de seu uso junto aos estudantes

Em seguida visando a validação da pesquisa buscou-se comparar a aplicação da proposta de abordagem a uma abordagem tradicional (Primeira Edição) para o ensino dos fundamentos de programação de computadores. Para isso elencou-se os seguintes critérios:

- cumprimento da ementa e objetivos da disciplina, observando se os tópicos previstos foram exercidos;
- comparação da época em que os assuntos foram introduzidos;
- comparação da época de realização de avaliações e provas;
- comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina;
- comparação do desempenho dos alunos;

Visando enriquecer os resultados foram adicionadas observações qualitativas do pesquisador a respeito do engajamento dos alunos e sua participação e comentários do professor titular da disciplina a respeito do desempenho dos estudantes.

Os resultados referentes a estes critérios estão no subitem Resultados da Primeira Edição.

Com base nos resultados foi possível observar que a proposta de abordagem cumpre com os tópicos presentes na ementa da maioria das disciplinas de introdução a computadores.

Os Resultados da Primeira Edição indicam também que a exposição dos estudantes mais cedo ao uso prático do computador e a assuntos que normalmente são vistos de maneira conceitual primeiramente na abordagem tradicional não interferem negativamente no desempenho dos estudantes.

Cabe observar que geralmente o escopo das disciplinas introdutórias em programação, inicialmente gira em torno da definição de conceitos, entretanto, durante esta aplicação em nenhum momento isso foi fornecido aos estudantes, estes eram instigados a realizar operações e atividades de alteração de códigos e partir disso eram questionados sobre o que acontecia, formulando seus próprios conceitos.

Desta forma, os Resultados da Primeira Edição da aplicação indicam que a não apresentação de forma teórica dos fundamentos de algoritmos e programação, bem como a sua representação por meio de fluxogramas e pseudocódigo não influenciou a capacidade dos estudantes assimilarem os conteúdos envolvidos em programação e também desenvolverem seus próprios códigos.

De acordo com a aprendizagem significativa: um dos pontos fundamentais consiste em determinar aquilo que o aprendiz já sabe ou conhece (MASINI; MOREIRA, 2001), ou seja, o estado atual da sua estrutura cognitiva (conjunto de ideias que, no aluno, preexistem à nova aprendizagem), para que a proposta de ensino seja baseada nestes conhecimentos.

No entanto para a maioria dos estudantes iniciantes, a programação de computadores é um assunto completamente novo (DIJKSTRA, 1989). Sendo assim, os estudantes na maioria das vezes não possuem os subsunçores necessários para que a aprendizagem ocorra de maneira significativa, fazendo com que a aprendizagem de programação de computadores ocorra com pouco sentido.

Nestes casos onde o aprendiz não possui o conjunto de conhecimentos prévios necessários sobre o novo conceito a ser aprendido, Ausubel sugere como estratégia de manipular a estrutura cognitiva, com organizadores prévios (MOREIRA; SOUSA, 1996). Organizadores prévios, tem a função de ponte entre o que o aprendiz sabe e o que deve saber, visam estabelecer relações entre ideias, proposições e conceitos já existentes na estrutura cognitiva, a fim de que a aprendizagem possa ser significativa.

Exemplificando, no presente caso a ideia de “repetição” já é conhecida dos estudantes. Entretanto, ao apresentar o computador realizando uma repetição, não se exige do estudante imaginar o funcionamento de um comando como FOR: a execução de uma repetição pela máquina adquire um caráter quase concreto. Ao mudar os valores limites do laço em seu computador quando questionado pelo professor (por exemplo, em lugar de 10 iterações realizar 20), o estudante pode ainda experimentar hipóteses e sedimentar mais os conceitos. Em comparação, o estudante na aula teórica só irá confirmar o que viu mais tarde - ou outro dia – em laboratório, quando poderá ainda ser frustrado por erros de sintaxe inesperados e se questionar se afinal entendeu ou não a matéria.

Na proposta da proposta de abordagem implementada onde os conteúdos a serem aprendidos pelos estudantes se deram por meio da apresentação de pequenas estruturas de códigos simples, que ao longo do processo vão se tornando maiores e mais complexas e por meio de atividades práticas, os estudantes puderam desenvolver seu próprio repertório de conhecimentos e habilidades para resolver problemas e representa-los no ambiente computacional.

Além disso a comparação da época em que os assuntos foram introduzidos e a comparação dos conteúdos abordados em avaliações e provas referentes a ementa da disciplina, por meio das avaliações do professor titular e professor pesquisador, apontou que a proposta de abordagem, propícia aos estudantes verem os conteúdos mais vezes.

Sendo assim, os conteúdos na abordagem proposta foram apresentados inter-relacionados e de maneira cíclica onde o estudante pode ganhar maturidade para assimilar estes conteúdos, o que geralmente não ocorre na abordagem tradicional, pois nesta os conteúdos são apresentados isoladamente como caixas, ou seja, variáveis uma caixa, estruturas de decisão outra e estruturas de repetição outra,

o que faz com que o aluno sinta dificuldades para combinar esses conceitos na solução de problemas quando necessário.

Alguns estudantes, demonstraram dificuldades na realização das atividades propostas durante a aplicação, este processo pode distanciar ainda mais o aluno dos objetivos destas disciplinas, no entanto, é crucial que o professor não forneça as respostas, mas desenvolva situações que visem oportunizar ao aluno maneiras de romper estes bloqueios.

Ainda com relação ao desempenho dos alunos, foi possível perceber também que num determinado momento alguns alunos apresentam um ganho de aprendizagem referente a estes conteúdos significativos e avançam sem problemas durante os conteúdos, e, no entanto, outros alunos mesmo se esforçando não conseguem chegar a tal ganho. Tal ocorrência também já foi percebida por Ambrósio et al. (2011).

A aprendizagem destes conteúdos não ocorre em tempo real, por isso, há necessidade de que um mesmo conteúdo seja revisto e aplicado por diversas vezes em momentos diferentes e situações diferentes, portanto, a abordagem deve ser feita na forma de uma cíclica e crescente, para que no correr do processo de aprendizagem o estudante adquira maturidade e não desenvolva repulsa ao conteúdo.

Por isso, a motivação é outro fator essencial para que os alunos aprendam, e por isso conseguir o engajamento dos alunos e que estes estejam realmente envolvidos com o processo pode ser outro fator de sucesso para retenção destes conhecimentos. Neste sentido o investimento do professor no ambiente de sala aula e a utilização de conteúdos de maior interesse pode colaborar para o envolvimento dos alunos com estes conteúdos.

Neste sentido, o pesquisador observou indícios de que as aulas práticas dos conteúdos introdutórios de programação de computadores podem produzir um engajamento maior por parte dos alunos e, além disso, podem diminuir a lacuna no quesito abstração, um dos grandes problemas apontados por pesquisas sobre o ensino/aprendizagem de programação de computadores.

O desempenho dos alunos na disciplina do professor titular e do professor pesquisador durante a Primeira Edição da aplicação, apresenta alguns resultados passíveis de discussão quanto aos seus critérios ou aspectos, entre estes aspectos podemos citar como exemplo a introdução de novos assuntos, visando ilustrar melhor este exemplo a seguir é apresentado o Quadro 25, comparando superficialmente a

introdução de novos assuntos referentes a ementa da disciplina entre o professor titular e o professor pesquisador.

Comparação de introdução de novos assuntos – Professor Titular X Professor Pesquisador			
Unidade de tempo	Professor titular	Professor pesquisador	Observações
1º Bimestre	Novos assuntos Programação sequencial	Novos assuntos Programação sequencial; Instruções de seleção Instruções de repetição Funções e procedimentos Implementação de problemas em uma linguagem de programação	Nesta unidade os dois professores introduziram assuntos novos, entretanto o professor titular deu ênfase a aspectos conceituais abordando somente um tópico referente a ementa e o professor pesquisador deu ênfase a prática abordando diversos tópicos.
2º Bimestre	Assuntos já introduzidos pelo professor pesquisador	Revisão e aprofundamento dos assuntos já introduzidos e implementação numa nova linguagem	Nesta unidade o professor titular abordou instruções de seleção e Implementação de problemas em uma linguagem de programação, assuntos já abordados anteriormente pelo professor pesquisador.
3º Bimestre	Assuntos já introduzidos pelo professor pesquisador	Revisão e aprofundamento dos assuntos já introduzidos e implementação numa nova linguagem; Novo assunto *Vetores*	Nesta unidade o professor titular abordou instruções de repetição, assunto já abordado anteriormente pelo professor pesquisador
4º Bimestre	Assuntos já introduzidos pelo professor pesquisador Novo assunto *Matrizes*	Revisão e aprofundamento dos assuntos já introduzidos; Novo assunto *Matrizes*	Nesta unidade os dois professores introduziram assuntos novos

Quadro 25: Comparação de introdução de novos assuntos – Professor Titular X Professor Pesquisador

Fonte: Autoria própria

Neste ponto cabe a ressalva que a comparação apresentada no Quadro 25, é muito superficial, uma vez que além dos tópicos presentes na ementa da disciplina foram adicionados outros tópicos também importantes para o aprendizado dos alunos.

Um outro exemplo de resultado passível de discussão seria por que a disciplina ministrada pelo professor pesquisador possuí mais alunos com conceitos “A” e “B” do que a disciplina do professor titular. Cabe observar que cada professor tem sua própria metodologia de aula, interação com os alunos, desenvolvimento de avaliações e critérios para correções. E além disso, podemos mencionar também que enquanto os alunos realizam avaliações escritas na disciplina do professor titular, as avaliações do professor pesquisador eram desenvolvidas no computador, desse modo os alunos poderiam testar suas repostas e trabalhar a fim de corrigir os erros observados, o que não acontece em avaliações escritas.

Ainda com relação ao desempenho dos alunos, cabe observar que apesar da não uniformidade de desempenho dos alunos nas disciplinas, o índice de aprovação dos alunos, 69% na disciplina do professor titular e 86% na disciplina do professor pesquisador, estão acima da média num contexto retratado por diversas pesquisas.

De fato, é no mínimo presumível que a disciplina ministrada pelo professor pesquisador influenciou no desempenho dos alunos na disciplina do professor titular, e também é questionável o quanto ela foi influenciada no desempenho dos alunos, neste aspecto podemos presumir que a disciplina do professor pesquisador influenciou no desempenho dos alunos, seja por que eles tiveram acesso aos assuntos referentes a ementa da disciplina mais cedo, mais vezes ou também pelo registro de comentários do professor titular a respeito do desenvolvimento dos alunos.

No entanto, mensurar este grau de influência da disciplina ministrada pelo professor pesquisador na disciplina ministrada pelo professor titular seria algo que nos afastaria do objetivo principal desta pesquisa e além disso, compreenderia outras aplicações dentro de um tempo não disponível. E por esse motivo foi realizada uma segunda aplicação, visando identificar não o quanto ela influenciou, mas o quanto ela foi influenciada.

Os Resultados da Segunda Edição da aplicação prática da proposta de abordagem, foram bem similares aos Resultados da Primeira Edição, entretanto esta aplicação ocorreu isolada em outra modalidade de ensino, ou seja, sem um outro professor ministrando concomitantemente a disciplina de maneira tradicional.

Estes resultados indicam que a Primeira Edição da aplicação prática da proposta de abordagem, não foi ou pouco foi influenciada, não sendo perceptível a este pesquisador. E isto foi reforçado pelas observações feitas pela aplicação paralela por outro docente em outra instituição.

Por fim cabe destacar alguns outros aspectos positivos referentes aos resultados presentes nas aplicações práticas da proposta de abordagem, como por exemplo: a da definição de conceitos, geralmente presentes no escopo da maioria das disciplinas introdutórias de programação e que não foi utilizada por esta aplicação, não influenciou a capacidade dos alunos na resolução de problemas e sua codificação.

Percebeu-se indícios de que a aplicação prática, como na apresentação do laço de repetição *FOR,* quando o aluno pode ver e analisar a resposta do computador mediante as suas entradas de comandos, ou seja, o aluno pode ver como a programação acontece de maneira concreta o que pode contribuir para diminuir o grau de abstração presente na matéria de programação, favorecendo entre outros aspectos a motivação dos estudantes (PELIZZARI; KRIEGL, 2002; PIVA JR; FREITAS, 2010).

Durante o ensino/aprendizagem dos fundamentos de programação de computadores, ao trabalhar com outras linguagens em lugar de BASIC, em nossa experiência de sala de aula, percebe-se que os estudantes concentram seus esforços nas características específicas de sintaxe e semântica da linguagem, e não em como realmente acontece a programação (a resolução do problema e sua formalização numa linguagem de programação), sendo assim, a utilização de linguagens de programação com sintaxe mais simples e clara, pode favorecer o processo (REBOUÇAS et al. 2010; MANSO; OLIVEIRA; MARQUES, 2009).

Alguns alunos podem ter dificuldades em identificar erros nos resultados produzidos pelos seus códigos inicialmente, estas dificuldades estão diretamente relacionadas a pouca experiência em programação, deste modo, a situação indicada pela proposta de abordagem, possibilita aos alunos verem os conteúdos mais vezes, o que pode contribuir para a aquisição de experiência em programação (AURELIANO; TEDESCO, 2012; BYRNE; LYONS, 2001; HAGAN; MARKHAM, 2000).

Como fator positivo pode-se destacar à carga horária utilizada, pois os conteúdos apresentados durante esta aplicação, quase atendem a todos os requisitos das disciplinas que contemplam carga horária muito superior.

Portanto podemos concluir que os resultados das aplicações colaboram na confirmação da hipótese sugerida por este estudo.

6 CONSIDERAÇÕES FINAIS

O processo de ensino/aprendizagem dos fundamentos de programação de computadores, tem se mostrado difícil para estudantes e professores. Isto é exposto pela literatura que referencia os elevados níveis de insucesso, as elevadas taxas de desistência e até mesmo abandono do curso.

As dificuldades envolvidas no processo de ensino/aprendizagem de computadores abrangem os quatro integrantes envolvidos no processo, sendo: a instituição, o professor, o aluno e os conteúdos, com diversas variáveis para cada um destes.

A literatura disponibiliza propostas que visam contribuir com o processo de aprendizagem de programação ou atenuar as dificuldades existentes no processo. Estas propostas geralmente se encaixam em três principais vertentes: ferramentas, estratégias e a combinação de ferramentas e estratégias.

Deste modo, esta pesquisa teve como objetivo elaborar uma proposta de abordagem prática baseada na teoria da aprendizagem significativa, enfatizando/valorizando a interação com a máquina e expondo os estudantes mais cedo ao uso prático do computador para o ensino de programação.

Elaborou-se uma proposta de abordagem para as disciplinas introdutórias de programação. Esta proposta foi submetida a um teste piloto, seguido de uma pesquisa experimental, onde foram conduzidas aplicações da proposta pelo professor pesquisador.

Os resultados das aplicações indicam que:

I. A proposta cumpre com os tópicos presentes na ementa da maioria das disciplinas de introdução a programação;

II. Carga horária utilizada para apresentação dos tópicos da disciplina foi menor comparativamente a carga horária habitual, desta forma, o professor pode aproveitar o tempo para trabalhar mais vezes estes conteúdos permitindo uma melhor fixação destes por parte do estudante;

III. A proposta oportuniza aos estudantes verem os conteúdos mais vezes de maneira inter-relacionados e de maneira cíclica onde o estudante pode ganhar maturidade para assimilar estes conteúdos;

IV. As aulas práticas dos conteúdos introdutórios de programação de computadores podem produzir um engajamento maior por parte dos alunos e podem

diminuir a lacuna no quesito abstração, um dos grandes problemas apontados por pesquisas sobre o ensino /aprendizagem de programação de computadores.

V. A exposição dos estudantes mais cedo ao uso prático do computador e a assuntos que normalmente são vistos de maneira conceitual primeiramente na abordagem tradicional não interferiram negativamente no desempenho dos estudantes.

Neste sentido, recorremos a Ausubel, o qual indica que cada disciplina tem uma estrutura articulada e hierarquicamente organizada de conceitos. No entanto, a ordem em que os principais conceitos e ideias da matéria de ensino são apresentadas muitas vezes não é a mais adequada para facilitar a interação com o conhecimento prévio do aluno.

Deste modo, os resultados das aplicações da proposta de abordagem, apresentam indícios que a estrutura articulada e hierarquicamente organizada de conceitos no ensino de programação pode não ser o mais adequado, ou seja, em nossa abordagem os conteúdos foram apresentados inter-relacionados e de maneira cíclica onde o estudante pode ganhar maturidade para assimilar, o que geralmente não ocorre na abordagem tradicional.

Por fim, devemos mencionar algumas limitações encontradas na realização da pesquisa, entre elas: a não possibilidade de realizar o estudo com formação de um grupo de controle, e também aplicação paralela com outro docente.

Entretanto hipótese foi confirmada, visto que as aplicações indicam resultados positivos comparativamente à abordagem tradicional.

6.1 ESTUDOS FUTUROS

Como trabalhos futuros sugere-se a replicação deste estudo utilizando a proposta de abordagem, com a formação de grupos de controle e o acompanhamento dos alunos participantes em disciplinas de programação subsequentes, com o objetivo de verificar os resultados posteriores da aplicação da proposta de abordagem.

Agregar a esta pesquisa aspectos motivacionais e/ou cognitivos;

Analisar detalhadamente como certas habilidade e competências influenciam a aprendizagem de programação.

E por fim, identificar quais as condições para que num determinado momento alguns alunos apresentem um ganho de aprendizagem significativo referente a programação de computadores e progridam sem problemas durante os conteúdos, e, no entanto, outros alunos mesmo se esforçando não conseguem chegar a tal ganho como também sugere Ambrósio *et al.*(2011).

Sugere-se também, em termos mais gerais, trabalhar os questionamentos:

Por que ainda reproduzimos métodos de ensino de programação que remetem ao século passado, onde grande parte das escolas e alunos não tinham acesso as ferramentas disponíveis atualmente?

Por que os conteúdos de programação geralmente ainda são apresentados de maneira que a solução do problema não é associada à sua representação em uma linguagem de programação?

Por que apesar de inúmeras pesquisas, o quadro retratado por Dijkstra em 1989, sobre a cruel realidade de ensinar ciência da computação ainda persiste?

Como ensinar programação eficazmente para os alunos atuais?

REFERÊNCIAS

ALMEIDA, E. S. et al. AMBAP: Um ambiente de apoio ao aprendizado de Programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 22., 2002, Florianópolis. **Anais...** Florianópolis: SBC, 2002. Disponível em: <http://200.17.137.110:8080/licomp/Members/jeanemelo/plonelocalfolderng.2006-04-10.7475913377/PEP/Aulas23/2002SbcAmbap.pdf>. Acesso em: 24 abr. 2015.

AMBRÓSIO, A. P. L. et al. Programação de Computadores: compreender as dificuldades de aprendizagem dos alunos. **Revista Galego-Portuguesa de Psicoloxía e Educación**, v. 19, n. 1, ano 16, p. 185–197, 2011. Disponível em: <http://repositorium.sdum.uminho.pt/handle/1822/15554>. Acesso em: 14 ago. 2014.

AURELIANO, V. C. O.; TEDESCO, P. C. A. R. Ensino-aprendizagem de Programação para iniciantes: uma revisão sistemática da literatura focada no SBIE e WIE. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 23., 2012, Rio de Janeiro. **Anais...** Rio de Janeiro, 2012

AUSUBEL, D.P. **Aquisição e retenção de conhecimentos**: uma perspectiva cognitiva. Lisboa: Plátano, 2003. Disponível em: <http://files.mestrado-em-ensino-de-ciencias.webnode.com/200000007-610f46208a/ausebel.pdf>. Acesso em: 14 ago. 2014.

AUSUBEL, D.P. **The psychology of meaningful verbal learning**. New York: Grune and Stratton, 1963. Disponível em: <http://psycnet.apa.org/psycinfo/1964-10399-000>. Acesso em: 10 jun. 2015.

AUSUBEL, D.P; NOVAK, JD; HANESIAN, H. **Educational psychology**: a cognitive view. New York:, Holt, Rinehart and Winston.1968. Disponível em: <http://www.spbkbd.com/english/art_english/art_51_030211.pdf>. Acesso em: 8 jun. 2015.

AUSUBEL, D.P; NOVAK, JD; HANESIAN, H. **Psicologia educacional**. Rio de Janeiro, Interamericana, 1980. Disponível em: <https://scholar.google.com.br/scholar?q=Psicologia+educacional&btnG=&hl=pt-BR&as_sdt=0%2C5#3>. Acesso em: 29 jun. 2015.

BARANAUSKAS, M. C. C. Procedimento, função, objeto ou lógica? Linguagens de programação vistas pelos seus paradigmas. In: José Armando Valente. (Org.). **Computadores e Conhecimento - Repensando a Educação**. Campinas, SP: Unicamp, 1994, p. 45-63. Disponível em: <http://www.nied.unicamp.br/publicacoes/arquivos/3XaQ1o8Nmk\nhttp://disciplinas.dcc.ufba.br/pub/MATA56/Exercicios/(Leitura_e_Resenha)_ArtigoDiscussaoParadigmas.pdf>. Acesso em: 29 jun. 2015.

BEN-ARI, M. Constructivism in computer science education. **Journal of Computers in Mathematics and Science Teaching**, v.20, n.1, p. 45-73, 2001.

BENNEDSSEN, J.; CASPERSEN, M. E. Abstraction ability as an indicator of success for learning computing science? In: In**: Proceedings of the ACM Workshop on International Computing Education Research, ICER 08**. Sydney, Australia, 2008. Disponível em: <http://dl.acm.org/citation.cfm?id=1404523>. Acesso em: 8 jun. 2015.

BERGERSEN, G. R.; GUSTAFSSON, J. E. Programming skill, knowledge, and working memory among professional software developers from an investment theory perspective. **Journal of Individual Differences**, v. 32, n. 4, p. 201–209, 2011. Disponível em: <http://econtent.hogrefe.com/doi/full/10.1027/1614-0001/a000052>. Acesso em: 8 jun. 2015.

BIGGS, J. Aligning teaching and assessing to course objectives. In: **Teaching and Learning in Higher Education:** New Trends and Innovations. University of Aveiro, 13-17 April, 2003. Disponível em: <https://www.dkit.ie/zh-hans/system/files/Aligning_Reaching_and_Assessing_to_Course_Objectives_John_Biggs.pdf>. Acesso em: 8 jun. 2015.

BIGGS, J. What the student does: teaching for enhanced learning. **Higher Education Research & Development**, v.18, n.1, p.57-75, 1999. Disponível em: < https://www.ntnu.no/documents/601374998/0/Biggs+(1999)%20What+the+Student+Does.pdf/02a63c23-929e-459a-aa0f-efc04521c32c>. Acesso em: 7 jun. 2015.

BORGES, M. A. F. Avaliação de uma metodologia alternativa para a aprendizagem de programação. In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO – WEI 2000, 8., 2000, Curitiba. **Anais...** Curitiba: [s.n.], 2000. Disponível em: <http://www.niee.ufrgs.br/eventos/SBC/2000/pdf/wei/relatos/selecionados/wei006.pdf>. Acesso em: 14 ago. 2014.

BRAATHEN, P.C. O processo ensino aprendizagem em disciplinas básicas do terceiro grau. **Revista Educação & Tecnologia (E&T)**, Belo Horizonte, v.8, n.1, p.34-41, jan./jun. 2003. Disponível em: <http://www.revista.cefetmg.br/index.php/revista-et/article/view/53>. Acesso em: 29 jun. 2015.

BRANDÃO, L. O.; BRANDÃO, A. A. F.; RIBEIRO, R. S. iVProg - uma ferramenta de programação visual para o ensino de algoritmos. In: CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (CBIE), 2012. **Anais...**, 2012. Disponível em: <http://ceie-sbc.tempsite.ws/pub/index.php/wcbie/article/view/1668>. Acesso em: 14 ago. 2014.

BRASIL. Ministério da Educação. Secretaria de Educação Superior. **Diretrizes Curriculares de Cursos da área de Computação e Informática.** Brasília: MEC/SESU, 2001.

BRASIL. Ministério da Educação. Secretaria de Educação Profissional e Tecnológica. **Catálogo Nacional de Cursos Técnicos**. Brasília: MEC/Setec, 2012..

BRUSILOVSKY, Peter et al. Teaching programming to novices: a review of approaches and tools. In: **Proceedings of ED-MEDIA'94 - World conference on educational multimedia and hypermedia**. Vancouver, Canada, p. 103-110, 1994. Disponível em: <http://www.computer.org/portal/web/csdl/doi/10.1109/FIE.1999.839268>. Acesso em: 8 jun. 2015.

BYRNE, P.; LYONS, G The Effect of Student Attributes on Success. **ACM SIGCSE Bulletin**, New York, NY. v.33, n.3, p.49-52, Sept. 2001. Disponível em: <http://dl.acm.org/citation.cfm?id=377467>. Acesso em: 24 abr. 2015.

CAMPOS, R. L. B. L. **ERM2C**: uma metodologia para melhoria do ensino-aprendizado de lógica de programação. 2010. Disponível em: <http://www.fejal.br/erbase2010/papers/weibase/65502.pdf>. Acesso em: 14 ago. 2014.

CÂNDIDA, L.; MARCELINO, M. J.; MENDES, A. J. The impact of learning styles in introductory programming learning.. In: INTERNATIONAL CONFERENCE ON ENGINEERING EDUCATION – ICEE 2007, Coimbra, Portugal. **Anais...** Coimbra, Portugal, 2007. Disponível em: <http://ineer.org/Events/ICEE2007/papers/432.pdf>. Acesso em: 8 jun. 2015.

CASPERSEN, M. E.; BENNEDSEN, J. Instructional design of a programming course: a learning theoretic approach. In**: Proceedings of the 3rd International Workshop on Computing Education Research - ICER**. Atlanta, GA: ACM, 2007. Disponível em: <http://dl.acm.org/citation.cfm?id=1288595>. Acesso em: 28 set. 2014.

CASTRO, T. H. C. Arquitetura SAAP: sistema de apoio à aprendizagem de programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 22., 2002. Florianópolis. **Anais...** Florianópolis: SBC/UFSC, 2002. Disponível em: <http://www.researchgate.net/profile/Davidson_Cury/publication/228782248_Arquitetura_SAAP_Sistema_de_Apoio__Aprendizagem_de_Programao/links/544fa40b0cf26dda089208ba.pdf>. Acesso em: 8 jun. 2015.

CATTERALL, J. Beyond the classroom: The effect of institutional factors on scholarly teaching and learning innovations. **Studies in Learning, Evaluation, Innovation and Development**, v.5, n.3, p.55-66, 2008. Disponível em: <https://scholar.google.com.br/scholar?q=Beyond+the+classroom%3A+The+effect+of+institutional+factors+on+scholarly+teaching+and+learning+innovations&btnG=&hl=pt-BR&as_sdt=0%2C5#0>. Acesso em: 8 jun. 2015.

CHATZOPOULOU, D. I.; ECONOMIDES, A. A. Adaptive assessment of student's knowledge in programming courses. **Journal of Computer Assisted Learning**, v. 26, n. 4, p. 258–269, 2010. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1111/j.1365-2729.2010.00363.x/full>. Acesso em: 8 jun. 2015.

COMPEAU, D.; HIGGINS, C. Computer Self-Efficacy: development of a measure and initial test. **Management Information Systems Quarterly**, v.19, n.2, p. 189–211, June, 1995. Disponível em: <http://www.jstor.org/stable/249688>. Acesso em: 8 jun. 2015.

COOPER, S; DANN, W; PAUSCH, R. Using animated 3D graphics to prepare novices for CS1. **Computer Science Education**, v.13, n.1, 2003. Disponível em: <http://www.tandfonline.com/doi/abs/10.1076/csed.13.1.3.13540>. Acesso em: 8 jun. 2015.

COOPER, S.; DANN, W.; PAUSCH, R. Alice: a 3-D tool for introductory programming concepts. **Journal of Computing Sciences in Colleges**, v.15, n.5, p.101-116, 2000. Disponível em: <http://dl.acm.org/citation.cfm?id=364161>. Acesso em: 8 jun. 2015.

DÉBORA, Ayla; REBOUÇAS, Dantas S; MARQUES, Diego Lopes. Aprendendo a ensinar programação combinando jogos e Python. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 21., 2010. João Pessoa, PB. **Anais...**, 2010. Disponível em: <http://www.ccae.ufpb.br/sbie2010/anais/Inicio.html>. Acesso em: 6 jun. 2015.

DELGADO, C. et al. Identificando competências associadas ao aprendizado de leitura e construção de algoritmos. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 25., 2005, São Leopoldo, RS. **Anais...** São Leopoldo: Sonopress, 2005. Disponível em: <http://200.169.53.89/download/CD congressos/2005/SBC 2005/pdf/arq0037.pdf>. Acesso em: 6 jun. 2015.

DELGADO, C. et al. Uma abordagem pedagógica para a iniciação ao estudo de algoritmos. In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO (WEI'2004), 12., 2004, Salvador, BA. **Anais...**, 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2004/0024.pdf>. Acesso em: 6 jun. 2015.

DENNING, P. J. et al. A discipline in crisis. **Communications of the ACM**, v.24, n.6, 370–374, 1981. Disponível em: <http://www.researchgate.net/profile/Anthony_Hearn/publication/220423882_A_Discipline_in_Crisis/links/02e7e52cb0525ad928000000.pdf>. Acesso em: 8 jun. 2015.

DEREMER, D. Improving the learning environment in CS I: experiences with communication strategies. **ACM SIGCSE Bulletin**, v.25, n.3, 1993. Disponível em: <http://dl.acm.org/citation.cfm?id=165418>. Acesso em: 7 jun. 2015.

DETERS, J. I. et al. O desafio de trabalhar com alunos repetentes na disciplina de algoritmos e programação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 19., 2008. Fortaleza, Ceará. **Anais...** 2008. Disponível em: <http://www.proativa.virtual.ufc.br/sbie/CD_ROM_COMPLETO/workshops/workshop 2/O Desafio de Trabalhar com Alunos Repetentes na.pdf>. Acesso em: 19 ago. 2014.

DIJKSTRA, Edsger W. On the cruelty of really teaching computer science. **Communications of the ACM**, v.32, n.12, Dec. 1989. Disponível em: <http://www.smaldone.com.ar/documentos/ewd/EWD1036_pretty.pdf>. Acesso em: 7 jun. 2015.

DIM, C. A.; ROCHA, F. E. L. APIN: Uma ferramenta para aprendizagem de lógicas e estímulo do raciocínio e da habilidade de resolução de problemas em um contexto computacional no ensino médio. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (CSBC), 31., 2011, Natal, RN. **Anais...**, 2011. Disponível em: <http://www.dimap.ufrn.br/csbc2011/anais/eventos/contents/WEI/Wei_Secao_6_Artigo_2_Dim.pdf>. Acesso em: 19 ago. 2014.

DUNICAN, E. Making the analogy : alternative delivery techniques for first year programming courses. In: **Proceedings from the 14º Workshop of the Psychology of Programming Interest Group**. Carlow: Brunel University, 2002. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.115.8440&rep=rep1&type=pdf>. Acesso em: 25 abr. 2015.

EACHUS, P.; CASSIDY, S. Developing the computer user self-efficacy (CUSE) scale: investigating the relationship between computer self-efficacy, gender and experience with computers. **Journal of Educational Computing Research**, v.26, n.2, p.133-153, 2002. Disponível em: <http://usir.salford.ac.uk/1169/>. Acesso em: 9 jun. 2015.

ECKERDAL, A.; THUNÉ, M.; BERGLUND, A. What does it take to learn'programming thinking'? In: **Proceedings of the First International Workshop on Computing Education Research, ICER 05.** New York: ACM, 2005. Disponível em: <http://dl.acm.org/citation.cfm?id=1089799>. Acesso em: 7 jun. 2015.

FALKEMBACH, G. A. M.; ARAUJO, F. V.; et al. Ambiente de aprendizagem adaptado para algoritmos (A4). In: TALLER INTERNACIONAL DE SOFTWARE EDUCATIVO, 8., 2003. Santiago. **Anais...** Santiago, Chile, 2003. Disponível em: <http://200.17.137.110:8080/licomp/Members/jeanemelo/plonelocalfolderng.2006-04-10.7475913377/PEP/PEP2009/Aula6/Grupo3/a4_trad.pdf>. Acesso em: 8 jun. 2015.

FALKEMBACH, G. A. M.; AMORETTI, M. S. M.; et al. Aprendizagem de algoritmos: uso da estratégia ascendente de resolução de problemas. In: TALLER INTERNACIONAL DE SOFTWARE EDUCATIVO, 8., 2003. Santiago. **Anais...** Santiago, Chile, 2003. Disponível em: <http://www.tise.cl/2010/archivos/tise2003/papers/aprendizagem_de_algoritmos.pdf>. Acesso em: 6 jun. 2015.

FARIA, E. S. J.; COELLO, J. M. A. Detectando diferenças significativas entre programas como auxílio ao aprendizado colaborativo de programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (CSBC), 24. 2004. Salvador, Bahia, **Anais...** Salvador: SBC, 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2004/009.pdf>. Acesso em: 10 jun. 2015.

FELDER, Richard M.; BRENT, Rebecca. Designing and teaching courses to satisfy the ABET Engineering Criteria. **Journal of Engineering Education**, v. 92, n. 1, p. 7-25, 2003. Disponível em: <http://onlinelibrary.wiley.com/doi/10.1002/j.2168-9830.2003.tb00734.x/abstract>. Acesso em: 6 jun. 2015.

FERNANDES, J. H. C. Ensino introdutório de computação e programação: uma abordagem concreta, construtivista, linguística e histórica. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (CSBC), 22., 2002, Florianópolis, SC. **Anais...** Porto Alegre, RS: SBC, 2002. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2002/0014.pdf>. Acesso em: 7 jun. 2015.

FINIZOLA, Antonio Braz et al. O ensino de programação para dispositivos móveis utilizando o MIT-App Inventor com alunos do ensino médio. In: WORKSHOP DE INFORMÁTICA NA ESCOLA (WIE 2014), 20., 2014, Dourados, MS., **Anais...**, Dourados, MS, p. 337–341, 2014. Disponível em: < http://www.br-ie.org/pub/index.php/wie/article/view/3116/2624>. Acesso em: 9 jun. 2015.

GARCIA, I. C.; REZENDE, P. J.; CALHEIROS, F. C. Astral: Um ambiente para ensino de estruturas de dados através de animações de algoritmos. **Revista Brasileira de Informática na Educação**, Florianópolis, SC, v. 1, p. 71-80, 1997.. Disponível em: <http://www.ic.unicamp.br/~rezende/garcia.htm>. Acesso em: 6 jun. 2015.

GIL, AC. **Como elaborar projetos de pesquisa**. São Paulo, 2002.

GIRAFFA, L. M. M.; MARCZAK, S. S.; ALMEIDA, G. O ensino de algoritmos e programação mediado por um ambiente na web. In: CONGRESSO NACIONAL DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 23., 2003, Campinas-SP. **Anais...** Campinas-SP, 2003. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2003/003.pdf>. Acesso em: 6 jun. 2015.

GOMES, A.; HENRIQUES, J.; MENDES, A. J. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. **Educação, Formação & Tecnologias**, v. 1, n. 1, p. 93–103, Maio, 2008. Disponível em: <http://www.eft.educom.pt/index.php/eft/article/view/23>. Acesso em: 14 ago. 2014.

GOMES, A.; MENDES, A. J. **Learning to program-difficulties and solutions**. In: International Conference on Engineering Education - ICEE, 2007. Coimbra, Portugal: [s.n.]. Disponível em: <http://ineer.org/Events/ICEE2007/papers/411.pdf>. Acesso em: 7 jun. 2015.

GOMES, C. C. C. et al. Uma proposta para auxiliar alunos e professores no ensino de programação: a ambiente AIIP. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 22., 2001, Aracaju, SE, **Anais...**, 2011. Disponível em: <https://dimap.ufrn.br/csbc2011/anais/eventos/contents/WEI/Wei_Secao_4_Artigo_2_Gomes.pdf>. Acesso em: 8 jun. 2015.

GOMES, R. C.; SCHIMIGUEL, J. Aplicação de teorias de aprendizagem construtivista e modelo instrucional em ambiente virtual para o ensino de algoritmos para os alunos do curso de ciências da computação. In: INTERNATIONAL CONFERENCE ON ENGINEERING AND TECHNOLOGY EDUCATION, 11., 2010, Ilhéus. **Anais...** Ilhéus, 2010. Disponível em: <http://proceedings.copec.org.br/index.php/intertech/article/view/1502>. Acesso em: 7 jun. 2015.

GRIES, D. What should we teach in an introductory programming course?. **ACM SIGCSE Bulletin**, v. 6, n. 1, p. 81–89, 1974.

HABERMAN, B.; MULLER, O. Teaching abstraction to novices: Pattern-based and ADT-based problem-solving processes. In: **38th ASEE/IEEE Frontiersin Education Conference**, IEEE, 2008. Disponível em: <http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4720415>. Acesso em: 8 jun. 2015.

HAGAN, D; MARKHAM, S. Does it help to have some programming experience before beginning a computing degree program? **ACM SIGCSE Bulletin**, v. 32, n. 3, p.25-28, 2000. Disponível em: <http://dl.acm.org/citation.cfm?id=343063>. Acesso em: 23 set. 2015.

HENDERSON, P. B. Modern introductory computer science. **ACM SIGCSE Bulletin**, v. 19, n. 1, p. 183–190, 1987. Disponível em: <http://dl.acm.org/citation.cfm?id=31756>. Acesso em: 6 jun. 2015.

HENRIKSEN, P; KÖLLING, M. Greenfoot: combining object visualisation with interaction. In: **Companion to the 19th annual ACM SIGPLAN conference on Object-oriented programming systems, languages, and applications (OOPSLA)**. Vancouver, BC, CANADA: ACM, pp. 73-82. 2004. Disponível em: <http://dl.acm.org/citation.cfm?id=1028701>. Acesso em: 8 jun. 2015.

HINTERHOLZ JUNIOR, O. Tepequém : uma nova ferramenta para o ensino de algoritmos nos cursos superiores em Computação. In: WORKSHOP SOBRE EDUCAÇÃO EM COMPUTAÇÃO (WEI), 7., 2009, , Bento Gonçalves, RS. **Anais...** Disponível em: <http://csbc2009.inf.ufrgs.br/anais/pdf/wei/st02_04.pdf>. Acesso em: 6 jun. 2015.

IEPSEN, Edécio Fernando; BERCHT, Magda; REATEGUI, Eliseo. Persona-Algo: personalização dos exercícios de algoritmos auxiliados por um agente afetivo. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 21., 2010. João Pessoa - PB. **Anais...**, 2010. Disponível em: <http://www.cbie.org.br/index.php/anaissbie>. Acesso em: 6 jun. 2015.

JAIN, J. et al. Experimental evaluation of animated-verifying object viewers for Java. In: **Proceedings of the ACM Symposium on Software Visualization (SoftVis)**, September 4-5, Brighton, UK, 2006. Disponível em: <http://portal.acm.org/citation.cfm?id=1148493.1148497>. Acesso em: 8 jun. 2015.

JENKINS, T. On the difficulty of learning to program. In: **Proceedings of the 3rd Annual LTSN-ICS Conference**. Loughborough University, United Kingdom, August 2002. Disponível em: <http://78.158.56.101/archive/ics/events/conf2002/tjenkins.pdf>. Acesso em: 24 abr. 2015.

JENKINS, T. The motivation of students of programming. In: **ACM SIGCSE Bulletin, Proceedings of the 6th Annual Conference on Innovation and Technology In Computer Science Education**, ITiCSE 2001, v.33, n.3, 2001. Disponível em: <http://dl.acm.org/citation.cfm?id=377472>. Acesso em: 7 jun. 2015.

KNUTH, Donald E. Computer programming as an art. **Communications of the ACM**, v. 17, n. 12, p. 667–673, 1974.

KOLIVER, C.; DORNELES, R. V.; CASA, M. E. Das (muitas) dúvidas e (poucas) certezas do ensino de algoritmos. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (CSBC), 24., 2004, Salvador, BA. **Anais...**, 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2004/008.pdf>. Acesso em: 6 jun. 2015.

KÖLLING, M; HENRIKSEN, P. Game programming in introductory courses with direct state manipulation. **ACM SIGCSE Bulletin**, v. 37, n.3, p.65-69, 2005. Disponível em: <http://dl.acm.org/citation.cfm?id=1067465>. Acesso em: 8 jun. 2015.

KOZAK, D. V.; EBERSPÄCHER, H. F. Uma abordagem para o ensino de programação nas engenharias. In: CONGRESSO BRASILEIRO DE ENSINO DE ENGENHARIA, 29., 2001, Porto Alegre. **Anais ...** Porto Alegre, RS: Pontifícia Universidade Católica do Rio Grande do Sul, 2001. Disponível em: <http://www.abenge.org.br/CobengeAnteriores/2000/artigos/397.PDF>. Acesso em: 7 jun. 2015.

LAUBER, J.R. Codecademy. **CHOICE: Current Reviews for Academic Libraries**, v. 50, n. 3, p. 519-520, 1 nov. 2012. Disponível em: <http://link.galegroup.com/apps/doc/A307526304/AONE?sid=googlescholar&linkaccess=fulltext>. Acesso em: 9 jun. 2015.

LEWANDOWSKI, G.; JOHNSON, E.; GOLDWEBER, M. Fostering a creative interest in computer science. **ACM SIGCSE Bulletin**, v. 37, n.1, p.535-539, 2005. Disponível em: <http://dl.acm.org/citation.cfm?id=1047512>. Acesso em: 8 jun. 2015.

LIMA, M. R.; LEAL, M. C. Motivação discente no ensino-aprendizagem de programação de computadores. **Educação & Tecnologia**, v.17, n.1, 2013. Disponível em: <http://www.marcinholima.com.br/artigos/motivacao_discente.pdf>. Acesso em: 14 ago. 2014.

LINN, M. C.; CLANCY, M. J. The case for case studies of programming problems. **Communications of the ACM**, v.35, n.3, p.121-132, 1992. Disponível em: <http://dl.acm.org/citation.cfm?id=131301>. Acesso em: 7 jun. 2015.

MANSO, A.; OLIVEIRA, L.; MARQUES, C. G. **Portugol IDE - uma ferramenta para o ensino de programação**. 2009. Disponível em: <http://orion.ipt.pt/~manso/papers/2009/Portugol_IDE_PAEE2009.pdf>. Acesso em: 8 jun. 2015.

LAKATOS, E. M.; MARCONI, M. A. **Fundamentos de metodologia científica**: técnicas de pesquisa. 7 ed. São Paulo: Atlas, 2010.

MARTINS, S. W.; MENDES, A. J.; FIGUEIREDO, A. D. Comunidades de Investigação em Programação: uma estratégia de apoio ao aprendizado inicial de programação. **IEEE-RITA**, v. 5, p. 39-46, 2010. Disponível em: <http://rita.det.uvigo.es/201002/uploads/IEEE-RITA.2010.V5.N1.A7.pdf>.

MASINI, EFS; MOREIRA, MA. **Aprendizagem significativa**: a teoria de David Ausubel. São Paulo: Centauro, 2001.

MCDOWELL, C. et al. The effects of pair-programming on performance in an introductory programming course. **ACM SIGCSE Bulletin**, v.34, n.1, 2002. Disponível em: <http://dl.acm.org/citation.cfm?id=563353>. Acesso em: 7 jun. 2015.

MENDES, A. J. N.; GOMES, A. J. Suporte à aprendizagem da programação com o ambiente SICAS. In: CONGRESSO IBERO-AMERICANO DE INFORMÁTICA EDUCATIVA, 5., 2000, Vinã del Mar, Chile, **Anais...**, 2000. Disponível em: <http://lsm.dei.uc.pt/ribie/docfiles/txt20037292359Suporte %C3%A0 aprendizagem.pdf>. Acesso em: 8 jun. 2015.

MOREIRA, M.A. **Teorias de aprendizagem**. São Paulo: EPU, 1999.

MOREIRA, M.A. **Uma abordagem cognitivista ao ensino da Física.** Porto Alegre: Editora de Universidade, 1983.

Moreira, M.A., SOUSA, C.M.S.G. **Organizadores prévios como recurso didático.** Porto Alegre, RS: Instituto de Física, 1996.

MOTA, M. P. et al. Ambiente integrado à Plataforma Moodle para apoio ao desenvolvimento das habilidades iniciais de programação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 20., 2009, Florianópolis, Sc. **Anais...** , 2009. Disponível em: <http://www.proativa.virtual.ufc.br/sbie2009/conteudo/artigos/completos/61591_1.pdf >. Acesso em: 6 jun. 2015.

NOBRE, I. A. M.; MENEZES, C. S. Suporte à cooperação em um ambiente de aprendizagem para programação (SAmbA). In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 13., 2002, SÃO LEOPOLDO, RS. **Anais...**, 2002. Disponível em: <http://br-ie.org/pub/index.php/sbie/article/view/195>. Acesso em: 6 jun. 2015.

NOVAK, J. D. **Aprender criar e utilizar o conhecimento:** mapas conceituais como ferramenta de facilitação em escolas e empresas. Lisboa: Plátano. 2000.

NUUTILA, E. et al. Learning programming with the PBL method - experiences on PBL cases and tutoring. In: BENNEDSEN, Jens, CASPERSEN, Michael E., KÖLLING, Michael (Eds.). **Reflections on the Teaching of Programming: methods and Implementations**, p. 47–67, 2008. Disponível em: <http://link.springer.com/chapter/10.1007/978-3-540-77934-6_5>. Acesso em: 6 jun. 2015.

PAPERT, Simoun. Mindstorms: Children, computers and powerful ideas. **New Ideas in Psychology**, v. 1, n. 1, p. 87, 1983. Disponível em: <http://dl.acm.org/citation.cfm?id=1095592>. Acesso em: 8 jun. 2015.

PARREIRA, M. O.; FORSTER, C. H. Q. Requisitos para a construção de um ambiente propício ao ensino de programação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 19., 2008, Fortaleza, CE, **Anais...**, 2002. Disponível em: <http://www.proativa.virtual.ufc.br/sbie/CD_ROM_COMPLETO/workshops/workshop2/Requisitos para a constru%E7%E3o de um ambiente prop%EDcio ao.pdf>. Acesso em: 6 jun. 2015.

PEARS, A. et al. A survey of literature on the teaching of introductory programming. **ACM SIGCSE Bulletin**, v. 39, n. 4, p. 204-223, Dez. 2007. Disponível em: <http://dl.acm.org/citation.cfm?id=1345441>. Acesso em: 11 nov. 2014.

PELIZZARI, A; et al. Teoria da aprendizagem significativa segundo Ausubel. **Revista PEC**, Curitiba, v.2, n.1, p.37-42, jul. 2001/2002. Disponível em: <http://files.gpecea-usp.webnode.com.br/200000393-74efd75e9b/MEQII-2013- TEXTOS COMPLEMENTARES- AULA 5.pdf>. Acesso em: 29 jun. 2015.

PEREIRA JÚNIOR, J. C. R. et al. AVEP – um Ambiente de Apoio ao Ensino de Algoritmos e Programação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO (CSBC, 26., 2006, Campo Grande, MT. **Anais...**, 2006. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2006/0017.pdf>. Acesso em: 6 jun. 2015.

PEREIRA JÚNIOR, J. C. R. et al. Ensino de algoritmos e programação: uma experiência no nível médio. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 25., 2005, São Leopoldo, RS. **Anais...**, 2005. Disponível em: < http://www.lbd.dcc.ufmg.br/colecoes/wei/2005/008.pdf >. Acesso em: 6 jun. 2015.

PEREIRA JÚNIOR, J. C. R.; RAPKIEWICZ, C. E. O Processo de ensino-aprendizagem de fundamentos de programação: uma visão crítica da pesquisa no Brasil. In: In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO RJ/ES, 1., 2004, Vitória - ES. **Anais...**, 2004. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/weirjes/2004/003.pdf>. Acesso em: 6 jun. 2015.

PIMENTEL, E. P.; FRANÇA, V. F.; OMAR, N. A caminho de um ambiente de avaliação e acompanhamento contínuo da aprendizagem em Programação de Computadores. In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO E

INFORMÁTICA DO ESTADO DE MINAS GERAIS, 2., 2003, Poços de Caldas, MG. **Anais...** , 2003. Disponível em: <http://200.17.137.110:8080/licomp/Members/jeanemelo/plonelocalfolderng.2006-04-10.7475913377/PEP/Aulas19/WEIMIG2003EdsonPimentelArtigo1-aula19.pdf>. Acesso em: 6 jun. 2015.

PIVA JR, D.; FREITAS, R. L. Estratégias para melhorar os processos de abstração na disciplina de Algoritmos. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 21., 2010. João Pessoa, PB. **Anais...**, 2010. Disponível em: <http://www.br-ie.org/pub/index.php/sbie/article/view/1464>. Acesso em: 6 jun. 2015.

RAABE, A. L. A.; SILVA, J. M. C. Um Ambiente para atendimento as dificuldades de aprendizagem de algoritmos. In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO (WEI). 13., 2005, São Leopoldo, RS. **Anais...**, 2005. Disponível em: <http://www.lbd.dcc.ufmg.br/colecoes/wei/2005/003.pdf>. Acesso em: 6 jun. 2015.

RAPKIEWICZ, C. E. et al. Estratégias pedagógicas no ensino de algoritmos e programação associadas ao uso de jogos educacionais. **Novas Tecnologias na Educação**, v. 4, n. 2, Dez., 2006. Disponível em: <http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:CINTED-UFRGS+Novas+Tecnologias+na+Educa??o#0>. Acesso em: 13 ago. 2014.

REBELO, B. J. **SICAS-COL**: um sistema colaborativo para aprendizagem inicial da programação. 2006. Dissertação (Mestrado em Engenharia Informática) - Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia de Coimbra. Coimbra, 2006. Disponível em: <https://eg.sib.uc.pt/handle/10316/13897>. Acesso em: 8 jun. 2015.

REBOUÇAS, Dantas S et al. Aprendendo a ensinar programação combinando jogos e Python. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 21., 2010. João Pessoa - PB. **Anais...**, 2010. Disponível em: <http://www.ccae.ufpb.br/sbie2010/anais/Inicio.html>. Acesso em: 6 jun. 2015.

RESNICK, M. et al. Scratch: programming for all. **Communications of the ACM**, v.52, n.11, p.60-67, 2009. Disponível em: <http://dl.acm.org/citation.cfm?id=1592779>. Acesso em: 8 jun. 2015.

RICHARDSON, R. J. **Pesquisa social**: métodos e técnicas. 3.ed. São Paulo: Atlas, 1999.

ROBERTS, E. An overview of MiniJava. **ACM SIGCSE Bulletin**, v.33, n.1, 2001. Disponível em: <http://dl.acm.org/citation.cfm?id=364525>. Acesso em: 8 jun. 2015.

ROBERTS, E. Conserving the seed corn: reflections on the academic hiring crisis. **ACM SIGCSE Bulletin**, v.31, n.4, Dez.1999. Disponível em: <http://www-cs.stanford.edu/people/eroberts/papers/ConservingSeedCorn.pdf>. Acesso em: 8 jun. 2015.

ROBINS, A.; ROUNTREE, J.; ROUNTREE, N. Learning and teaching programming : a review and discussion. **Computer Science Education**, v.13, n.2, p. 137-172, 2003. Disponível em: <http://www.tandfonline.com/doi/abs/10.1076/csed.13.2.137.14200>. Acesso em: 18 ago. 2014.

ROCHA, P. S. et al. Ensino e aprendizagem de programação: análise da aplicação de proposta metodológica baseada no Sistema Personalizado de Ensino. **Renote**, v. 8, n. 3, p. 1-11, 2010. Disponível em: <http://seer.ufrgs.br/renote/article/view/18061>. Acesso em: 14 ago. 2014.

RODRIGUES JÚNIOR, M. C. Experiências positivas para o ensino de algoritmos. In: In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO, 4. Feira de Santana, BA. **Anais...** Sergipe, Bahia: SBC, 2004. Disponível em: <http://www.uefs.br/erbase2004/documentos/weibase/Weibase2004Artigo001.pdf>. Acesso em: 14 ago. 2014.

RODRIGUES, M. C. Como ensinar programação?. **Informática – Boletim Informativo**, Ano 1, n.1, ULBRA, Canoas, RS, 2002. Disponível em: <https://scholar.google.com.br/scholar?q=Rodrigues%2C+M.+C.+Como+Ensinar+Programa%C3%A7%C3%A3o%3F.&btnG=&hl=pt-BR&as_sdt=0%2C5#0>. Acesso em: 6 jun. 2015.

SANTOS, R. P.; COSTA, H. A. X. Análise de metodologias e ambientes de ensino para algoritmos, estruturas de dados e programação aos iniciantes em computação e informática. **INFOCOMP - Journal of Computer Sciense**, v. 5, n. 1, p. 41–50, 2006. Disponível em: <http://www.dcc.ufla.br/infocomp/artigos/v5.1/art06.pdf\nhttp://www.academia.edu/download/30554771/art06.pdf>. Acesso em: 19 ago. 2014.

SANTOS, R. P.; COSTA, H. A. X. TBC-AED e TBC-AED/Web: Um desafio no ensino de algoritmos, estruturas de dados e programação. In: WORKSHOP DE EDUCAÇÃO EM COMPUTAÇÃO E INFORMÁTICA DO ESTADO DE MINAS GERAIS (WEIMIG), 4., 2005, Varginha, MG, **Anais...**, 2005. Disponível em: <http://www.cos.ufrj.br/~rps/pub/completos/2005/WEIMIG.pdf>. Acesso em: 19 ago. 2014.

SCAICO, Pasqueline Dantas et al. Ensino de programação no ensino médio: uma abordagem orientada ao design com a linguagem Scratch**. Revista Brasileira de Informática na Educação**, v. 21, n. 2, p. 92, 2013. Disponível em: < http://www.br-ie.org/pub/index.php/rbie/article/view/2364/2132>. Acesso em: 6 jun. 2015.

SELLTIZ, C. et al. **Métodos de pesquisa nas relações sociais**. São Paulo: EPU, 1974.

SHEARD, J. et al. Analysis of research into the teaching and learning of programming. In: **Proceedings of the fifth international workshop on Computing education research (ICER) '09**, Ago. 2009. Disponível em: <http://portal.acm.org/citation.cfm?doid=1584322.1584334>. Acesso em: 23 abr. 2015.

SHELL, Duane F. et al. Associations of students creativity, motivation, and self-regulation with learning and achievement in college computer science courses. In: **FRONTIERS IN EDUCATION CONFERENCE (FIE 2013) IEEE**, Oklahoma City, OK, p. 1637-1643, 2013.

SLEEMAN, D. The challenges of teaching computer programming. **Communications of the ACM**, v. 29, n. 9, p. 840–841, 1986. Disponível em: <http://dl.acm.org/citation.cfm?id=214913>. Acesso em: 6 jun. 2015.

SOLOWAY, E. et al. What do novices know about programming ? In: BRADE, A., SHNEIDERMAN, B (Eds.). **Directions in human-computer interactions**, Norwood, NJ: Ablex, p. 27–54, 1983. Disponível em: <https://scholar.google.com.br/scholar?q=What+do+novices+know+about+programming%3F&btnG=&hl=pt-BR&as_sdt=0%2C5#0>. Acesso em: 7 jun. 2015.

SOUZA, C. M. VisuAlg - Ferramenta de apoio ao ensino de programação. **Revista TECCEN**, v. 2, n. 2, p. 1-9, set. 2009. Disponível em: <http://www.uss.br/pages/revistas/revistateccen/V2N22009/ArtigoVisuAlgSOUZA.pdf>. Acesso em: 8 jun. 2015.

TOBAR, C. M. et al. Uma arquitetura de ambiente colaborativo para o aprendizado de programação. In: SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO (SBIE), 12, 2001, Vitória, ES. **Anais...**Vitória: UFES, 2001. Disponível em: <https://www.icmc.usp.br/~joaoluis/sbie-01.pdf>. Acesso em: 13 ago. 2014.

WILLIAMS, L. et al. Strengthening the case for pair programming. **IEEE Software**, v.17, n.4, p.19-25, 2000. Disponível em: <http://www.computer.org/csdl/mags/so/2000/04/s4019.pdf>. Acesso em: 7 jun. 2015.

WILSON, B. C.; SHROCK, S. Contributing to success in an introductory computer science course: a study of twelve factors. **ACM SIGCSE Bulletin**, v.33, n.1, p.184-188, 2001. Disponível em: <http://dl.acm.org/citation.cfm?id=364581>. Acesso em: 7 jun. 2015.

WIRTH, N. **Algoritmos e estruturas de dados**. Rio de Janeiro: LTC, 1989.

WOLBER, David. App Inventor and Real-World Motivation. In: **Proceedings of the 42nd ACM technical symposium on Computer science education**, March 09-12, 2011, Dallas, TX, USA, p. 601–606, 2011. Disponível em: <http://dl.acm.org/ft_gateway.cfm?id=1953329&ftid=938728&dwn=1&CFID=528680861&CFTOKEN=18575345> Acesso em: 6 jun. 2015.

YAMAMOTO, Flavio S. et al. Interdisciplinaridade no ensino de ciência da computação. In: CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 25., 2005, São Leopoldo, RS, **Anais...**, 2005. Disponível em: <http://www.unisinos.br/_diversos/congresso/sbc2005/_dados/anais/pdf/arq0040.pdf>. Acesso em: 6 jun. 2015.

ZAMBON, Melissa Picchi; SOUZA, Deisy Das Graças de; ROSE, Tânia Maria Santana de. Autoeficácia e experiência de professores no uso de tecnologias de informática. **Revista Brasileira de Informática na Educação**, v. 20, n. 2, p. 44–53, 2012. Disponível em: <http://www.br-ie.org/pub/index.php/rbie/article/view/1360>. Acesso em: 15 ago. 2014.

ANEXO A - Diário de Classe da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014
Professor Titular – Registros de Conteúdos

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	Professor Titular
Bimestre	1o Bimestre

Aula	Data	Tema	Observações
1	13/02/2014	Apresentação da disciplina, história do computador	Sem obs.
2	20/02/2014	Introdução: Algoritmos, características Exemplos.	Sem obs.
3	27/02/2014	Representações de algoritmos: Linguagem natural e Fluxograma.	Sem obs.
4	06/03/2014	Representações de algoritmos: Pseudocódigo (Portugol) exercícios para fixação.	Sem obs.
5	13/03/2014	Constantes; Variáveis; Tipos de dados; Entrada e saída de dados.	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

6	20/03/2014	Atribuição e operadores Aritméticos, exercícios para fixação.	Sem obs.
7	27/03/2014	Operadores: Lógico, Relacional, exercícios para fixação.	Lista de exercícios 1.
8	03/04/2014	Revisão	Sem obs.
9	10/04/2014	Avaliação Bimestral 1	Entrega da lista de exercícios 1.
10	17/04/2014	Correção da lista de exercícios e avaliação.	Sem obs.

Telêmaco Borba, 17 de abril de 2014

Professor Titular
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	Professor Titular
Bimestre	2o Bimestre

Aula	Data	Tema	Observações
11	24/04/2014	Porque aprender a programar computadores; Conceitos básicos de LP: 1.linguagem de máquina; 2.linguagem simbólica; 3.linguagem de alto nível.	Sem obs.
12	08/05/2014	Apresentação do ambiente de programação VisuAlg. Implementação de algoritmos no ambiente.	Atividades em laboratório de informática.
13	15/05/2014	Estruturas de controle simples (Se então)	Lista de exercícios 2.
14	22/05/2014	Estrutura de controle simples (Se então) exercícios para fixação no ambiente VisuAlg	Atividades em laboratório de informática.
15	29/05/2014	Estruturas de controle composta (Senão), exercícios para fixação	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

16	05/06/2014	Estrutura de controle composta (Senão) exercícios para fixação no ambiente VisuAlg	Atividades em laboratório de informática.
17	12/06/2014	Estruturas de controle aninhada, exercícios para fixação	Sem obs.
18	26/06/2014	Estrutura de controle aninhadas exercícios para fixação no ambiente VisuAlg	Atividades em laboratório de informática.
19	03/07/2014	Revisão	Sem obs.
20	10/07/2014	Avaliação Bimestral 2	Entrega da lista de exercícios 2.

Telêmaco Borba, 10 de julho de 2014

Professor Titular
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	Professor Titular
Bimestre	3o Bimestre

Aula	Data	Tema	Observações
21	31/07/2014	Estruturas de repetição Enquanto, Repita até	Sem obs.
22	07/08/2014	Estruturas de repetição (Enquanto, Repita até) exercícios para fixação no ambiente VisuAlg	Atividades em laboratório de informática.
23	14/08/2014	Estruturas de repetição Para faça	Sem obs.
24	21/08/2014	Estrutura de repetição (Para faça) exercícios para fixação no ambiente VisuAlg	Atividades em laboratório de informática.
25	28/08/2014	Introdução ao PHP, Breve histórico; Instalação e configuração do PHP; Configurando o PHP no Apache, Cliente Servidor.	Atividades em laboratório de informática.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

26	04/09/2014	Características da linguagem: sintaxe, variáveis, entrada e saída de dados.	Sem obs.
27	11/09/2014	Implementação de estruturas de controle em PHP (IF e ELSE)	Atividades em laboratório de informática.
28	18/09/2014	Implementação de estruturas de Repetição em PHP (While, Do while, For)	Atividades em laboratório de informática.
29	25/09/2014	Revisão	Sem obs.
30	02/10/2014	Avaliação Bimestral 3	Sem obs.

Telêmaco Borba, 02 de outubro de 2014

Professor Titular
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	Professor Titular
Bimestre	4o Bimestre

Aula	Data	Tema	Observações
31	09/11/2014	Introdução a Vetores.	Sem obs.
32	16/11/2014	Introdução a Matrizes.	Sem obs.
33	23/11/2014	Implementação de Vetores e Matrizes em PHP, exercícios para fixação.	Atividades em laboratório de informática.
34	30/11/2014	Introdução a Sub-rotinas (Funções e procedimentos).	Projeto: Programar um site utilizando PHP, HTML, CSS - o site deve conter pelo menos 1 cadastro, 1 relatório, e 1 formulário de contato. O site deverá ser apresentado na última aula do bimestre.
35	06/11/2014	Implementação de sub-rotinas em PHP, exercícios para fixação.	Atividades em laboratório de informática.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

36	13/11/2014	Funções passagem de parâmetros por valor e por referencia em PHP.	Atividades em laboratório de informática.
37	20/11/2014	PHP conexão com banco de dados MySql; Operações básicas de manipulação de dados PHP MySql	Atividades em laboratório de informática.
38	27/11/2014	Avaliação Bimestral 4	Avaliação escrita em sala de aula.
39	04/12/2014		Avaliação prática em laboratório.
40	11/12/2014	Apresentação do projeto.	Alguns alunos faltaram e não apresentaram o projeto.

Telêmaco Borba, 11 de Dezembro de 2014

Professor Titular
Siape:

ANEXO B - Diário de Classe da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014
Professor Pesquisador – Registros de Conteúdos

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	PROFESSOR PESQUISADOR
Bimestre	1o Bimestre

Aula	Data	Tema	Observações
1	13/02/2014	Apresentação da disciplina, a metodologia de trabalho, o download da ferramenta a ser utilizada (Decimal Basic) e apresentação do ambiente.	Sem obs.
2	20/02/2014	Comando PRINT com caracteres e números; expressões matemáticas simples; uso de variáveis inteiras e o laço de repetição FOR.	Sem obs.
3	27/02/2014	Revisão dos comandos apresentados na aula anterior; Apresentação das variações do comando PRINT (pular linha, imprimir na sequência e tabulação), a variante STEP para o laço FOR; Desafio semanal I	Sem obs.
4	06/03/2014	Resolução do Desafio Semana I; Comando INPUT, para a captação pelo teclado; Comandos IF e ELSE.	Sem obs.
5	13/03/2014	Revisão dos comandos apresentados em aulas anteriores; Contador e acumulador; Desafio semanal II.	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

6	20/03/2014	Resolução do Desafio Semana II; Comandos RANDOMIZE, RND, IF e ELSE.	Sem obs.
7	27/03/2014	Revisão dos comandos apresentados em aulas anteriores; Comandos DO LOOP, DO WHILE LOOP, EXIT DO, EXIT FOR; Desafio semanal III.	Sem obs.
8	03/04/2014	Resolução do Desafio Semana III, Janela de plotagem disponibilizada pelo ambiente; Comando de plotagem PLOT LINES.	Sem obs.
9	10/04/2014	Sub-rotinas Comandos CALL e SUB; Desenvolvimento de uma aplicação simples jogo da forca, empregando os conhecimentos adquiridos.	Sem obs.
10	17/04/2014	Avaliação Bimestral (prática); conteúdos: estrutura de sequência, decisão e repetição	Sem obs.

Telêmaco Borba, 17 de Abril de 2014

PROFESSOR PESQUISADOR
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	PROFESSOR PESQUISADOR
Bimestre	2o Bimestre

Aula	Data	Tema	Observações
11	24/04/2014	Ambiente PascalZim; download da ferramenta e exploração do ambiente.	Sem obs.
12	08/05/2014	Comandos Write e Writeln com caracteres e números; expressões matemáticas simples; uso de variáveis inteiras e o laço de repetição FOR.	Sem obs.
13	22/05/2014	Revisão dos comandos apresentados na aula anterior; Comando Read; Laço de repetição FOR com comando Read; Desafio semanal IV.	Sem obs.
14	29/05/2014	Resolução do Desafio Semanal IV; Vetores; Variáveis caracteres (Strings); Armazenamento e manipulação de Strings.	Sem obs.
15	05/06/2014	Revisão dos comandos apresentados em Pascal; Comandos IF THEN e ELSE; Contador e acumulador; Desafio Semanal V.	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

16	12/06/2014	Resolução do Desafio Semanal V; Comandos WHILE DO; REPEAT.	Sem obs.
17	26/06/2014	Sub-rotinas Procedures e Functions; Desafio Semanal VI.	Sem obs.
18	03/07/2014	Desenvolvimento de Jogo de operações matemáticas utilizando os comandos apresentados até o momento.	Sem obs.
19	03/07/2014	Avaliação Bimestral (prática); conteúdos: estrutura de sequência, decisão e repetição	Sem obs.
20	10/07/2014	Análise do desempenho dos alunos; revisão dos conteúdos; comparação dos comandos em Basic e Pascal	Sem obs.

Telêmaco Borba, 10 de Julho de 2014

PROFESSOR PESQUISADOR
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	PROFESSOR PESQUISADOR
Bimestre	3o Bimestre

Aula	Data	Tema	Observações
21	31/07/2014	Ambiente DevC++; download da ferramenta e exploração do ambiente.	Sem obs.
22	07/08/2014	Estrutura de programação C, Bibliotecas e funções.	Sem obs.
23	14/08/2014	Transição para linguagem C, aspectos de sintaxe, comparação com Basic e Pascal	Sem obs.
24	21/08/2014	Comandos printf e scanf com caracteres e números; expressões matemáticas simples; uso de variáveis inteiras, reais e o laço de repetição FOR.	Sem obs.
25	28/08/2014	Revisão dos comandos apresentados na aula anterior; Comandos IF e ELSE; Contador e acumulador.	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

26	04/09/2014	Vetores; Variáveis caracteres (Strings); Armazenamento e manipulação de Strings; Percorrer vetores utilizando laço de repetição FOR. Desafio Semanal VII (resolver problemas de 1 a 10 do site urionlejudge)	Sem obs.
27	11/09/2014	Comandos WHILE; DO WHILE.	Sem obs.
28	18/09/2014	Sub-rotinas: Funções: parâmetros e tipos de retorno	Sem obs.
29	25/09/2014	Desenvolvimento de sistema simples para cálculo de várias formas geométricas utilizando sub-rotinas e os comandos apresentados até o momento. Desafio Semanal VII (resolver problemas de 11 a 20 do site urionlejudge)	Sem obs.
30	02/10/2014	Avaliação Bimestral (prática); conteúdos: estrutura de sequência, decisão, repetição, vetores, funções.	Sem obs.

Telêmaco Borba, 02 de Outubro de 2014

PROFESSOR PESQUISADOR
Siape:

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LINGUAGEM DE PROGRAMAÇÃO I - TÉCNICO EM INFORMÁTICA PARA INTERNET INTEGRADO AO ENSINO MÉDIO

Professor	PROFESSOR PESQUISADOR
Bimestre	4o Bimestre

Aula	Data	Tema	Observações
31	09/10/2014	Análise do desempenho dos alunos; revisão dos conteúdos; comparação dos comandos em Basic X Pascal X C.	Sem obs.
32	16/10/2014	Revisão tipos de variáveis, vetores; Variáveis Globais e Local.	Sem obs.
33	23/10/2014	Matrizes, armazenamento e manipulação; Percorrer matriz utilizando laço FOR	Sem obs.
34	30/10/2014	Exercícios, desenvolvimento de programas simples; Soluções e discussão.	Sem obs.
35	06/11/2014	Revisão Funções; passagem por valores e referencia	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

36	13/11/2014	Introdução recursividade	Sem obs.
37	20/11/2014	Introdução a ponteiros	Sem obs.
38	27/11/2014	Introdução a Fila e Pilha	Sem obs.
39	04/12/2014	Revisão para avaliação	Sem obs.
40	11/12/2014	Avaliação Bimestral (prática); conteúdos: estrutura de sequência, decisão, repetição, matriz, funções	Sem obs.

Telêmaco Borba, 11 de Dezembro de 2014

PROFESSOR PESQUISADOR
Siape:

ANEXO C - Avaliações e Provas da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Titular

INSTITUTO FEDERAL
PARANÁ
Campus Telêmaco Borba

MINISTÉRIO DA EDUCAÇÃO

Técnico em Informática para Internet Integrado ao Ensino Médio
Disciplina: Linguagem de Programação I – 1º Bimestre
Professor: Titular
Data: 10/04/2014

Aluno(a): _____

Avaliação Escrita

1. Defina Algoritmos e explique por que os Algoritmos são importantes?

2. Quais as características de um Algoritmo? Explique cada uma delas.

3. Quais as principais formas de representação de um Algoritmo? Explique cada uma delas.

4. Escreva um algoritmo em linguagem natural contendo os procedimentos necessários para vir até escola?

5. Escreva um Pseudocódigo para o cálculo da área do triangulo? *Fórmula A=(b*h)/2*

6. Relacione os tipos de dados abaixo:

(a) Inteiro	( ) verdadeiro
(b) Caractere	( ) 7
(c) Real	( ) 'MARIA'
(d) Lógico	( ) 11.50

7. Considere as variáveis A, B, C, D são tipo inteiro, A=10; B=10; C=30; D=40, qual o resultado da seguinte expressão: D=A*B+C

8. Considere as variáveis soma, num1 e num2 são do tipo inteiro, se soma:=0; num1:=10; num2:=20; soma:=num1; soma:= num2; ao término destas operações qual o valor armazenado na variável soma:
   a) 0
   b) 10
   c) 20
   d) 30
   e) Nda

9. Coloque Verdadeiro (V) ou Falso (F), na avaliação das expressões abaixo:

( ) 6<=7	( ) 3+5 >= 2*3	( ) 'joão'<> 'maria'
( ) 10-4 > 7	( ) 10.0<= 10	( ) 8*3 = 3*8

10. Assinale abaixo o item que **não** corresponde a um operador Relacional
    a) >
    b) <
    c) >=
    d) =
    e) Nda.

INSTITUTO FEDERAL
PARANÁ
Campus Telêmaco Borba

MINISTÉRIO DA EDUCAÇÃO

Técnico em Informática para Internet Integrado ao Ensino Médio
Disciplina: Linguagem de Programação I – 2º Bimestre
Professor: Titular
Data: 10/07/2014

Aluno(a): _____

Avaliação Escrita

1. Por que é importante aprender a programar computadores?
2. Explique as diferenças entre linguagem de máquina e linguagem de alto nível?
3. Escreva um Pseudocódigo que leia o valor do raio e calcule e imprima a área do círculo. *Fórmula A=(3.14159*(raio*raio))*
4. Escreva um Pseudocódigo que leia quatro notas de um aluno, calcule a média e imprima se ele foi aprovado ou reprovado, considere a média para aprovação 6.0
5. Escreva um Pseudocódigo para calcular o aumento salarial de um empregado. Leia o salarioAtual do empregado e imprima o salarioNovo. Por padrão, o aumento será de 15%. Entretanto, deve ser aplicada uma regra diferente para cada faixa salarial. Regras:
    a) se 1.500,00 <= salarioAtual < 1.750,00: aumento igual a 12%
    b) se 1.750,00 <= salarioAtual < 2.000,00: aumento igual a 10%
    c) se 2.000,00 <= salarioAtual < 3.000,00: aumento igual a 7%
    d) se salarioAtual acima de 3.000,00: aumento igual a 5%. Relacione os tipos de dados abaixo.
6. Escreva um Pseudocódigo que leia 4 valores inteiros A, B, C e D. A seguir, se B for maior do que C e se D for maior do que A e a soma de C com D for maior que a soma de A e B e se C e D, ambos, forem positivos e se a variável A for par escrever a mensagem "Valores aceitos", senão escrever "Valores não aceitos".
7. Assinale abaixo o item que **não** corresponde a um operador lógico.
    a) E
    b) OU
    c) NÃO
    d) <>
    e) Nda.
8. Assinale abaixo o item que corresponde aos operadores utilizados para avaliar expressões em estruturas de controles:
    a) Operadores lógicos
    b) Operadores relacionais
    c) Operadores aritméticos
    d) Operadores lógicos e/ ou relacionais
    e) Nda.

9. Dado o Pseudocódigo abaixo, qual será a saída do programa (instrução da linha 12)?

```
1. algoritmo “TESTE1”
2. var
3.    a, b, c: real
4. inicio
5.    a := 10
6.    b := 20
7.    se (a + b < 30) entao
8.      c := (a + b) / 2
9.    senao
10.     c := (a + b) * 2
11.   fimse
12.   escreval (c)
13. fimalgoritmo
```

10. Dado o Pseudocódigo abaixo, e caso o usuário informe os seguintes valores: n1=5, n2=6, n3=7, faltas=20. Qual será a saída produzida pelo programa?

```
1. algoritmo "TESTE2"
2. var
3.    n1, n2, n3, media: real
4.    faltas: inteiro
5. inicio
6.    leia (n1)
7.    leia (n2)
8.    leia (n3)
9.    leia (faltas)
10.   media := ((n1+n2+n3)/3)
11.   escreval ("A média do Aluno é: ", media)
12.   se (media<4) ou (faltas>20) entao
13.     escreval ("A situação atual do aluno é REPROVADO")
14.   senao
15.     se (media<7) entao
16.       escreval ("A situação atual do aluno é EXAME")
17.     senao
18.       escreval ("A situação atual do aluno é APROVADO")
19.     fimse
20.   fimse
21. fimalgoritmo
```

INSTITUTO FEDERAL
PARANÁ
Campus Telêmaco Borba

MINISTÉRIO DA EDUCAÇÃO

Técnico em Informática para Internet Integrado ao Ensino Médio
Disciplina: Linguagem de Programação I – 3º Bimestre
Professor: Titular
Data: 02/10/2014

Aluno(a): _____

Avaliação Escrita

1. Qual a diferença em Scripts executados em clientes e Scripts executados em servidor?

2. Qual a função da variável $_POST em PHP?

3. Escreva um Pseudocódigo que pergunte ao usuário quantos números ele deseja somar. Em seguida, leia a quantidade informada de números e apresentar o valor da soma, quantos números são maiores que 7 e quantos números são maiores que 10.

4. A série de Fibonacci inicia com os números 1 e 1, e cada número posterior equivale à soma dos dois números anteriores. Exemplo: caso o número 9 seja informado, o resultado será 1, 1, 2, 3, 5, 8, 13, 21, 34. Escreva em Pseudocódigo e também o código PHP que calcula a série de Fibonacci para um número informado pelo usuário.

5. Sergio tem 1,50 metros e cresce 2 centímetros por ano, enquanto Odete tem 1,10 metros e cresce tem 3 centímetros por ano. Escreva o código PHP que calcule e mostre quantos anos serão necessários para que Odete seja maior que Sergio.

6. Relacione os tipos de dados abaixo:

(a) integer	( ) 7.89	( ) Caractere
(b) float	( ) -12	( ) Real
(c) string	( ) “OPA”	( ) Inteiro

7. Associe os comandos Pseudocódigo e PHP abaixo:

(a) se então / senão	( ) while
(b) para faça	( ) if / else
(c) repita até	( ) do / while
(d) enquanto	( ) for

8. Assinale abaixo o item que **não** corresponde a um comando de estrutura de Repetição.
   a) Para / Faça
   b) While
   c) If / else
   d) For
   e) Nda.

9. Dado o Pseudocódigo abaixo, e caso o usuário informe os seguintes valores: n1=1; n2=10. Qual será a saída produzida pelo programa (instrução da linha 13)?

```
1. algoritmo "TESTE1"
2. var
3.    n1, n2, i, soma: inteiro
4. inicio
5.    leia (n1)
6.    leia (n2)
7.    soma := 0
8.    Para i de n1 ate n2 faca
9.       Se (i mod 2) <> 0 entao
10.         soma := soma +i
11.      fimSe
12.   fimPara
13.   escreval (soma)
14. fimalgoritmo
```

10. Dado o código PHP abaixo, qual será a saída do programa.

```
1.  <?php
2.    $premiado = 3;
3.    if( $premiado < 100 ){
4.      for( $i = 1; $i <= 100; $i++ ){
5.        if( $i == $premiado ){
6.          echo $i . ' é o número premiado ' . PHP_EOL;
7.          break;
8.        } else {
9.          if ( $i % 2 == 0 ){
10.           echo $i . ' é par ' . '<br />' . PHP_EOL;
11.         } else {
12.           echo $i . ' é impar ' . '<br />' . PHP_EOL;
13.         }
14.       }
15.   }
16.  } else {
17.    echo 'O número deve ser menor que 100';
18.  }
19. ?>
```

INSTITUTO FEDERAL
PARANÁ
Campus Telêmaco Borba

MINISTÉRIO DA EDUCAÇÃO

Técnico em Informática para Internet Integrado ao Ensino Médio
Disciplina: Linguagem de Programação I – 4º Bimestre
Professor: Titular
Data: 27/11/2014

Aluno(a): _____

Avaliação Escrita

1. O que é possível ser feito com a função mysqli_query()

2. Escreva um Pseudocódigo ou código PHP, que leia um vetor de 10 valores inteiros. Em seguida, mostre o maior e menor valor armazenado.

3. Dadas as tarefas de sistemas web listadas abaixo, marque com **C** aquelas que considerar *client-side scripts* e com **S** aquelas que considerar *server-side scripts*.

   ( ) Validação de CPF
   ( ) Validação de dados de cartão de crédito
   ( ) Cadastro de pessoas
   ( ) *Scripts* desenvolvidos em javascript
   ( ) *Scripts* desenvolvidos em PHP

4. Com relação a linguagem PHP, assinale com **V** as afirmativas que julgar verdadeiras e com **F** as falsas.

   ( ) PHP é uma siga recursiva que significa PHP Hypertext Preprocessor
   ( ) Assim como a linguagem ASP, PHP é uma linguagem de código proprietário (não é livre).
   ( ) PHP não necessita de código HTML para exibir informações.
   ( ) Para execução dos scripts feitos com PHP, é necessário no mínimo a instalação de um servidor web, como o Apache, e de um servidor PHP
   ( ) PHP permite trabalhar com uma variedade de Sistemas Gerenciadores de Bancos de Dados, entre eles MySQL e PostgreSQL.
   ( ) É possível exibir informações no PHP através das funções echo e print, que funcionam basicamente da mesma maneira
   ( ) Antes de utilizar uma variável PHP, é necessário declará-la no início do código, sem esquecer de declarar o tipo da mesma, como int, float e char.

5. A linguagem PHP trabalha basicamente com duas instruções de seleção: ***if.. else*** e ***switch***. Qual a diferença entre estas instruções?

6. Ao criar um formulário no HTML cujas informações devem ser enviadas para um script PHP, não devem ser esquecidos os atributos ***method*** e ***action***. Quais as funções destes atributos?

7. $_POST é uma variável pré-definida do PHP que tem a função de fazer o que?

8. Para que servem os comandos PHP ***include*** e ***require***? Qual a diferença entre eles?

9. Dado o seguinte código PHP, qual será a saída do programa (instrução da linha 10)?

```
1. <?php
2.      $a = 10;
3.      $b = 20;
4.      if($a + $b < 50) {
5.          $c = ($a + $b) / 2;
6.      }
7.      else {
8.          $c = ($a + $b) * 2;
9.      }
10.     echo $c;
11.?>
```

10. Observe o código HTML abaixo, que cria um formulário.

```
<form name="form1" method="post" action="acao.php">
    Valor Inicial
    <input name="txtValorInicial">
    Valor Final
    <input name="txtValorFinal">
    Quantidade
    <input name="txtQuantidade">
    <input type="submit" value="Enviar">
</form>
```

Dado o código HTML acima, desenvolva o script do arquivo **acao.php**, que deve realizar as seguintes tarefas:

a) Receber os dados dos três campos de texto, atribuindo cada valor para sua respectiva variável.
b) Exibir *N* números aleatórios entre o valor Inicial e o valor final informados (campos **txtValorInicial** e **txtValorFinal**, respectivamente), onde *N* é o valor informado no campo **txtQuantidade**. ***[Nota:** Para gerar o número aleatório utilize a função **rand** e para realizar a repetição utilize o comando **for** ou **while]***

Exemplo: Caso o usuário informe os seguintes valores:

- Valor Inicial: **5**
- Valor Final: **100**
- Quantidade: **10**

Uma possível saída para o arquivo **acao.php** seria:

```
45 13 22 7 99 77 32 56 12 90
```

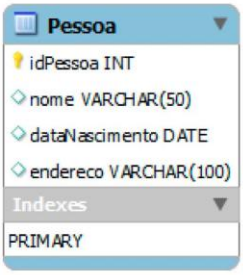
INSTITUTO FEDERAL
PARANÁ
Campus Telêmaco Borba

MINISTÉRIO DA EDUCAÇÃO

Técnico em Informática para Internet Integrado ao Ensino Médio
Disciplina: Linguagem de Programação I – 4º Bimestre
Professor: Professor Titular
Data: 04/12/2014

Avaliação Prática

Utilizando o phpMyAdmin, crie a tabela representada abaixo, respeitando rigorosamente os nomes dos campos e seus respectivos tipos de dados. ***[Nota:** A chave primária da tabela deve ser definida como auto incremento**]***

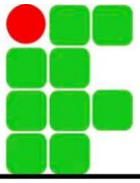


Para a tabela criada no phpMyAdmin, desenvolva um formulário com os elementos adequados para cada campo da tabela. Após criar o formulário, desenvolva um script PHP que possibilite realizar inserção e atualização (*insert* e *update*) na mesma.

Observação: No formulário, para campos do tipo DATE, utilize:

```
<input type="date" name="..." />
```

ANEXO D - Avaliações e Provas da disciplina Linguagem de Programação I (LPI) do Curso Técnico em Informática para Internet Integrado ao Ensino Médio - 2014 Professor Pesquisador



IFPR – Instituto Federal do Paraná
Câmpus Telêmaco Borba

AVALIAÇÃO BIMESTRAL – 1º Bimestre

ALUNO:

DATA: 10/04/2014

DISCIPLINA: **Linguagem de Programação I - LPI** PROFESSOR: **Pesquisador** CURSO: **Técnico em Informática para Internet Integrado ao Ensino Médio**	**CONCEITO**

Orientações:

- *Avaliação Prática individual.*
- *Crie uma pasta na área de trabalho com seu nome;*
- *Salve cada exercício conforme o modelo ExQ1, ExQ2...*
- *Ao término da avaliação chame o professor.*

1. Escreva um programa que imprima na tela 300 vezes a seguinte frase **"ISHIIIII VOU TER DE APRENDER A PROGRAMAR"**

2. Escreva um programa que leia do teclado dois números e guarde nas variáveis **A** e **B**, troque os valores (**A deve receber B e B deve receber A)**, e imprima na tela os valores de **A** e **B.**

3. Escreva um programa que leia do teclado dois números **X** e **Y**, e realize a soma do intervalo informado pelo usuário, e imprima na tela o valor da soma.

4. Escreva um programa que leia do teclado três valores (a, b, c) e imprima na tela estes valores em ordem crescente.

5. Escreva um programa que leia do teclado a idade de dez pessoas e ao término imprima:

   a. Valor da idade do mais Jovem.
   b. Valor da idade do mais velho

IFPR – Instituto Federal do Paraná
Câmpus Telêmaco Borba

AVALIAÇÃO BIMESTRAL – 2º Bimestre

ALUNO:

DATA: 03/07/2014

DISCIPLINA: **Linguagem de Programação I - LPI**

PROFESSOR: **Pesquisador**

CURSO: **Técnico em Informática para Internet Integrado ao Ensino Médio**

CONCEITO

Orientações:

- *Avaliação Prática individual.*
- *Crie uma pasta na área de trabalho com seu nome;*
- *Salve cada exercício conforme o modelo ExQ1, ExQ2...*
- *Ao término da avaliação chame o professor.*

1. Escreva um programa que peça dois números inteiros e imprima na tela a soma desses dois números.

2. Escreva um programa que leia um valor em metros e imprima na tela o valor convertido em centímetros.

3. Escreva um programa que leia a quantidade de dias, horas, minutos e segundos do usuário, e calcule e imprima na tela o total em segundos.

4. Escreva um programa que calcule o tempo de uma viagem de carro. O programa deve ler a distância a percorrer e a velocidade média esperada para a viagem. O programa deve imprimir na tela o tempo estimado para viagem.

5.

URI Online Judge | 1142

PUM

Adaptado por Neilor Tonin, URI Brasil

Timelimit: 1

Escreva um programa que leia um valor inteiro N. Este N é a quantidade de linhas de saída que serão apresentadas na execução do programa.

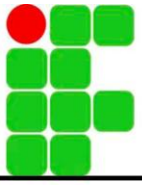
Entrada

O arquivo de entrada contém um número inteiro positivo N.

Saída

Imprima a saída conforme o exemplo fornecido.

Exemplo de Entrada	Exemplo de Saída
7	1 2 3 PUM 5 6 7 PUM 9 10 11 PUM 13 14 15 PUM 17 18 19 PUM 21 22 23 PUM 25 26 27 PUM



IFPR – Instituto Federal do Paraná
Câmpus Telêmaco Borba

AVALIAÇÃO BIMESTRAL – 3º Bimestre

ALUNO:

DATA: 03/07/2014

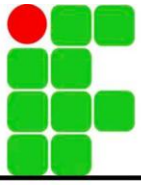
DISCIPLINA: **Linguagem de Programação I - LPI** PROFESSOR: **Pesquisador** CURSO: **Técnico em Informática para Internet Integrado ao Ensino Médio**	**CONCEITO**

Orientações:

- *Avaliação Prática individual.*
- *Crie uma pasta na área de trabalho com seu nome.*
- *Escolha um dos dois exercícios abaixo para resolver.*
- *Ao término da avaliação chame o professor.*

1. Escreva um programa utilizando funções que:
   a. Receba o nome e o sexo de 10 pessoas;
   b. Mostre na tela um menu para impressão da listagem de pessoas por sexo;
   c. Mostre na tela a listagem selecionada pelo usuário ordenando os nomes das pessoas em ordem crescente.

2. Escreva um programa utilizando funções que:
   a. Leia um valor de ponto flutuante com duas casas decimais. Este valor representa um valor monetário.
   b. Calcule o menor número de notas e moedas possíveis no qual o valor pode ser decomposto.
   c. Considere notas de 100, 50, 20, 10, 5, 2 e moedas de 1, 0.50, 0.25, 0.10, 0.05 e 0.01.
   d. Mostre na tela a relação de notas e moedas necessárias.



IFPR – Instituto Federal do Paraná
Câmpus Telêmaco Borba

AVALIAÇÃO BIMESTRAL – 4º Bimestre

ALUNO:

DATA: 11/12/2014

DISCIPLINA: **Linguagem de Programação I - LPI** PROFESSOR: **Pesquisador** CURSO: **Técnico em Informática para Internet Integrado ao Ensino Médio**	**CONCEITO**

Orientações:

- *Avaliação Prática individual.*
- *Crie uma pasta na área de trabalho com seu nome.*
- *Ao término da avaliação chame o professor.*

1. **Considere o sistema descrito abaixo:**

   Um cinema precisa de um sistema de controle de cadeiras para suas salas.

   Cada sala possui cinquenta fileiras de cadeiras, onde dessas, 10 fileiras são da classe especial. Na classe especial são quatro cadeiras por fileira e nas outras são seis.

2. Com essas informações e sabendo que é necessário a ocupação de todas as cadeiras da sala de cinema seja armazenada numa única estrutura de dados homogênea, desenvolva um programa com um menu que realize as seguintes funções:
   a. Preencher uma vaga com os seguintes parâmetros: Classe, Número da Fileira, Número da Cadeira.
   b. Liberar uma vaga, que será usado em caso de desistência, com os mesmos parâmetros acima.
   c. Relatório de fileiras que estão vazias, ou seja, as fileiras que não tenham nenhuma cadeira ocupada.
   d. Relatório de porcentagem de cadeiras que estão desocupadas.
   e. Relatório de lucro conforme a descrição abaixo.

3. Para o relatório de lucro considere que a sala de cinema não lucra se não tiver uma taxa de ocupação de pelo menos 50% do valor unitário das cadeiras ocupadas, além disso, o valor de uma cadeira na classe especial é igual a 5 vezes o valor unitário de uma comum, no relatório de lucro apresente as seguintes informações:
   a. Se a sala de cinema estiver lucrando mostre na tela “LUCRO”;
   b. Se a sala de cinema NÃO estiver lucrando, mostre na tela “PREJU”, e informe abaixo quantas cadeiras comuns devem ser vendidas (prioritariamente), seguido se necessário de quantas mais cadeiras de classe especial devem ser vendidas para lucrar.

ANEXO E - Diário de Classe da disciplina de Lógica de Programação do Curso Programador Web – PRONATEC 2014
Professor Pesquisador – Registros de Conteúdos

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

IFPR - Instituto Federal do Paraná - Campus Telêmaco Borba
Direção de Ensino, Pesquisa e Extensão
Coordenação de Ensino

Registro de Conteúdos - LÓGICA DE PROGRMAÇÃO - PROGRAMADOR WEB - PRONATEC

Professor	PROFESSOR PESQUISADOR
Bimestre	1o Bimestre

Aula	Data	Tema	Observações
1	15/09/2014	Apresentação da disciplina, a metodologia de trabalho, instalação da ferramenta a ser utilizada (IDLE-PHYTON-3.3), comandos PRINT com caracteres e números; expressões matemáticas simples; uso de variáveis, Identação, e o laço de repetição FOR, RANGE	Sem obs.
2	22/09/2014	Revisão dos comandos apresentados na aula anterior. Variações do comando PRINT (pular linha, imprimir na sequência). Comando INPUT, INPUT com FOR. Desafio semanal I	Sem obs.
3	29/09/2014	Resolução do Desafio Semana I; Comandos IF e ELSE com os operadores relacionais e lógicos.	Sem obs.
4	06/10/2014	Revisão dos comandos apresentados. Contador e acumulador; Desafio semanal II.	Sem obs.
5	13/10/2014	Resolução do Desafio Semana II; operador % em suas funções. Operador % com IF e ELSE; Reforço na questão da identação.	Sem obs.

INSTITUTO FEDERAL
PARANÁ
Câmpus Telêmaco Borba

Registro de Conteúdos

Ministério da Educação

6	20/10/2014	Revisão dos comandos apresentados. Desenvolvimento de uma aplicação simples para o cálculo das despesas domésticas. Atividade de desenvolvimento de uma aplicação simples pelos alunos. Desafio semanal III.	Sem obs.
7	03/11/2014	Resolução do Desafio Semana III; comandos LOOP, WHILE, BREAK.	Sem obs.
8	10/11/2014	Revisão dos comandos apresentados. Funções def; desenvolvimento de uma aplicação simples para cálculos geométricos utilizando Funções.	Sem obs.
9	17/11/2014	Avaliação Bimestral (prática); conteúdos: estrutura de sequência, decisão e repetição.	Sem obs.
10	24/11/2014	Análise do desempenho dos alunos, revisão dos tópicos necessários.	Sem obs.

Telêmaco Borba, 24 de Novembro de 2014

PROFESSOR PESQUISADOR
Siape: