

Joinville/SC – 26 a 29 de Setembro de 2017
UDESC/UNISOCIESC
“Inovação no Ensino/Aprendizagem em
Engenharia”



COBENGE 2017
XLV CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA

MÉTODOS E RECURSOS PARA O ENSINO DE PROGRAMAÇÃO: EXPERIÊNCIA COM FOCO NO SISTEMA COMPUTACIONAL EM UM CURSO DE ENGENHARIA

Fabio Rafael Segundo - fabio.segundo@ufsc.br

Departamento de Engenharias
Rua João Pessoa, 2750 – Sala B-117
89036-256 – Blumenau – SC

***Resumo:** Este trabalho relaciona os principais métodos encontrados na literatura para o ensino de programação e apresenta alguns recursos potenciais. É descrita uma estratégia de ensino de programação com foco no sistema computacional e um caso da sua aplicação em três semestres na disciplina de Introdução às Ciências da Computação em um curso de engenharia. Os resultados apontam vantagens do método e considerações sobre os recursos utilizados, baseando-se na pesquisa respondida pelos alunos e no seu desempenho na disciplina.*

***Palavras-chave:** ensino de programação, programação para engenharias, sistema computacionais.*

1. INTRODUÇÃO

Os sistemas computacionais com seus programas são utilizados em diversos campos e áreas do conhecimento, desde as tarefas mais simples até as mais complexas. Por isso, a programação de computadores já faz parte do currículo básico de diversos cursos técnicos e de formação superior. Quando os alunos desses cursos têm pouco ou nenhum conhecimento prévio sobre o assunto, o ensino de programação torna-se mais desafiador, exigindo estratégias de ensino mais elaboradas (CHEN, 2011).

Uma disciplina introdutória desse assunto, de modo geral, objetiva que o aluno obtenha entendimento básico sobre a solução de problemas apoiado no uso de uma sequência de instruções logicamente organizadas chamada programa. Ao entender a programação básica de um sistema computacional, os alunos têm condições de criar programas básicos e programar aplicativos, tais como o de computação numérica, propondo soluções que poderão ser integradas às suas áreas de formação.

A programação está muito relacionada com conceitos matemáticos, principalmente, relacionados a algorítmica e a álgebra booleana. Um programador deve formar representações abstratas de um processo, expressá-las na forma de estruturas lógicas e traduzir essas estruturas em um código correto em uma linguagem de programação (MOHOROVICIC, 2011).

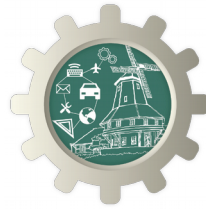
Para tanto, aprender a programar consistentemente, mesmo que de um modo básico, exige o aprendizado sobre a resolução de problemas de forma lógica, além do funcionamento

Organização



Promoção





dos sistemas computacionais e das instruções para sua programação. A resolução lógica de problemas é um dos aprendizados que os alunos frequentemente encontram mais dificuldades (JACKSON, 2009; CRUZ, 2013; LOOI, 2014).

Diante disso, vários trabalhos propõem o emprego de métodos e recursos para melhorar a aprendizagem da programação, tais como método focando a resolução de problemas (LOOI, 2014), a aprendizagem por padrões (MOHOROVICIC, 2011), a abordagem construtivista (BORGES, 2000), entre outros.

Este trabalho relaciona métodos encontrados na literatura para o ensino de programação e apresenta alguns recursos potenciais. É apresentado um caso com aplicação de diferentes estratégias de ensino em três semestres consecutivos na disciplina de Introdução às Ciências da Computação. Além disso, são analisados e apresentados dados quantitativos de uma pesquisa exploratória com os alunos em um levantamento sobre o potencial de aprendizagem dos recursos utilizados nesse período.

2. O ENSINO DE PROGRAMAÇÃO

O desafio do ensino de programação consistente envolve a capacidade de resolução de problemas e de utilização dos recursos de uma linguagem de programação para a codificação da solução modelada. Esse desafio fomenta a proposição de diferentes métodos de ensino. Segundo Looi (2014), não basta que os estudantes entendam a teoria relevante, pois esse conhecimento somente é de fato compreendido se for aplicado na solução de problemas reais. Essa questão não é potencializada no método tradicional de ensino de programação, pois trabalha pouco o aprendizado da resolução lógica de problemas e o entendimento adequado das ações do programa no sistema computacional. Focando apenas o programa e seu objetivo final, o ensino tradicional acaba não fornecendo meios para que o aluno entenda consistentemente o problema e sua solução computacional, falhando na generalização de conceitos para resolver outros problemas (BORGES, 2000).

Figura 1 – Exemplos de sistemas computacionais.

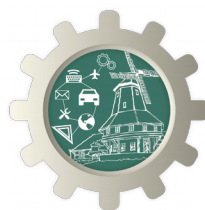


Organização



Promoção





Ressalta-se assim, a importância de pesquisar métodos e recursos que fomentem o aprendizado de programação levando em consideração o sistema computacional, principalmente para o ensino introdutório de programação. A Figura 1 ilustra exemplos de sistemas computacionais normalmente conhecidos dos alunos, caracterizados por terem programas, processador e memória como elementos comuns.

2.1. Métodos de Ensino de Programação

O método tradicional ou instrucionista de ensino de programação é baseado na exposição gradual de comandos, sua função e aplicação em um programa. À medida que o aluno é apresentado a mais comandos, os programas apresentados pelo professor vão sendo mais complexos (BORGES, 2000). Frequentemente acaba desmotivando os alunos que não conseguem relacionar adequadamente as instruções dos programas à resolução do problema. Isso fica evidente na proposição pelo aluno de uma solução adotada em um programa anterior para resolver um problema com solução diferente. Acaba assim exigindo maior esforço dos alunos e capacidade de busca de outros meios para aprenderem a programar.

De outra forma, a Aprendizagem Baseada em Problemas (*Problem-Based Learning – PBL*) traz uma metodologia educativa caracterizada pelo ensino através da aplicação de problemas. Os alunos são organizados em grupos para analisar e propor soluções aos problemas de forma sistematizada. Essa abordagem é centrada em tipos principais de problemas que os profissionais encontrarão rotineiramente e o estudante assume papel ativo. A PBL é focada em sete passos e existem modelos diferentes (MOHOROVICIC, 2011).

Essa metodologia foi usada com sucesso no trabalho de Looi (2014) no ensino de programação. Contudo, os resultados são baseados nas avaliações realizadas em grupo, não sendo apresentadas informações sobre o desempenho individual. Uma vez que os alunos devem pesquisar a solução de um problema real, os alunos acabam buscando informações de como o programa interagirá com o sistema computacional.

Outro método é o aprendizado baseado em Quebra-cabeças (*Puzzle-based learning – PZBL*) (FALKNER, 2010). Esse método ajuda os alunos a reconhecer e integrar partes de um programa para alcançar a solução desejada, tal como partes de um quebra-cabeças encaixadas para obter a figura final. O método é organizado em vários passos: enunciação do problema; apresentação das linhas de códigos do programa separadas, podendo ser agrupadas de acordo com a função, sendo que uma linha de código é a menor parte do quebra-cabeças; e então, a proposição da reconstrução do programa.

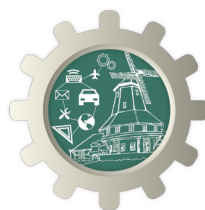
Um método apoiado em três técnicas, uso de exemplos, de padrões e de exercícios, foi apresentado no trabalho de Chen (2011). Nesse método são utilizados exemplos para se analisar a descrição de cada problema de programação e diferenciar cada solução relacionando o problema. Os padrões são usados para que se aprenda a lidar com novos problemas de programação baseando-se em partes básicas/comuns de programas, integrando-as para a sua solução. Diferente do PZBL, os padrões constituem partes que normalmente resolvem genericamente uma situação e não são linhas de código isoladas (Falkner, 2010). Por fim, são usados exercícios para praticar a escrita correta dos programas, diminuindo as chances de que o aluno modele um algoritmo e não consiga codificar uma solução correta. Chen (2011), usou um sistema *web on-line* com *feedback* sobre os exercícios para dar

Organização



Promoção





autonomia aos alunos. Essa forma de proporcionar autonomia também foi explorada no método adotado por Rocha (2010).

No método de Programação em par (*Pair Programming* – PP), os programadores trabalham em pares na modelagem do programa e na codificação cada um assume sua função: guia ou navegador. Enquanto o guia codifica as linhas do programa, o navegador observa e orienta o guia para a construção correta do programa. Assim, tem-se um trabalho colaborativo que resulta na diminuição dos erros e do tempo de codificação. Os papéis podem ser trocados e observa-se que pode ser difícil manter-se no seu papel (MOHOROVÍČIĆ, 2011).

A preparação cuidadosa e técnica dos materiais de aula e a disponibilização aos alunos é utilizada no método de Leituras Pré-gravadas (*Prerecorded Lectures* – PL).

Normalmente é apresentado em material multimídia on-line com narração das notas de aula. Para alguns alunos, o tempo para as aulas e para a resolução dos exercícios acaba não sendo suficiente para aprender, assim a PL ajuda o aluno a rever as explicações de aula e se concentrar nos conceitos mais importantes com explicações pré-gravadas, organizadas e revisadas pelo professor. Apesar do retorno positivo dos alunos quanto esse método, Smith (2008) não observou melhoria significativa na aprendizagem nos seus ensaios.

Outro método que pode ser citado é a Programação Baseada em Jogos (*Game-Themed Programming* – GTP), o qual utiliza uma pequena aplicação de jogo como tema para o aprendizado de conceitos de programação. As partes da implementação do jogo são organizadas em tarefas atribuídas a cada aluno. Esse método proporciona motivação e entusiasmo aos alunos quando eles conseguem entender o funcionamento geral do jogo, suas tarefas e a interação entre elas. Contudo, esse entendimento pode exigir um tempo considerável. Esse método proporciona maior aprendizado do que aulas tradicionais, mas o aluno deve aprender mais programação (SUNG, 2010).

Por fim, Borges (2000) apresenta um método construtivista para ensino de programação, o qual se diferencia por focar na resolução do problema e ir apresentando a teoria relacionada de acordo com a necessidade encontrada pelo aluno na busca de uma solução. O professor deve estar mais focado na motivação dos alunos, uma vez que o trabalho fica mais individualizado e, como consequência, também deve ser aplicado em turmas menores. Uma restrição a essa abordagem é a cultura instrucionista já adquirida pelos alunos, dificultado o trabalho na lógica construtivista.

3. RECURSOS UTILIZADOS PARA O APRENDIZADO DE PROGRAMAÇÃO

Na aplicação de um método de ensino é importante adotar os recursos mais adequados para favorecer a aprendizagem quanto a diversos aspectos. O principal aspecto é a devida compreensão do conteúdo, mas a quantidade de alunos atingidos, a facilidade de acesso, o favorecimento da colaboração, o aprofundamento técnico, a autonomia, entre outros, podem ser considerados. Alguns recursos são descritos a seguir (ROCHA 2010; CHEN, 2011; CRUZ, 2013):

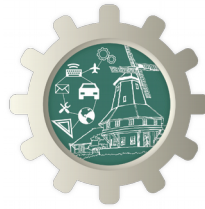
- Algoritmo informal: é uma sequência de passos gerais informais que permite organizar uma solução para um problema, mas abstraindo detalhes desse processo. Isso ajuda os alunos na interpretação do enunciado do problema e a organizar mentalmente estruturas abstratas em sequências lógicas no sentido da solução.

Organização



Promoção





- Fluxogramas: são representações gráficas que permitem modelar um algoritmo, avaliar o fluxo de programa e sua lógica, favorecendo o seu entendimento e a sua verificação. Os aplicativos Raptor¹ e Flowgorithm² permitem avaliar o algoritmo em execução e traduzi-lo em linguagens.
- Teste de mesa: é um método para avaliar o comportamento lógico de um algoritmo, no qual as variáveis são relacionadas em colunas de uma tabela e segue-se o fluxo de programa alterando-se o valor da variável quando necessário, instrução por instrução.
- Listas de exercícios: contem os enunciados dos problemas e podem ou não apresentar a solução dos exercícios de acordo com a estratégia de ensino do professor.
- Trabalhos em grupos: favorecem a colaboração e cooperação para atingir a resolução do problema, procurando envolver todos os alunos na atividade.
- Websites com informações: permitem o acesso facilitado às informações do curso pelo aluno, possibilitando revisitar conteúdos e resolver exercícios. Exemplos são os ambientes virtuais de aprendizagem (AVA).
- Sistemas automáticos de programação: são ambientes em que o aluno pode resolver o exercício no próprio aplicativo. Além do enunciado, alguns são preparados para mostrar os erros, dar *feedbacks*, mostrar a solução e registrar as tentativas. O módulo de programação do Moodle³ e o Blockly⁴ Games são exemplos.
- Monitoria: depois dos horários das aulas os alunos podem solicitar ajuda de um monitor que tenha experiência em programação. Deve-se verificar a disponibilidade de horários da monitoria para não conflitar com a do curso e de outras disciplinas do aluno.
- Aplicativos de programação facilitada: são programas que apresentam uma forma de programação facilitada, normalmente visual, através de blocos que se encaixam e com atores. Um exemplo de aplicativo disso é o Scratch⁵.
- Linguagem de programação: o modo como a linguagem permite organizar a programação deve seguir um paradigma, como por exemplo o procedimental e o orientado a objetos. O modo procedimental é uma forma de programação mais fácil para iniciantes e por isso frequentemente adotado em cursos de introdução à programação.

Jackson (2009) compreende que a escolha da linguagem é uma das questões menos importantes para proporcionar o aprendizado. Todavia, Mohorovičić (2011) acredita que a linguagem escolhida tem um papel relevante na aprendizagem da programação. De modo geral, a preocupação dos pesquisadores do tema concentra-se na melhoria da capacidade de resolução de problemas.

1 Aplicativo com licença de uso gratuita. Mais informações em “<http://raptor.martincarlisle.com/>”.

2 Aplicativo com licença de uso gratuita. Mais informações em “<http://www.flowgorithm.org/>”.

3 Plugin para o AVA Moodle. Mais informações em “https://moodle.org/plugins/mod_vpl/”.

4 Aplicativo web sem controle de acesso. Mais informações em “<https://blockly-games.appspot.com/>”.

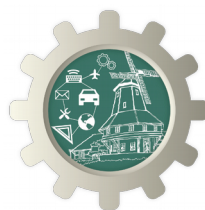
5 Aplicativo com licença de uso gratuita. Mais informações em “<http://scratch.mit.edu/>”.

Organização



Promoção





4. ESTUDO DE CASO NA DISCIPLINA DE INTRODUÇÃO À CIÊNCIA DA COMPUTAÇÃO

Um estudo de caso foi realizado com a disciplina de Introdução à Ciência da Computação durante três semestres consecutivos de um curso público de graduação em engenharia. Uma pesquisa exploratória foi realizada com os alunos quanto a sua percepção da aprendizagem obtida na disciplina por assunto e recurso utilizado. A partir disso, são apresentadas questões quanto ao método de ensino empregado, a prática e a utilização dos recursos em aula.

4.1. O Método de Ensino Utilizado

A metodologia adotada combina diferentes métodos. Utiliza elementos da abordagem instrucionista, da construtivista, da abordagem de Chen (2011) e da PBL. Os elementos de cada abordagem foram:

- abordagem instrucionista: adoção de exercícios com dificuldade gradual, facilitando a adaptação dos alunos tendo em vista sua cultura prévia;
- abordagem de Chen (2011): uso de exemplos, de padrões e de exercícios, fomentando a prática de resolução de problemas e sua generalização;
- abordagem construtivista: os exercícios vão se complementando até obter a solução de um enunciado de dificuldade maior, sendo o aluno responsável em fazer as escolhas na construção de uma solução;
- abordagem de PBL: resolução de problemas em jogos e de um problema real utilizando-se de grupos.

Cada exercício é baseado no enunciado de um problema. Os problemas são apresentados dos mais simples aos mais complexos, estes sendo uma construção baseada nos exemplos mais simples.

A solução para o problema é organizada iniciando-se com a análise do enunciado e a representação de uma ideia de solução descrevendo-a como um algoritmo informal. Uma vez que adequada para se alcançar a solução, realiza-se a modelagem em fluxograma, o teste de mesa, a codificação em uma linguagem de programação e os testes de execução. Esse processo de desenvolvimento de uma solução algorítmica para o problema enunciado é apresentada na Figura 2. Nessa figura, as etapas estão em quadros e as setas indicam os possíveis resultados da etapa. Dependendo do resultado da etapa pode ser necessário voltar para a etapa anterior.

Esse processo apresenta melhores resultados quando o desenvolvedor da solução consegue imaginar ou usar a abstração para alcançar uma ideia de solução que possa ser considerada adequada e que conseqüentemente possa levar a uma sequência informal de passos na direção da solução do problema. Para isso, o método utilizado adota como suporte explicações sobre os sistemas computacionais, fornecendo meios para fortalecer a idealização do sistema computacional e do programa em funcionamento.

Neste sentido, deve ser ensinada primeiramente a estrutura dos sistemas computacionais e a interação dos programas com as partes do sistema, utilizando como exemplos equipamentos do dia a dia dos alunos. As estruturas de controle de repetição e divisão de fluxo são ensinadas em seguida, remetendo-se aos sistemas computacionais exemplificados. Jogos podem ser usados para ajudar o aluno a entender as estruturas de controle e o fluxo de programa.

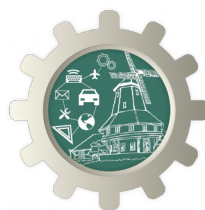
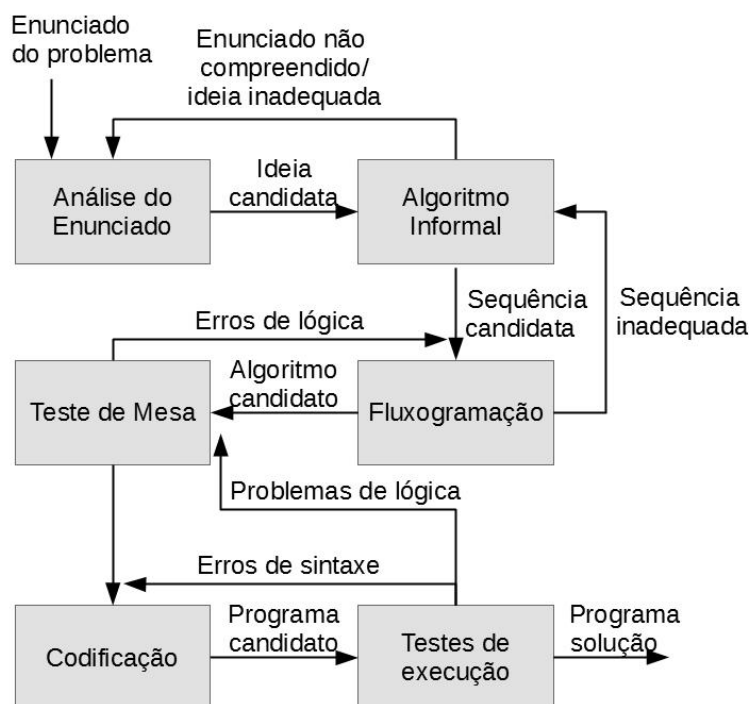


Figura 2 – Processo utilizado para a resolução dos enunciados de problemas.



Em cada semana o professor deve resolver um ou mais enunciados de problemas de mesmo nível de dificuldade, utilizando todos os passos do processo ilustrado na Figura 2. Em laboratório, os alunos devem repetir a resolução dos enunciados já apresentados e desenvolver soluções para outros enunciados de problemas do mesmo nível. A cada nova semana o nível de dificuldade dos problemas é incrementado, sendo que os problemas já resolvidos servem de base para a construção das novas soluções.

Depois da prática da resolução de programas básicos, tal como o cálculo da média de três números e verificação do triângulo equilátero, são apresentados padrões de programas para ajudar a solucionar partes dos problemas, mesmo que os problemas sejam diferentes, incentivando a comparação entre problemas e soluções. Esses padrões são mostrados como técnicas de programação e repetidos cada vez que foram necessários. Exemplos dessas técnicas são a utilização de *flags* para controlar laços de repetição, o uso de variável para salvamento do maior ou do menor valor e a variável auxiliar para realizar troca de valores entre posições de vetores.

Após o aluno ter praticado a resolução dos exercícios, é apresentado um trabalho prático e individual de programação de uma série de problemas contextualizados em um ambiente real e que pode ser realizado com auxílio de livros, dos colegas e/ou da monitoria.

4.2. Prática e Utilização dos Recursos em Sala de Aula

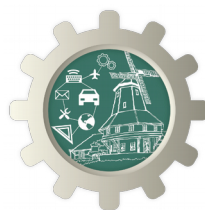
Os primeiros passos dos alunos do curso na programação foram realizados de forma lúdica com as atividades de resolução de problemas com o uso do Blockly Games. Os jogos

Organização



Promoção



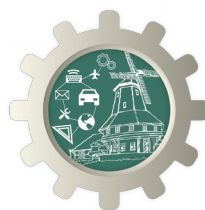


disponíveis exigem lógica para a montagem de sequências de passos com estruturas de repetição, condicionais e variáveis. Nesta atividade foi possível identificar os alunos que precisariam de mais suporte para aprender a programar.

Tabela 1 – Exemplos de enunciados de problemas como exercícios, um possível algoritmo informal e padrões/técnicas utilizadas.

Enunciado do problema	Algoritmo informal	Padrões/técnicas de programação
A1) Receba três números informados pelo usuário e apresente a média aritmética simples desses números.	Receba o primeiro número digitado pelo usuário. Receba o segundo número digitado. Receba o terceiro número digitado. Some os três números e divida por 3. Apresente o resultado obtido na tela. Fim do programa.	- Nenhum
A2) Receba mil números informados pelo usuário e apresente a média aritmética simples desses números.	Receba um número digitado pelo usuário. Repita mil vezes o passo anterior e vá somando os valores informados. Divida a soma por mil. Apresente o resultado obtido na tela. Fim do programa.	- Repetição com variável contador; - Acumulador em laço de repetição.
A3) Uma máquina gera um estado de funcionamento (aqui será obtido pelo teclado). Quando esse estado for -9 deve-se apresentar na tela a mensagem “máquina parada” e terminar o programa.	Receba um número digitado pelo usuário (status da máquina). Repita o passo anterior enquanto o valor seja diferente de -9. Apresente na tela a mensagem: “máquina parada”. Fim do programa	- Repetição com <i>flag</i> de saída.
B1) Receba números informados pelo usuário e apresente a média aritmética simples desses números. O programa deve parar quando o usuário informar -9.	Receba um número digitado pelo usuário. Repita três o passo anterior enquanto o valor não for -9, incremente uma variável contador e acumule a soma dos valores. Calcule a média dividindo a soma acumulada pela variável contador. Apresente o resultado obtido na tela. Fim do programa	- Repetição com <i>flag</i> de saída. - Contador em laço de repetição. - Acumulador em laço de repetição.

Ao introduzir os primeiros conceitos sobre programação e iniciar o uso de algoritmos informais, foi verificada a necessidade de um suporte para que os alunos pudessem idealizar o



programa em funcionamento. Dessa forma, o aluno teria melhores condições de imaginar algum encaminhamento no sentido de uma solução para o enunciado. Isso pois, mesmo nos casos em que o aluno havia entendido o problema enunciado, ele não conseguia entender como se dava uma solução algorítmica através de programas. Além disso, alguns alunos não sabiam quais os procedimentos tinham a sua disposição e quais poderiam ser utilizados para compor uma solução para o problema. Vamos tomar como exemplo os exercícios enunciados na Tabela 1, onde A1, A2 e A3 são enunciados básicos e B1 é um enunciado em um nível imediatamente superior que utiliza padrões utilizados em A2 e A3. Para muitos alunos iniciantes, não é fácil entender que a solução do problema enunciado em A2 passa pelo uso de uma instrução que recebe dados do teclado, que existe uma repetição controlada até mil vezes e o uso de um acumulador, e que após a repetição há uma instrução de cálculo e uma instrução que envia informações para o monitor de vídeo do computador.

Uma forma utilizada para favorecer esse entendimento foi explicar a arquitetura geral dos sistemas computacionais conforme organizada na Figura 3. Para mostrar o papel da linguagem de programação e do programa no sistema computacional foi explicada a sua interação com cada uma das partes gerais do sistema: processador, memória e periféricos (Figura 3). Todavia, por não ser objetivo da disciplina e do curso, não se entrou em detalhes de hardware e outros do baixo nível computacional.

Tomando-se como exemplos os equipamentos mostrados na Figura 1, é enfatizada a obtenção e envio de dados de acordo com seus periféricos de entrada e saída e como os programas se relacionam com isso. Assim, utilizou-se do entendimento da lógica do funcionamento sistemático do sistema computacional, que está intimamente relacionado com as soluções algorítmicas, para fomentar ideias para resolução dos problemas. Na Figura 3, as cores colaboram com a explicação sobre três sistematizações que normalmente geram questões relevantes realizadas pelos alunos:

1. (Em vermelho) Onde está o programa com suas instruções e como estas fazem as partes do computador funcionar.
2. (Em amarelo) De onde vem os dados de entrada e para onde vão os dados de saída.
3. (Em verde) Por que há memórias diferentes.

As interações entre o programa e as outras partes do sistema computacional foram sendo explicadas no decorrer do curso com a utilização da linguagem de programação C. Apesar de ser uma linguagem de alto nível, faz com que o programador deva ficar atento às limitações do sistema computacional, reforçando a integração existente entre suas partes.

A utilização de modelagem dos algoritmos com fluxogramas ajudou os alunos a obter uma versão visual do programa antes de codificá-lo em uma linguagem de programação. Uma questão importante para os alunos foi entender a ligação dos fluxogramas com a linguagem de programação. Essa ligação é favorecida quando é possível criar uma correspondência fácil dos símbolos e setas do fluxograma para as instruções de uma linguagem de programação. Para tanto foram utilizadas convenções no uso dos algoritmos, tais como a obrigatoriedade de inicialização de variáveis no início do algoritmo e a definição de uma estrutura de controle de repetição padronizada.

Organização



Promoção



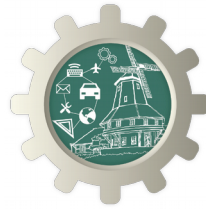
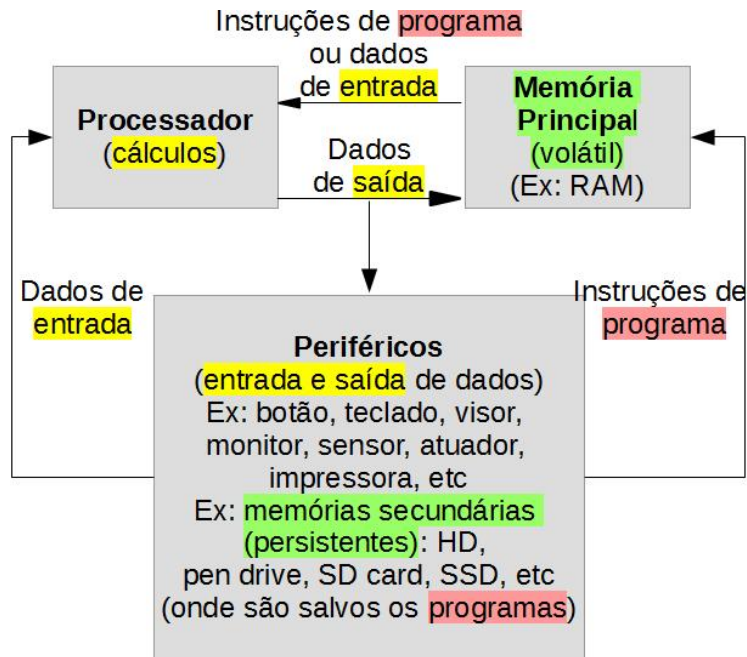


Figura 3 – Arquitetura geral dos sistemas computacionais comentada.



O uso de um Ambiente Integrado de Desenvolvimento (Integrated Development Environment - IDE) de programas com marcador colorido de código ajudou os alunos a indentificarem os blocos de comandos, funções, expressões condicionais e outros elementos da linguagem de programação.

As listas de exercícios foram sendo apresentadas para os alunos aumentando-se a dificuldade dos problemas a cada semana. Na semana, os alunos tiveram metade das aulas teóricas e outra metade com prática em laboratório.

Um exemplo do aumento do nível de dificuldade e da resolução de problemas baseado em padrões é apresentado na Tabela 1. Isso pode ser verificado ao avaliar que a solução do enunciado B1 passa pela construção de uma solução utilizando-se o conhecimento trabalhado em A1 e em A2.

A atividade final do curso foi baseada em um contexto real em que as máquinas de um chão de fábrica geram arquivos com seus registros de operação (logs). Apesar de o trabalho ser individual, os alunos deveriam trocar informações em grupos para desenvolver programas que solucionassem os problemas enunciados comuns para cada grupo. Alguns desses problemas envolveram a geração de vetores, gráficos e cálculos para avaliar o comportamento das máquinas, o que é interessante para um aluno de engenharia. Um exemplo de enunciado pedia a geração de um gráfico semanal com o número de paradas de uma máquina logo após a carga de uma determinada matéria-prima.

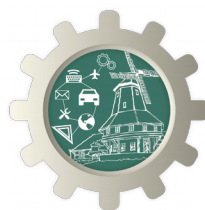
A avaliação do curso foi feita sobre uma prova individual teórica, uma prova individual prática, o trabalho final e uma nota de avaliação formativa, em que o professor registrava o envolvimento do aluno na realização das atividades propostas.

Organização



Promoção





5. RESULTADOS

O método e práticas apresentadas surgiram diante das dificuldades encontradas no ensino e no acompanhamento da aprendizagem obtidas pelos alunos. Um resumo das dificuldades encontradas, das abordagens e dos recursos utilizados é apresentado na Tabela 2.

Para avaliar essa metodologia e prática descritas, uma pesquisa foi realizada em três semestres de oferta da disciplina. No primeiro semestre não foi abordada a integração dos programas com o sistema computacional e foi dada uma ênfase mais instrucionista ao ensino, apesar de também haver a resolução de um trabalho final com problemas do contexto real. Apenas no segundo semestre utilizou-se o Scratch, além do Blockly Games. A pesquisa objetivou avaliar o potencial de aprendizagem do método e dos recursos na visão dos alunos e contrastar com o a avaliação realizada pelo professor.

Tabela 2 – Dificuldades encontradas e estratégias com recursos e abordagens utilizadas nas aulas.

Dificuldades trabalhadas	Estratégias utilizadas
Adaptação à cultura prévia dos alunos	Utilização de elementos da abordagem instrucionista com exemplos e exercícios aumentando-se a dificuldade.
Entendimento/imaginação de um encaminhamento para uma possível solução do enunciado	Generalização dos sistemas computacionais com exemplos do dia-a-dia e explicação das suas partes e questões importantes.
Sistematização da solução	Realização de todas as etapas do processo de desenvolvimento de uma solução por programa, conforme a Figura 2. Utilização de algoritmo informal, fluxogramas e teste de mesa.
Entendimento da lógica de programação	Utilização de resolução de problemas de forma lúdica com o Blockly Games. Ele foi escolhido entre outros, pois ao final do jogo apresenta a solução em código de programa e isso é utilizado para relacionar com os exercícios em linguagem C. Utilização da abordagem baseada no uso de exemplos, de padrões e de exercícios (Chen, 2011).
Entendimento de blocos de instruções, estruturas de controle, variáveis e funções.	Utilização uma IDE que facilite a identificação dos blocos de instrução e outros elementos da linguagem de programação.
Manutenção do interesse/motivação	Aproximação com a realidade dos alunos e listas de exercícios com dificuldade gradual. Utilização de elementos da abordagem construtivista. Realização de um trabalho final não trivial, baseado na resolução de problemas de em um contexto real de um ambiente fabril.



O instrumento de pesquisa foi um questionário enviado para os alunos por correio eletrônico. De 104 alunos que concluíram a disciplina, 27 (25,96%) responderam o questionário, sendo 7 do primeiro, 10 do segundo e 10 do terceiro semestre. A idade média geral destes foi de 21 anos, sendo que apenas 30% tinham mais de 19 anos.

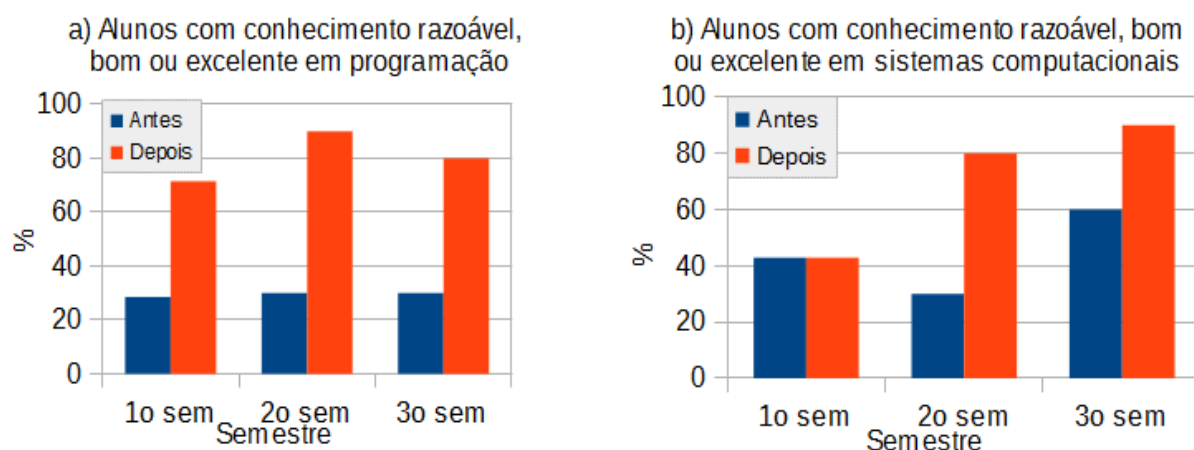
No questionário, as questões avaliaram o conhecimento do aluno em programação e em sistemas computacionais, antes e depois de cursar a disciplina, apresentando a escala: não sei informar; nada; muito pouco; pouco; razoável; bom e excelente. Também foi avaliada a contribuição para o aprendizado do aluno de diversos recursos empregados. O gráfico da Figura 4a mostra o percentual de alunos que responderam sobre seu conhecimento em programação.

O gráfico da Figura 4b mostra o percentual de alunos que responderam sobre seu conhecimento em sistemas computacionais.

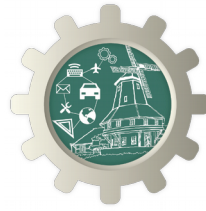
Observa-se que sobre o aprendizado de sistemas computacionais no 1o semestre, o qual não foi abordado o tema, o gráfico não mudou. Nos semestres seguintes, o conhecimento sobre os sistemas computacionais aumentou, o que era de se esperar, mas também aumentou o conhecimento em programação, colaborando com uma visão positiva do método utilizado no segundo e terceiro semestre.

Do primeiro ao terceiro semestre o percentual de aprovação obtida pelos alunos foi de 50,00%, 58,82% e 76,66% respectivamente. Assim, pode-se verificar que houve um aumento do número de aprovações acompanhando um aumento do conhecimento obtido sobre o sistema computacional e programação, o que poderia apontar uma vantagem do método que aumenta o foco no sistema computacional.

Figura 4 – Gráficos percentual de alunos com sua percepção de conhecimento em a) programação; b) sistemas computacionais, antes e depois da disciplina..



Nas respostas do questionário, quanto aos recursos utilizados pelos alunos, pode-se verificar que o uso dos fluxogramas foi de muita utilidade para a aprendizagem da programação. Já quanto ao uso da programação visual em blocos, os alunos responderam de



modo geral que o Scratch, utilizado no segundo semestre avaliado, não contribuiu muito com o aprendizado da programação, pois os alunos não conseguiram generalizar os conceitos de programação e aplicá-los posteriormente na linguagem C. Também não se pode confirmar se houve uma contribuição significativa no aumento da aprendizagem com o uso do Blockly Games. De outra forma, o estudo com os colegas se mostrou de muita importância para o aprendizado, próximo a importância das listas de exercícios, aulas práticas e explicações do professor, recursos que obtiveram maior importância. É importante informar que o Blockly Games e o Scratch foram utilizados em poucas aulas, já o fluxograma foi utilizado pelos alunos em vários encontros e isso pode ter influenciado o resultado.

6. CONSIDERAÇÕES FINAIS

Este trabalho apresentou uma avaliação do ensino de programação por três semestres de um curso de engenharia e uma revisão de alguns métodos e recursos para isso.

O ensino da programação mais atrelada ao ensino do sistema computacional objetivou trazer mais elementos para diminuir a abstração dos alunos na elaboração de soluções para os exercícios de programação e aproximá-los do mundo real. Isso foi considerado um diferencial importante para o entendimento do problema pelo aluno.

Os números relativos ao percentual de aprovação dos alunos e o aumento do conhecimento que eles informaram ter obtido tanto na programação quanto ao aprendizado sobre os sistemas computacionais apontam para um potencial do método utilizado para melhoria da aprendizagem de programação. Todavia, mais resultados são necessário para se comprovar a sua eficácia.

Os resultados obtidos quanto ao potencial de aprendizagem dos recursos utilizados no curso devem ser objeto de mais estudos, pois a forma que o recurso é empregado com os alunos pode interferir. Neste sentido, além da forma de aplicação, a própria frequência de uso, pode influenciar os alunos a considerarem um recurso de maior contribuição para sua aprendizagem do que outro.

Agradecimentos

Agradecimento ao grupo de pesquisa EDUMIDIA pelo apoio para a realização da pesquisa.

REFERÊNCIAS BIBLIOGRÁFICAS

Livros:

CRUZ, A. J. G. "Informática para Eng. Ambiental:... Algoritmos e programação". 2013.

Trabalhos em eventos:

BORGES, M. A. "Avaliação de uma metodologia alternativa para a aprendizagem de programação." VIII Workshop de Educação em Computação–WEI. 2000.

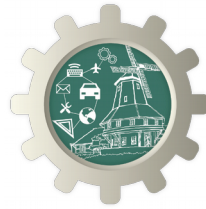
Organização



Promoção



Joinville/SC – 26 a 29 de Setembro de 2017
UDESC/UNISOCIESC
“Inovação no Ensino/Aprendizagem em Engenharia”



COBENGE 2017
XLV CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA

MOHOROVICIC, S., and Vedran S. "An Overview of Computer Programming Teaching Methods." Central European Conf. on Infor. and Intelligent Systems. 2011.
SMITH G, Fidge C. "On the efficacy of prerecorded lectures for teaching introductory programming". In Proceedings of the tenth conference on Australasian computing education-Volume 78 (pp. 129-136). Australian Computer Society, Inc. 2008.

Artigos de Periódicos:

ROCHA, P. S., et al. "Ensino e aprendizagem de programação: análise da aplic. De proposta ... baseada no sistema personalizado de ensino." RENOTE. 2010.
CHEN, W. J., X. Li, and W. D. Liu. "Teaching computer programming to non-computer science students." Proceedings of 3rd ACE. Katahira: IAFOR Publications. 2011.
FALKNER N., Sooriamurthi R., Michalewicz Z. "Puzzle-Based Learning for Engineering and Computer Science". Journal Computer, Vol. 43, Issue 4, pp. 20-28. 2010.
JACKSON, D; Rob M. "A new approach to teaching programming." Cambridge. 2009.
LOOI, H. C., and Afzaal H. S. "Problem-based learning: An analysis of its ... to the teaching of programming." Int. Proc. of Economics Dev. and Research. 2014.
SUNG K, et al. "Game-Themed Programming Assignment Modules: A Pathway ... Programming Courses". IEEE Trans. on Education, Vol. PP, Issue 99. 2010.

**METHODS AND RESOURCES FOR TEACHING OF PROGRAMMING:
A CASE FOCUSING COMPUTATIONAL SYSTEMS IN AN
ENGINEERING COURSE**

Abstract: *This work lists the main methods found in the literature for teaching of programming and presents some potential resources. It presents a programming teaching strategy focusing the computational system and a case of its application on three semesters in the subject of Introduction to Computer Science in an engineering course. The results point out advantages and disadvantages of the method and used resources, based on the research answered by the students confronted with their performance in the course.*

Key-words: *programming teach, programming for engineering, computational systems*

Organização



Promoção

