UNIVERSIDADE ESTADUAL PAULISTA“ JÚLIO DE MESQUITA FILHO”

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUCIANA MIEKO SUNANO

**OBJETO DE APRENDIZAGEM NO APOIO AO ENSINO DE LÓGICA COMPUTACIONAL**

Bauru, SP

2015

LUCIANA MIEKO SUNANO

**OBJETO DE APRENDIZAGEM NO APOIO AO ENSINO DE LÓGICA COMPUTACIONAL**

Trabalho de Conclusão de Curso de graduação apresentado à disciplina Projeto e Implementação de Sistemas do curso de Bacharelado em Ciência da Computação da Faculdade de Ciências da Universidade Estadual Paulista “Júlio de Mesquita Filho”, como requisito parcial para obtenção do título de bacharel.

Orientador: Prof. Dr. Wilson Massashiro Yonezawa

Bauru, SP

2015

LUCIANA MIEKO SUNANO

**OBJETO DE APRENDIZAGEM NO APOIO AO ENSINO DE LÓGICA COMPUTACIONAL**

Trabalho de Conclusão de Curso de graduação apresentado à disciplina Projeto e Implementação de Sistemas do curso de Bacharelado em Ciência da Computação da Faculdade de Ciências da Universidade Estadual Paulista “Júlio de Mesquita Filho”, como requisito parcial para obtenção do título de bacharel.

Aprovado em \_\_\_/\_\_\_/\_\_\_\_

BANCA EXAMINADORA

Prof. Dr. Wilson Massashiro Yonezawa (orientador)

Prof. Dr. Aparecido Nilceu Marana

Prof. Dr. Renê Pegoraro

AGRADECIMENTOS

Agradeço à UNESP e todos os seus funcionários, uma vez que sem vocês esta faculdade não teria se tornado minha segunda casa em Bauru.

À minha mãe Graça, por tudo o que me ensinou e por sempre estar disposta a me aconselhar da melhor maneira possível, com muito amor e com muito apoio em todas as minhas decisões.

Ao meu pai Luiz que, mesmo à distância, nunca esteve longe, sempre me ajudando quando eu mais precisava e por sempre colocar os meus estudos em primeiro lugar até nos momentos mais difíceis.

Ao meu irmão Lucio, principalmente pela amizade, pelos conselhos e também pelas constantes broncas que tanto me incentivaram ao longo desta minha jornada.

À minha amada tia Dete (*in memorian*), por todo o amor e todo o carinho, mesmo que você não esteja mais comigo eu sei que está cuidando de mim como sempre cuidou.

Aos meus animais de estimação, Neguinha, Mingau, Byte e Pixel (*in memorian*) que só por estarem presentes, já faziam o meu dia melhor.

Ao meu orientador Prof. Dr. Wilson M. Yonezawa, por me aceitar como orientanda, pela incrível paciência, por todo o conhecimento passado e, além disso, por todos os conselhos dados, vou levá-los por toda a minha vida.

À minha amiga Carol, irmã de consideração que faz parte da minha família em Bauru, que sempre esteve ao meu lado desde o começo da faculdade, sempre me ajudando e cuidando de mim nos momentos em que eu mais precisei.

Ao meu amigo Mateus, pelo companheirismo e conselhos dados dentro e fora deste trabalho.

À minha amiga Julia, pela irmandade criada principalmente nos últimos anos da faculdade, mesmo à distância.

À minha amiga Lya que se tornou uma cúmplice nesses últimos anos de faculdade e também uma irmã.

Ao meu amigo Saito que se tornou um parceiro durante a faculdade, me trazendo muita felicidade, principalmente quando fizemos intercâmbio juntos.

Ao meu amigo Bruno, que sempre esteve ao meu lado e me ensinou a ser uma pessoa mais compreensiva.

A todos os meus amigos da República Toca e agregados, por se tornar a minha família de Bauru e por fazerem de mim uma pessoa mais aberta e compreensiva, lembrarei de vocês para sempre, literalmente.

Ao Prof. Dr. Humberto Ferasoli Filho, por ser um pai de consideração, sempre me dando os melhores conselhos e também por me apresentar a sua filha Guta, que se tornou uma irmã para mim.

Ao meu grande mestre e amigo Kaor Tanida (*in memorian*), que por mais que não esteja fisicamente comigo, sempre esteve ao meu lado me apoiando.

À minha querida e amada amiga Olympia, por todas as conversas, risadas, cartas trocadas e todas as orações.

À minha amiga e segunda mãe Dora, por todos os conselhos e orações.

A todos os professores, mestres e doutores por todo o conhecimento passado à mim.

Ao Prof. Dr. Augusto Ronchi Junior, por me ajudar a realizar o sonho de participar de um intercâmbio acadêmico.

“Na alma sempre uma chama acesa

Que a luz do luar nos traga inspiração”

(Mulan, 1998)

resumo

A velocidade com a qual as mudanças tecnológicas acontecem cresce a cada dia. É visível a facilidade que as gerações mais recentes têm de lidar com as novas tecnologias, sejam elas para o lazer ou até mesmo para a educação, diferentemente das gerações anteriores, que podem não compreender totalmente o modo como essa nova geração trabalha. Além disso, especificamente em cursos de computação, existe uma grande dificuldade do aluno em conceitos introdutórios ligados à programação, que possivelmente poderia ser minimizada caso esses alunos tivessem sido introduzidos aos conceitos do pensamento computacional logo na infância, ou até mesmo durante o início da educação superior. Este trabalho é um protótipo de um Objeto de Aprendizagem que tem como público alvo estudantes do ensino básico e, também, estudantes do ensino superior, principalmente os de cursos de computação, sendo dividido em atividades com um ou mais exercícios com o propósito de uma aprendizagem progressiva do aluno. Seu objetivo é, através do uso de analogias, que facilitam o aprendizado de conteúdos novos e desconhecidos, desenvolver o pensamento computacional nesses alunos e, consequentemente, o raciocínio lógico, a fim de contribuir com que eles possam criar seus próprios caminhos em busca de soluções em diversas situações da vida.

**Palavras-chave:** Objeto de Aprendizagem, Software no Ensino, Educação, Pensamento Computacional, Analogia.

abstract

Each day, we can see how faster and faster the changes and advances in technology happen. It’s visible the easiness in which the new generations learn how to use the most recent technologies, for fun or for educational purposes, at the same time that people from previous generations may not understand fully how this generation works. In another point, specifically on computation related courses, there is a big difficulty from the students to learn introductory concepts linked to programming that possibly could have been minimized if these students were presented to concepts of computational thinking as children, or even during the very beginning of superior education. This work is a prototype of a Learning Object that has as target audience students from basic school as well as superior educational students, especially those from computation courses. It is separated in activities with one or more exercises with the purpose of a progressive learning by the student. It’s goal is to, by using analogies, that make the learning of new and unknown subjects easier, to develop the computational thinking on those students and, thereafter, the logical thinking, and so helping them to create their own ways when searching for solutions on many different situations in life.

**Key-words:** Learning Object, Teaching Software, Education, Computational Thinking, Analogy.

**LISTA DE FIGURAS**

Figura 1 - Infográfico sobre a geração Z. 7

Figura 2 – Descrição geral das gerações 9

Figura 3 - O pensamento computacional. 17

Figura 4 - Estrutura do computador IAS. 22

Figura 5 - O computador: estrutura de alto nível. 25

Figura 6 - Componentes do computador: visão global. 26

Figura 7 - A unidade central de processamento (CPU). 27

Figura 8 - Ciclo de instruções básico. 28

Figura 9 - Um esquema de computador 30

Figura 10 - Relação de comandos. 31

Figura 11 - Trecho de programa codificado. 31

Figura 12 - O trecho do programa representado na memória. 32

Figura 13 – O processo da analogia. 33

Figura 14 - Protótipo de analogia 1.0. 39

Figura 15 - Protótipo de analogia 1.0, dividido em 4 blocos. 40

Figura 16 - Protótipo de analogia 2.0. 41

Figura 17 - Protótipo de analogia 2.0 divido em blocos. 43

Figura 18 - O agente do protótipo de analogia 2.0. 44

Figura 19 - A mesa de trabalho do protótipo de analogia 2.0. 45

Figura 20 - Protótipo de analogia 2.1. 47

Figura 21 – Protótipo do Objeto de Aprendizagem. 48

Figura 22 – Etapas do Objeto de Aprendizagem. 49

Figura 23 – Tela de boas-vindas. 49

Figura 24 – Apresentação da personagem. 50

Figura 25 – Explicação: Bloco RELEMBRAR EXERCÍCIO. 51

Figura 26 – Explicação: Bloco QUADRO. 51

Figura 27 – Explicação: Bloco ARMÁRIO. 52

Figura 28 – Explicação: Bloco MESA. 52

Figura 29 – Explicação: Bloco BOTÕES. 53

Figura 30: Fluxograma da Atividade 1. 54

Figura 31 – Atividade 1. 55

Figura 32 – Atividade 1: Exercício 1. 55

Figura 33 – Atividade 1: Vídeo tutorial do Exercício 1. 56

Figura 34 – Tela do Exercício 1. 57

Figura 35 – Bloco RELEMBRAR EXERCÍCIO expandido. 57

Figura 36 – Tela de erro. 58

Figura 37 – Tela de sucesso. 59

Figura 38 – Atividade 1: Exercício 2. 60

Figura 39 – Tela do Exercício 2. 60

Figura 40 – Atividade 1: Exercício 3. 61

Figura 41 – Tela do Exercício 3. 61

Figura 42 - Fluxograma da Atividade 2. 62

Figura 43 - Atividade 2. 63

Figura 44 - Atividade 2: Exercício 1. 63

Figura 45 – Tela do Exercício 1. 64

Figura 46 – Os comandos divididos em abas. 64

Figura 47 – Atividade 2: Exercício 2. 65

Figura 48 – Tela do Exercício 2. 65

Figura 49 – Atividade 2: Exercício 3. 66

Figura 50 - Tela Exercício 3. 67

Figura 51 - Fluxograma da Atividade 3. 68

Figura 52 - Atividade 3. 69

Figura 53 - Atividade 3: Exercício 1. 69

Figura 54 – Tela do Exercício 1. 70

Figura 55 – Atividade 3: Exercício 2. 70

Figura 56 – Tela do Exercício 2. 71

Figura 57 – Atividade 3: Exercício 3. 72

Figura 58 - Tela Exercício 3. 72

Figura 59 - Fluxograma da Atividade 4. 73

Figura 60 - Atividade 4. 74

Figura 61 - Atividade 4: Exercício 1, o exercício final. 74

Figura 62 – Tela do Exercício 1, o exercício final. 75

Figura 63 – Tela Final. 75

**LISTA DE TABELAS**

[Tabela 1 – Geração de Computadores. 21](#_Toc441832649)

SUMáRIO

1 Introdução 1

1.1 Problematização e Justificativa 3

1.2 Objetivos 4

2 Fundamentação Teórica 6

2.1 A educação e o pensamento computacional 6

2.2 O que são objetos de aprendizagem 19

2.3 O Computador e a Analogia 20

3 métodos e materiais 36

3.1 Métodos 36

3.2 Materiais 36

4 Desenvolvimento 38

4.1 Analogia 38

4.2 O Objeto de Aprendizagem e suas Atividades 48

5 conclusão 77

5.1 Trabalhos futuros 78

Referências 79

# 1 Introdução

O computador é uma ferramenta indispensável para a sociedade moderna nas mais diversas instâncias, desde a medicina, agricultura e até à aviação, inclusive na área educacional, em todas as etapas do sistema de ensino para alunos de todas as idades. Porém, nos primórdios de sua história, não se podia prever o quanto o computador estaria presente no nosso dia-a-dia, mesmo sendo uma tecnologia relativamente recente, se comparada a outras, já que a Segunda Guerra Mundial é considerada seu marco inicial, época na qual foram construídos os primeiros computadores digitais, revolucionando o mundo tecnológico (FONSECA FILHO, 2007).

Inicialmente, os computadores eram máquinas gigantescas, chegando a pesar toneladas, e seu propósito era resolver operações matemáticas complexas, a fim de atender as expectativas de grandes empresas e instituições públicas e privadas. Porém, a partir de 1975, aproximadamente, a história do computador foi revolucionada com a disseminação dos circuitos integrados e a chegada do PC (*Personal Computer —* Computador Pessoal), que foram possíveis através da diminuição dos custos e a facilidade de acesso, rompendo uma barreira com relação ao público-alvo do computador, transformando-o em um bem de consumo pessoal e estimulando a produção em massa e a pesquisa para que seu uso fosse estendido para as mais diversas áreas (FONSECA FILHO, 2007).

Com o passar dos anos o computador se tornou um importante aliado na educação, principalmente em conjunto com a popularização da internet, pois trouxe para a sociedade novas possibilidades de comunicar, produzir e difundir informação. Agora, o computador deixara de ser apenas uma ferramenta para realizar tarefas específicas para, juntamente com a internet, se transformar em uma grande fonte de disseminação de conhecimento e compartilhamento de informações.

Como resultado dessa grande transformação tecnológica, os alunos que ingressam atualmente nas universidades são conectados tecnologicamente desde a infância. Porém, os educadores que lecionam nessas mesmas instituições de ensino não tiveram, em sua maioria, esse mesmo contato com a tecnologia desde cedo na vida, e muitas vezes encontram dificuldades em adaptar-se às rápidas evoluções tecnológicas. Acontece então, dentro da sala de aula, o encontro entre gerações distintas: alunos, chamados Nativos Digitais, e professores, chamados Imigrantes Digitais. Essa diferença de gerações pode trazer uma dificuldade na comunicação e no ensino, uma diferença entre a mensagem passada e a mensagem recebida, já que a linguagem usada pelos indivíduos Nativos Digitais inclui a linguagem tecnológica, com a qual os Imigrantes Digitais não estão completamente familiarizados ou possuem certas dificuldades.

Além da dificuldade de interação causada pela diferença geracional existente entre alunos e educadores, há também uma dificuldade de compreensão por parte dos alunos em cursos relacionados, especificamente, à computação, principalmente nos conceitos mais abstratos da programação, como em algoritmos e lógica computacional.

Como uma das soluções possíveis para este problema, tem-se o pensamento computacional, conceito que tem como base a decomposição, abstração, reconhecimento de padrões e algoritmo, como uma forma de construir um raciocínio lógico para solucionar um problema por um caminho eficaz (BITESIZE; [201-], tradução nossa). A prática do pensamento computacional, segundo Sica (2011), deveria ser incluída desde o ensino básico, para que o pensamento lógico do indivíduo seja desenvolvido juntamente com as outras habilidades ensinadas às crianças. Quanto aos alunos ingressantes dos cursos na área de computação, o ensino do pensamento computacional num primeiro contato com a faculdade poderia desmistificar certas disciplinas e facilitar o entendimento do aluno em relação à programação durante todo o curso, formando profissionais mais capacitados.

Com isso, cada vez mais, surgem pesquisas e discussões envolvendo ferramentas que possam auxiliar a comunicação entre educadores e alunos, quebrando essa barreira de gerações e gerando uma melhor compreensão dos conceitos. Um exemplo são os Objetos de Aprendizagem, que são ferramentas que visam o auxílio da passagem de conhecimento dentro e fora da sala de aula.

Portanto, este projeto visa a criação de um Objeto de Aprendizagem específico para apoio ao ensino de conceitos de lógica computacional e algoritmos. A partir de pesquisas realizadas na área de computação e educação, este Objeto de Aprendizagem foi implementado com o objetivo de facilitar a interação entre professores e alunos ao redor de assuntos conhecidos como muito complicados ou demasiado técnicos, transformando-os, através de analogias, em conceitos palpáveis e entendíveis, principalmente no momento de primeiro contato com a área.

## 1.1 Problematização e Justificativa

“Nossos alunos mudaram radicalmente. Os alunos de hoje não são os mesmos para os quais o nosso sistema educacional foi criado” (PRENSKY, 2001, p. 01).

É evidente que o sistema educativo enfrenta problemas diretamente ligados à discrepância do contato com a tecnologia entre alunos, indivíduos nascidos na era digital atual, totalmente tecnológica, e professores, que vêm das eras pré-digitais e não estão habituados a lidar com as tecnologias existentes ou têm dificuldade em acompanhar a velocidade do movimento de atualização tecnológica, em sua maioria. Esse conflito de geração reflete a necessidade de uma atualização do sistema educacional vigente, que não segue a velocidade do desenvolvimento tecnológico e, além disso, mostra-se desatualizado em diversas instâncias, uma vez que este foi criado em uma realidade completamente diferente da que temos hoje em dia e, em sua maioria, não sofreu os ajustes necessários para se adequar a ela. Logo, a linguagem dos educadores na sala de aula se torna estranha em relação a linguagem digital em que nasceram os alunos desta nova geração (GECK, 2006; PRENSKY, 2001).

Como consequência, os alunos possuem uma certa dificuldade em resolver problemas genéricos e conceitos mais abstratos, os quais exigiriam um raciocínio lógico que não é componente do currículo atualmente adotado no sistema educacional brasileiro. Para que este problema fosse minimizado, seria necessário que os conceitos do pensamento computacional estivessem presentes na grade escolar dos alunos desde a educação básica. Atualmente, a escola ensina a criança a solucionar problemas através de caminhos pré-determinados, como no ensino da leitura, escrita ou matemática, ao invés de ensiná-las que uma solução pode ser encontrada por diversos caminhos, que podem ser descobertos ou criados por elas mesmas. Este tipo de raciocínio seria o pensamento mais aberto necessário para facilitar o desenvolvimento do raciocínio lógico, que colaboraria não só com o ensino da lógica computacional mas também com a solução de outros problemas da vida adulta, que não vêm com fórmulas ou tabelas (BLIKSTEIN, 2008; HINTERHOLZ; CRUZ, 2015).

Uma das consequências a longo prazo deste problema mais pertinente a este projeto é a evasão em cursos de ensino superior ligados à tecnologia. No âmbito mais específico da computação, de acordo com Giraffa e Mora (2013), existe uma grande evasão nos cursos de computação, principalmente nas disciplinas iniciais, que vem sendo estudada por diversos pesquisadores, uma vez que é uma grande perda, tanto para o sistema educacional quanto para o aluno.

Segundo Chella, Oliveira e Silva (2014), a desistência de estudantes nesses casos é uma consequência do grande indíce de reprovação existente em disciplinas relacionadas, principalmente, à programação. Uma das maiores dificuldades desses alunos na programação é o aprendizado de conceitos e habilidades que são tidas como pré-requisitos quando o objetivo é aprender algoritmo, o qual requer a capacidade de raciocínio lógico e de resolução de problemas genéricos, que, como mostrado anteriormente, não é um problema restrito a computação (GIRAFFA; MORA, 2013).

Este projeto se justifica na busca de uma ferramenta de fácil acesso e fácil entendimento que mostre que a lógica computacional pode ser ensinada de forma prática e simples, de maneira a implementar o pensamento lógico desde cedo na educação, seja em fases de aprendizado inicial ou em ambiente universitário onde os indivíduos estão entrando em contato com os conceitos da computação pela primeira vez. Desta maneira, busca-se diminuir a evasão dos cursos da área assim como a simplificação dos conceitos base para que os alunos possam aplicá-los futuramente com mais facilidade nas disciplinas mais complexas que exijam o conhecimento sólido de tais conceitos.

## 1.2 Objetivos

O objetivo geral desse projeto é arquitetar e implementar um Objeto de Aprendizagem de apoio ao professor no ensino da lógica computacional e conceitos de algoritmo, por meio de analogias.

Em conjunto com este objetivo geral, tem-se os objetivos específicos deste trabalho. Primeiramente foi realizado um levantamento bibliográfico sobre todos os conceitos que seriam utilizados ao longo do processo e, com isso, foi feito um levantamento sobre a problematização existente na educação básica e também na educação superior, mais especificamente em cursos de computação. E, então, o Objeto de Aprendizagem foi esquematizado e implementado nas linguagens de programações que foram previamente escolhidas após um levantamento das possíveis plataformas para este projeto.

# 2 Fundamentação Teórica

Para aprofundar e entender o estudo que dará a base ao Objeto de Aprendizagem de apoio ao professor em disciplinas e conceitos básicos de algoritmo e lógica computacional que será produto deste trabalho, é preciso compreender melhor quais são os públicos envolvidos na educação atual e quais seus papéis como protagonistas do processo de aprendizagem.

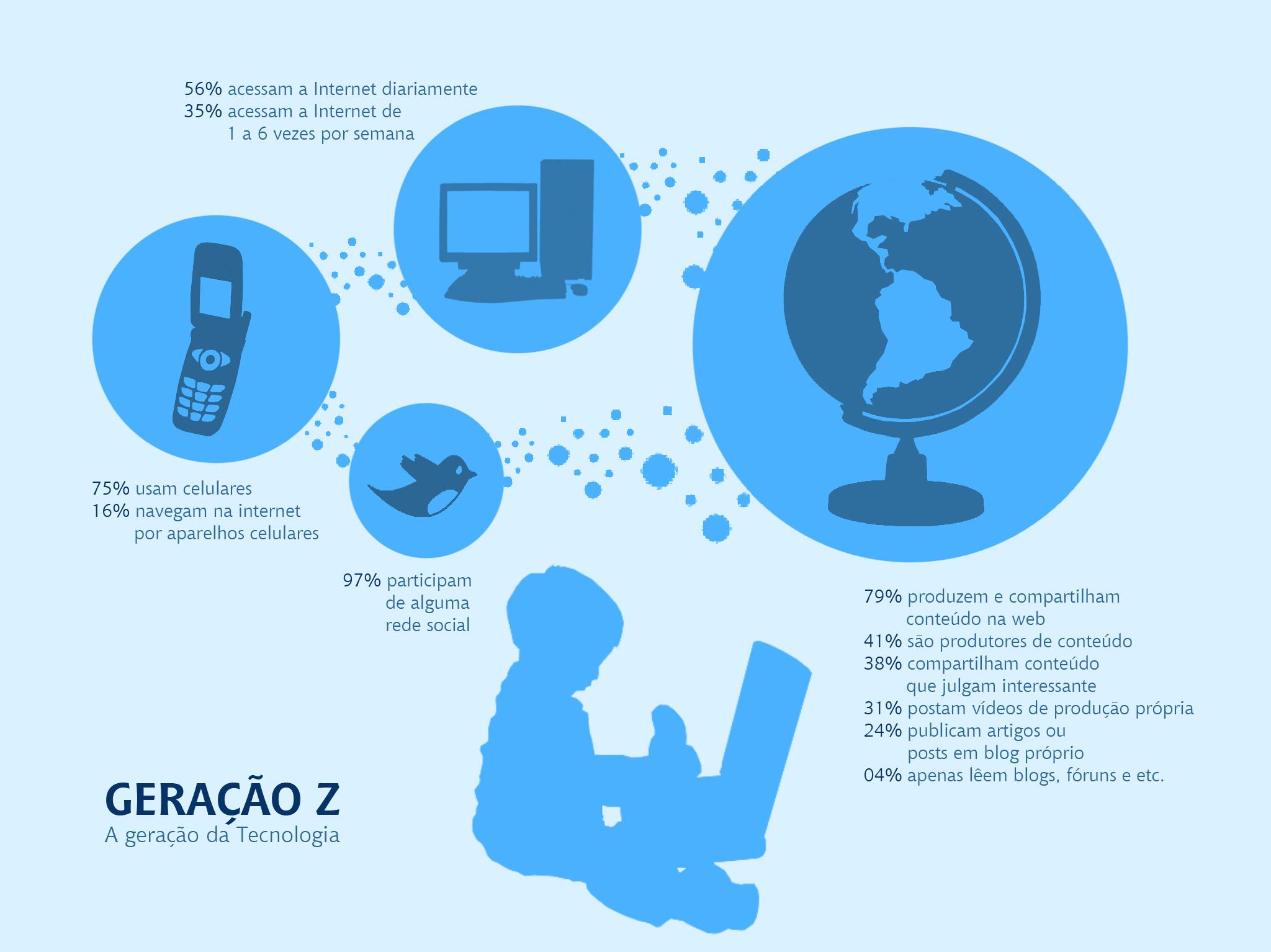
## 2.1 A educação e o pensamento computacional

Os alunos de hoje em dia não são os mesmos para os quais o sistema educacional atual foi criado. Esses alunos, frutos da chegada e da rápida difusão da tecnologia digital no final do século XX e começo do século XXI, não apenas mudaram e continuam mudando seus hábitos em maior velocidade, houve também uma descontinuidade, ou seja, as coisas estão tão mudadas que não há volta, segundo Prensky (2001). Com isso, tem-se um conflito no âmbito educacional atual entre duas gerações distintas: os alunos e os educadores.

Segundo Prensky (2001), as mudanças pelas quais esses alunos foram submetidos foram muitas, tanto em termos de estilos, gírias, roupas, etc, como em relação à velocidade dos avanços tecnólogicos, se comparados à épocas passadas. Geck (2006) diz que a geração dos alunos atuais, ou geração Z, são “falantes nativos” da linguagem digital dos computadores, vídeo games e internet, sendo composta por indivíduos nascidos a partir de 1990, conhecidos como Nativos Digitais. O surgimento dessa geração coincidiu com o nascimento da Web mais gráfica, que é o resultado de grandes avanços na área, principalmente em relação à interface com o usuário, muito parecida com a que temos atualmente, gerando condições únicas para o crescimento e desenvolvimento desses indivíduos. Como produto dessa era tecnológica, esses indivíduos são os mais conectados desde a criação dos componentes eletrônicos, e desde crianças estão em constante contato com os navegadores Web gráficos, notebooks, celulares, videogames, banda larga, entre outros, trabalhando também a capacidade de realizar mais de uma tarefa ao mesmo tempo. Com isso, os estudantes dessa geração estão acostumados a sempre estar conectados a algum dispositivo eletrônico, seja ele um computador, celular, tablet ou outros, realizando tarefas desde trabalhos escolares até escutar música ou trocar mensagens instantânea com os amigos, utilizando diversos aplicativos para tais atividades.

A Figura 1, criada pela agência Vogg Branded Content, representa um estudo realizado pela empresa de pesquisa de mercado “Quest Inteligência de Mercado”, que apresenta alguns dados sobre a geração Z ligados ao comportamento e à conexão que essa geração demonstra em relação às tecnologias atuais.

Figura 1 - Infográfico sobre a geração Z.

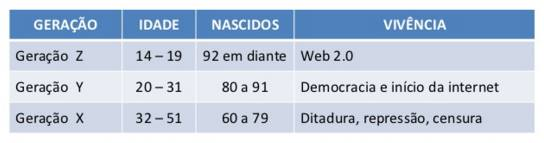


Fonte: cia de resultados[[1]](#footnote-1)

Os docentes que lecionam a geração atual são indivíduos das gerações anteriores, principalmente da geração X. Esta geração, descendente dos baby-boomers, compreende os indivíduos nascidos entre cerca de 1960 à 1079. Segundo Maldonado (2009), “As pessoas da geração X tendem a ser individualistas, irreverentes, autoconfiantes; valorizam mais a lealdade a si mesmas, já que a aspiração de conseguir um emprego por toda a vida deixou de existir”. Esta geração cresceu acompanhando o desenvolvimento tecnológico com a popularização dos meios de comunicação em massa e a exigência de maior independência do indivíduo, porém também enfrentou as repressões da ditadura e a instabilidade econômica do país. Na época do nascimento da geração Y, a geração de transição, quando começaram as maiores revoluções tecnológicas e o avanço acelerado das inovações nesse campo, os indivíduos da geração X já eram adultos formados, o que dificultou sua adaptação para se inserir no meio tecnológico e entender tanto a linguagem quanto os hábitos que acompanharam essa transição.

O conflito geracional, apesar de existente, é menor quando os professores são indivíduos da geração Y, que são os nascidos entre 1980 e 1991, como mostra a Figura 2, o que os torna imigrantes no ciberespaço, e “vivem os dilemas e desafios de um tempo de transição. Eles foram formados na cultura oralista e presencial, acostumados a olhar o outro e interagir no mesmo meio físico de forma síncrona” (MARTINS; GIRAFFA, 2009). Ou seja, ainda Martins e Giraffa (2009), essa geração nasceu em um outro meio e aprendeu a construir o conhecimento de uma forma distinta da geração atual e, por isso, vive em um momento de transição e tenta se adaptar à essa nova realidade. Prensky (2001) diz que por serem imigrantes, esses educadores aprendem a se adaptar no novo ambiente, porém eles sempre mantêm seu “sotaque“ em certo grau, ou seja, seu pé no passado.

Figura 2 – Descrição geral das gerações



Fonte: Quest Inteligência de Mercado[[2]](#footnote-2)

Ainda segundo Prensky (2001), na prática, a geração Z está acostumada a receber informações muito rapidamente, além de processar mais de uma tarefa por vez e realizar multitarefas. Os alunos de hoje em dia preferem primeiramente os gráficos e depois o texto, trabalham melhor quando ligados a uma rede de contatos e têm sucesso com gratificações instantâneas e recompensas frequentes. Em contrapartida, os professores de outras gerações têm dificuldade em se adaptar a essas habilidades adquiridas e aperfeiçoadas desde a infância pela geração atual, já que o processo de aprendizagem no sistema de ensino de sua época era mais lento e não utilizava dos artifícios disponíveis nos dias de hoje. Muitos professores consideram tais aparelhos como uma distração prejudicial à concentração e não acreditam que seus alunos possam aprender o conteúdo com êxito ao mesmo tempo que assistem televisão ou escutam música, por exemplo.

Por isso, um dos maiores problemas na metodologia da educação atual é o conflito entre essas gerações, os Nativos e os Imigrantes Digitais (nascidos antes da explosão das tecnologias), que acontece já que essas gerações trabalham de maneiras bem diferentes em relação ao comportamento. Muitas vezes, os filhos da era digital, que atualmente já estão ingressando nas faculdades, se deparam com uma linguagem ultrapassada dos educadores (da era pré-digital), acarretando em uma desconexão entre a mensagem a ser passada e a mensagem a ser recebida, segundo o mesmo autor. Levando este problema em consideração, Martins e Giraffa (2009) acreditam que uma das saídas viáveis para os docentes seria explorar o desenvolvimento de certas competências e habilidades levando em consideração aspectos técnicos e pedagógicos com o objetivo de conseguir quebrar essa barreira entre gerações.

Contudo, existe uma parcela da geração Z que enfrenta mais uma dificuldade durante o processo de sua formação. Santos e Costa (2006) afirmam que, atualmente, os estudantes que optam por cursos de Computação e Informática se deparam com uma barreira logo no ínicio da sua formação acadêmica, pois sentem uma grande dificuldade durante o primeiro contato com o estudo de algoritmo, lógica computacional e, consequentemente, a programação.

Antes de entender claramente a origem e a razão dessa dificuldade que os docentes sentem de conduzir o aluno à entender conceitos mais abstratos em aulas de conteúdos introdutórios da computação, é necessário definir alguns conceitos que envolvem a problematização citada anteriormente. Segundo Medina e Fertig (2005), apesar do termo algoritmo ser comumente associado à computação, o conceito deste termo não é restrito somente a ela, podendo também ser utilizado em outras áreas, como engenharia, administração, entre outras. Ainda segundo os autores, a origem da palavra algoritmo vem do nome do matemático iraniano Abu Abdullah Mohammad Ibn Musa al-Khawarizmi, nascido em Khawarizm (Kheva) no século XVII, ou seja, a denominação desse conceito aconteceu antes mesmo do primeiro computador eletrônico ser criado.

Antes de definir o conceito de algoritmo é necessário entender o conceito de ação, que é “um acontecimento que, a partir de um estado inicial, após um perídodo de tempo finito, produz um estado final previsível e bem definido” (FARRER et al., 1989, p. 15). Assim, um algoritmo nada mais é do que “a descrição de um conjunto de comandos que, obedecidos, resultam numa sucessão finita de ações” (FARRER et al., 1989, p. 15).

Já, Medina e Fertig (2005, p. 13) definem o termo algoritmo, de um outro ponto de vista, como “um procedimento passo a passo para a solução de um problema” ou também “uma sequência detalhada de ações a serem executadas para realizar alguma tarefa”. Os autores ainda dão como exemplo algumas aplicações existentes fora da computação, como as ações necessárias para se fazer um balancete ou uma receita culinária, mostrando que, realmente, o conceito de algoritmo não é útil apenas na computação.

Dentro do entendimento de algoritmo, tem-se a divisão em algumas categorias e segundo, Ascencio e Campos (2008), as três principais são:

* descrição narrativa é, após analisar o enunciado, escrever os passos a serem seguidos na linguagem natural, por exemplo, a língua portuguesa. A vantagem é que o aluno não necessita aprender nenhum conceito novo, pois a linguagem natural já é a nativa. Porém, esta categoria abre brechas para diferentes compreensões, o que pode causar dificuldades no momento em que o aluno for transformar o algoritmo em programa.
* fluxograma é, após analisar o enunciado, simbolizar o algoritmo através de gráficos predefinidos. A vantagem é que, por ser uma abordagem visual, o entendimento é mais natural. Entretanto, é necessário o aprendizado da simbologia dos fluxogramas e, além disso, esta metodologia não possui muitos detalhes, o que também dificulta na transformação do algoritmo em programa.
* pseudocódigo ou portugol consiste em, após analisar o enunciado, escrever o algoritmo por meio de regras predefinidas por essas categorias. A vantagem é que, ao contrário das outras duas modalidades apresentadas, a passagem de algoritmo para programa é quase que instântanea, pois só é necessário reconhecer os comandos da linguagem que serão utilizados. Porém, é necessário aprender as regras do pseudocódigo, para manter uma padronagem no algoritmo e um melhor entendimento.

Além do foco no aprendizado de algoritmo, há também o conceito de lógica. Segundo Feitosa e Paulovich (2005), a lógica surgiu na antiguidade como uma ciência, tendo como objetivo a análise do raciocínio. O ensino de lógica matemática faz parte da base de qualquer curso de computação, pois é essencial no aprendizado de programação. Souza (2015) afirma que “uma das razões para se estudar Lógica é que ela nos confere a capacidade de análise crítica dos argumentos mentais utilizados na organização das ideias e dos processos criativos.”

Ambos os conceitos de lógica e algoritmo são base no aprendizado de programação, porém não somente. Para Santos e Costa (2006), a lógica e o algoritmo estão completamente ligados, já que o ensino interligado destes conceitos formam dois dos pilares das ciências exatas. Isso mostra que a lógica e o algorítmo são úteis além dos limites da computação, essenciais para despertar o raciocínio matemático-lógico, que é usado na resolução de problemas científicos porém também em qualquer outro problema genérico, ou seja, essenciais na criação e desenvolvimento do raciocínio lógico usado em diversas instâncias da vida do indivíduo.

O conhecimento de algoritmos e lógica computacional é um requisito quando o objetivo é programar. Santos e Costa (2006) definem a programação como sendo “um conjunto de princípios, técnicas e formalismos que visam o desenvolvimento de produtos de software confiáveis bem estruturados”, sendo necessário o conhecimento prévio de conceitos como os princípios da abstração e do encapsulamento e as técnicas de modularização e programação estruturada. Portanto o estudo de programação não se restringe somente ao estudo de linguagens de programação, mas também a todos os conceitos abstratos ligados à elas e se realizado de maneira efetiva, pode resultar em indivíduos mais esclarecidos e, portanto, produtos de melhor qualidade.

Juntamente com o conflito existente entre as gerações na educação atual, existe também uma crescente dificuldade dos alunos ao entrarem em contato pela primeira vez com conceitos de algoritmo e lógica computacional durante a faculdade. De acordo com Gomes et al. (2008) são várias as razões responsáveis pela dificuldade no aprendizado de programação e reunem alguns conjuntos de fatores que, segundo os autores, interferem nesse conhecimento. São eles:

* Os métodos de ensino que geralmente são utilizados pelos educadores não atendem por completo a necessidade dos alunos. Um dos motivos é que o ensino não é personalizado, ou seja, o aluno não recebe uma supervisão adequada à sua necessidade individual e, geralmente, as estratégias usadas pelos educadores não atendem aos estilos de aprendizagem de cada aluno, forçando-os a uma aprendizagem uniforme, desconsiderando o fato de que cada indivíduo tem seu próprio ritmo para aprender. No âmbito da programação, muitas vezes o educador tende a desvalorizar os conceitos base e a focar mais no estudo de certa linguagem específica, ao invés de usá-la como uma ferramenta para solução de problemas, o que não é facilitado quando um conceito dinâmico, como a programação, é ensinado através de métodos estáticos, como explicações verbais, diagramas, textos e assim por diante.
* Os métodos de estudo que muitos alunos estão acostumados a utilizar muitas vezes não são apropriados para o aprendizado de programação. A programação, diferentemente de outros conteúdos, necessita de um estudo prático e intensivo, em outras palavras, é programando que se aprende a programar. Além disso, os alunos não estudam o suficiente, pois a programação exige muito mais do que assistir uma aula ou ler um livro, são necessários muito treino e reflexões sobre os conceitos aprendidos.
* Algumas habilidades e atitudes dos alunos também atrapalham o aprendizado de programação. Um grande exemplo disso é a dificuldade de muitos em resolver problemas genéricos, como quando o objetivo é produzir um algoritmo e, logo, programar. Além disso, a deficiência em conhecimentos matemáticos e lógicos e a falta do dominínio sobre o funcionamento completo de estruturas básicas na programação também atrapalham o aprendizado em questão.
* Algumas características da natureza da programação também geram dificuldades durante o ensino. Essa aprendizagem requer algumas competências por parte do aluno, como um elevado nível de abstração, generalização, transferência e pensamento crítico. O problema surge pois o aluno tem dificuldade em aplicar conceitos abstratos para solucionar problemas reais. Além da abstração, existe uma grande dificuldade no aprendizado de linguagens de programação, já que a sintaxe é muito complexa, ou seja, quando o estudante está programando, ele precisa se concentrar simultaneamente no algoritmo em si e também nas regras de sintaxe da linguagem que está sendo implementada.
* E, por fim, alguns aspectos psicológicos tendem a influenciar negativamente o estudante que visa aprender a programar, como, por exemplo, a falta de motivação para estudar e aprender, muitas vezes gerada pela conotação negativa que é dada à programação ou pelo fato de ter a popularidade de ser uma atividade difícil. Além disso, geralmente a programação é ensinada logo no início do curso superior, que é uma época de transição entre escola e faculdade, e muito instável para os alunos, dificultando ainda mais o aprendizado.

A deficiência desse aprendizado dentro do ensino superior não só se reflete no estudo de programação, mas também, segundo Chella, Oliveira e Silva (2014), na evasão desses alunos em cursos relacionados à computação, muitas vezes devido a um grande índice de reprovação em disciplinas de programação, o que afeta diretamente o sistema educacional atual por não gerar um retorno ao mesmo. Para Rapkiewicz et al. (2006), essas disciplinas são consideradas gargalos em cursos de computação, pois com altos índices de reprovação e evasão, acabam dificultando ou até mesmo impedindo a continuidade dos alunos em cursos de computação, particularmente.

De acordo com Giraffa e Mora (2013), a perda de alunos durante a graduação é um problema muito grave para o sistema educacional, pois um aluno que inicia mas não termina um curso é considerado um desperdício social, acadêmico e econômico, tanto no ensino privado, que significa uma perda de receitas, quanto no público, que significa recursos públicos investidos sem retorno. Além disso, o próprio aluno possui um alto investimento para fazer um curso superior, não só financeiro como também emocional, pessoal e social, ou seja, a evasão é um problema para a educação, para a sociedade, para as instituições e para o indivíduo.

Ainda de acordo com as autoras, a desistência que acontece nos cursos relacionados à Ciência da Computação acontece principalmente no primeiro ano de curso. Esse fato também é colocado em vista por Gomes et al. (2008), que afirmam que o problema começa logo no início do aprendizado, pois o aluno tem dificuldade na compreensão de alguns conceitos básicos e abstratos na programação, como em estruturas de dados ou até em criar algoritmos que resolvam problemas concretos. Portanto é necessário focar na melhoria de competências que deveriam ter sido adquiridas logo no começo do curso, como a capacidade de resolução de problemas, de raciocínio, de lógica, entre outras, e não apenas no desenvolvimento de habilidades específicas de programação, como o aprendizado de linguagens.

Então, para Rapkiewicz et al. (2006), é necessário desenvolver o raciocínio lógico do aluno, pois sem ele o aluno não consegue entender a lógica de um problema, logo, não consegue solucioná-lo através de uma sequência lógica ou algoritmo. Gomes et al. (2008, p.164) complementa que as dificuldades que alunos de ensino superior “[...] sentem ao programar prendem-se com a incapacidade de conceber algoritmos, e estamos convictos que estas se devem principalmente à incapacidade de resolver problemas”, Sica (2011) traz como solução que se o pensamento computacional, ou *computational thinking*, e a lógica fossem ensinados no ensino básico, a capacidade do aluno de dedução e conclusão de problemas aumentaria consideravelmente:

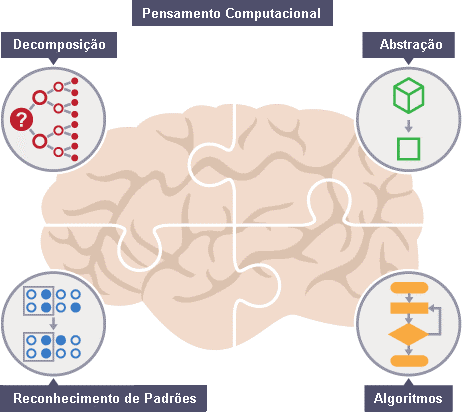
* É importante salientar que devemos primar pela qualidade do ensino em todos os níveis da cadeia de formação de recursos humanos. Entendemos que a Computação deva ser ensinada desde o ensino fundamental, a exemplo de outras ciências como Física, Matemática, Química e Biologia. Esses são pontos muito importantes para que no futuro tenhamos recursos humanos qualificados para enfrentar os desafios que advirão. (SBC, link).

E diferentemente do que se pode pensar, o pensamento computacional é um conjunto de habilidades que não se restringe aos profissionais da área de computação, mas que é essencial para todos, já que são competências que podem ser aplicadas em diferentes ambientes, não só em conceitos de Computação (WING, 2006, tradução nossa). Ainda segundo a autora, o pensamento computacional é uma competência fundamental na formação de uma criança e deveria ser ensinada no ensino básico, juntamente com o aprendizado de leitura, escrita e aritmética. Ela ainda define que esse conceito possui algumas características:

* Conceituar, não programar. O Pensamento computacional requer uma capacidade de pensar em diferentes níveis de abstração, ou seja, ser capaz de transformar problemas aparentemente impossíveis em problemas simples e possíveis de resolver.
* Habilidade fundamental, não mecânica. O pensamento computacional é uma habilidade fundamental e não mecânica ou utilitária, que todo indivíduo deveria saber e todo estudante deveria desenvolver.
* Uma maneira na qual pessoas, não computadores, pensam. Pensamento computacional é uma maneira de pessoas serem capazes de resolver problemas e não de pensar como um computador, que deve ser usado como uma ferramenta à imaginação e não para substitui-lá.
* Complementa e combina a matemática e a engenharia*.* A matemática e a engenharia são inerentes à computação, ou seja, a computação se apoia nos conceitos de pensamento matemático e de engenharia, reconhecendo e trazendo suas particularidades para compor seu conjunto de habilidades.
* Ideias, não artefatos. A computação não deve necessariamente ter como produto um artefato como um hardware ou software, pois utilizando conceitos fundamentais da computação, o pensamento computacional conduz o indivíduo a lidar e resolver problemas do cotidiano.
* Para todos, em qualquer lugar. O pensamento computacional será uma realidade quando ele estiver intrínseco à sociedade, sendo parte do dia-a-dia de cada indivíduo e utilizado em diversas aplicações.

O Bitesize ([201-], tradução nossa), que é um recurso de suporte ao estudo para estudantes da educação básica do Reino Unido produzido pela BBC (acrônimo para *British Broadcasting Corporation* — Corporação Britânica de Radiodifusão), define que “o pensamento computacional possibilita que indivíduo, ao se deparar com um problema complexo, seja capaz de entender o problema e desenvolver possíveis soluções para o mesmo”, e, ainda, divide esse conceito em quatro pilares principais: decomposição, reconhecimento de padrões, abstração e algoritmos, como pode-se observar na Figura 3:

Figura 3 - O pensamento computacional.



Fonte: Adaptado de BBC[[3]](#footnote-3)

* Decomposição: fragmentar um sistema ou problema muito complexo em partes menores e mais gerenciáveis.
* Reconhecimento de padrões: procurar por semelhanças entre problemas distintos.
* Abstração: focar apenas em informações de extrema importância, ignorando os detalhes que forem irrelevantes.
* Algoritmos: desenvolver uma solução passo-a-passo para o problema ou desenvolver as regras a serem seguidas para alcançar a resolução do problema.

Segundo a Sociedade Internacional de Tecnologia em Educação (ISTE — *International Society for Technology in Education*) (2011, tradução nossa) juntamente com a Associação de Professores de Ciência da Computação (CSTA — *Computer Science Teachers Association*) (2011, tradução nossa) o pensamento computacional é um processo para a resolução de problemas que inclui algumas características:

* Formalizar problemas de um modo que nos permita usar o computador e outras ferramentas para resolvê-los.
* Organizar logicamente os dados e analisá-los.
* Representar os dados por meio de abstrações como modelos e simulações.
* Automatizar soluções através do pensamento algorítmico (uma série de passos ordenados).
* Identificar, analisar e implementar soluções possíveis com a meta de realizar a combinação mais eficiente e eficaz de passos e recursos.
* Generalizar e transferir este processo de solução de problema a uma grande variedade de problemas.

Ainda segundo a ISTE e CSTA (2011, tradução nossa), com o aprendizado do pensamento computacional o aluno desenvolve algumas disposições e atitudes realçadas pelas características citadas anteriormente, sendo elas:

* Confiança ao lidar com complexidade.
* Persistência ao trabalhar com problemas difíceis.
* Tolerância com a ambiguidade.
* Capacidade de lidar com problemas em aberto.
* Capacidade de se comunicar e trabalhar em equipe para atingir uma meta em comum ou uma solução.

Portanto, diferentemente do que muitos podem pensar, o pensamento computacional não se trata de saber navegar na internet, enviar e-mails ou usar um editor de texto (BLINKSTEIN, 2008). Essas habilidades, inclusive, já são altamente desenvolvidas nos indivíduos da geração atual. Ainda segundo o autor, o que é feito atualmente, ao invés do aprendizado do pensamento computacional, é um “adestramento digital”, no qual o aluno aprende que a tecnologia deve ser utilizada para reorganizar informações distintas já existentes, e não criar novos conhecimentos que, como citado anteriormente, seria o ideal.

## 2.2 O que são objetos de aprendizagem

Levando esses estudos e definições em consideração, pode-se dizer que os métodos de ensino tradicionais já não são tão eficazes se implementados na era digital. Por isso surgem, cada vez mais, pesquisas e discussões sobre ferramentas que possam ser usadas para trazer a tecnologia como aliada ao ensino dentro e fora da sala de aula, tanto para o professor quanto para o aluno.

Atualmente, “com o avanço das tecnologias da informação e comunicação (TIC), bem como do crescimento do uso da Internet para fins educativos, surge um novo conceito de recursos didáticos: os Objetos de Aprendizagem” (SILVA; CAFÉ; CATAPAN, 2010, p. 94). Com isso, segundo uma pesquisa realizada pelo Centro de Estudos sobre as Tecnologias da Informação e da Comunicação CETIC (2012 apud BRAGA, 2014, p. 23) o uso dessas tecnologias tem crescido a cada ano no Brasil.

Pode-se definir um Objeto de Aprendizagem como:

* uma entidade, digital ou não-digital, que pode ser usada, reusada ou referenciada durante o ensino com suporte tecnológico. Exemplos de ensino com suporte tecnológico incluem sistemas de treinamento baseados no computador, ambientes de aprendizagem interativos, sistemas instrucionais auxiliados por computador, sistemas de ensino a distância e ambientes de aprendizagem colaborativa. Exemplos de objetos de aprendizagem incluem conteúdo multimídia, conteúdos instrucionais, objetivos de ensino, software instrucional e software em geral e pessoas, organizações ou eventos referenciados durante um ensino com suporte tecnológico. (IEEE, 2000 apud TAROUCO et al., 2006, p. 2).

Willey (2000 apud BRAGA, 2014, p. 21) também define os objetos de aprendizagem como, basicamente, qualquer recurso digital com a capacidade de ser reutilizado com o intuito de apoiar a aprendizagem. Para Hay e Knaack (2007 apud AMIEL; OREY; WEST, 2011), os objetos de aprendizagem são ferramentas web interativas de apoio ao aprendizado de conceitos específicos, incrementando, guiando ou ampliando o processo cognitivo de quem aprende. Ou seja, segundo Prata e Nascimento (2007), um Objeto de Aprendizagem tem como principal objetivo favorecer a interação entre o aluno e o professor na aprendizagem de um determinado conteúdo a ser ministrado, contribuindo para o processo de ensino e aprendizado no cotidiano.

Os Objetos de Aprendizagem podem estar presentes na sala de aula de diversas formas: como arquivos digitais, imagens, vídeos, referências a sites ou outros materiais digitais que possam ser usados como suporte dentro e fora da sala de aula, segundo Silva, Café e Catapan (2010). Para Tortosa (2006 apud SILVA; CAFÉ; CATAPAN, 2010, p. 95) essas ferramentas digitais são essenciais no processo de construção do material docente e espera-se que, com a fragmentação em unidades modulares independentes, esses objetos possam ser reutilizados em ambientes distintos. Presume-se que os Objetos de Aprendizagem possuam as seguintes características:

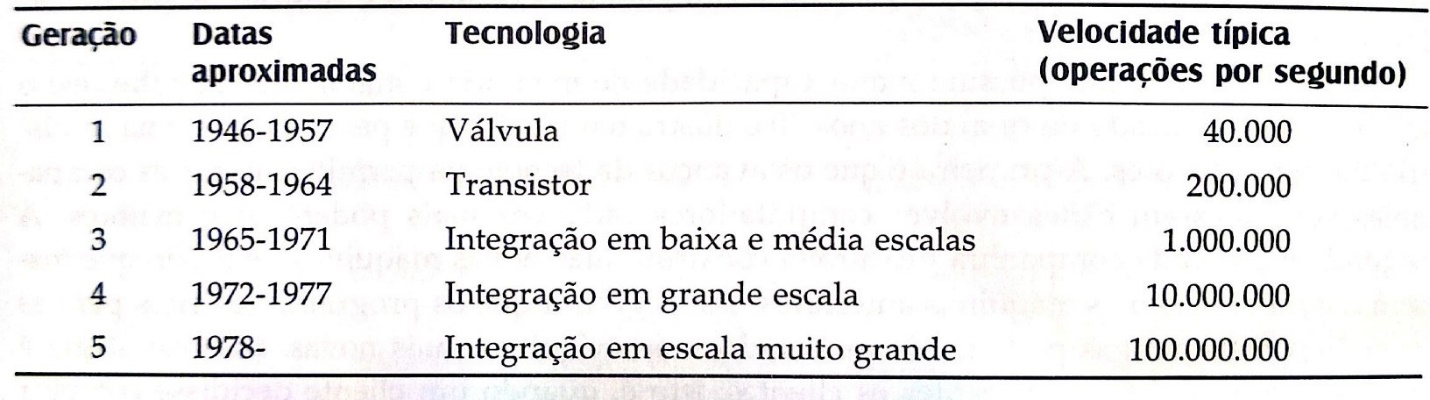
* **acessibilidade**: devem possuir uma identificação padronizada que garanta a sua recuperação;
* **reusabilidade**: devem ser desenvolvidos de forma a compor diversas unidades de aprendizagem;
* **interoperabilidade**: devem ser criados para serem operados em diferentes plataformas e sistemas;
* **portabilidade**: devem ser criados com a possibilidade de se mover e se abrigar em diferentes plataformas;
* **durabilidade**: devem permanecer intactos perante as atualizações de software e hardware. (SILVA; CAFÉ; CATAPAN, 2010, p. 96).

A finalidade dessas ferramentas de ensino, que são elaboradas com o objetivo de apoiar os educadores dentro e fora da sala de aula, é obter uma qualidade de ensino elevada, com cada vez mais contribuições, com o objetivo de aprimorar o ensino superior e aumentar a formação de profissionais melhor qualificados para o mercado, segundo Santos e Costa (2006). Paralelamente, espera-se que este Objeto de Aprendizagem rompa essa descontinuidade surgida entre a geração dos Nativos Digitais e os Imigrantes Digitais, criando um canal de comunicação mais eficiente entre eles.

## 2.3 O Computador e a Analogia

Para atingir o objetivo de fortalecer a base intelectual dos alunos ingressantes em cursos voltados para a programação com conceitos de lógica computacional e algoritmos, o Objeto de Aprendizagem em questão irá simular a estrutura básica de um computador. A história dos computadores, de acordo com Stallings (2002), é comumente dividida em gerações, como mostra a Tabela 1, mais especificamente em cinco gerações que são divididas segundo a tecnologia de hardware empregada e a cada evolução desses grupos de computadores com tecnologias base semelhantes nota-se maior velocidade, maior capacidade de memória e menor tamanho se comparados com suas gerações antecedentes.

Tabela 1 – Geração de Computadores.

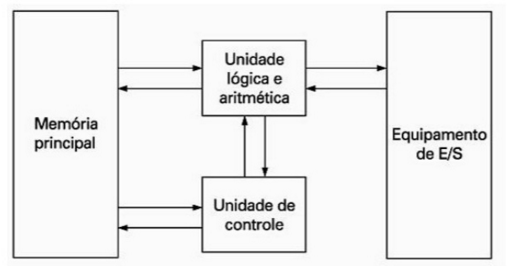
Fonte: STALLINGS, 2002, p. 21.

Historicamente, de acordo com Stallings (2002), em 1943 a primeira proposta de um computador eletrônico digital foi apresentada, aceita e iniciada em 1946. Esse projeto foi finalizado e entregue, dando início à primeira geração de computadores, que eram arquitetados com válvulas eletrônicas. O computador em questão foi batizado de ENIAC (Computador e Integrador Numérico Eletrônico — *Electronic Numerical Integrator and Computer*), teve como criadores John Mauchly e John Presper Eckert da Universidade de Pensilvânia. O ENIAC ocupava um espaço de aproximadamente 140 metros quadrados com suas mais de 18 mil válvulas, pesava aproximadamente 30 toneladas e sua operação consumia cerca de 140 quilowatts de energia elétrica, porém era muito mais veloz do que qualquer computador eletromecânico existente na época, já que conseguia executar 5 mil adições por segundo, aproximadamente. O autor diz que, inicialmente, ele foi projetado para ser utilizado durante a guerra pelo Laboratório de Pesquisas Balísticas do Exército americano (Army’s Ballistics Research Laboratory — BRL), no entanto, ao ser finalizado em 1946, já era tarde demais para ser usado durante a guerra e o BRL decidiu utilizá-lo para outros fins até 1955, ano em que o ENIAC foi desativado.

Como consequência ao fato do ENIAC ter a necessidade de carregar e modificar um programa, o que tornava a experência tediosa, surge um outro projeto, o EDVAC (Computador Variável Discreto Eletrônico — Electronic Discrete Variable Computer). O EDVAC foi proposto e formulado por um matemático chamado John von Neumann, que também era um dos consultores do projeto antecessor a este (STALLINGS, 2002). Segundo Murdocca e Heuring (2000) com essa nova proposta, von Neumann trouxe um novo conceito que veio em oposição aos computadores que armazenavam seus programas em memórias externas, o Programa Armazenado, que consiste em armazenar um programa na memória do computador, juntamente com os dados a serem processados pelo mesmo.

Posteriormente, em 1946, von Neumann juntamente com o Instituto de Estudos avançados de Princeton (IAS — Institute of Advanced Study), do qual era integrante, começou o projeto de mais um computador que foi batizado de IAS (Figura 4), assim como o Instituto, que acabou se tornando um protótipo para todos os computadores de propósitos geral subsequentes.

Figura 4 - Estrutura do computador IAS.



Fonte: STALLINGS, 2002, p. 20.

Cada componente da imagem consiste em:

* Uma memória principal, que armazena dados e instruções.
* Uma unidade lógica e aritmética (ULA), capaz de realizar operações com dados binários.
* Uma unidade de controle, que interpreta e executa instruções armazenadas na memória.
* Dispositivos de entrada e saída (E/S), operados pela unidade de controle. (STALLINGS, 2002, p. 20).

Atualmente, praticamente todos os projetos de computadores, com raras exceções, são fundamentados nos conceitos então desenvolvidos por John von Neumann no Instituto Avançado de Princeton e são conhecidos como Arquitetura von Neumann, que é baseada em três conceitos:

* Os dados e as instruções são armazenados em uma única memória de leitura e escrita.
* O conteúdo dessa memória é endereçado pela sua posição, independentemente do tipo de dados nela contidos.
* A execução de instruções ocorre de modo sequencial (exceto quando essa sequência é explicitamente alterada de uma instrução para a seguinte). (STALLINGS, 2002, p. 55)

Visando uma outra tecnologia base que, diferentemente das válvulas, ocupasse menos espaço, fosse mais barata e dissipasse menos calor, surgiu então, segundo Stallings (2002), a segunda geração de computadores, com a substituição das válvulas por transistores. Essa tecnologia foi inventada pela Bell Laboratories em 1947, revolucionando a indústria eletrônica da época, mas somente no final da década de 50 que os computadores totalmente compostos por transistores começaram a ser oficialmente comercializados. Além da mudança nos componentes eletrônicos, na segunda geração os computadores incluíam software de sistema e começaram a operar com linguagens de programação de alto nível, que “[...] são mais próximas da linguagem natural e guardam pouca similaridade com a linguagem da máquina em que serão executadas” (MEDINA; FERTIG, 2005, p.15).

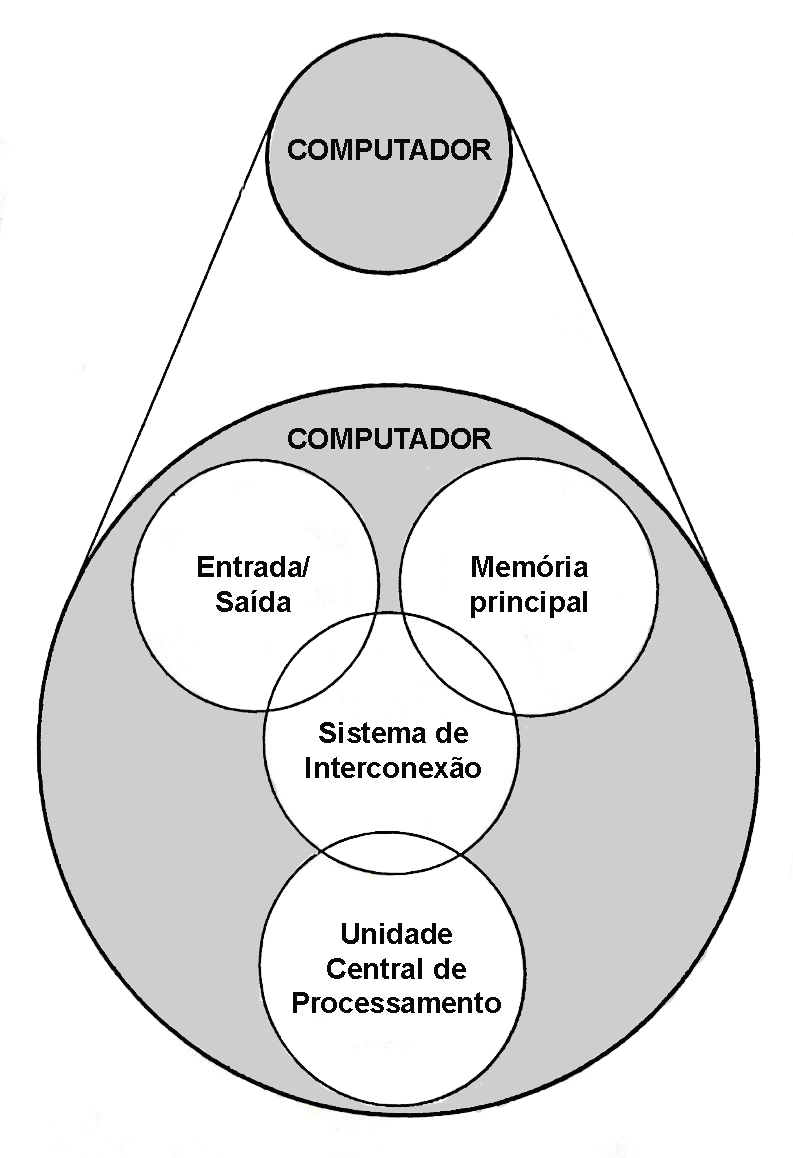
Entretanto o processo de fabricação dos dispositivos eletrônicos dessa geração, que eram composto por transistores que posteriormente seriam soldados à placas de circuito, era caro e incômodo. Depois de um certo tempo, esse tipo de fabricação começou a trazer problemas para a indústria tecnológica da época, já que cada vez mais os computadores exigiam mais transistores a fim de atender suas respectivas necessidades. Até que em 1958 uma nova técnica revolucionou a eletrônica com a invenção do circuito integrado, iniciando a era da microeletrônica, literalmente ‘eletrônica pequena’, e também a terceira geração dos computadores (STALLINGS, 2002).

Ainda segundo o autor, como o circuito integrado pode ser fabricado com base em um único semicondutor, por exemplo o silício, ao invés de um circuito ser formado por componentes feitos de peças de silício separadas, é possível produzir milhares de transistores, simultaneamente, utilizando somente uma placa de silício.

A partir da quarta e quinta gerações existe menos consenso sobre as definições que separam cada geração, mas elas podem ser distinguidas pela chegada, respectivamente, da integração em grande escala (large scale integration — LSI), na qual era possível colocar mais de mil componentes em somente uma pastilha, e da integração em escala muito grande (very-large-scale integration — VLSI), que atingia mais de 10 mil componentes por pastilha. Já atualmente as pastilhas VLSI suportam mais de 100 mil componentes. Portanto, desde então a tecnologia vem avançando rapidamente e a separação e criação de novas gerações se tornou menos significativa, já que ainda vivemos os resultados dessas últimas gerações.

Segundo Tanenbaum (2007), o computador (Figura 5) é uma máquina capaz de resolver problemas para as pessoas executando instruções que lhe são dadas. Stallings (2002) define que “o computador é composto de um pequeno conjunto de componentes lógicos básicos, que podem ser combinados de vários modos para armazenar dados binários e executar operações aritméticas e lógicas sobre esses dados”, representado na Figura 5. Não há uma regra ou quantidade certa de componentes a serem usados em um sistema computacional específico, para cada aplicação particular pode-se obter uma combinação de componentes lógicos que se adeque melhor à situação.

Figura 5 - O computador: estrutura de alto nível.

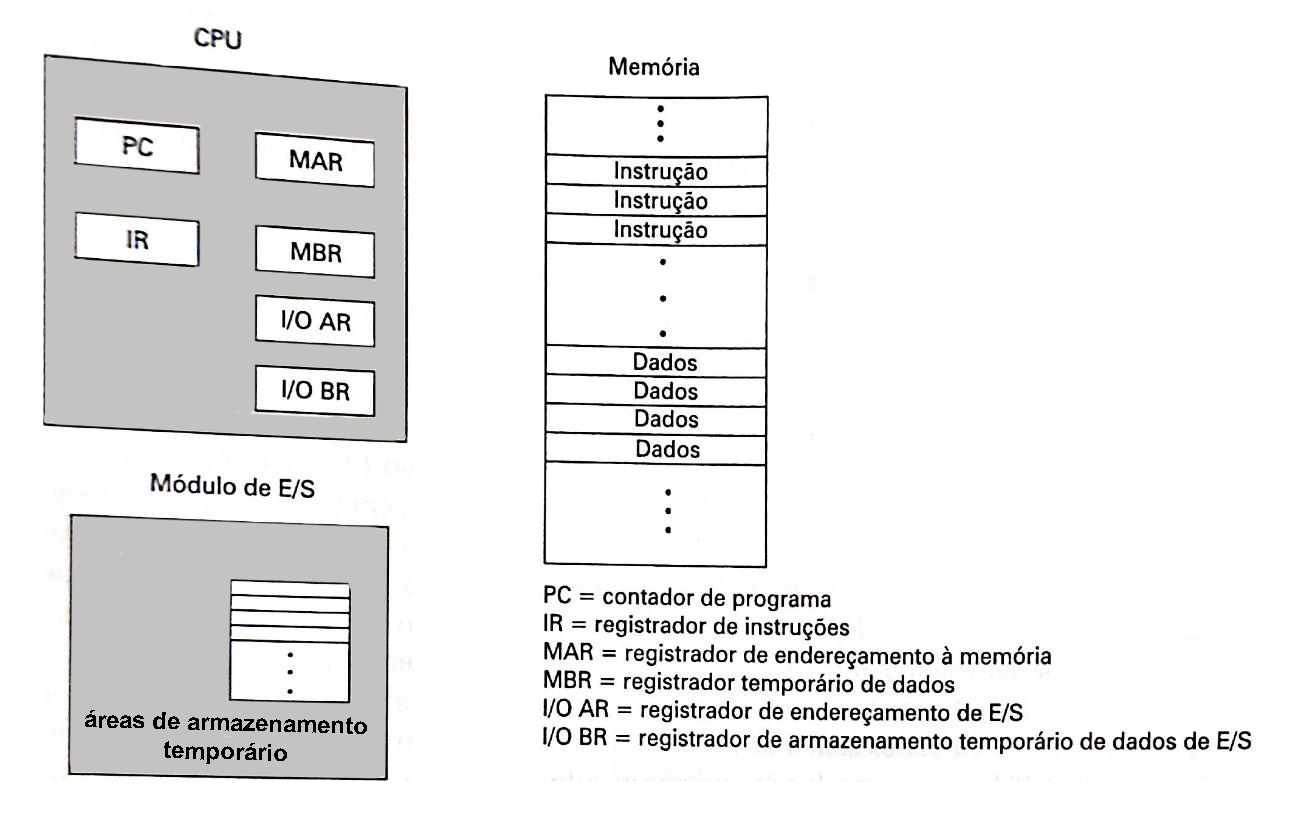


Fonte: Adaptado de STALLINGS, 2002, p. 10.

Os principais elementos de um computador, ou sistema computacional, são a unidade central de processamento (*central processing unit* - CPU), a memória principal, o subsistema de E/S (entrada e saída) e os mecanismos de interconexão entre esses componentes, sendo que cada um deles possui suas respectivas funções, como define Stallings (2002). Segue a definição de cada componente, segundo o autor, e a Figura 6 que representa o sistema computacional em questão:

* Unidade central de processamento (CPU): controla a operação do computador e desempenha funções de processamento de dados. É muitas vezes chamada, simplesmente, de *processador*.
* Memória principal: armazena dados.
* E/S: transfere dados entre o computador e o ambiente externo.
* Sistema de interconexão: mecanismos que estabelecem a comunicação entre a CPU, a memória principal e os dispositivos de E/S (entrada/saída). (STALLINGS, 2002, p. 10).

Figura 6 - Componentes do computador: visão global.

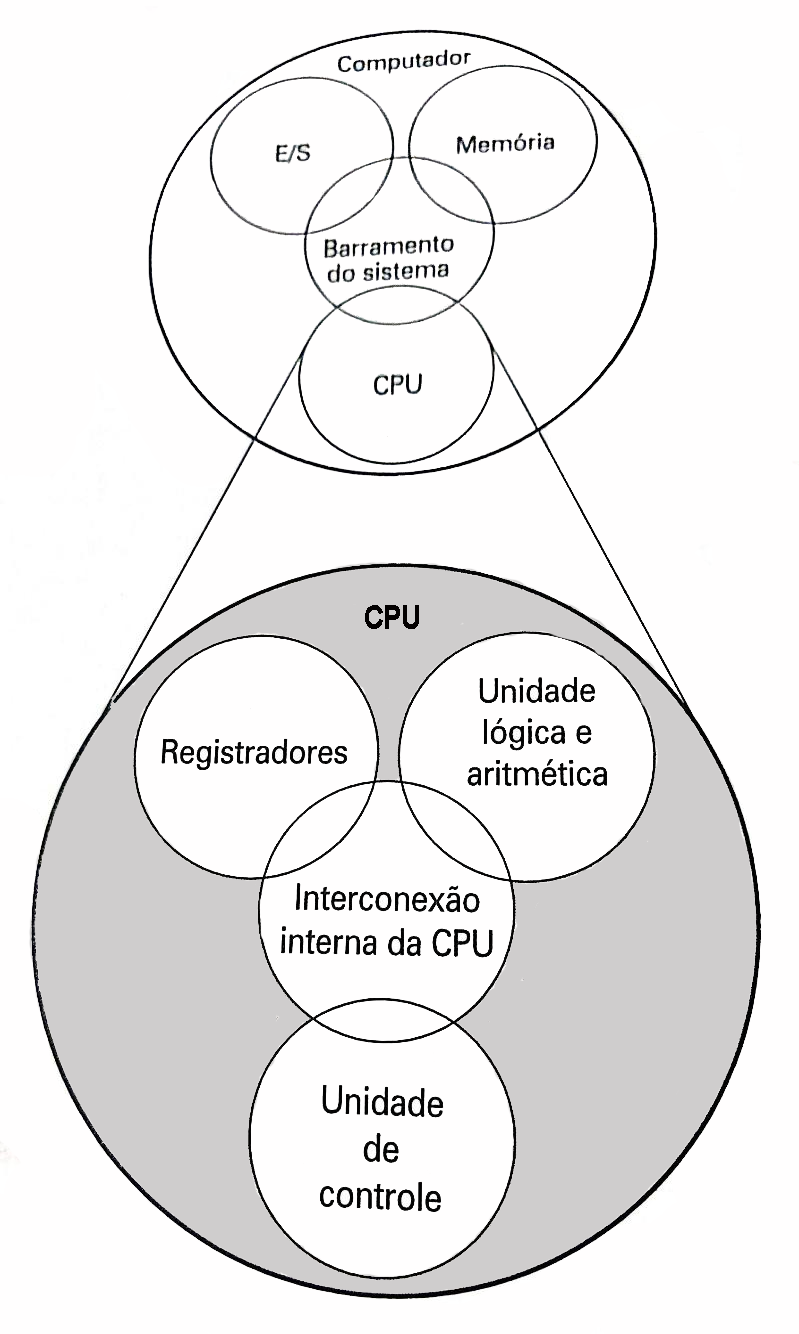


Fonte: Adaptado de STALLINGS, 2002, p. 58.

Segundo Stallings (2002, p.11) “um sistema computacional pode ter um ou mais de cada um dos componentes relacionados anteriormente. Os sistemas mais tradicionais são compostos de uma única CPU”. Ainda segundo o autor, a unidade central de processamento é, de certa forma, o componente mais complexo do sistema. Os principais componentes do processador e a Figura 7 que o representa são os seguintes:

* Unidade de controle: controla a operação da CPU e, portanto, do computador.
* Unidade lógica e aritmética (ULA): desempenha as funções de processamento de dados do computador.
* Registradores: fornecem o armazenamento interno de dados da CPU.
* Interconexão da CPU: mecanismo que possibilita a comunicação entre a unidade de controle, a ULA e os registradores. (STALLINGS, 2002, p. 11).

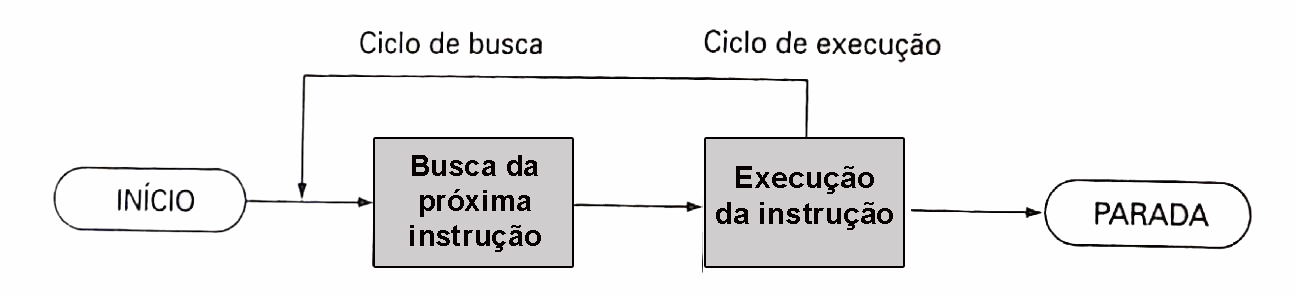
Figura 7 - A unidade central de processamento (CPU).



Fonte: Adaptado de STALLINGS, 2002, p. 12.

O autor ainda define que esses componentes, juntos, tem a função básica de executar um programa constituído por um conjunto de instruções armazenadas na memória. Basicamente, para que essas instruções sejam executadas, pode-se dizer que há dois passos: a CPU interpreta (busca) instruções na memória, uma a uma na sequência, e, a seguir, executa cada uma. Portanto, segundo Stallings (2002, p. 57) “a execução de um programa consiste na repetição desse processo de busca e execução de instruções”, como mostra a Figura 8, ou também conhecidos como ciclo de busca e ciclo de execução. A execução de programas se encerra, ou, no caso, ocorre a parada, somente se o sistema for desligado, se ocorrer no programa algum tipo de erro irrecuperável ou se, durante o programa, for executada alguma instrução de programa que pare a operação do contador.

Figura 8 - Ciclo de instruções básico.



Fonte: Adaptado de STALLINGS, 2002, p. 58.

Segundo Stallings (2002, p. 58), após começar a execução de um programa, “no início de cada ciclo de instrução, o processador busca uma instrução da memória” até que chegue ao fim do programa. Para que esse ciclo ocorra, em um processador típico, há um registrador chamado contador de instruções, ou também conhecido como contador de programa (*program* *counter* - PC), que é usado para armazenar o endereço da próxima instrução a ser buscada na memória do programa. O contador de programa, normalmente, é incrementado pelo processador a cada busca de instrução, o que faz com que a próxima instrução esteja sempre na sequência, ou seja, ela está no próximo endereço de memória.

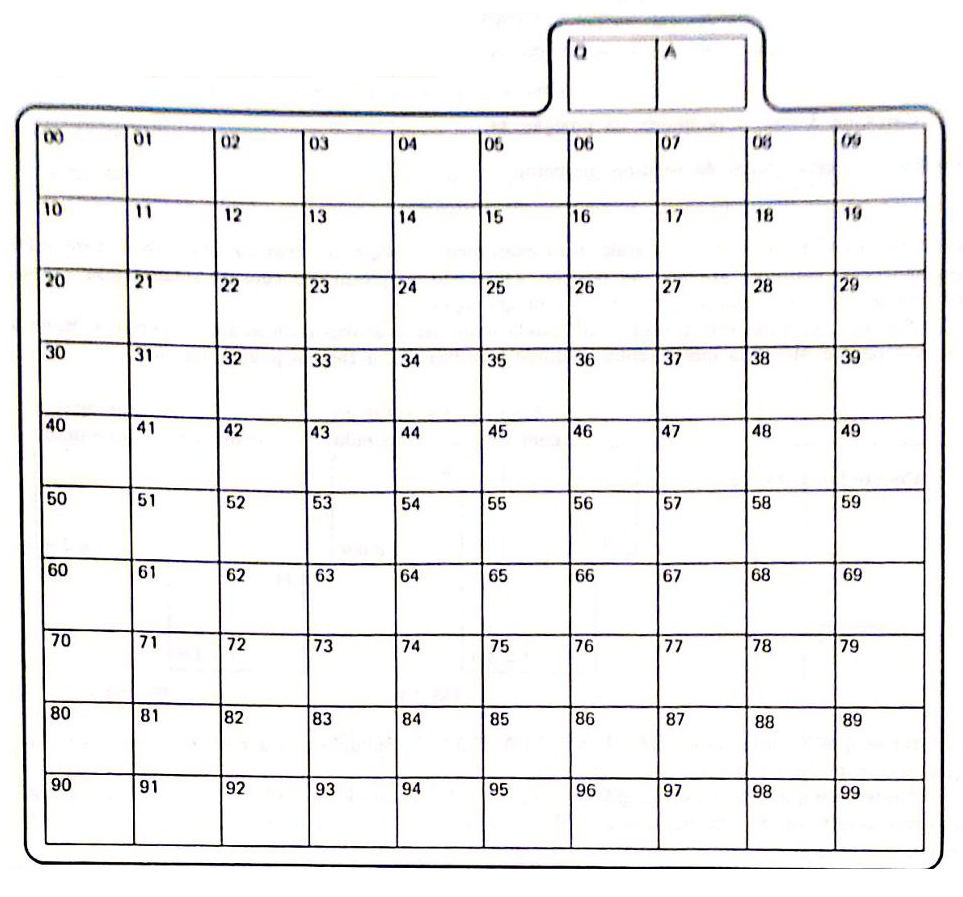
Além do PC, Stallings (2002) diz que a unidade central de processamento também possui o registrador de instruções (*instruction register* - IR) que é onde a instrução buscada no processador é carregada, ou seja, esse registrador armazena os bits que especificam a ação que o processador deve executar. O terceiro e último registrador do processador é o registrador de armazenamento de dados, denominado acumulador (AC).

Ainda segundo o autor, as ações que o processador executa podem ser classificadas em quatro categorias:

* **Processador-memória**: transferência de dados do processador para a memória ou da memória para o processador.
* **Processador-E/S:** transferência de dados entre o processador e um dispositivo periférico, por meio de um módulo de E/S.
* **Processamento de dados**: execução de operações aritméticas ou lógicas sobre dados.
* **Controle:** determinadas instruções podem especificar que a sequência de instruções seja alterada. Por exemplo, o processador pode buscar uma instrução de posição de memória de endereço 149, que especifica que a próxima instrução a ser executada é aquela contida na posição de memória de endereço 182. A execução dessa instrução consiste em armazenar o endereço 182 no PC. Assim, no próximo ciclo de busca, a instrução será obtida do endereço 182, e não do endereço 150. (STALLINGS, 2002, p. 59).

Utilizando estes conceitos previamente explicados sobre o que é um computador e o seu funcionamento, Crawford JR e COPP (1974) propõem um esquema de um computador mais simplificado, como mostra a Figura 9.

Figura 9 - Um esquema de computador

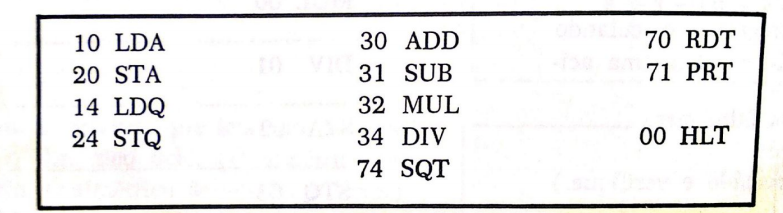


Fonte: CRAWFORD JR; COPP, 1974, p. 01.

Os autores propõem um computador com cento e dois compartimentos, sendo que cem deles representam posições de memória, cada uma com um endereço no canto superior esquerdo. Em cada compartimento de memória está gravado um número contendo quatro dígitos no intervalo de -9999 a 9999, mesmo que não estejam visíveis. Os outros dois compartimentos restantes representam os registradores, ou acumuladores, “Q” e “A”.

Para armazenar uma linha de comando na memória, o autor representa os comandos por números de dois dígitos. Por exemplo, o número 30 simboliza o comando “ADD”, como pode-se observar na Figura 10:

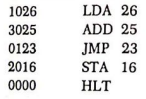
Figura 10 - Relação de comandos.



Fonte: CRAWFORD JR; COPP, 1974, p. 68.

Então, por exemplo, um trecho de programa poderia ter a seguinte representação por números (Figura 11).

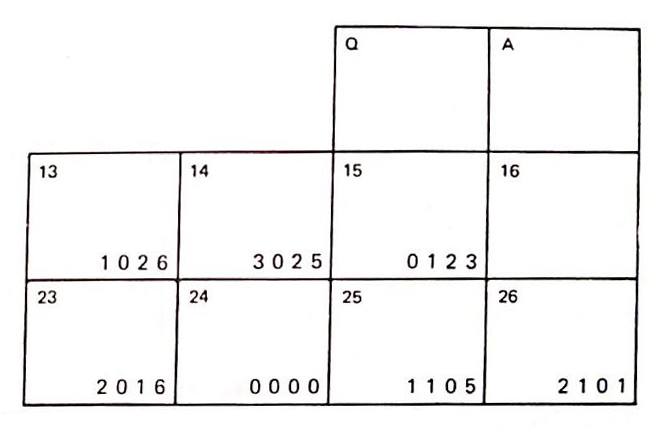
Figura 11 - Trecho de programa codificado.



Fonte: CRAWFORD JR; COPP, 1974, p. 69.

O mesmo trecho de programa seria representado na memória da seguinte forma, como mostra a Figura 12:

Figura 12 - O trecho do programa representado na memória.



Fonte: CRAWFORD JR; COPP, 1974, p. 69.

Portanto, ao introduzir para o aluno estes conceitos sobre o funcionamento completo de um computador e um modelo simplificado como mostra Crawford JR e COPP (1974) no Objeto de Aprendizagem, espera-se um maior desenvolvimento do pensamento computacional, tanto para alunos do ensino básico, que com o pensamento computacional almeja-se uma maior facilidade na resolução de problemas genéricos, quanto para alunos de graduação em cursos relacionados à computação, que com o pensamento computacional almeja-se um melhor entendimento de certos conceitos no primeiro contato com a lógica computacional e algoritmos, aplicando a capacidade de resolver problemas genéricos em problemas específicos da área.

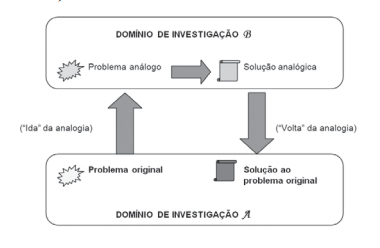
Entretanto, a proposta não é ensinar tais conceitos com a representação de um computador, pois a principal estratégia a ser adotada no Objeto de Aprendizagem que será fruto dessa pesquisa será o uso de analogias, utilizando como base o modelo de um computador simplificado de Crawford JR e COPP (1974). Com isso, espera-se que o aluno possa compreender melhor alguns conceitos que muitas vezes parecem abstratos no primeiro contato em sala de aula.

De forma a entender melhor a escolha deste recurso para a execução do Objeto de Aprendizagem, é preciso primeiro definir o conceito, as vantagens e desvantagens das analogias. A palavra analogia descende do grego, sugerindo uma ideia de proporção, que muitos aliam à proporção matemática, mais especificamente. Posteriormente, o conceito de analogia se expandiu para campos além da matemática, e, atualmente, pode ser entendido como uma comparação entre estruturas de domínios diferentes de conhecimento, baseada em proximidades, uma conhecida (estrutura análoga) e outra pouco ou não conhecida (estrutura alvo) (OLIVEIRA, 2012; DUARTE, 2005; PERELMAN; OLBRECHTS-TYTECA, 2005; TERRAZZAN et al., 2000).

* O processo da analogia é fundamentalmente o seguinte. A analogia científica vale-se de um trânsito de ida e volta entre dois domínios de investigação diferentes. A solução ao problema científico original — sendo indisponível ou insatisfatório uma solução por “primeiros princípios” — é buscada em uma solução que já seja conhecida para um problema correspondente em um âmbito de investigação diferente. (BEZERRA, 2011, p. 596)

O diagrama da Figura 13, demonstra este processo de ida e volta que envolve uma analogia entre dois domínios de investigação, no qual o sistema B representa a estrutura análoga e o sistema A representa a estrutura alvo, sendo que a primeiro é utilizada para ajudar a encontrar a solução para a segunda (BEZERRA, 2011).

Figura 13 – O processo da analogia.



Fonte: BEZERRA, 2011, p. 597.

Essa estratégia a ser utilizada não pressupõe, portanto, a existência de uma igualdade simétrica e sim uma relação que se assemelha a uma outra relação, visando o esclarecimento, estruturação e avaliação do desconhecido a partir do que se conhece. Porém, uma analogia não é uma simples relação, ou identidade parcial, pois é uma semelhança de relação e não uma relação de semelhança (OLIVEIRA, 2012; DUARTE, 2005; PERELMAN; OLBRECHTS-TYTECA, 2005; TERRAZZAN et al., 2000). O dicionário Michaelis (2009) coloca as seguintes definições para analogia:

* **1** Qualidade de análogo. **2** Proporção matemática ou igualdade de razões. **3** Semelhança de propriedades. **4** Semelhança em algumas particularidades, de funções etc., sem que haja igualdade atual ou completa: ***Não há como negar a analogia entre o coração e uma bomba aspirante-premente***. **5*****Filos*** Identidade de relação entre seres de natureza diferente. **6** No ocultismo, método lógico intermediário entre dedução e a indução, pertencente à teoria. ***Antôn: diferença. Raciocinar por analogia:*** julgar pelas semelhanças que existem entre fatos. (MICHAELIS, 2009)

De acordo com Duarte (2005), o uso de analogias pode trazer algumas potencialidades, elas podem:

* Ativar o raciocínio analógico, organizar a percepção e desenvolver capacidades cognitivas como a criatividade e a tomada de decisões.
* Favorecer o entendimento e a visualização de conceitos mais abstratos.
* Facilitar a evolução ou a mudança de conceitos.
* Permitir a percepção de concepções alternativas.
* Ser utilizadas para avaliar o aprendizado do aluno.

Porém a autora ainda traz algumas dificuldades que podem surgir com a utilização de analogias:

Pode não ajudar a atingir o objetivo proposto, pois a analogia pode ser entendida como o conceito alvo.

* Pode não ativar o raciocínio analógico.
* Pode não ser interpretada como uma analogia.
* Pode fazer com que o aluno foque apenas no lado positivo da analogia, esquecendo as limitações da mesma.

A analogia é geralmente utilizada com o objetivo de elucidar algum conceito de uma disciplina ou área, introduzindo esse conceito em um novo campo de maior conhecimento, além de ser de extrema importância no auxilio da explicação de conceitos obscuros, segundo Oliveira (2012). Com isso, o emprego de analogias é comumente colocado em prática pelos educadores, a fim de auxiliar no aprendizado de diversas áreas.

* O emprego de analogias [na educação] busca, além da inovação pedagógica representada por uma forma dinâmica e adaptativa de se trabalhar a estruturação de conceitos com o aluno, uma aceitação crescente do recurso à intuição básica [...]. (NAGEM et al., 2001, p. 197)

Por fim, espera-se que através do levantamento bibliográfico e das análises elaboradas até o momento, em complemento com as futuras pesquisas que serão aprofundadas ao longo do processo, sobre conteúdos multidisciplinares assim como os específicos da computação, seja possível realizar com êxito um produto que possa atingir os objetivos desse projeto, baseado em conceitos de um referencial teórico consistente.

# 3 métodos e materiais

## 3.1 Métodos

Como o propósito deste trabalho é um produto, o tipo de pesquisa a ser utilizada em relação aos objetivos gerais é a exploratória, uma vez que o objetivo proposto é a criação de uma ferramenta educacional que auxilie o educador no ensino de conceitos relacionados ao pensamento computacional, a fim de solucionar certos problemas gerados devido a deficiência do sistema educacional atual no que se diz respeito à habilidades ditas essenciais, como o raciocínio lógico e a capacidade de resolução de problemas genéricos. Segundo Gil (2002), essa classificação de pesquisa em específico é utilizada com o propósito de planejar e estruturar uma hipótese a partir do conhecimento e entendimento de um certo problema a ser estudado.

## 3.2 Materiais

No início do projeto foram decididas quais linguagens de programação seriam utilizadas na implementação deste Objeto de Aprendizagem, que são as seguintes: HTML (*HyperText Markup Languange —* Linguagem de Marcação de Hipertexto), CSS (*Cascading Style Sheets* — Folhas de Estilo em Cascata) e Javascript. Foram levados em consideração alguns aspectos como acessibilidade, modularização e também o nível de afinidade e experiência da autora com estas linguagens de programação.

A linguagem HTML, que é uma linguagem estruturada através de marcações, é a base de todo o conteúdo existente na *World Wide Web* (WWW), que é uma das principais fontes de conhecimento atualmente, principalmente entre estudantes. É necessário que o usuário utilize um navegador de internet (*Web Browser*), que é a ferramenta capaz de interpretar estas marcações de estrutura contidas nesta linguagem (BROOKS, 2007, tradução nossa).

Com o HTML responsável pela a estrutura do conteúdo, o CSS é uma linguagem de estilo simples e revolucionária, responsável pela especificação da aparência, como cor, tamanho, fonte, etc, dos diversos elementos demarcados por uma linguagem de marcação, como o HTML. Esta linguagem de estilo foi concebida com o objetivo de separar o estilo de uma aplicação e sua estrutura em partes diferentes, ou seja, o HTML é responsável somente pela estrutura e o CSS encarrega-se do estilo, e, com isso, tem-se um código menos poluído visualmente (GRANNEL; SUMNER; SYNODINOS, 2007, tradução nossa).

O JavaScript é uma linguagem de programação interpretada comumente utilizada no desenvolvimento para Web, diferentemente do Java que é uma linguagem compilada. Essa ferramenta tem suas origens com as linguagens C/C++ porém não é independente, ou seja, ela geralmente é utilizada em conjunto ao HTML com o objetivo de construir páginas mais interativas, permitindo a modificação do conteúdo e a aparência dos elementos de forma dinâmica. O Javascript se desenvolveu em uma linguagem de programação completa, capaz de manipular e lidar com textos e operações matemáticas, sendo útil na resolução de diversos possíveis problemas dentro da computação (BROOKS, 2007, tradução nossa).

Além da linguagem JavaScript pura, uma de suas bibliotecas, conhecida como jQuery, foi escolhida para ser utilizada neste projeto. Segundo Silva (2008), esta biblioteca foi criada por John Resig e pode ser utilizada gratuitamente, uma vez que é um software livre e aberto. Essa biblioteca “[...] é uma maneira simples e fácil de escrever JavaScript [...] Simplicidade é a palavra-chave que resume e norteia o desenvolvimento com jQuery” (SILVA, 2008, p. 26).

O editor de códigos utilizado durante toda a implementação deste Objeto de Aprendizagem para as três linguagens de programações escolhidas foi o Sublime Text 2, que “possui uma interface bastante limpa, e o fundo preto da tela não atrapalha a leitura em nada” (YUNG, 2013).

# 4 Desenvolvimento

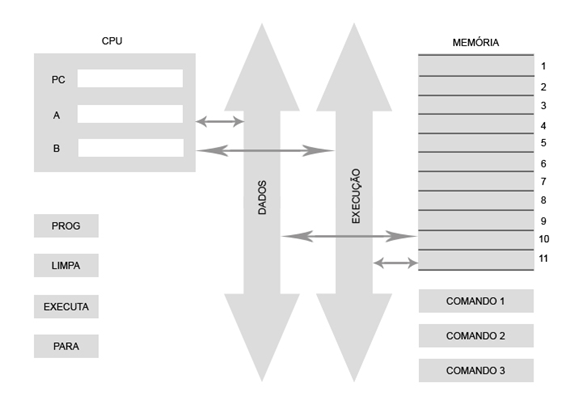
Neste capítulo, os detalhes relacionados ao desenvolvimento do produto desta pesquisa, ou seja, do Objeto de Aprendizagem, serão descritos através da analogia proposta e sua estrutura, utilizando os conceitos estudados durante o levantamento bibliográfico.

## 4.1 Analogia

Esse projeto propõe o uso de uma analogia na interface do Objeto de Aprendizagem, a fim de facilitar o entendimento de conceitos ligados ao pensamento computacional.

A analogia utilizada neste projeto não foi idealizada de forma direta. O primeiro protótipo da analogia projetado pela autora teve como base a definição de um esquema computador simplificado feito por Crawford JR e COPP (1974). O protótipo inicial do Objeto de Aprendizagem desta monografia foi criado, baseado na concepção de codificar programas utilizando um protótipo simplificado de computador. A Figura 14 apresenta o primeiro protótipo de analogia para este Objeto de Aprendizagem:

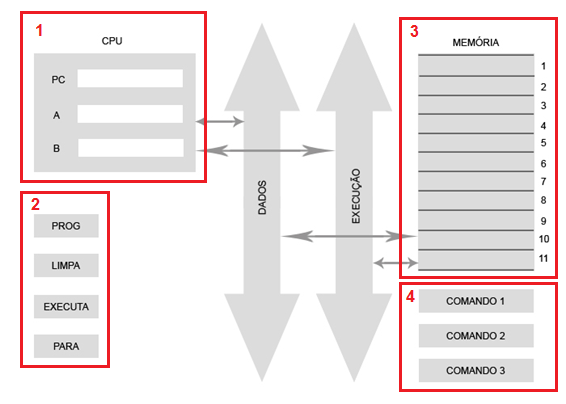
Figura 14 - Protótipo de analogia 1.0.



Fonte: Elaborada pela autora.

O protótipo mostrado na Figura 15 é uma representação de um computador que pode ser dividido, basicamente, em quatro blocos principais (Figura 15):

Figura 15 - Protótipo de analogia 1.0, dividido em 4 blocos.



Fonte: Elaborada pela autora.

1. O primeiro bloco representa uma CPU, contendo o PC (*Program Counter*) e os dois registradores “A” e “B”.
2. O segundo bloco possui os botões de ação do programa, nos quais o aluno pode começar o programa (PROG), limpar a memória (LIMPA), executar o programa implementado (EXECUTA) e interromper a execução do programa (PARA).
3. O terceiro bloco representa a memória deste protótipo, que é o lugar no qual o usuário implementa o programa.
4. O último e quarto bloco apresenta quais comandos estão disponíveis para que o usuário utilize em seu programa.

Entretanto, este protótipo inicial pode ser intimidador, caso o usuário não esteja familiarizado com os termos utilizados nessa analogia, como CPU e a Memória, por exemplo. Os conceitos utilizados neste protótipo necessitam de um conhecimento básico do assunto a fim de que o usuário seja capaz de compreender a funcionalidade de cada elemento, dado que eles não são intuitivos para indivíduos que não estão acostumados com a interface de um computador. Por essa razão, esse protótipo foi avaliado como inviável, já que esta necessidade de conhecimento específico prévio entra em conflito com os pré-requisitos de uma analogia, uma vez que para utilizá-la com sucesso é necessário, obrigatoriamente, que a solução análoga, que representa o problema original, seja um conteúdo conhecido pelo público-alvo.

Com isso, uma nova analogia foi elaborada, visando uma solução para o problema do primeiro protótipo, com uma situação mais intuitiva e uma interface mais gráfica, representando um ambiente de trabalho. A Figura 16 representa o segundo modelo proposto para o Objeto de Aprendizagem fruto deste trabalho, apresentando uma nova situação análoga ao problema original:

Figura 16 - Protótipo de analogia 2.0.



Fonte: Elaborada pela autora.

O segundo protótipo produzido conta com uma interface mais gráfica e, consequentemente, mais amigável ao usuário. A analogia inserida neste segundo modelo pode ser considerada mais robusta e menos intimidadora, visto que ela simula uma situação real e corriqueira, e não apenas uma interface sem relação com qualquer abstração conhecida pelo senso-comum. Ou seja, as chances do usuário se identificar com o novo ambiente proposto como situação análoga são superiores. Além disso, provavelmente todos os componentes utilizados no modelo são previamente conhecidos por uma grande parte dos usuários, tornando desnecessária a explicação teórica de cada bloco existente no novo ambiente proposto, algo que não ocorria no protótipo descartado.

Esta nova proposta conta com novas representações mais gráficas para os blocos que foram inseridos no primeiro modelo, como mostra a Figura 17.

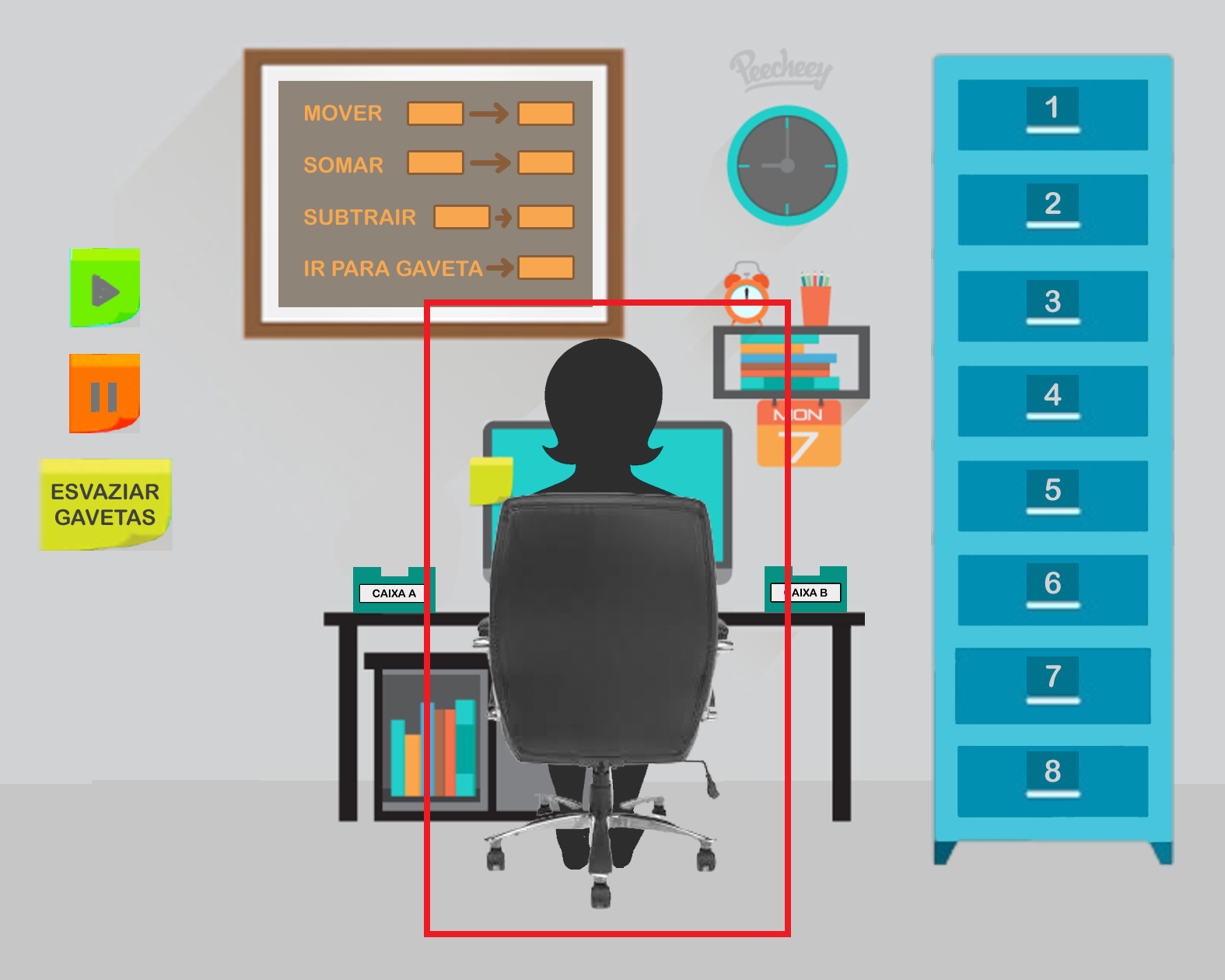
Figura 17 - Protótipo de analogia 2.0 divido em blocos.



Fonte: Elaborada pela autora.

1. Este conjunto de elementos representa a CPU, de acordo com o primeiro protótipo, contendo o PC (*Program Counter*) e os dois registradores “A” e “B”.
2. Os post-it’s representam os botões e ação do programa, nos quais o aluno pode executar o programa implementado (►), interromper a execução do programa ( ▍▍) e limpar a memória (esvaziar gavetas),
3. As gavetas representam a memória deste protótipo, que é o lugar no qual o usuário armazena os dados e as instruções.
4. O quadro apresenta quais comandos estão disponíveis para utilização do usuário em seu programa, assim como no primeiro protótipo.

Figura 18 - O agente do protótipo de analogia 2.0.



Fonte: Elaborada pela autora.

Esta analogia também possui a representação de um agente de ações (interpretando o fluxo de ações que existe em um computador), no caso, representado por uma personagem (Figura 18) que literalmente executará os comandos de acordo com as ordens do usuário, da maneira que foram descritos anteriormente como ilustrado na Figura 20.

Figura 19 - A mesa de trabalho do protótipo de analogia 2.0.

Fonte: Elaborada pela autora

A Figura 19 simboliza a mesa de trabalho do agente da analogia. Nela, se comparado com a primeira analogia, temos:

1. As caixas “A” e “B” representando os registradores do computador “A” e “B”, respectivamente.
2. O monitor representando o PC (Contador de Programa), indicando que a execução está na primeira gaveta do armário.

Esta analogia consiste no seguinte funcionamento:

1. Uma atividade é proposta ao usuário (Por exemplo, mover o número 20 para o registrador “A”).
2. Com base no objetivo da atividade, o usuário escolhe um comando que ele espera que seja útil para tal proposta (Por exemplo, o comando “MOVER”).
3. Feito isso, ele precisa escolher um valor para mover para algum lugar, no caso, um registrador ou uma gaveta (Por exemplo: mover o número 20 para a caixa “A”).
4. Depois de preencher os dados no quadro, o usuário deve arrastar essa linha para alguma das gavetas. (Por exemplo: o usuário arrastou a ação de mover o número 20 para a caixa “A” na gaveta 1)
5. Se o usuário julgar que as ações que ele colocou nas gavetas já solucionam o problema proposto, então ele aperta o botão para executar suas ações.
6. A cada começo de ações a personagem agente do programa verificará a sua tela de computador para saber qual gaveta ela deve olhar, no caso estará mostrando a gaveta 1.
7. Então, ela se dirige à gaveta 1, abre e encontra a seguinte ordem “mover o número 20 para a caixa A”.
8. A agente volta à mesa, escreve o número 20 em um papel e coloca dentro da pasta “A”.
9. Como o usuário tenha completado a tarefa com sucesso, ele receberá uma gratificação e poderá passar para a próxima fase, caso houver mais.

Durante o desenvolvimento, foi verificado que o bloco de comandos poderia ter a necessidade de aumentar em altura durante o uso, principalmente nas fases final em que o usuário tem ao seu dispor mais comandos, tornando inviável a área em que ele está disposto. Com isso, houve uma pequena mudança na disposição dos blocos a fim de amenizar esse problema, como pode-se observar na Figura 20, na qual os blocos de comandos e de ações inverteram de posições.

Figura 20 - Protótipo de analogia 2.1.

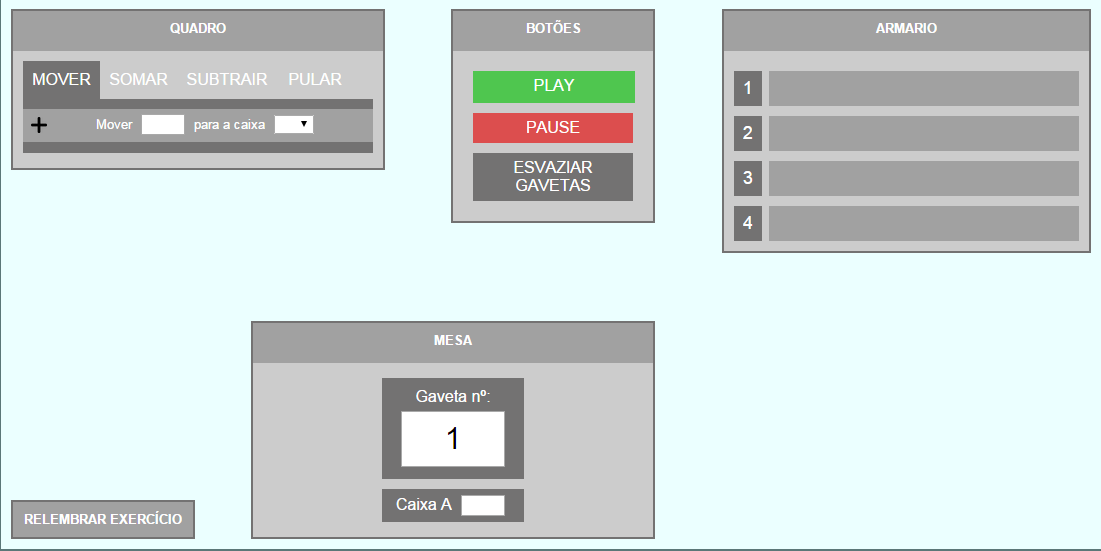


Fonte: Elaborada pela autora.

No entanto, o produto final de fato programado por esta autora não terá a aparência desta última analogia desenvolvida, devido à falta de recursos para a implementação do projeto, já que tal feito necessitaria do trabalho de um designer, além do trabalho do programador aqui desenvolvido. Porém, o protótipo realizará as mesmas atividades, da mesma maneira, podendo conter uma interface mais amigável caso o projeto seja levado adiante e de fato implementado.

Os estudos acima mostram como uma analogia mais amigável e mais relacionada com a realidade pode facilitar o uso do Objeto de Aprendizagem e o entendimento dos conceitos por ele passados. Entretanto, a programação do Objeto de Aprendizagem fruto deste trabalho é um protótipo, que, por razões técnicas, não possui a interface gráfica proposta para a analogia, uma vez que, para que isto fosse possível, seria necessário que a implementação fosse feita em conjunto com um designer gráfico. Portanto, este trabalho apresenta o funcionamento do projeto, porém sem a interface análoga (Figura 21), que poderá ser incluída no futuro, caso este projeto seja continuado.

Figura 21 – Protótipo do Objeto de Aprendizagem.



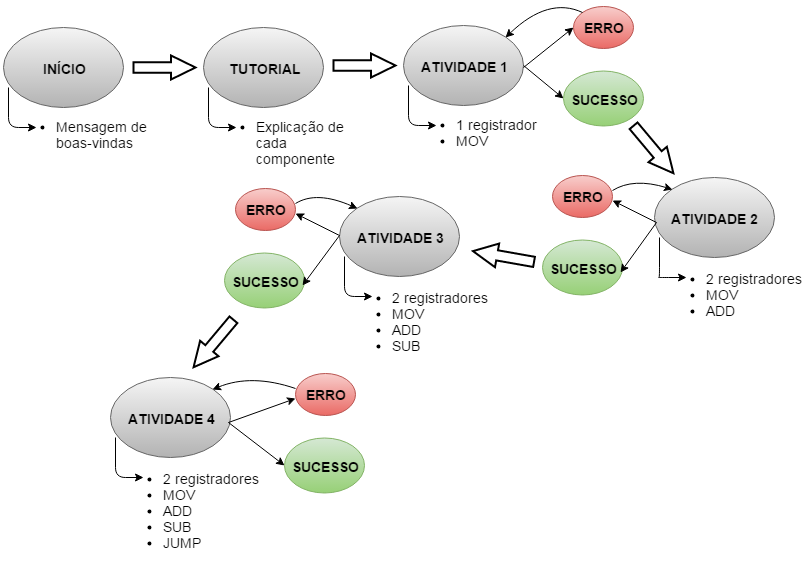
Fonte: Elaborada pela autora.

Como pode-se observar, apesar de não conter a interface amigável proposta para implementações futuras, este protótipo apresentado também contém os quatro blocos principais, assim como na interface análoga (Figura 20), além de um quinto bloco onde o usuário pode relembrar o enunciado do exercício.

## 4.2 O Objeto de Aprendizagem e suas Atividades

Este Objeto de Aprendizagem é composto por três etapas principais: início, tutorial e as atividades, como mostra a Figura 22:

Figura 22 – Etapas do Objeto de Aprendizagem.



Fonte: Elaborada pela autora.

Primeiramente, o usuário é recebido com uma tela de boas-vindas (Figura 23).

Figura 23 – Tela de boas-vindas.



Fonte: Elaborada pela autora.

No momento em que o usuário estiver preparado para iniciar as próximas etapas, ele deve clicar no botão “COMEÇAR >”. Feito isso, ele será apresentado à personagem principal desta analogia (Figura 24), juntamente com uma breve história sobre ela, que, no caso, é inserida com o objetivo de tornar este aprendizado uma experiência mais lúdica.

Figura 24 – Apresentação da personagem.



Fonte: Elaborada pela autora.

Após esta fase inicial de boas-vindas, ao clicar no botão “>>”, o usuário será direcionado à uma explicação bloco a bloco deste Objeto de Aprendizagem, como mostram as Figuras 25, 26, 27, 28 e 29.

Figura 25 – Explicação: Bloco RELEMBRAR EXERCÍCIO.



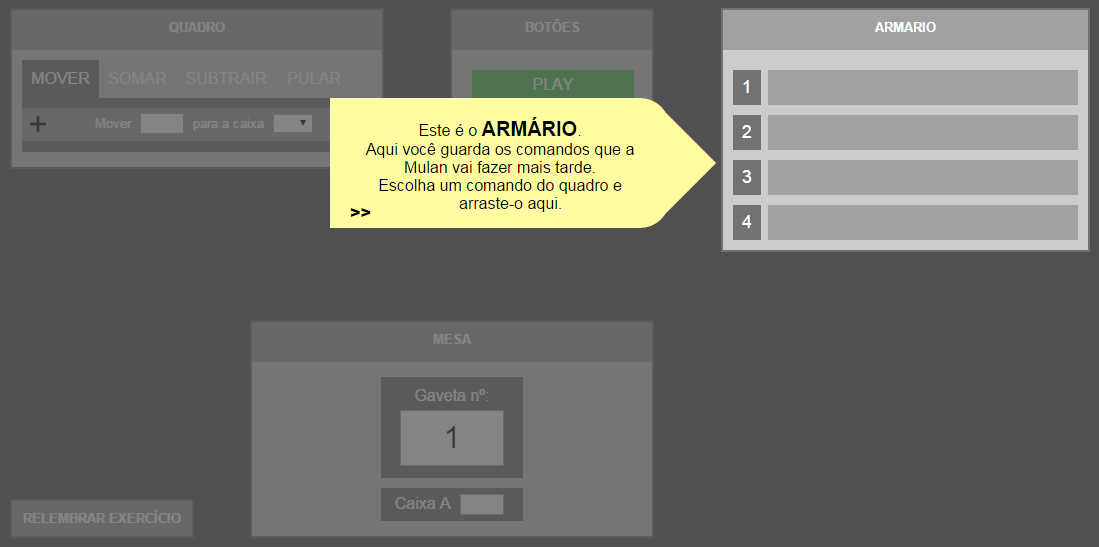
Fonte: Elaborada pela autora.

Figura 26 – Explicação: Bloco QUADRO.



Fonte: Elaborada pela autora.

Figura 27 – Explicação: Bloco ARMÁRIO.



Fonte: Elaborada pela autora.

Figura 28 – Explicação: Bloco MESA.



Fonte: Elaborada pela autora.

Figura 29 – Explicação: Bloco BOTÕES.



Fonte: Elaborada pela autora.

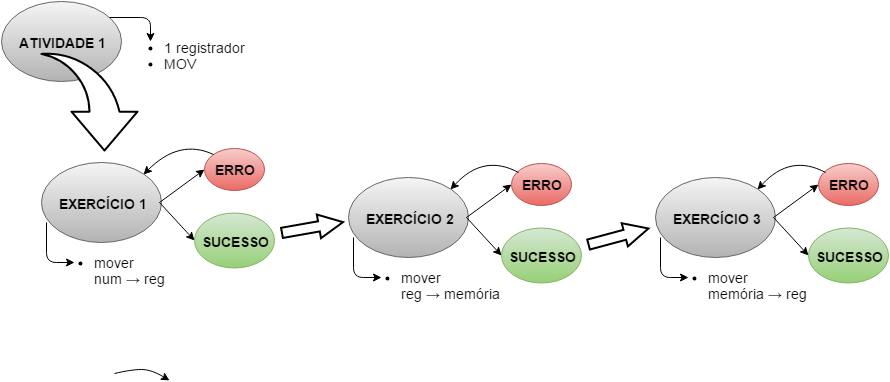
Após o fim da última explicação de Blocos do Objeto de Aprendizagem (Figura 29), o usuário chega à etapa das Atividades, que é o foco principal de aprendizagem deste projeto. Cada atividade é dividida em forma de exercícios que submetem o jogador a um desenvolvimento progressivo de sua aprendizagem. Em cada atividade, que contém um ou mais exercícios, haverá a introdução de novas ferramentas para o usuário, que podem ser comandos novos a serem utilizados, ou uma quantidade maior de caixas (registradores) ou, também, uma maior quantidade de gavetas (espaços de memória), como mostrado anteriormente no fluxograma da Figura 22.

Portanto, as atividades são divididas, basicamente, no aprendizado de quatro comandos principais:

* **Primeira atividade:** o comando MOVER.
* **Segunda atividade:** comando SOMAR.
* **Terceira atividade:** comando SUBTRAIR.
* **Quarta atividade:** comando PULAR.

### 4.2.1 Primeira atividade

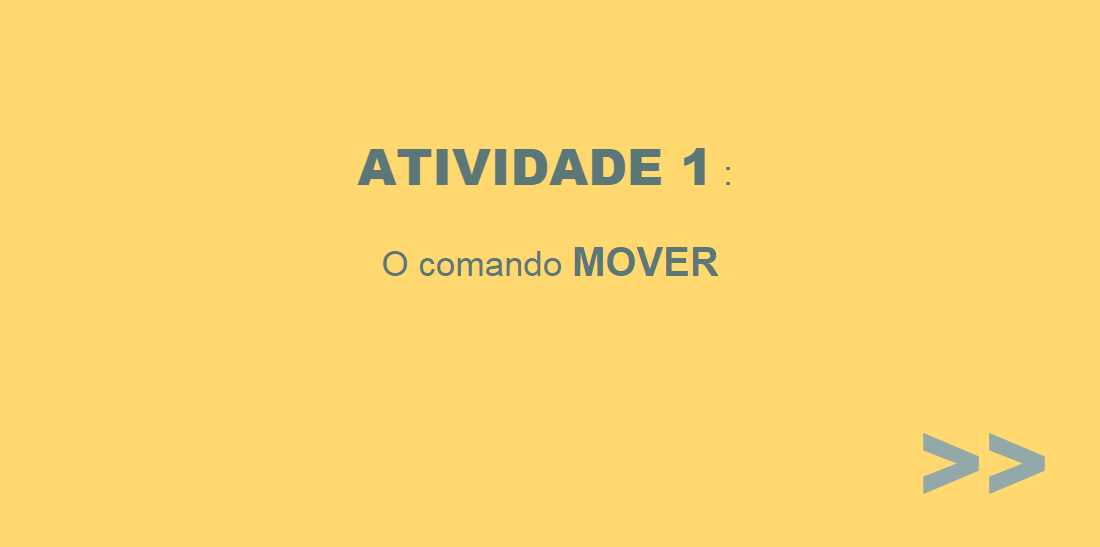
Em seguida, após o usuário passar pelas telas iniciais, familiarizando-o com a estrutura deste Objeto de Aprendizagem, será proposto a ele a primeira Atividade, que, no caso, contém 3 exercícios, como pode ser observado no fluxograma apresentado na Figura 30.

Figura 30: Fluxograma da Atividade 1.

Fonte: Elaborada pela autora.

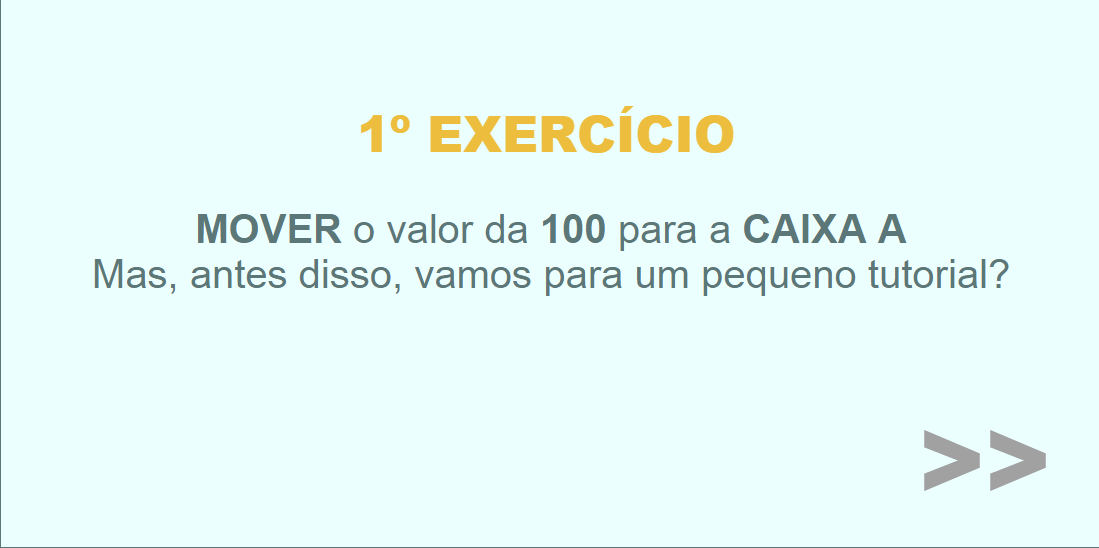
Após passar pela etapa de explicação dos Blocos, o usuário visualizará uma tela inicial da atividade, mostrando qual comando ele terá disponível para essa fase, no caso, o comando MOVER (Figura 31). Dando continuidade, o usuário será redirecionado para a próxima tela, contendo o enunciado do primeiro exercício desta atividade (Figura 32).

Figura 31 – Atividade 1.



Fonte: Elaborada pela autora.

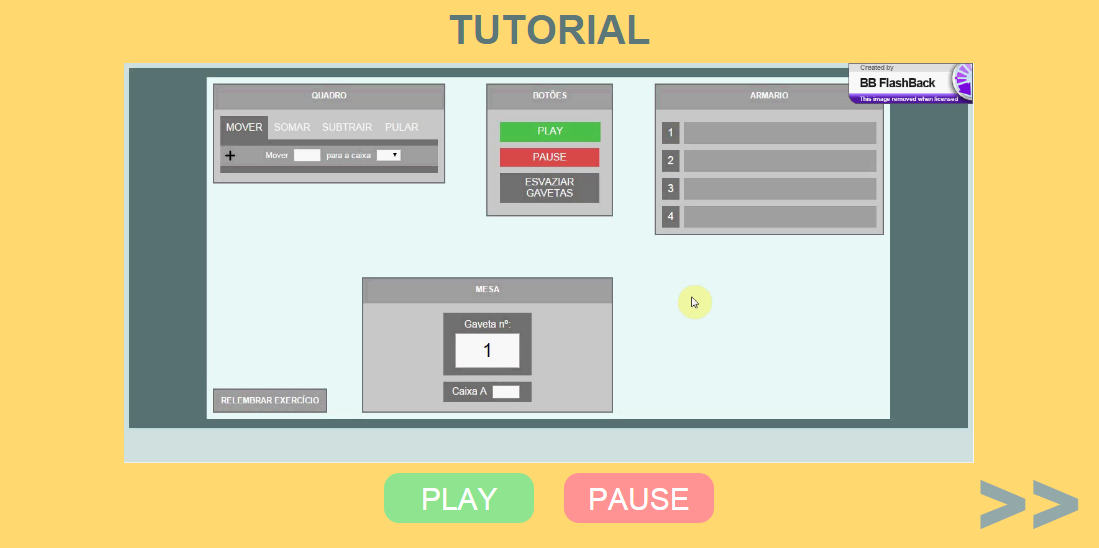
Figura 32 – Atividade 1: Exercício 1.



Fonte: Elaborada pela autora.

Porém, como este exercício é o exercício introdutório do Objeto de Aprendizagem, o usuário será apresentado a um vídeo que mostra o funcionamento do Objeto de Aprendizagem resolvendo este primeiro exercício, antes que ele mesmo possa resolvê-lo, como mostra a Figura 33. Com este vídeo, espera-se que o usuário possa entender melhor o funcionamento na prática dos blocos que foram explicados previamente a ele.

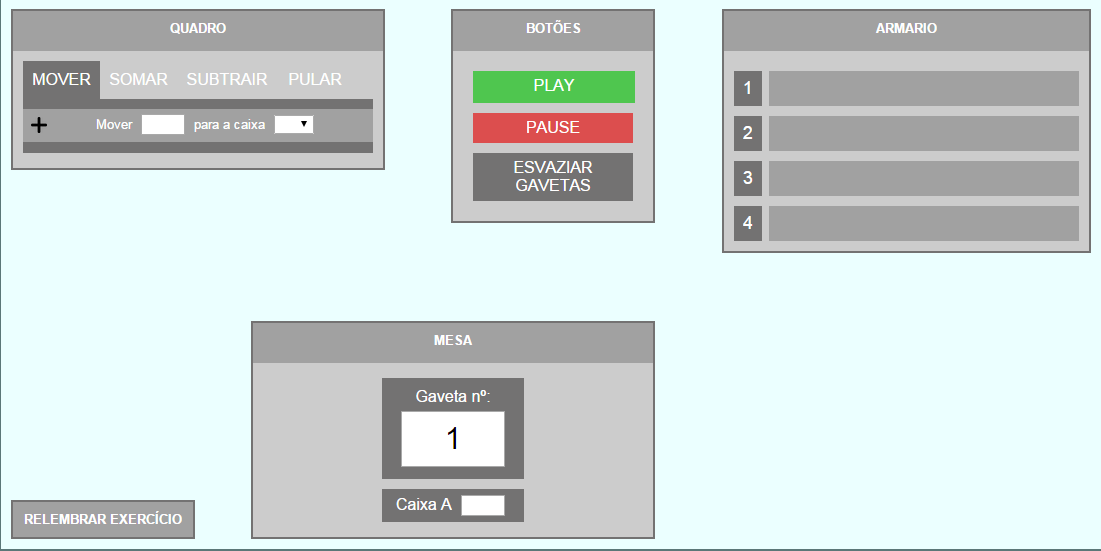
Figura 33 – Atividade 1: Vídeo tutorial do Exercício 1.



Fonte: Elaborada pela autora.

E então, passando para a próxima tela, o usuário será apresentado ao Objeto de Aprendizagem em si, contendo o comando de mover um valor para uma caixa, e terá que resolver o exercício conforme dita o enunciado, como mostra a Figura 34:

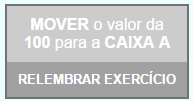
Figura 34 – Tela do Exercício 1.



Fonte: Elaborada pela autora.

Caso o usuário tenha esquecido qual o objetivo do exercício em que ele está, ao clicar no bloco “RELEMBRAR EXERCÍCIO”, este se expandirá e exibirá o enunciado, como mostra a Figura 35:

Figura 35 – Bloco RELEMBRAR EXERCÍCIO expandido.

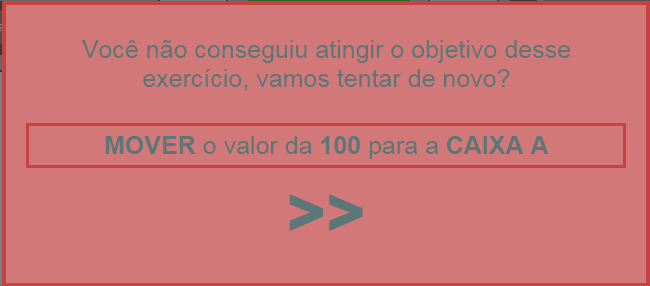


Fonte: Elaborada pela autora.

No Bloco BOTÕES, como mostra a Figura 34, o usuário tem a disposição o botão ESVAZIAR GAVETAS, no qual ao apertá-lo, todas as gavetas serão automaticamente esvaziadas caso o usuário perceba que colocou comandos errados ou que não irá usar. Ainda no mesmo Bloco, o usuário tem a disposição o botão PAUSE, no qual ele pode parar a execução dos comandos que foram previamente colocados nas gavetas, caso ele perceba que não chegará no resultado esperado e, então, automaticamente todos os blocos serão reiniciados para seus estados iniciais.

Depois de terminar a execução de todas as gavetas, mesmo as vazias, o programa checará se o usuário atingiu o objetivo proposto no exercício. Caso as respostas estejam incorretas, aparecerá para o usuário uma tela de erro (Figura 36) mostrando que ele não atingiu o objetivo e relembrando o enunciado. Se o usuário clicar no botão “>>”, ele poderá voltar para a tela do exercício com os blocos reiniciados para tentar de novo.

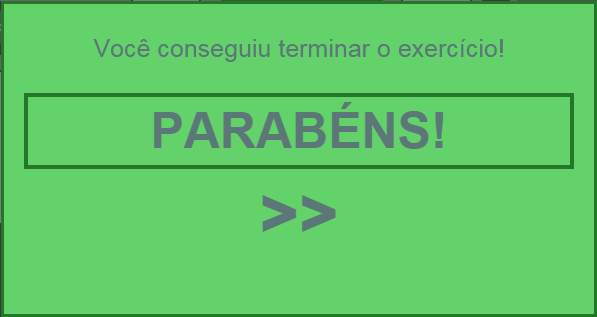
Figura 36 – Tela de erro.



Fonte: Elaborada pela autora.

Porém, se após a execução de todas as gavetas o programa detectar que as respostas estão corretas, aparecerá a tela de parabéns (Figura 37), parabenizando o usuário por completar o exercício e permitindo que ele passe para o próximo.

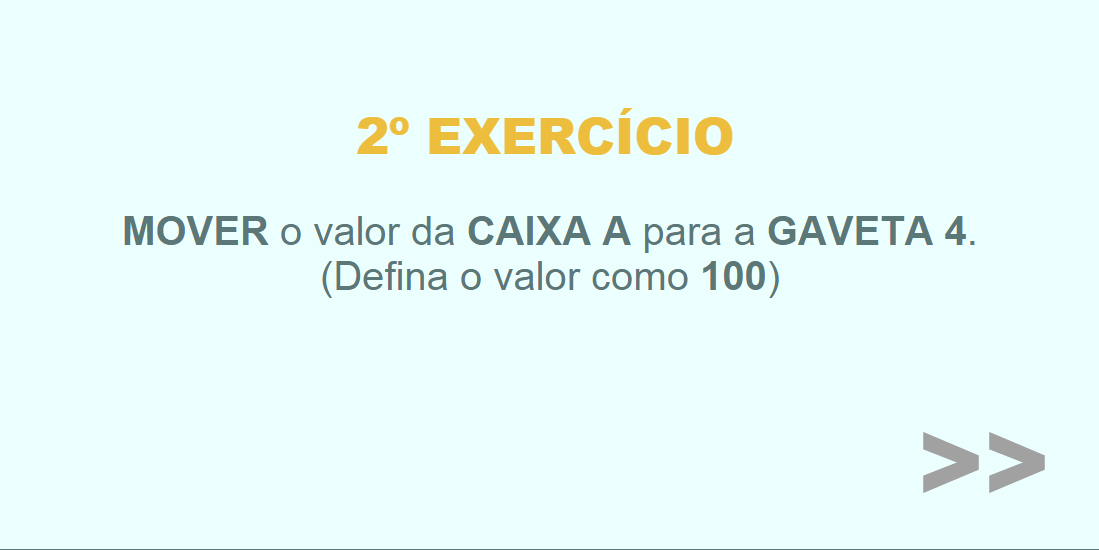
Figura 37 – Tela de sucesso.



Fonte: Elaborada pela autora.

Em seguida, o usuário será redirecionado para a tela do enunciado do Exercício 2 (Figura 38), e posteriormente para a tela onde ele resolverá exercício em si (Figura 39), no qual ele terá à disposição mais um tipo do comando MOVER: o comando de mover o valor de uma caixa para uma das gavetas.

Figura 38 – Atividade 1: Exercício 2.



Fonte: Elaborada pela autora.

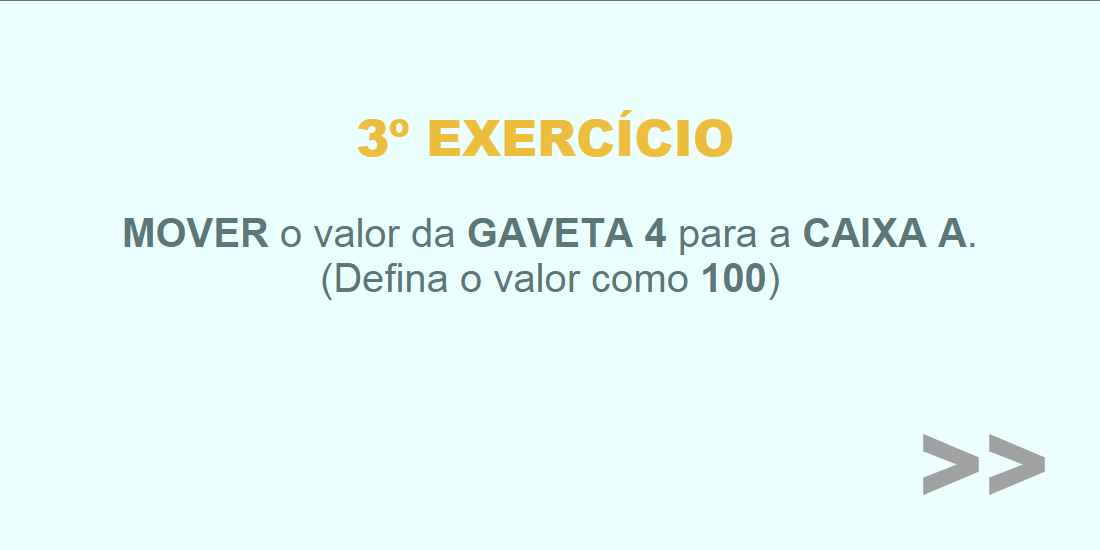
Figura 39 – Tela do Exercício 2.



Fonte: Elaborada pela autora.

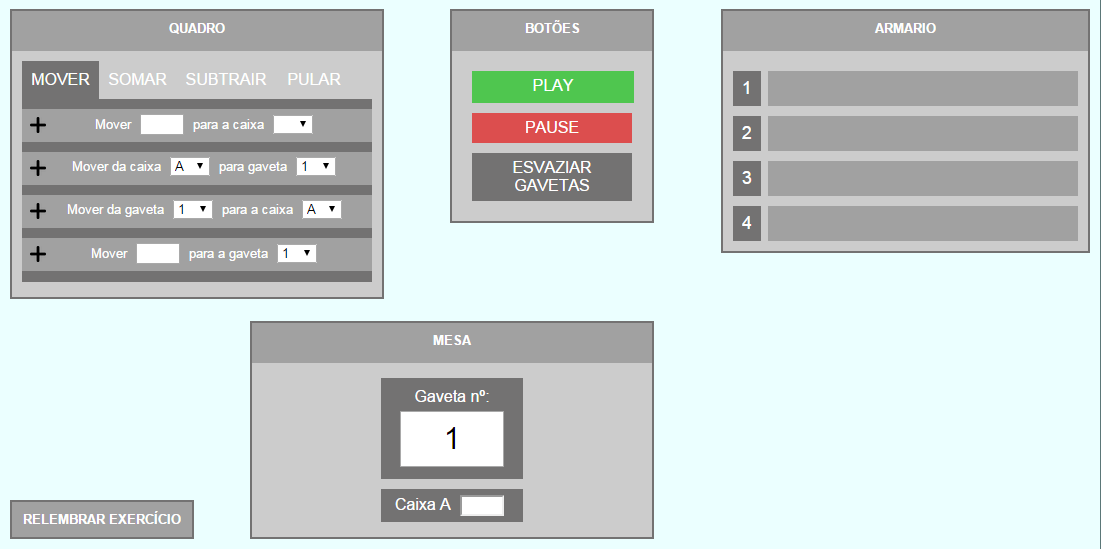
Caso o usuário acerte o exercício, ele será redirecionado para a tela de sucesso e, em seguida, para a tela do enunciado do Exercício 3 (Figura 40), e posteriormente para a tela onde ele resolverá exercício em si (Figura 41), no qual ele terá à disposição mais dois tipos do comando MOVER, o comando de mover o valor de uma gaveta para uma caixa e o comando de mover um valor para uma das gavetas.

Figura 40 – Atividade 1: Exercício 3.



Fonte: Elaborada pela autora.

Figura 41 – Tela do Exercício 3.



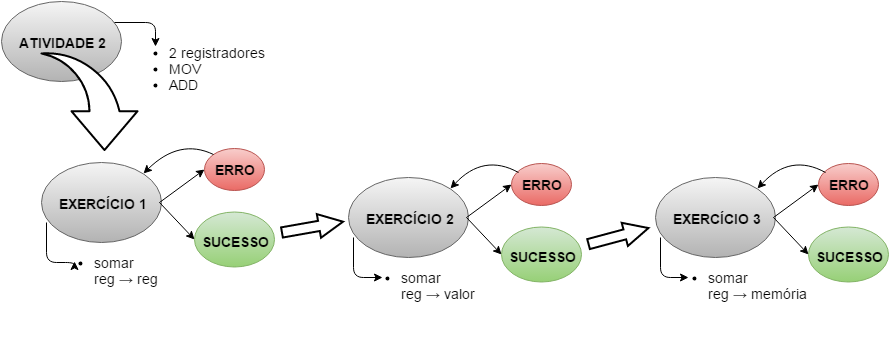
Fonte: Elaborada pela autora.

Caso o usuário acerte o exercício, ele será redirecionado para a tela de sucesso e chegará ao final da Atividade 1. Com isso, espera que ele tenha compreendido por completo o conceito do comando MOVER e seus quatro diferentes modos de ser utilizado, a fim de poder utilizá-los nas atividades posteriores.

### 4.2.2 Segunda atividade

Com a primeira atividade finalizada, espera que o usuário esteja com o conceito do comando MOVER bem claro e definido. Agora, o usuário será apresentado à atividade 2, que, assim como a primeira, possui 3 exercícios, como mostra o fluxograma da Figura 42:

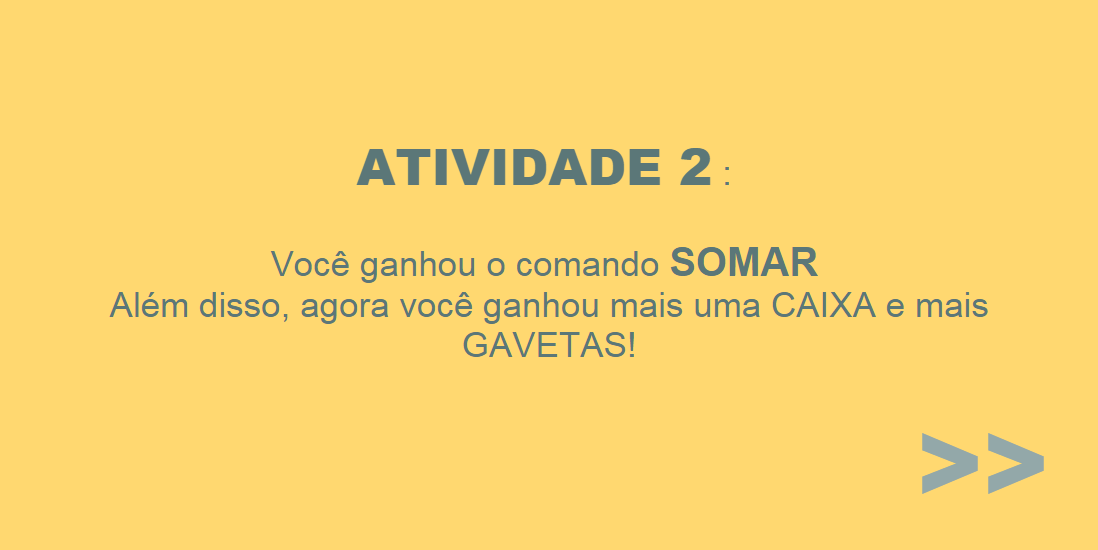
Figura 42 - Fluxograma da Atividade 2.



Fonte: Elaborada pela autora.

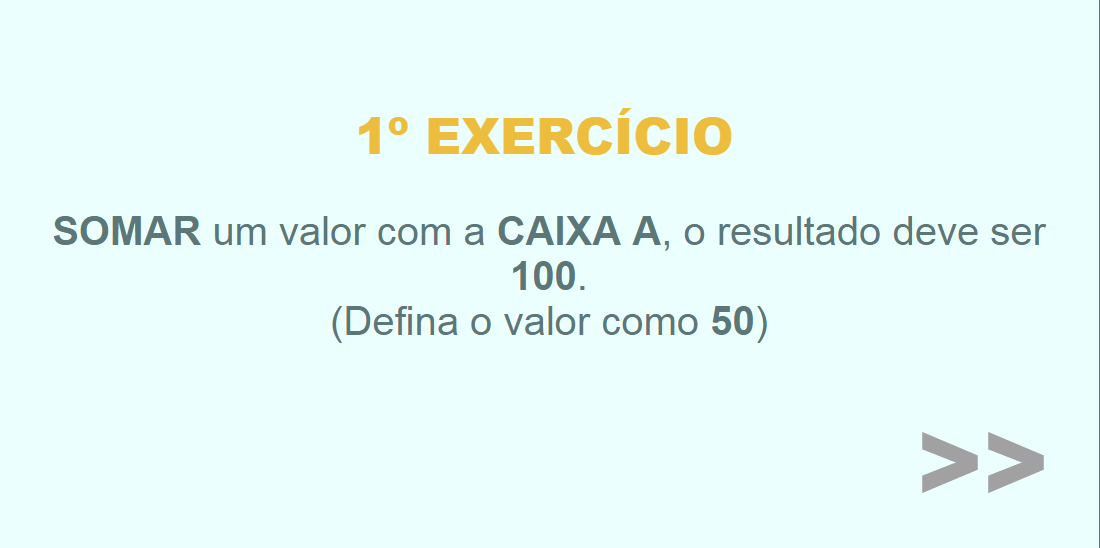
Primeiramente, será apresentada para o usuário a tela da atividade 2, mostrando as três novas recompensas que lhe foram dadas: uma nova caixa “B”, para que ele possa realizar operações mais complexas, mais algumas gavetas e o novo comando SOMAR (Figura 43) e, em seguida, ele será apresentado ao primeiro exercício desta atividade (Figura 44). Após, ele será direcionado para a tela onde ele resolverá exercício em si (Figura 45), no qual ele terá à disposição um tipo do comando SOMAR, o comando de somar o valor de uma das caixas com um valor colocado por ele.

Figura 43 - Atividade 2.



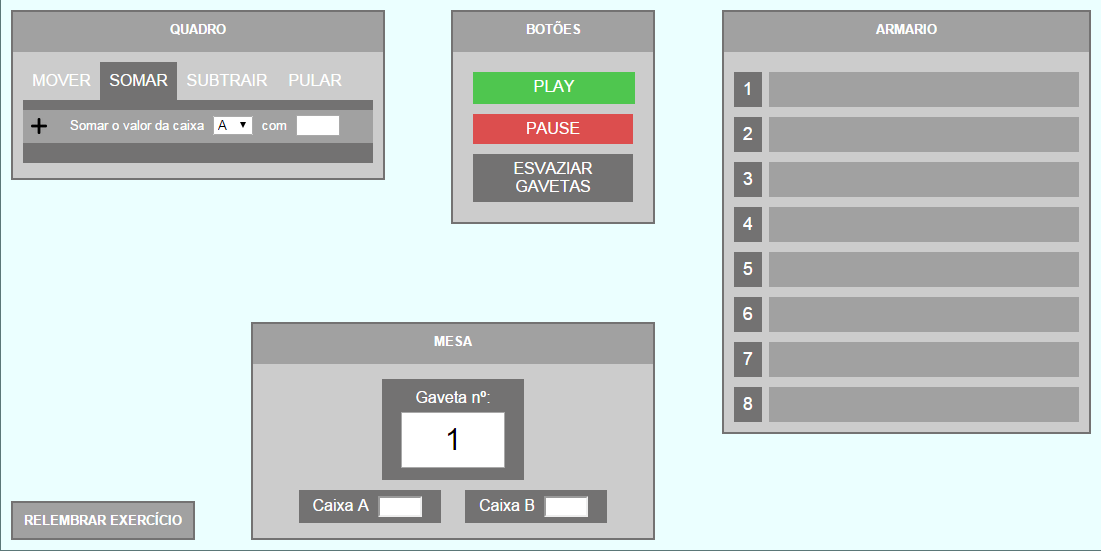
Fonte: Elaborada pela autora.

Figura 44 - Atividade 2: Exercício 1.



Fonte: Elaborada pela autora.

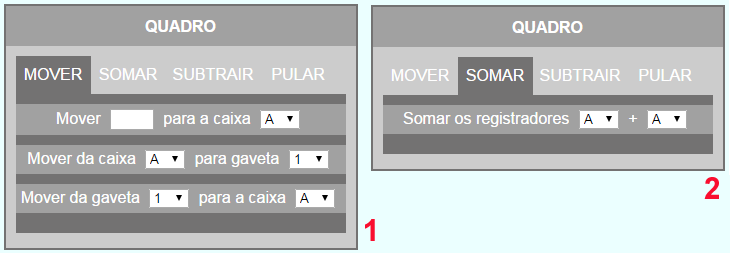
Figura 45 – Tela do Exercício 1.



Fonte: Elaborada pela autora.

Com a inserção de um novo tipo de comando, no caso o SOMAR, existe a necessidade de algum tipo de armazenamento ou agrupamento dos comandos, já que agora o usuário possui mais de um tipo de comando e colocá-los em sequência seria inviável em consequência do tamanho que eles ocupariam. Por isso, os comandos, agora, serão organizados em um formato de abas (Figura 46), que é um conceito no qual a maioria dos jovens usuários estão acostumados, uma vez que é utilizado pela maioria dos principais navegadores de internet atuais.

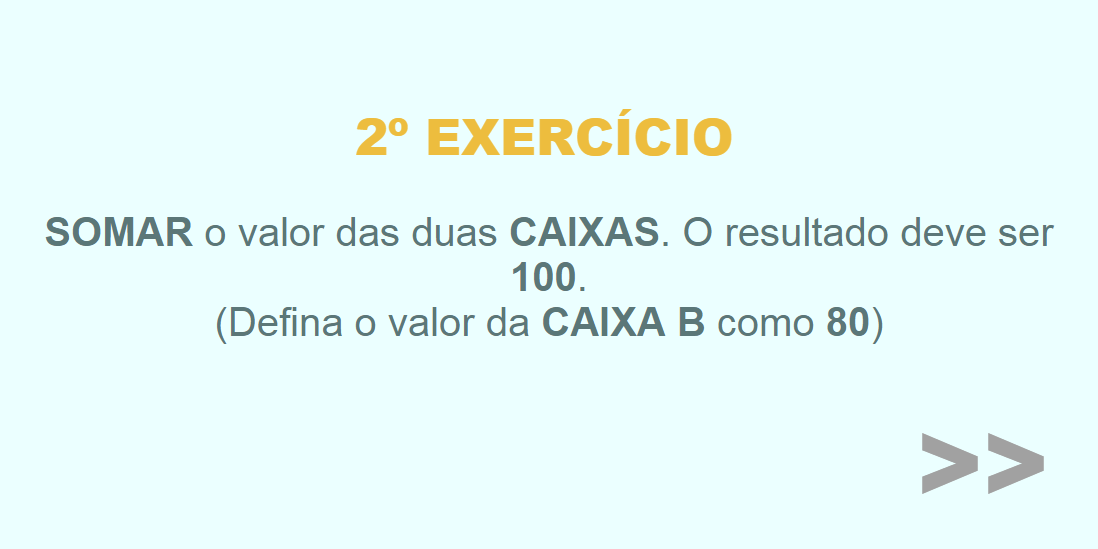
Figura 46 – Os comandos divididos em abas.



Fonte: Elaborada pela autora.

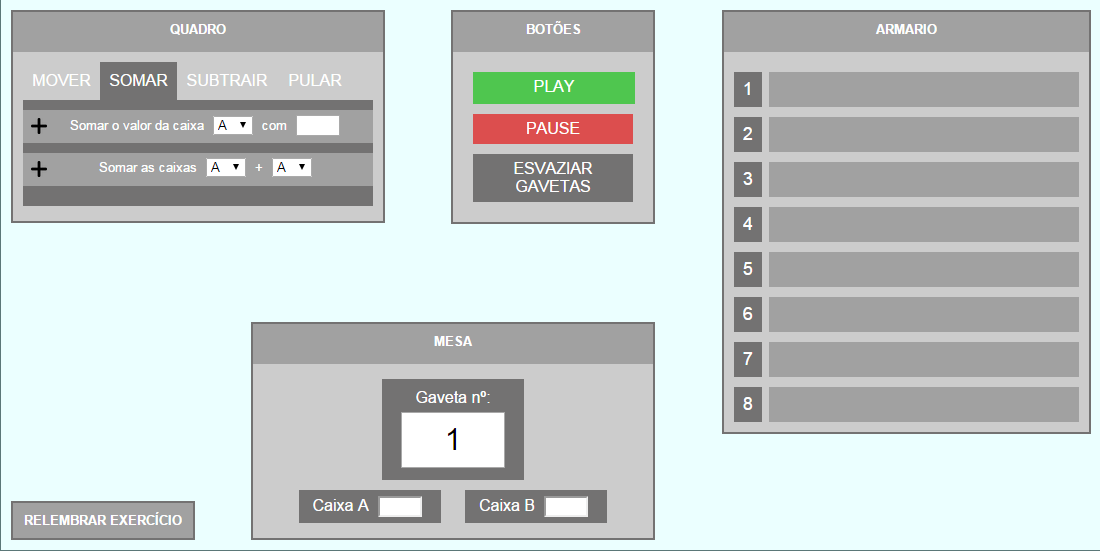
Caso o usuário acerte o exercício, ele será direcionado para a tela de sucesso e, em seguida, para a tela do enunciado do Exercício 2 (Figura 47), e posteriormente para a tela onde ele resolverá exercício em si (Figura 48), no qual ele terá à disposição mais um tipo do comando SOMAR: o comando de somar o valor de duas caixas.

Figura 47 – Atividade 2: Exercício 2.



Fonte: Elaborada pela autora.

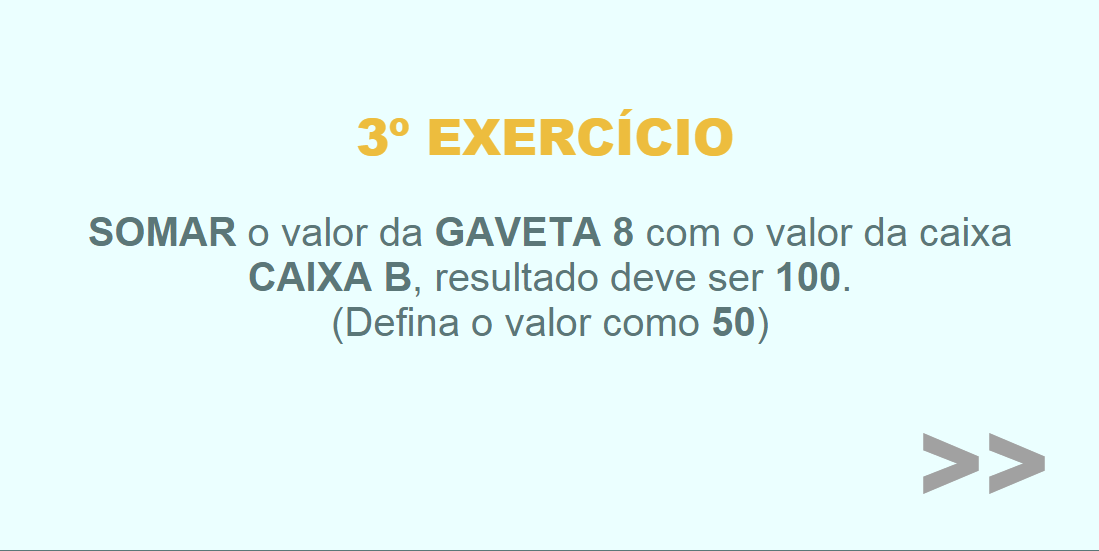
Figura 48 – Tela do Exercício 2.



Fonte: Elaborada pela autora.

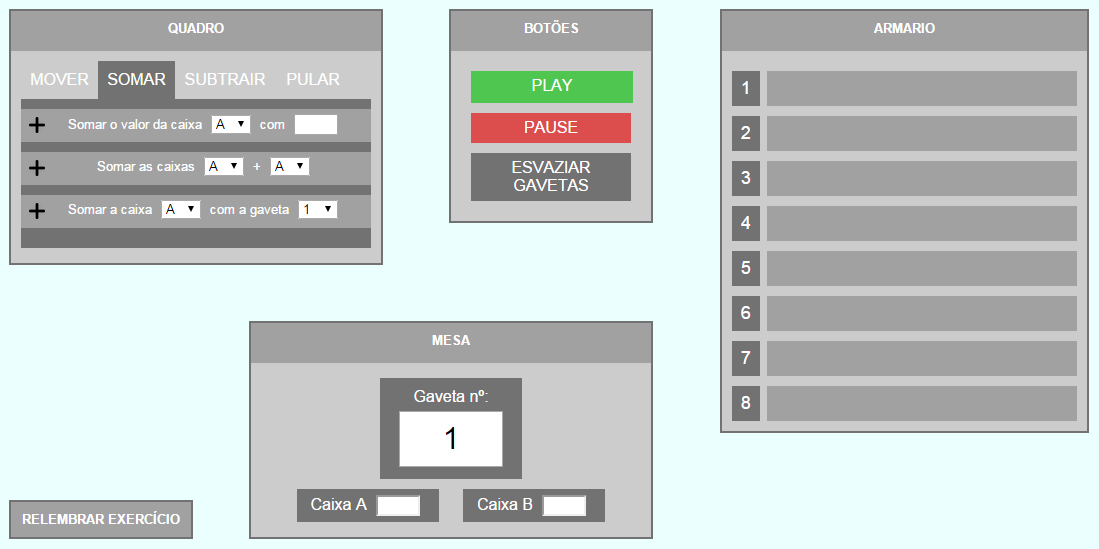
Assim, se o usuário atingir o objetivo deste exercício, ele será direcionado para a tela de sucesso e, em seguida, para a tela do enunciado do Exercício 3 (Figura 49), e posteriormente para a tela onde ele resolverá exercício em si (Figura 50), no qual ele terá à disposição mais um tipo do comando SOMAR, o comando de somar o valor de uma caixa com o valor de uma das gavetas.

Figura 49 – Atividade 2: Exercício 3.



Fonte: Elaborada pela autora.

Figura 50 - Tela Exercício 3.



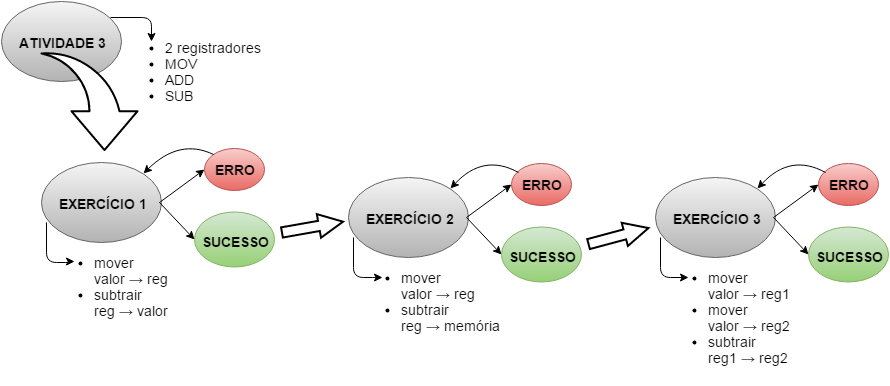
Fonte: Elaborada pela autora.

Se o usuário concluir com êxito o objetivo deste exercício ele será direcionado para a tela de sucesso. Com isso, ele, consequentemente, também finalizará a Atividade 2. Ao encerrar essa atividade, espera-se que o usuário domine os conceitos dos comandos das atividades previamente concluídas, que são os comandos MOVER e SOMAR, para, então, prosseguir com as próximas atividades.

### 4.2.3 Terceira atividade

A terceira atividade tem como objeto ensinar ao aluno a utilizar comando SUBTRAIR, que é semelhante ao comando SOMAR, ensinado anteriormente. Esta atividade possui 3 exercícios, uma vez que pretende-se ensinar 3 modalidades diferentes do comando SUBTRAIR. O fluxograma da Figura 51 representa a atividade 3 e seus exercícios.

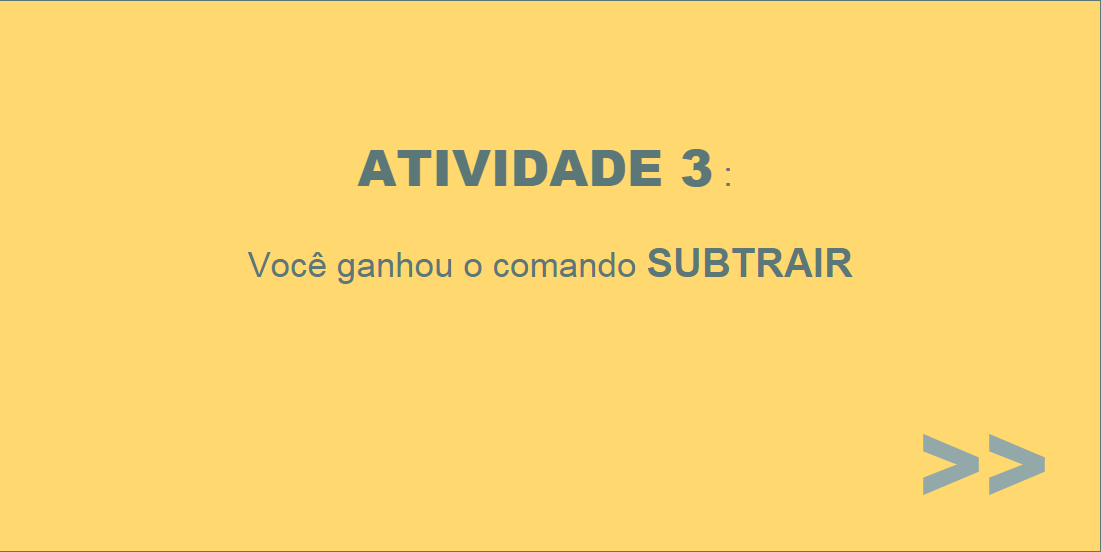
Figura 51 - Fluxograma da Atividade 3.



Fonte: Elaborada pela autora.

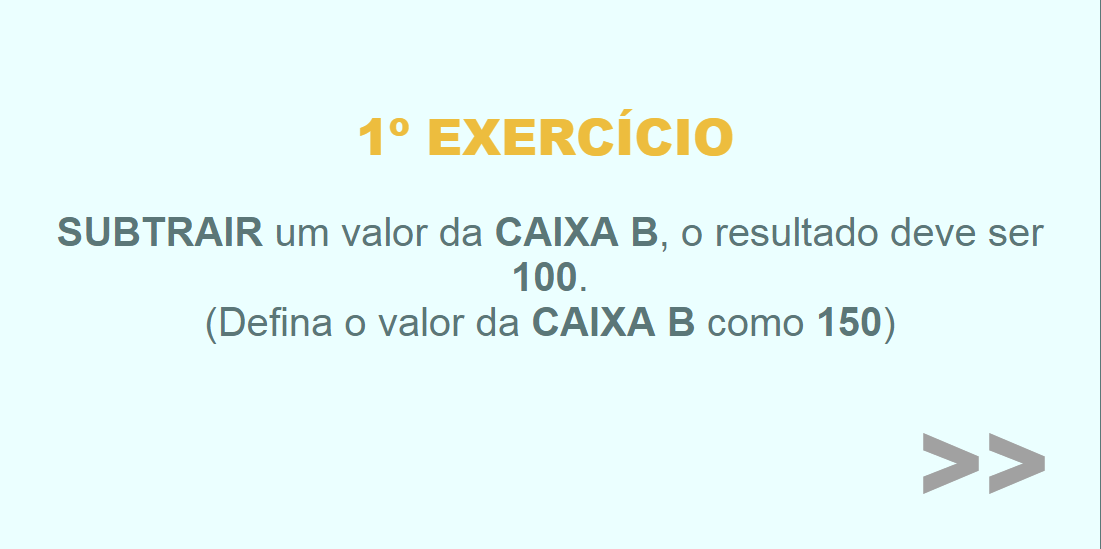
Primeiramente, será apresentada para o usuário a tela da atividade 3, mostrando sua nova recompensa que lhe foi dada: o comando SUBTRAIR (Figura 52) e, em seguida, ele será apresentado ao primeiro exercício desta atividade (Figura 53). Posteriormente, ele será direcionado para a tela onde ele resolverá o exercício em si (Figura 54), no qual ele terá à disposição um tipo do comando SUBTRAIR: o comando de subtrair um valor do valor de uma das caixas.

Figura 52 - Atividade 3.



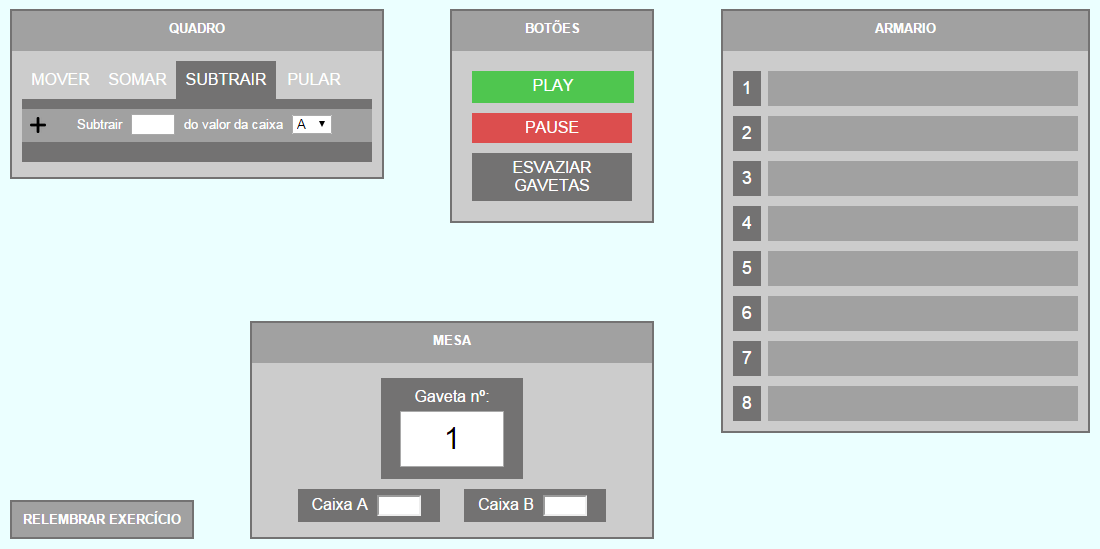
Fonte: Elaborada pela autora.

Figura 53 - Atividade 3: Exercício 1.



Fonte: Elaborada pela autora.

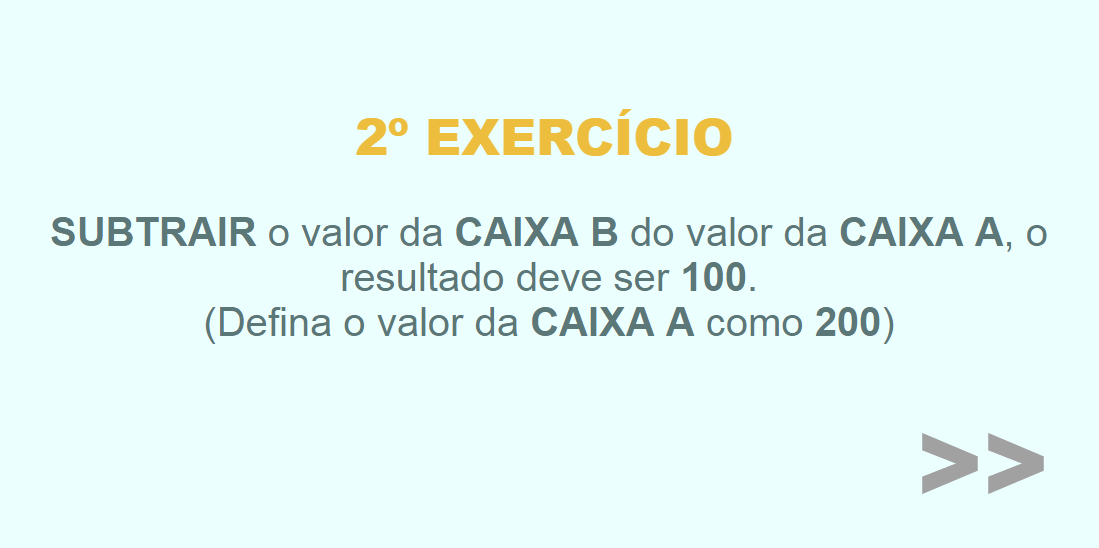
Figura 54 – Tela do Exercício 1.



Fonte: Elaborada pela autora.

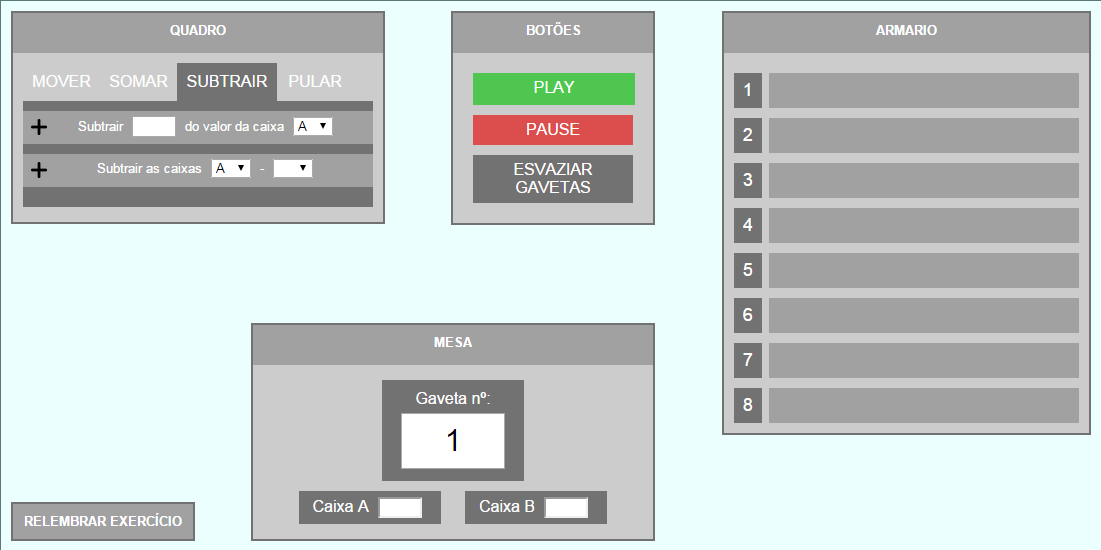
Caso o usuário acerte o exercício, ele será direcionado para a tela de sucesso e, em seguida, para a tela do enunciado do Exercício 2 (Figura 55), e posteriormente para a tela onde ele resolverá exercício em si (Figura 56), no qual ele terá à disposição mais um tipo do comando SUBTRAIR, o comando de subtrair o valor de uma das caixas do valor de uma das gavetas.

Figura 55 – Atividade 3: Exercício 2.



Fonte: Elaborada pela autora.

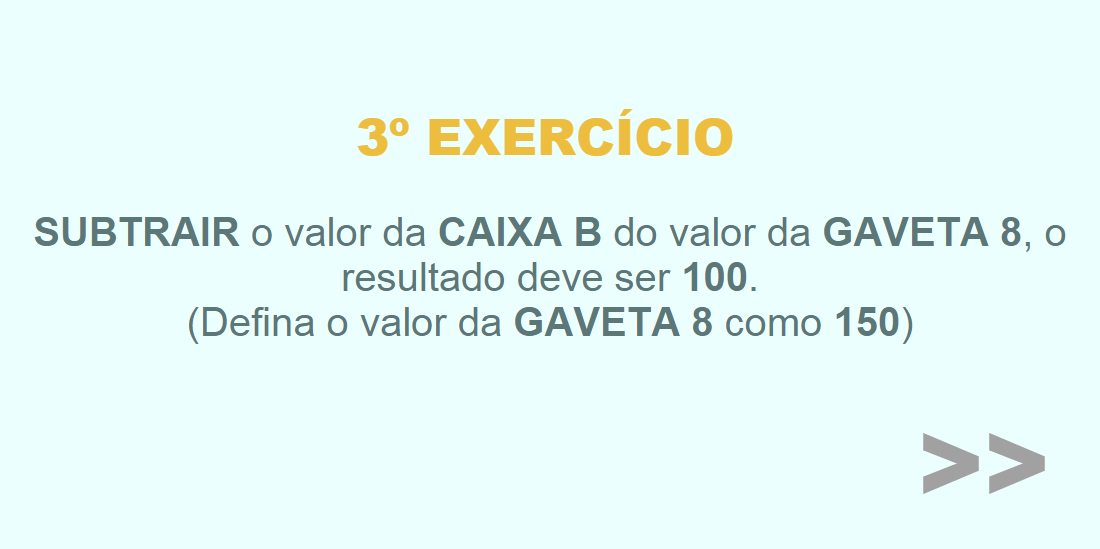
Figura 56 – Tela do Exercício 2.



Fonte: Elaborada pela autora.

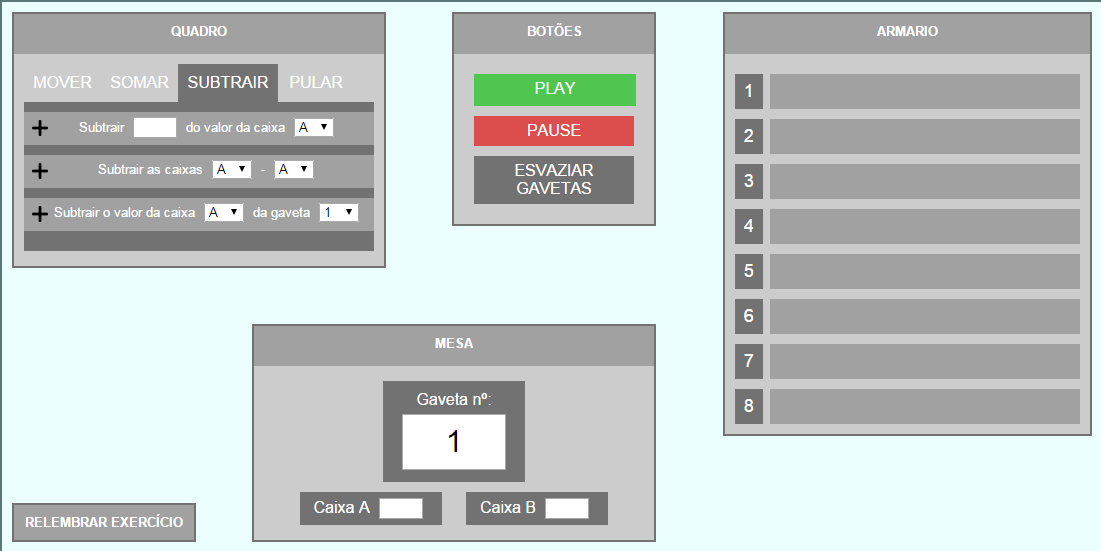
Se o usuário atingir o objetivo deste exercício, ele será direcionado para a tela de sucesso e, em seguida, para a tela do enunciado do Exercício 3 (Figura 57), e, posteriormente, para a tela onde ele resolverá exercício em si (Figura 58), no qual ele terá à disposição mais um tipo do comando SUBTRAIR, o comando de subtrair o valor de uma das caixas do valor de uma das gavetas.

Figura 57 – Atividade 3: Exercício 3.



Fonte: Elaborada pela autora.

Figura 58 - Tela Exercício 3.



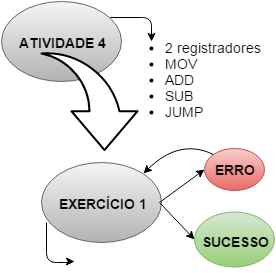
Fonte: Elaborada pela autora.

Com isso, ao estudante finalizar com sucesso este exercício, encerra-se também a Atividade 3. Portanto, supõe-se que o aluno, agora, compreenda os conceitos dos comandos MOVER, SOMAR e SUBTRAIR e seja capaz de utilizá-los em problemas variados.

### 4.2.4 Quarta atividade

Por fim, após passar por 3 atividades, cada uma com 3 exercícios, totalizando 9 exercícios, o aluno entra na última atividade, a qual contém apenas um exercício, que tem como objetivo ensinar o comando de PULAR, como pode-se observar no fluxograma da Figura 59. Para isso, mais uma aba no bloco de comandos é habilitada para abrigar o modelo do novo comando.

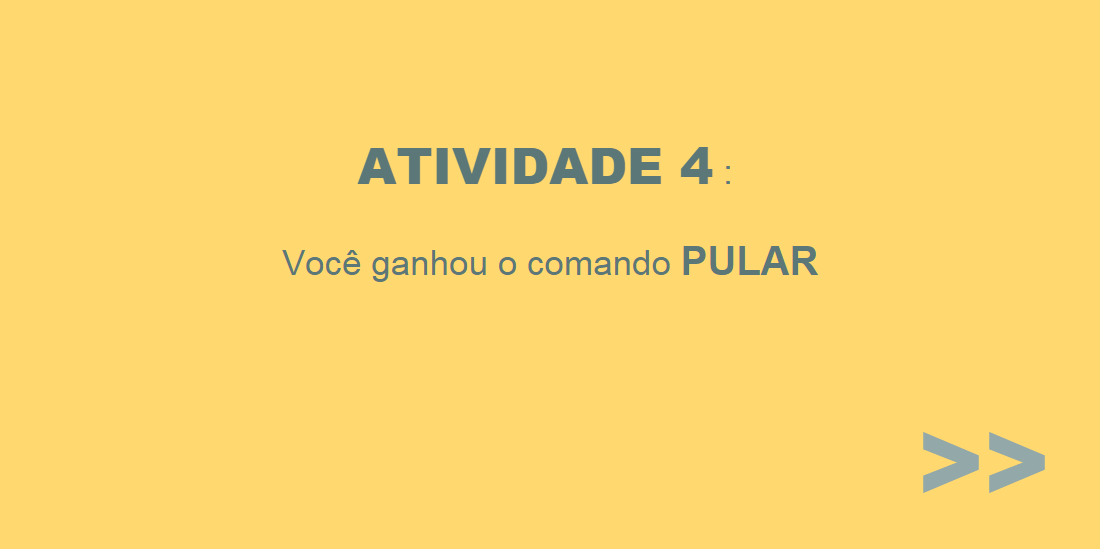
Figura 59 - Fluxograma da Atividade 4.



Fonte: Elaborada pela autora.

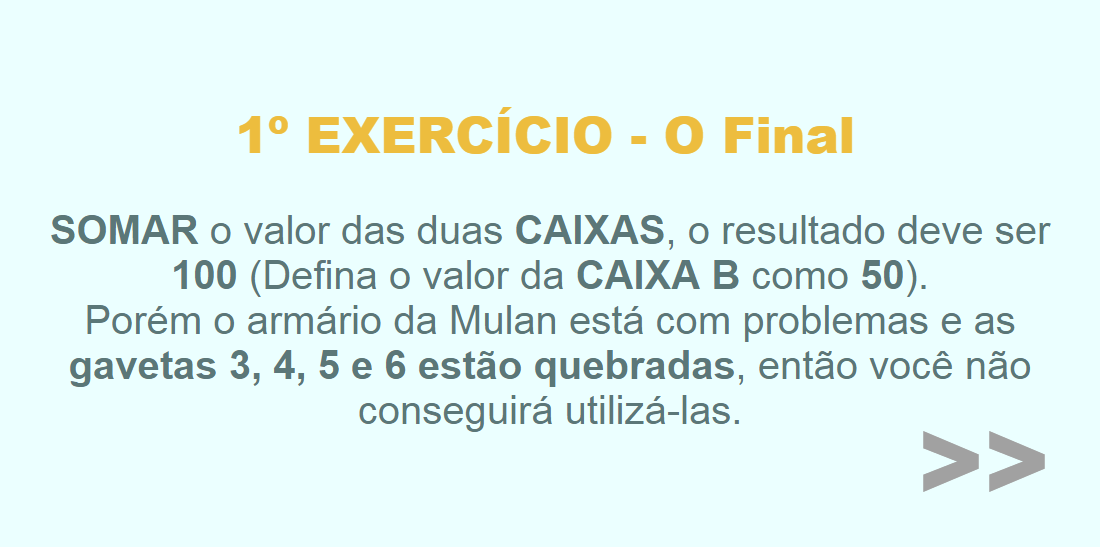
Primeiramente, será apresentada para o usuário a tela da atividade 4, mostrando sua nova recompensa que lhe foi dada: o comando PULAR (Figura 60) e, em seguida, ele será apresentado ao único exercício desta atividade, no qual diz que o usuário não terá todas as gavetas à disposição uma vez que algumas estão quebradas (Figura 61). Posteriormente, ele será direcionado para a tela onde ele resolverá exercício em si (Figura 62), no qual ele terá à disposição um tipo do comando PULAR: ele pode usá-lo para pular da gaveta em que está para uma outra gaveta abaixo desta.

Figura 60 - Atividade 4.



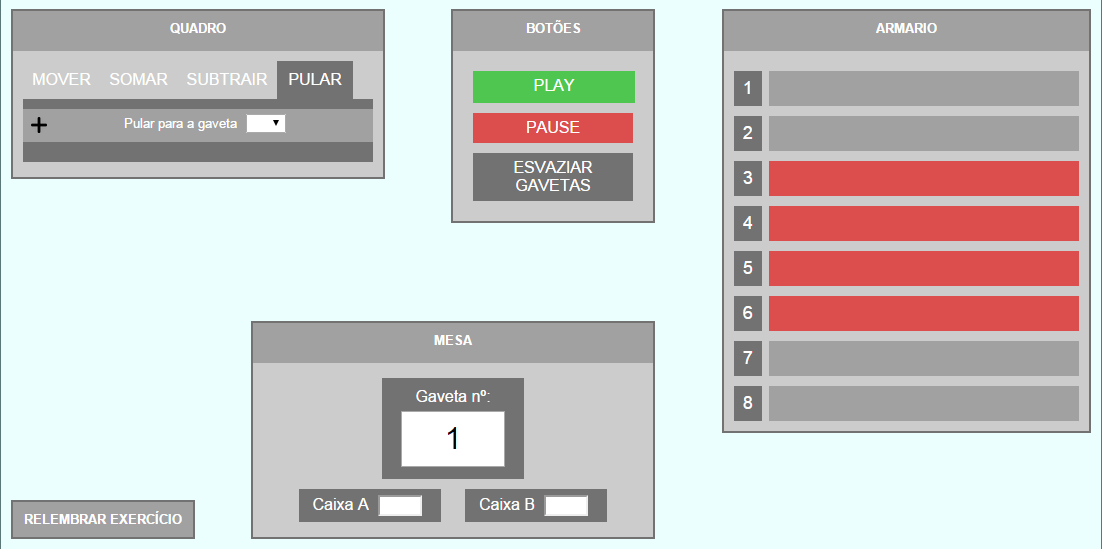
Fonte: Elaborada pela autora.

Figura 61 - Atividade 4: Exercício 1, o exercício final.



Fonte: Elaborada pela autora.

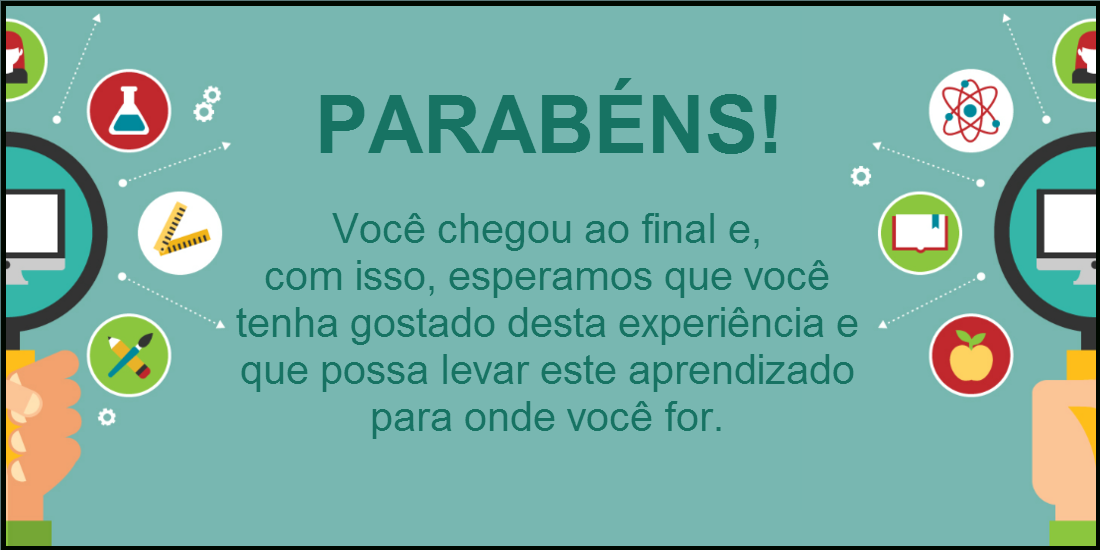
Figura 62 – Tela do Exercício 1, o exercício final.



Fonte: Elaborada pela autora.

Ao resolver com êxito este exercício, o usuário encerra esta última atividade e, consequentemente, o Objeto de Aprendizagem em si. Ele será direcionado para a tela final deste Objeto de Aprendizagem, parabenizando-o (Figura 63).

Figura 63 – Tela Final.



Fonte: Elaborada pela autora.

Por fim, com a execução de todas estas atividades e exercícios, espera-se que o aluno compreenda melhor alguns conceitos lógicos como MOVER, SOMAR, SUBTRAIR e PULAR, utilizar o conceito de algoritmo, mesmo que ele não saiba o que é, além treinar a habilidade de, utilizando o pensamento computacional, solucionar problemas genéricos que, à primeira vista, podem parecem muito complexos e abstratos, mas que se disponibilizados aos poucos fazem com que o aluno aprenda gradativamente.

# 5 conclusão

Este trabalho tem como objetivo implementar um protótipo de um Objeto de Aprendizagem no apoio ao ensino de lógica computacional, a fim de diminuir a barreira existente entre alunos e educadores, dado a grande diferença existente entre estas gerações em relação à afinidade tecnológica de cada uma delas , e contribuir com a aprendizagem dos alunos sobre o pensamento computacional, seja na educação básica ou na educação superior, uma vez que o pensamento computacional é essencial no desenvolvimento do raciocínio lógico de um indivíduo.

Para atingir tais objetivos, este Objeto de Aprendizagem conta com uma interface simplificada de uma analogia que inicialmente foi baseada no funcionamento de um computador, a fim de inserir um ambiente mais intuitivo ao aluno, ou seja, o objetivo de colocar uma analogia neste Objeto de Aprendizagem é justamente utilizar um conceito provavelmente conhecido pelo aluno para ensinar algo que é desconhecido para ele.

O Objeto de Aprendizagem em si é dividido em atividades com um ou mais exercícios que procuram estimular o aprendizado progressivo do aluno. Essas atividades e exercícios foram baseadas no conceito de certos comandos lógicos e alguns de seus diferentes modos de serem usados.

Com isso, espera-se que haja um desenvolvimento do pensamento computacional no aluno, seja ele um aluno do ensino básico ou do ensino superior, uma vez que essa habilidade é essencial qualquer indivíduo. No âmbito da computação, o desenvolvimento desta habilidade pode acarretar em uma menor dificuldade dos alunos em conceitos básicos ligados à programação, diminuindo o número de reprovas nestas disciplinas introdutórias e, consequentemente, uma redução na evasão de alunos em cursos de computação.

Por fim, pode-se concluir que este Objeto de Aprendizagem tem a possibilidade de cumprir os objetivos propostos caso seja realmente inserido dentro da sala de aula como uma ferramenta de apoio ao educador no ensino de conceitos ligados ao pensamento computacional e, consequentemente, o raciocínio lógico.

## 5.1 Trabalhos futuros

Este Objeto de Aprendizagem aqui desenvolvido é um protótipo. Com isso, espera-se que futuramente, este protótipo seja implementado em conjunto com a analogia proposta para, assim, ser efetivamente utilizado em sala de aula a fim de constatar se este trabalho realmente atinge os objetivos aqui propostos.

Além disso, planeja-se que, futuramente, algumas funcionalidades sejam incluídas neste Objeto de Aprendizagem, a fim de melhorar a interação do educador com o programa, tendo como objetivo tornar esta ferramenta mais flexível. Isto provavelmente seria possível com a criação de uma interface especifica para o professor, no qual ele poderia elaborar as atividades propostas de acordo com os conceitos que ele deseja ensinar e não só seguir um roteiro pronto, abrindo, assim, uma diversidade de possibilidades no aprendizado em sala de aula.

# Referências

AMIEL, T.; OREY, M.; WEST, R. **Recursos Educacionais Abertos (REA):** modelos para localização e adaptação.In: **ETD – Educação Temática Digital** Campinas, SP, v. 12, p. 112-125, mar.2011. ISSN-2592. Disponível em: <http://periodicos.bc.unicamp.br/ojs/index.php/etd/article/view/1206>. Acesso em: 24 mai. 2015.

ASCENCIO, A. F. G.; CAMPOS, E. A. V. **Fundamentos da Programação de Computadores.** 3. ed. São Paulo: Editora da UFABC, 2014.

BEZERRA, V. A. **Estruturas conceituais e estratégias de investigação:**  modelos representacionais e instancias, analogias e correspondência. **Scientia Estudia**, São Paulo, SP, v. 9, n. 3, p. 585-609, 2011. Disponível em: <http://www.scielo.br/scielo.php?pid=S1678-31662011000300007&script=sci\_arttext>. Acesso em: 20 mai. 2015.

BITESIZE. **Introduction to computacional thinking**. [201-]. Disponível em: < http://www.bbc.co.uk/education/guides/zp92mp3/revision/1>. Acesso em: 15 dez. 2015.

BLIKSTEIN, P. **O pensamento computacional e a reinvenção do computador da educação**. Dez. 2008. Disponível em: <http://www.blikstein.com/paulo/documents/online/ol\_pensamento\_computacional.html>. Acesso em: 25 nov. 2015.

BRAGA, J. **Objetos de Aprendizagem:** Metodologia e Desenvolvimento. 1. ed. Norristown: Springer Science & Business Media, 2007.

BROOKS, D. R. **An Introduction to HTML and JavaScript:** for Scientists and Engineers. 3. ed. Norristown: Springer Science & Business Media, 2007.

CHELLA, M. T.; OLIVEIRA, C. M.; SILVA, J. C. O. **Simulador de Robô para Auxílio ao Ensino de Programação**. In: **EATI – Encontro Anual de Tecnologia da Informação.** 4., 2014, Campinas, p. 96-102, nov.2014. ISSN-2236-8604. Disponível em: <http://www.eati.info/eati/2014/assets/anais/artigo11.pdf>. Acesso em: 10 abri. 2015.

CRAWFORD JR, R. A; COPP, D. H. **Introdução à programação de computadores**. Tradução: Luiz Ignácio Pio de Almeida. Porto Alegre: Editora Globo, 1974.

DICIONÁRIO Michaelis. 2009. Disponível em: <http://michaelis.uol.com.br/moderno/portugues/index.php?lingua=portugues-portugues&palavra=analogia>. Acesso em: 20 out. 2015.

DUARTE, M. C. **Analogias na Educação em Ciências Contributos e Desafios,** 2004. Instituto de Educação e Psicologia, Universidade do Minho, Braga, Portugal. Disponível em: <http://www.if.ufrgs.br/public/ensino/vol10/n1/v10\_n1\_a1.htm>. Acesso em: 26 mai. 2015.

FARRER, H. et al. **Programação estruturada de computadores:** algoritmos estruturados. 3. ed. Rio de Janeiro: Livros Técnicos e Científicos, 1989.

FONSECA FILHO, C. **História da Computação:** O caminho do pensamento e da tecnologia. Porto Alegre: Livros Técnicos e Científicos, 2007. Disponível em: <<http://www.pucrs.br/edipucrs/online/historiadacomputacao.pdf>>. Acesso em: 20 abri. 2015.

GECK, C. **The Generation Z Connection**: Teaching Information Literacy to the Newest Net Generation. Fev. 2006. Disponível em: <http://www.redorbit.com/news/technology/397034/the\_generation\_z\_connection\_teaching\_information\_literacy\_to\_the\_newest/ >. Acesso em: 25 mai. 2015.

GIL, A. C. **Como elaborar Projetos de Pesquisa**. 4. ed. São Paulo: Editora Atlas, 2002.

GIRAFFA, L. M. M.; MORA, M. C. **Evasão da disciplina de algoritmo e programação:** Um estudo a partir dos fatores intervenientes na perspectiva do aluno. In: **III CLABES – Tercera Conferencia Latinoamericana sobre el abandono em la educación superior.** 3., 2013, Ciudad de México, nov.2013. Disponível em: <http://www.alfaguia.org/www-alfa/images/ponencias/clabesIII/LT\_1/ponencia\_completa\_136.pdf>. Acesso em: 28 abri. 2015.

GOMES, A. et al. **Aprendizagem de programação de computadores:**  dificuldades e ferramentas de suporte. **Revista Portuguesa de Pegadogia**, Coimbra, v. 42, n. 2, p. 161-179, 2008. Disponível em: <http://iduc.uc.pt/index.php/rppedagogia/article/view/1242>. Acesso em: 25 mai. 2015.

GRANNEL, C; SUMNER, V; SYNODINOS, D. **The Essential Guide to HTML5 and CSS3 Web Design**. friends of ED, 2012.

HINTERHOLZ, L. T.; CRUZ, M. E. K. **Desenvolvimento do Pensamento Computacional:** Um relato de atividade junto ao Ensino Médio, através do Estágio Supervisionado em Computação III. In: **XXI WIE – Workshop de Informática na Escola.** Maceió, p. 137-146, 2015. Disponível em: <http://br-ie.org/pub/index.php/wie/article/view/5008>. Acesso em: 15 dez. 2015.

ISTE, International Society for Technology in Education; CSTA, Computer Science Teachers Association. **Operation Definition of Computational Thinking**. 2011. Disponível em: <https://www2.cjf.jus.br/jspui/handle/1234/5522>. Acesso em: 15 dez. 2015.

MALDONADO, M. T. **A geração Y no trabalho:** Um desafio para os gestores. **BIGJus – Boletim de Informações Gerenciais da Justiça Federal**, v. 3, n. 9, mai. 2009. Disponível em: <https://www2.cjf.jus.br/jspui/handle/1234/5522>. Acesso em: 10 mai. 2015.

MARTINS, C. A.; GIRAFFA, L. M. M. **Formação do Docente Imigrante Digital para atuar com Nativos Digitais no Ensino Fundamental**. Porto Alegre: Pontifícia Universidade Católica do Rio Grande do Sul,2009. Disponível em: <http://www.pucpr.br/eventos/educere/educere2008/anais/pdf/132\_220.pdf>. Acesso em: 26 mai. 2015.

MEDINA, M; FERTIG, C. **Algoritmo e Programação:**  Teoria e Prática. 4. ed. São Paulo: Novatec Editora, 2005.

MURDOCCA, M. J; HEURING, V. P. **Introdução à Arquitetura de Computadores**. Tradução: Sergio Vale Aguiar Campos. 5. ed. São Paulo: Prentice Hall, 2000.

NAGEN, R. L.; CARVALHAES, D. O.; DIAS, J. A. Y. T. **Uma proposta de metodologia de ensino com analogias**. **Revista Portuguesa de Educação**, Braga, v. 14, n. 1, p. 197-213, 2001. Disponível em: <http://files.ensinodecienciasnaamazonia.webnode.com/200000028-d2c54d3bf4/Nagem%20et%20al\_2001\_Proposta\_de\_ensino\_atraves\_de\_analogias.pdf>. Acesso em: 20 out. 2015.

OLIVEIRA, H. R. **Argumentação no Ensino de Ciências:** O uso de analogias como recurso para a construção do conhecimento. 2012. 130 f. Dissertação (Pós-Graduação em Educação) - Universidade Federal de Juiz de Fora, Juiz de Fora, 2012.

PERELMAN, C.; OLBRECHTS-TYTECA, L. **Tratado de Argumentação**: nova retórica. 2. ed. São Paulo: Martins Fontes, 2005.

PRATA, C. L.; NASCIMENTO, A. C. A. A. **Objetos de Aprendizagem:** Uma proposta de recurso Pedagógico. Brasília: MEC, SEED, 2007.

PRENSKY, M. **Nativos Digitais, Imigrantes Digitais**. Tradução: Roberta de Moraes Jesus de Souza. Horizon: NBC Press, v. 9, n. 5, out. 2001. Disponível em: <http://poetadasmoreninhas.pbworks.com/w/file/fetch/60222961/Prensky%20-%20Imigrantes%20e%20nativos%20digitais.pdf>. Acesso em: 25 abr. 2015.

QUEST Intelingência de Mercado. **Geração Z é a mais conectada, fuma menos e lê pouco, revela pesquisa**. Ago. 2011. Disponível em: < http://www.questmkt.com.br/questblog/?s=gera%C3%A7%C3%A3o+z>. Acesso em: 15 dez. 2015.

RAPKIEWICZ, C. E. et al. **Estratégias pedagógicas no ensino de algoritmo e programação associadas ao uso de jogos educacionais**. Porto Alegre: Renote, 2006.

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**. In: **INFOCOMP – Journal of Computer Science**, v. 5, n. 1, p. 41-50, jan. 2006. Disponível em: <http://professores.dcc.ufla.br/ojs/index.php/INFOCOMP/article/view/121>. Acesso em: 10 jun. 2015.

SICA, C. **Ciência da computação no ensino básico**. Out. 2011. Disponível em: <http://blogs.odiario.com/carlossica/2011/10/07/ciencia-da-computacao-no-ensino-medio/>. Acesso em: 28 nov. 2015.

SILVA, M. S. **jQuery:** A Biblioteca do Programador JavaScript. São Paulo: Editora Novatec, 2008.

SILVA, E. L.; CAFÉ, L; CATAPAN, A. H. **Os objetos educacionais, os metadados e os repositórios na sociedade da informação**. In: **Ciência da Informação**, Brasília, v. 39, n. 3, p. 93-101, dez. 2010. Disponível em: <http://professores.dcc.ufla.br/ojs/index.php/INFOCOMP/article/view/121>. Acesso em: 5 jun. 2015.

SOCIEDADE BRASILEIRA DE COMPUTAÇÃO. **Plano de Gestão 2013 – 2015**. [2013]. Disponível em: <http://www.sbc.org.br/index.php?option=com\_content&view=article&id=82:descricao&catid=215:plano-acao&Itemid=82>. Acesso em: 27 set. 2015.

SOUZA, J. N. **Lógica para ciência da computação e áreas afins**. 3. ed. Rio de Janeiro: Elsevier, 2015.

STALLINGS, William. **Arquitetura e Organização de Computadores**. Tradução: Carlos Camarão de Figueiredo e Lucília Camarão de Figueiredo. 5. ed. São Paulo: Prentice Hall, 2002.

TANENBAUM, A. S. **Organização estruturada de computadores.** Tradução: Wagner Luiz Zucchi. 5. ed. São Paulo: Pearson Prentice Hall, 2007.

TAROUCO, L. M. R. et al. **Formação de professores para produção e uso de objetos de aprendizagem.** In: **RENOTE – Revista Novas Tecnologias na Educação**, v. 4, n. 1, p. 1-10, jul. 2006. Disponível em: <http://www.seer.ufrgs.br/renote/article/download/13886/7802>. Acesso em: 21 jun. 2015.

TERRAZZAN, E. A. et al. **Analogias do ensino de ciência:**  resultados e perspectivas. In: **Anais do III Seminário de Pesquisa em Educação da Região Sul**, 2000. Disponível em: <http://www.portalanpedsul.com.br/admin/uploads/2000/Ensino\_e\_curriculo/Comunicacao/08\_23\_33\_2224.pdf>. Acesso em: 15 abri. 2015.

WING, J. M. **Computational Thinking**. In: **Communications of the ACM**, v. 49, n. 3, p. 33-35, mar. 2006. Disponível em: <http://www.seer.ufrgs.br/renote/article/download/13886/7802>. Acesso em: 20 dez. 2015.

YUNG, R. **Sublime Text 2**. Jul. 2013. Disponível em: <http://codigofonte.uol.com.br/ferramentas/sublime-text-2>. Acesso em: 25 jan. 2016.

1. Disponível em:< http://ciaderesultados.com.br/aprenda-a-lidar-com-a-geracao-z/ > Acesso em 24 abr. 2015 [↑](#footnote-ref-1)
2. Disponível em:< <http://www.questmkt.com.br/questblog/?s=gera%C3%A7%C3%A3o+z> > Acesso em 26 abr. 2015 [↑](#footnote-ref-2)
3. Disponível em: < <http://www.bbc.co.uk/education/guides/zp92mp3/revision/1> > Acesso em 13 dez. 2015 [↑](#footnote-ref-3)