UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA – UESB
DEPARTAMENTO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LUCAS RODRIGUES FERREIRA

UMA PROPOSTA PARA O ENSINO DOS CONCEITOS INTRODUTÓRIOS DE PROGRAMAÇÃO UTILIZANDO LINGUAGEM VISUAL EM BLOCOS E METODOLOGIA ATIVA

Vitória da Conquista

2018

Lucas Rodrigues Ferreira

UMA PROPOSTA PARA O ENSINO DOS CONCEITOS INTRODUTÓRIOS DE PROGRAMAÇÃO UTILIZANDO LINGUAGEM VISUAL EM BLOCOS E METODOLOGIA ATIVA

Trabalho de Conclusão de curso, apresentado ao curso de Bacharelado em Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, em Vitória da Conquista, como requisito para obtenção do título de Bacharel em Ciências da Computação.

Orientadora: Prof. Dra. Maísa Soares dos Santos Lopes

Coorientadora: Prof. Dra. Alzira Ferreira da Silva

Vitória da Conquista

2018

Este trabalho é dedico à minha família e amigos

AGRADECIMENTOS

Mais uma etapa da jornada chega ao fim, e compartilhar esse sentimento com quem está perto e admiramos é excelente. Nesses 4 anos aprendi sobre computação, mas além disso aprendi a ser uma pessoa melhor.

Agradeço a Deus, pelo auxílio, força e ânimo que me conduziu durante esse tempo, porque por Ele e para Ele são todas as coisas. Agradeço à toda minha família, principalmente meus pais, que abriram mão de alguns sonhos para auxiliar nos alcance dos meus. Ao meu irmão, que me ensinou sobre a perseverança nos estudos durante essa caminhada.

Agradeço aos meus amigos do curso, em especial à Dalton, Gustavo, Pablo, Raphael e Matheus Sales, que enfrentaram comigo os desafios dessa etapa e estão comigo desde o primeiro semestre; a todos do laboratório LINDALVA (Laboratório de Inteligência em Dispositivos de Arquitetura Livre e Veículos Autônomos) e do LARA (Laboratório Remoto em AVA); Aos formados Iago, Yan, Matheus Lima e Matheus Thiago que ajudaram nos desafios desse tempo e são exemplos para mim; A Celina, que é uma pessoa sem igual no curso de computação; A todos os professores do curso, em especial as professoras Maísa e Alzira, por toda orientação durante o trabalho final e por ter me acompanhado durante essa etapa da vida, parabéns pelas professoras que são. Aos professores, equipe e amigos do Centro Juvenil de Ciência e Cultura, que me acolheram e me ensinaram muito sobre a educação e a vida, em especial a professora Elmara um grande exemplo pra mim; Ao meu amigo Caio por pela força e acompanhamento antes e durante essa caminhada; e a minha amada Heloísa, que desde o início, esteve presente, mostrando compreensão e conforto nas partes mais desafiadoras desse percurso, ela colocou mais brilho nos meus dias, e me ajudou a conhecer mais de mim mesmo e me tornar melhor, obrigado.

Enfim, vocês fizeram parte de toda essa jornada e dessa conquista. Muito obrigado à todos!

RESUMO

A lógica de programação tem um papel de suma importância nos cursos de computação, por isso é uma disciplina obrigatória oferecida logo nos semestres iniciais. Desse modo, surge vários obstáculos no ensino/aprendizagem de programação, enfrentados pelos estudantes e professores. Ao longo do anos, pesquisas têm sido realizadas, e uma gama de literatura produzida sobre o tema. Contudo, as dificuldades do componente curricular ainda persistem nos cursos de computação, o que pode ser percebido pelos índices de reprovação e desistência pelos discentes. Nesse sentido é necessário trabalhar esses conceitos de uma forma diferente, possibilitando o aluno desenvolver habilidades para interpretação e resolução de problemas, de modo que consiga compreender a lógica e aplicar nas linguagens de programação. Para isso é necessário utilizar ferramentas que minimizem as dificuldades relacionadas a sintaxe e possibilite um melhor desenvolvimento do raciocínio lógico pelo discente. As metodologias ativas também contribuem para o aprendizado, uma vez que o coloca o estudante como membro ativo no processo de ensino/aprendizagem. Este trabalho avalia o uso de ferramentas em programação em blocos e o uso de metodologia ativa para o ensino de programação no curso de ciência da computação, da Universidade Estadual do Sudoeste da Bahia campus Vitória da Conquista.

Palavras-chaves: Aluno. Aprendizagem. Metodologia ativa. Programação em blocos. Problemas.

ABSTRACT

The logic of programming has a role of paramount importance in computer courses, so it is a compulsory subject offered in the initial semesters. In this way, several obstacles in the teaching / learning of programming arise, faced by students and teachers. Over the years, research has been conducted, and a range of literature produced on the subject. However, the difficulties of the curricular component persist in the courses of computation, which can be perceived by the indexes of disapproval and withdrawal by the students. In this sense it is necessary to work on these concepts in a different way, allowing the student to develop skills for interpretation and problem solving, so that he can understand the logic and apply in programming languages. For this it is necessary to use tools that minimize the difficulties related to syntax and enable a better development of the logical reasoning by the student. Active methodologies also contribute to learning, since it places the student as an active member in the teaching / learning process. This work evaluates the use of tools in block programming and the use of an active methodology for teaching programming in the course of computer science, Universidade Estadual do Sudoeste da Bahia, campus Vitória da Conquista.

Key words: Student. Learning. Active Methodology. Programming in blocks. Problems.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

A computação está presente nos diversos aspectos da vida humana. Ela permeia as atividades domésticas, empresariais, artísticas, tecnológicas, dentre outras. Nesse sentido, pode-se observar que a computação contribui nas várias áreas de conhecimento, auxiliando na resolução de problemas computacionais (FRANÇA e AMARAL, 2013). Diante disso, é necessário um bom preparo dos estudantes da área de computação, principalmente no desenvolvimento do raciocínio lógico e na resolução de problemas através da prática de programação. Isso porque o ensino de programação visa despertar no aluno as competências necessárias para desenvolver sistemas que solucionam problemas reais.

Contudo, guiar o discente para o aprendizado dos conceitos introdutórios nesse campo de estudo, nem sempre é uma tarefa fácil. Segundo Gomes et al. (2008), os níveis de insucesso nas disciplinas introdutórias têm sido objeto de pesquisas e discussões ao longo dos tempos. De acordo com Efoupoulos et al. (2005), a programação, aparentemente, é umas das habilidades mais difíceis de se dominar, por ser uma matéria que requer maior esforço e disciplina do aprendiz.

Em um estudo realizado em nove instituições, de seis países, os pesquisadores identificaram que muitos alunos passavam do primeiro ou segundo ano do curso sem saber programar (MCCRACKEN et al., 2001). Umas das principais barreiras apresentadas no processo de aprendizagem mostra que os alunos possuem dificuldades em dividir o problema em subproblemas, achar uma solução e uni-las, de modo a formar uma solução completa (GOMES, 2010).

Além disso, os estudos em programação tendem a exigir do estudante a formulação de conceitos abstratos, tais como variáveis, estruturas condicionais, estruturas de repetição e tipos de dados, que permitem encontrar a solução para um problema real. Porém, compreender esses conceitos e modelá-los de forma computacional, pensando no mundo real, não é fácil. Nesse sentido, alguns ambientes de programação propendem a deixar o aprendizado difícil, por conta de todas as características próprias. Inclusive, as próprias linguagens de programação, por conta das estruturas sintáticas, dificultam o processo de aprendizagem (GOMES, HENRIQUES e MENDES, 2008). Nesse sentido, McCracken et al. (2001) demonstram em sua pesquisa, que os alunos tendem a focar na implementação e muitas vezes se perdem na sintaxe, o que atrapalha o aprendizado no planejamento e busca para a solução de um determinado problema.

Dessa forma, os obstáculos impostos no desenvolvimento de habilidades para programar trazem como consequência a falta de interesse e desânimo no processo de aprendizagem. Isso gera altos índices de reprovação nas disciplinas introdutórias de programação, como demonstrado por Santos et al. (2015) na Universidade Federal do Rio Grande do Norte (UFRN). Nesse sentido, de acordo com Efopoulos et al. (2005), o aprendizado dos conceitos introdutórios e fundamentais de algoritmos e programação, não devem ser prejudicados pelo tempo empenhado no estudo de uma nova linguagem e sua sintaxe.

Desse modo, diversos tipos de ambientes e ferramentas visuais, como é o caso da programação em blocos, têm sido desenvolvidos para auxiliar os estudantes de computação no progresso com o aprendizado da lógica e a desenvolver a capacidade de solucionar problemas, como por exemplo, o Scratch, code.org e o app inventor (GOMES et al., 2008).

Não apenas tais ferramentas têm sido produzidas, como o uso de laboratórios virtuais se tornou comum no ensino de conteúdo das diversas áreas de conhecimento, incluindo a ciência da computação. Esses ambientes virtuais oferecem suporte de aprendizagem, permitindo ao aluno a prática dos assuntos vistos na sala de aula (TRENTIN, PÉREZ e SANTOS, 2002).

Além dos laboratórios virtuais, existem os laboratórios remotos, que também oferecem meios de aprendizagem através da aplicação do conhecimento adquirido nas aulas. Conforme Lopes (2017), os laboratórios remotos diferem dos virtuais porque utilizam componentes reais em um lugar diferente de onde são manipulados. Dessa forma, o discente observa como o aprendizado obtido nas aulas é aplicado na prática, verificando o que acontece no experimento real.

Entretanto, é necessário, aliado com as ferramentas e ambientes interativos, aplicar uma metodologia de ensino que facilite ao aluno a sua aprendizagem em programação. Desse modo, empregar uma metodologia adequada é importante para o processo de aprendizado do estudante. Assim, os métodos de ensino na programação, por parte dos professores, devem possuir um aspecto dinâmico (GOMES, HENRIQUES e MENDES, 2008), de modo a proporcionar ao discente um maior envolvimento com os conceitos estudados, como é o caso da metodologia Problem Based Learning (PBL) ou, em tradução livre, aprendizagem baseada em problema, que leva o aluno a ter um papel mais ativo no seu aprendizado (AMBRÓSIO e COSTA, 2010). Dessa forma, buscar novos caminhos para o ensino de programação possibilita superar várias dificuldades.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Avaliar o uso da linguagem visual em blocos junto com uma metodologia ativa no ensino/aprendizagem de programação para alunos dos cursos de computação.

1.1.2 Objetivos Específicos

- Identificar quais são as dificuldades no ensino/aprendizagem de programação.
- Verificar e analisar o uso das ferramentas de programação em blocos existentes.
- Verificar e analisar o uso de metodologias ativas no ensino de programação.
- Utilizar uma ferramenta de programação em blocos combinada a uma metodologia ativa de ensino para turma de alunos repetentes da disciplina de programação.
- Analisar os resultados obtidos com a aplicação da linguagem visual em blocos e a metodologia escolhida.

1.2 JUSTIFICATIVA

Com a velocidade com que as informações têm sido produzidas e repassadas para a sociedade, a exigência por sistemas que possam resolver problemas de forma rápida e eficiente aumenta. Com isso, é necessário uma boa qualificação do estudante de computação para desenvolver programas que impactem de forma positiva o mundo real. Por isso, ensinar os fundamentos da programação, seguindo as estruturas lógicas adequadas e fazendo o aluno pensar no problema e buscar uma solução, deve fazer parte dos principais objetos de estudo em cursos da área de computação.

Em vista disso, é necessário vencer as barreiras que o ensino de programação apresenta. Berssanette e Francisco (2018) destacam algumas das dificuldades que são enfrentadas, dentre elas o baixo nível de abstração dos discentes e a falta de competências necessárias para interpretação e solução de problemas. Um estudo realizado com 112 alunos do curso de Sistemas de Informação, na Faculdade de Educação Tecnológica do Estado do Rio de Janeiro, no Campus Paracambi, identificou as principais dificuldades dos acadêmicos

na disciplina Algoritmo e Linguagem de Programação 1 (LIMA, J., VIEIRA, C. e VIEIRA, P., 2015).

Um dos grandes problemas constatados, foi que boa parte dos alunos que participaram da pesquisa, cerca de 31%, sentem dificuldade no desenvolvimento do raciocínio lógico. Ligado a isso, os problemas relacionados aos conteúdos também é algo que traz dificuldades ao aprendizado dos alunos. As estruturas de dados, matrizes e vetores, totalizam 57% das dificuldades de aprendizagem dos estudantes, seguido por estruturas de repetição e estruturas de condição, que apresentam, respectivamente, 13% e 8% dos obstáculos nessa área de estudo (LIMA, J., VIEIRA, C. e VIEIRA, P., 2015).

De acordo com Lima, Vieira C. e Vieira P. (2015), a base inicial dos conceitos da disciplina, que envolve entrada, processamento e saída de dados, não causa muitas dificuldades para os discentes. Porém, quando as estruturas de decisão, repetição e sequenciais são apresentadas, os problemas de aprendizagem aparecem de forma bem clara (LIMA, J., VIEIRA, C. e VIEIRA, P., 2015, p.13).

Combinado a isso, tem-se a barreira de construir programas bem estruturados e ao mesmo tempo dar atenção à sintaxe da linguagem, o que constitui-se em motivo de desânimo nas fases introdutórias. Por esse motivo, Gomes, Henriques e Mendes (2008) concluem que quando o foco são as exigências da sintaxe, alguns alunos se sentem incapazes de escrever um programa bem sucedido. Diante disso, as dificuldades retratadas têm se mostrado obstáculos a serem vencidos por estudantes e professores, uma vez que o processo de ensino/aprendizagem nessa disciplina é uma tarefa complexa. (BERSSANETTE e FRANCISCO, 2018).

Sendo assim, é preciso apresentar ao discente desafios que o levem a desenvolver habilidades e competências inerentes aos estudos de programação, adotando ferramentas e metodologias que podem contribuir para o ensino/aprendizagem nessa área.

O presente trabalho apresenta o uso da linguagem visual em blocos com uma metodologia ativa de ensino na primeira unidade da disciplina de Algoritmos e Programação I - API, na Universidade Estadual do Sudoeste da Bahia - UESB, para os alunos repetentes. Sendo assim, com o uso de métodos diferentes na abordagem dos conceitos introdutórios de programação, o trabalho oferece um dinamismo maior às aulas, estimulando os discentes a avançarem nos estudos da disciplina e da computação, minimizando as dificuldades encontradas e vivenciadas por estudantes e professores.

1.3 METODOLOGIA

Para a execução dos objetivos propostos, foram identificadas as principais dificuldades de aprendizagem dos alunos nas disciplinas de programação com pesquisas bibliográficas. Além disso, foi realizado um levantamento sobre as ferramentas em blocos existentes, que são utilizadas para o ensino da lógica de programação e de que forma elas contribuem para o aprendizado. Ocorrendo, também, pesquisas, estudos e análise sobre as metodologias ativas e suas aplicações em computação.

Uma vez analisadas as barreiras no processo de ensino/aprendizagem, as linguagens visuais em blocos e as metodologias ativas que são utilizadas, foi compreendido as perspectivas de professores e estudantes sobre a disciplina Algoritmos e Programação I, com o uso de algumas entrevistas. O levantamento de dados contou também com um questionário, que foi aplicado antes do início da disciplina. Através deste questionário foi avaliada as dificuldades dos alunos repetentes e se já utilizaram ou conhecem alguma das ferramentas de programação em blocos, assim como alguma metodologia ativa. Logo após isso, com o decorrer da disciplina, utilizar a linguagem visual em blocos com uma metodologia ativa, de modo a despertar os alunos para o desenvolvimento de competências necessárias para interpretação e resolução de problemas, propondo várias atividades que os levem a ter um papel ativo no aprendizado. Após a obtenção dessas informações, será realizada a aplicação prática da programação em blocos e da metodologia ativa na disciplina Algoritmos e Programação I com os alunos repetentes.

Dessa forma, na conclusão da disciplina, pretende-se aplicar um novo questionário para avaliar se os discentes conseguiram vencer suas dificuldades e se o uso da plataforma e metodologia contribuíram para seu desenvolvimento em computação.

Os resultados dos questionários serão analisados de forma quantitativa e qualitativa, avaliando as perguntas abertas e de múltipla escolha com a escala Likert. Após essa análise, será identificado como a programação em blocos e a metodologia empregada auxiliaram no ensino/aprendizagem, abordando as vantagens e desvantagens encontradas, e o que pode ser feito em trabalhos futuros.

Com isso, o presente trabalho mostra as contribuições geradas com o uso de linguagem em blocos e metodologia ativa nos conceitos iniciais da disciplina de programação com uma turma de repetentes, uma vez que essa forma de metodologia não foi usada nas turmas anteriores e pode auxiliar de forma significativa no ensino-aprendizagem dos estudantes, minimizando as complexidades sintáticas iniciais e desenvolvendo o raciocínio

lógico através das resoluções dos problemas apresentados. Além do mais, trabalhar com alunos repetentes possibilita compreender as dificuldades que tiveram ao iniciar os estudos dessa matéria, e como auxiliar-lhes e a futuros discentes a aprenderem de forma mais eficiente os conceitos de programação apresentados logo no início do curso.

1.4 ESTRUTURA DO DOCUMENTO

Além deste capítulo, o trabalho está divido na seguinte estrutura:

- No **Capítulo 2** é apresentado o referencial teórico que servirá como base para desenvolvimento do trabalho.
- No **Capítulo 3** são apresentadas as informações obtidas com entrevistas de alunos e professores e o questionário para os alunos matriculados na disciplina, afim de ter uma perspectiva de como a disciplina é estruturada.
- No **Capítulo 4** é mostrado como se deu a aplicação da metodologia ativa e a utilização da programação em blocos com a descrição de cada atividade realizada nas aulas.
- No **Capítulo 5** os resultados obtidos são demonstrados e analisados.
- No último capítulo é realizada a conclusão e a proposta para trabalhos futuros.

2 PROGRAMAÇÃO

A programação está, intimamente, ligada à computação, e é mais do que escrever código, é uma ciência e também uma arte (GOMES, HENRIQUES e MENDES, 2008). Assim, ensinar a programar envolve conceitos fundamentais, que possibilitam ao estudante raciocinar de forma lógica, compreender os caminhos possíveis, e procurar a melhor solução para o problema. Dessa forma, programação engloba desenvolver habilidades e usar os processos lógicos para desenvolvimento de técnicas, proporcionando a produção de novas tecnologias e sistemas (FORBELLONE e EBERSPÄCHER, 2005), de forma a contribuir para a sociedade.

Segundo Bilabila (2017), programação envolve criar e elaborar programas de computador com estruturas bem definidas. Por isso, esse estudo envolve o raciocínio lógico e resolução de problemas, que são necessários para a construção de novas habilidades exigidas dos estudantes nessa área.

2.1 DIFICULDADES DE APRENDIZAGEM NOS ESTUDOS DE PROGRAMAÇÃO

Segundo Jekins (2002), poucos estudantes de programação acreditam que programar seja uma tarefa fácil. Nesse sentido, é essencial entender quais são as dificuldades enfrentadas por eles, a fim de propor um ambiente de aprendizagem que possibilite vencer as barreiras encontradas e ter motivação para prosseguir com o aprendizado nessa área.

Jekins (2002), levanta alguns fatores que influenciam na aprendizagem do aluno e que podem dificultar o processo de aprendizado na programação, incluindo os fatores cognitivos, que é divido em estilos de aprendizado, e a motivação dos discentes para programar. Além disso, a programação necessita de diversas habilidades do programador, especialmente a de analisar o problema e encontrar uma solução eficiente. O estudante ainda enfrenta as dificuldades de conhecer e entender novos conceitos, que exigem uma abstração maior dos problemas.

Nesse sentido, desenvolver essas habilidades não é uma tarefa simples, pois exige um desenvolvimento gradual por parte do aluno, que precisa, na maioria das vezes, aprender sobre a sintaxe e logo em seguida compreender as estruturas semânticas da programação para aplicá-las em conjunto. Gomes, Henriques e Mendes (2008), relatam que a preocupação principal deveria ser despertar o aluno para desenvolver a capacidade de resolver problemas, usando a linguagem de programação como um veículo para aplicação dos conceitos já

aprendidos, porém, a prática comum é trabalhar mais na linguagem de programação e nos detalhes sintáticos.

Além do mais, por causa do foco nas estruturas sintáticas e na leitura de livros sobre elas, e não na compreensão das estruturas essenciais de programação, o discente enfrenta mais uma barreira para aprender sobre o assunto, já que, muitas vezes, a prática de programação é colocada em segundo plano. Ao contrário disso, de acordo com Jekins (2002), o desenvolvimento de habilidades necessárias acontece programando, não somente na leitura de livros.

Jekins (2002) ainda destaca que a linguagem de programação adotada, na maioria das vezes, não é escolhida com o foco da aprendizagem do aluno, mas porque é a mais usada pela indústria, o que dificulta o processo de aprendizagem. Nesse sentido, Gomes, Henriques e Mendes (2008), afirmam que a prática comum no ensino de programação, é ensinar as estruturas sintáticas aos discentes, ao invés de ajudá-los a entender a utilidade de programar e os processos dinâmicos dessa disciplina, tais como as estruturas que fazem parte dela e a resolução de problemas. Um estudo realizado por Oliveira et al. (2017), mostra que a maior parte dos alunos sente mais dificuldade na sintaxe do que na estrutura semântica. A pesquisa foi realizada com 51 alunos do ensino superior da área de computação, dos quais, 49% encaram a sintaxe como tópico mais complicado, 37,3% consideram os detalhes semânticos mais difíceis, e 13,7% sentem dificuldade em ambos os assuntos.

Com isso, diferente de outros componentes curriculares, programação envolve conceitos mais dinâmicos e deve ser trabalhado com métodos que se adequam ao dinamismo envolvido, o que muitas vezes não ocorre (GOMES, HENRIQUES e MENDES, 2008). O foco em estruturas sintáticas, os processos estáticos de ensino, as múltiplas habilidades que precisam ser desenvolvidas e a dificuldade em analisar e resolver problemas, constituem as principais barreiras no aprendizado de programação por parte dos estudantes.

2.2 FERRAMENTAS E ESTRATÉGIAS PARA AUXILIAR NAS DIFICULDADES DE APRENDIZAGEM DE PROGRAMAÇÃO

Com as diversas dificuldades existentes no ensino de programação, é importante buscar estratégias que possam auxiliar os estudantes a desenvolver as habilidades necessárias e avançarem nos estudos de programação. Dessa forma, algumas ferramentas têm sido propostas com a finalidade de contribuir para o ensino dessa área, afim de diminuir a complexidade e facilitar a compreensão por parte dos discentes. Sendo assim, foram

propostos ambientes que evitassem a complexidade das linguagem de programação e possibilitassem o uso de uma linguagem mais clara, onde o iniciante na área de computação pudesse compreender as estruturas básicas, representação algorítmica e codificar usando o pseudocódigo, ou portugol. Tais estratégias e ambientes possuem o intuito de reduzir os problemas enfrentados pelos alunos (GOMES et al, 2008).

Uma dessas ferramentas que propõe levar o aluno a pensar sobre o problema e transformá-lo em um algoritmo, sem necessidade de se preocupar com uma linguagem de programação, é o VisuAlg. O VisuAlg permite a criação de algoritmos, possibilitando, também, a análise das estruturas utilizadas, verificação dos eventuais erros e utilização dos comandos de forma fácil, tendo o idioma português como base (LEITE et al., 2013). As principais vantagens que podem ser destacadas dessa ferramenta, é a linguagem clara, que por ter a base do idioma português facilita para que o aluno compreenda a estrutura, e uma interpretação em pseudocódigo, que permite o estudante visualizar como o algoritmo produzido se comporta na resolução do problema. (LEITE et al, 2013).

Apesar das vantagens práticas, o VisuAlg possui algumas desvantagens quando se trata da organização no uso de estruturas de decisão e de repetição, como as delimitações do início e final de um comando condicional ou no incremento de uma variável dentro da repetição (LEITE et al., 2013). Contudo, o VisuAlg compreende apenas uma pequena parte das ferramentas produzidas que contemplam o uso do pseudocódigo como uma linguagem clara para o aprendizado inicial. Outros exemplos de ferramentas que usam estes princípios são o Portugol IDE e o Portugol Viana. Ambos permitem utilizar a estrutura do portugol para execução dos algoritmos e criação de fluxogramas, possuindo características diferenciadas na interface e desvantagens quanto à organização das ferramentas e lentidão no uso de alguns recursos (MASSAM, 2012).

Apesar desse tipo de ferramenta ser bem utilizado, de acordo com Gomes et al. (2008), as ferramentas disponíveis parecem não suprir as reais dificuldades dos alunos, principalmente se tratando em analisar, resolver e construir algoritmos para representar a solução dos problemas propostos. Por isso alguns trabalhos focam na resolução de problemas e procuram auxiliar os alunos nessas atividades. Um exemplo é o Sistemas Interativo para Construção de Algoritmos e sua Simulação - SICAS, que possibilita a edição e solução do problema por parte do discente, permitindo-lhe experimentar e testar a execução da resposta encontrada (GOMES e MENDES, 2001). Diferente das abordagens com pseudocódigo, o SICAS utiliza uma estrutura visual para os estudantes representarem a solução dos problemas, que são os fluxogramas, e permite acompanhar o andamento do algoritmo após executá-lo

(GOMES e MENDES, 2001). Esse ambiente ainda proporciona a verificação dos erros e permite que o aluno proponha uma nova correção, além do mais, possibilita a comparação de diferentes algoritmos para o mesmo problema e analisar qual é o que tem um melhor desempenho (GOMES e MENDES, 2001).

Apesar de possibilitar vários recursos, como prática nas análises dos problemas e exercício da lógica, o SICAS não possui mecanismos para auxiliar os estudantes em alguns obstáculos que enfrentam ao analisar um problema e ultrapassarem alguns impasses que são encontrados no percurso (GOMES et al, 2008). Existem outros ambientes com as mesmas propostas de resolução de problemas, como é o caso do ProGuide, que baseado no SICAS, procura trazer melhorias para estudantes com maiores dificuldades (AREIAS e MENDES, 2006). Esse ambiente apresenta melhorias no que concerne ao acompanhamento do aluno na sua interpretação do problema e sua solução, oferecendo uma comunicação mais ativa com o aluno, de forma a guia-lo no passo a passo da soluções propostas (AREIAS e MENDES, 2006). Contudo, possui limitações quanto ao uso de uma linguagem natural, usada na comunicação com o estudante para ajudá-lo a encontrar a solução do problema e criar o algoritmo adequado (AREIAS e MENDES, 2006).

Acrescentando a essas estratégias, a robótica, também, tem ganhado espaço no ensino de programação. Isso porque ela é considerada um campo de estudo multidisciplinar (PADIR e CHERNOVA, 2013). Nesse sentido, propostas como o uso de laboratórios remotos robóticos tem surgido para permitir aos estudantes observar na prática o uso de programação, dentro e fora da sala de aula. Lopes et al. (2017), destaca que o uso desses robôs proporcionam a aplicação dos conteúdos abstratos que estão inerentes nos estudos de programação e motiva o estudante de programação a ter um aprendizado de maneira mais lúdica. Desse modo, foi proposto um curso do Laboratório Remoto em Ambiente Virtual de Aprendizagem (LARA), desenvolvido por estudantes e professores da Universidade Estadual do Sudoeste da Bahia – UESB (LOPES et al., 2017), que teve como objetivo ensinar os conteúdos básico e iniciais da programação (LOPES, 2017). O curso semipresencial, intitulado Curso Lara de Introdução à Programação (CLIP), utilizou um robô remoto do LARA (L1R2), para propor as atividades práticas, e os estudantes perceberem na prática como cada conceito da programação acontece (LOPES, 2017).

No ambiente de programação online, utilizado no CLIP, foi usada uma biblioteca própria do LARA, onde os estudantes poderiam testar os comandos em alto nível (LOPES, 2017), sem a necessidade de conhecer os detalhes da implementação para executar uma função, como por exemplo, direcionar o robô remoto para frente.

Ambientes como esse, que procuram levar o aluno a ver na prática como a programação ocorre, têm sido adotados para um melhor aproveitamento no ensino e aprendizagem de programação. Todavia, os laboratórios remotos possuem limitações quanto ao uso, pois além de cada aluno precisar reservar um horário para a utilização, apresentam complexidades quando se trata de realizar mudanças no *hardware* e estabelecer comunicação com o *software.* (POTKONJAK, 2016). Sendo assim, surgiu os laboratório virtuais, que procuram trazer vantagens que os laboratórios remotos não possuem, como múltiplos acessos de usuários e sem custos de manutenção com hardware, já que o software desempenha o papel principal. Potkonjak (2016), destaca ainda que com mudanças na configuração de parâmetros do sistema é possível adicionar ou substituir motores, tendo um custo menor que modificações no hardware. Apesar dessas vantagens, alguns laboratórios virtuais podem exigir recursos de modelagem dinâmica 3D e uma complexidade maior na sua configuração, o que Potkonjak (2016), apresenta como um dos problemas desses ambientes.

Com isso, outras opções, também, têm sido buscadas para auxílio no ensino dessa etapa dos cursos de computação, como por exemplo, linguagens baseadas em ícones, representações visuais de algoritmos e sistema com uso de tutores inteligentes (GOMES et al, 2008). Essas estratégias desenvolvidas, que propõe o uso da robótica na programação, elimina a dificuldade que as atuais linguagens de programação podem trazer nos estudos iniciais de programação e o foco nas interpretações, análises e soluções dos problemas, são relevantes. Contudo, é necessário possibilitar que o discente aprenda sobre os diversos aspectos da programação, de modo que ele não se distancie da compreensão dos conteúdos e nem das habilidades necessárias para um programador.

2.3 PROGRAMAÇÃO EM BLOCOS

Devido às dificuldades encontradas no ensino de programação, surgi uma posposta de ensino e aprendizagem utilizando programação gráfica, conhecida como programação em blocos. Essa programação funciona sem a escrita de linhas de códigos; as estruturas são formadas arrastando e encaixando os blocos de acordo com a resolução encontrada para o problema. Nesse sentido, o iniciante em lógica de programação, não precisa se preocupar em escrever linhas de código e cometer erros sintáticos, permitindo a ele pensar melhor no problema e sua solução.

2.3.1 Ferramentas de Programação em Blocos

Várias ferramentas de programação em blocos surgiram como alternativa para o ensino e aprendizagem da lógica de programação, oferecendo um ambiente onde várias pessoas, mesmo não sendo estudantes de computação, podem aprender os princípios básicos dessa área.

2.3.1.1 Code.org

Criado pelo Hadi Partovi (ZANATTA, 2015) e lançado em 2013, o Code.org é um ambiente de ensino e aprendizagem de programação online e gratuito, que possibilita o aprendizado dos conceitos básicos de programação para um público iniciante. (CAVALCANTE, COSTA e ARAÚJO, 2016). O ambiente é organizado por cursos, cada um direcionado a um público específico. Atualmente, o Code.org conta com quatro cursos principais, o curso 1 direcionados aos alunos de alfabetização, o curso 2 dirigido a estudantes de ensino fundamental ou médio, e os cursos 3 e 4 aprofundam o que foi dado no curso 2, com um nível de complexidade maior, de acordo com o avanço do aluno (CAVALCANTE, COSTA e ARAÚJO, 2016).

Cada curso possui diversas fases, onde nelas são abordados diversos conceitos da programação, como estruturas de repetição, estruturas condicionais, variáveis, entre outros. As fases são divididas em etapas, que são apresentadas como desafios, à medida que os desafios vão sendo finalizados, a complexidade vai aumentando. A interface do ambiente de ensino possui dois espaços, um para programação, utilizando os blocos fornecidos, e outro para execução e visualização da proposta de resolução produzida (Figura 1).

Figura 1 – Ambiente de programação do Code.org

Fonte: http://www.code.org (2018)

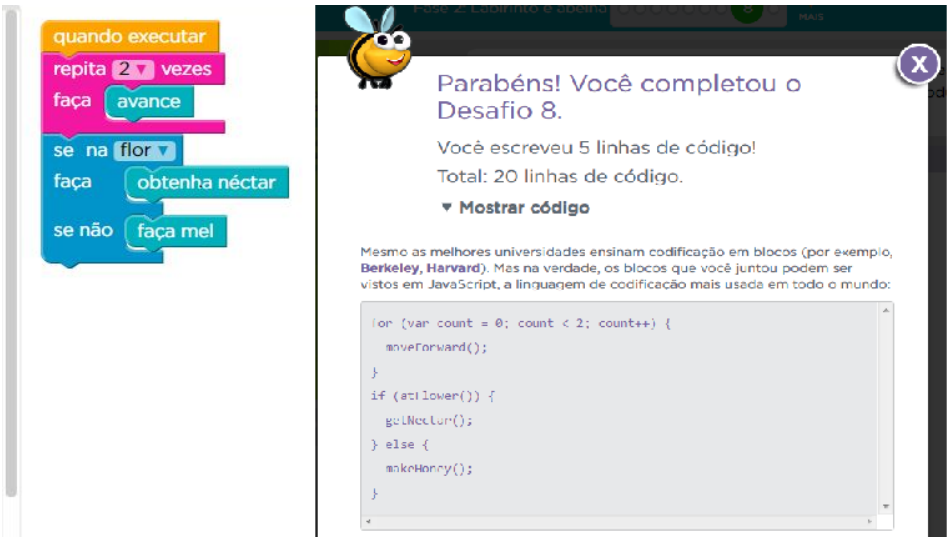
O discente pode usar os blocos, que estão disponíveis, arrastando para a área de programação, e logo em seguida, clicando no botão “Executar”, ver a solução na prática, ou ainda avaliar o passo a passo do algoritmo no botão “Passo”. A medida que o estudante vai avançado nas etapas propostos do curso, vai sendo apresentado novos blocos que permitem aprendizagem de novos conceitos. Na figura 1, é mostrado uma das etapas que utiliza-se a estrutura de repetição. Em certos momentos, de acordo com os avanços das fases e etapas, são inseridas as estruturas condicionais (Figura 2).

Figura 2 – Ambiente de programação do Code.org com estrutura condicional

Fonte: http://www.code.org (2018)

Com isso o estudante pode ir aprendendo várias estruturas inerentes aos conceitos básicos de lógica de programação ao longo dos desafios. Outra característica positiva desse ambiente, está em oferecer a possibilidade de observar a resolução em linha de código na linguagem JavaScript, logo após a conclusão do desafio (Figura 3).

Figura 3 – Código mostrado pelo ambiente do Code.org



Fonte: http://www.code.org (2018)

Sendo assim, os iniciantes na programação podem construir a solução para o problema, sem se preocupar com a sintaxe da linguagem, e logo em seguida observar como é organizada uma linguagem de programação e quais são as estruturas sintáticas que a compõem, fazendo a comparação da programação em blocos com o código textual. Apesar do ambiente do Code.org oferecer uma gama de recursos, o objetivo de ensino foca nas séries iniciais do ensino fundamental e médio, propondo a difusão desse ensino nas escolas (ZANATTA, 2015). Por isso, a plataforma não fornece explicações semânticas sobre os assuntos (CAVALCANTE, COSTA e ARAÚJO, 2016), ou os conceitos que vão sendo apresentados nos desafios, como é o caso das estruturas condicionais e variáveis. Mesmo com alguns vídeos de rápida explicação sobre o conteúdo, o ambiente não aprofunda na utilização dos conceitos. Além disso, o Code.org não concede ao estudante a liberdade de testar alguns comandos ou trabalhar determinados conceitos, isso por causa das opções pré-determinadas que são apresentadas na tela (CAVALCANTE, COSTA e ARAÚJO, 2016).

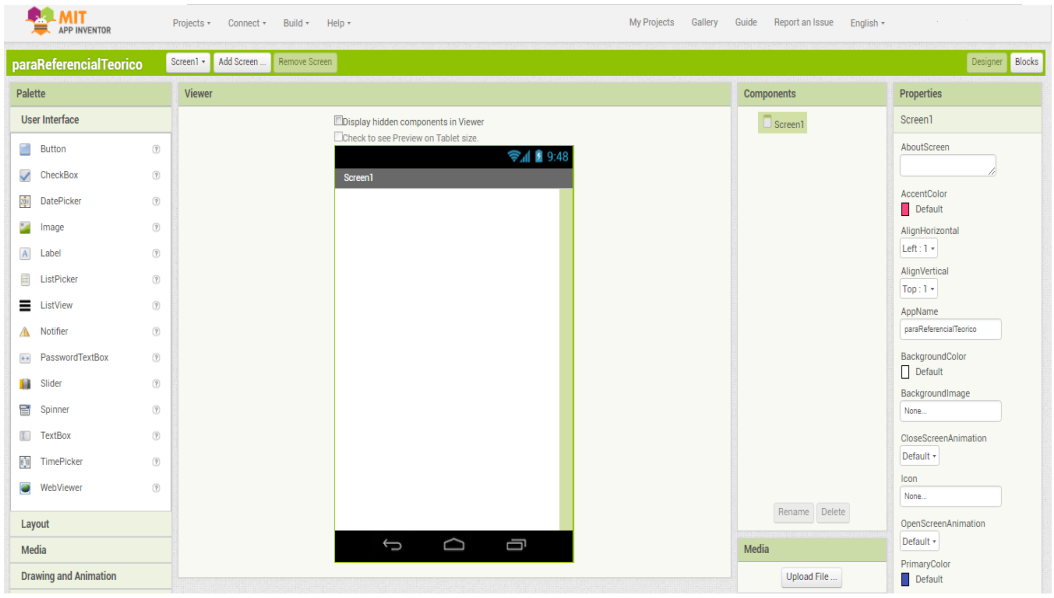
Diante disso, para uma aplicação no ensino superior, é desejável uma ferramenta que possibilite maior abrangência da semântica junto com uma metodologia que permita-o aprender e entender de forma sólida os conteúdos propostos.

2.3.1.2 App inventor

O App Inventor é uma linguagem visual de programação, online, desenvolvida pela Google e Instituto Tecnológico de Massachusetts - MIT, que permite a criação de aplicativos para dispositivos com sistema Android (GOMES e MELO, 2013). Essa ferramenta possibilita a construção de aplicativos com diversos recursos, tais como o uso de mapas, imagens, base de dados, sensores, entre outros. O ambiente permite que os estudantes criem suas aplicações usando a criatividade (GOMES e MELO, 2013), não colocando limitações quanto ao uso dos blocos, e não apresenta desafios para que sejam seguidos, deixando o aluno livre para construção dos seus aplicativos.

Por conta da programação ser realizada por blocos, a aprendizagem através da utilização do App Inventor, na produção de aplicativos, se torna melhor do que as linguagens em linha de código, isso porque, para alguns iniciantes em programação, é desencorajador digitar o código e logo em seguida receber inúmeras mensagens de erros, sem saber identifica-los (WOLBER et al., 2014). A plataforma é dividida em dois espaços. A primeira área é a de desenho da tela do aplicativo, ou seja, a interface (Figura 4). Nela é possível arrastar e soltar os componentes na tela, que pode ser observada como um smartphone ou tablet. Além da facilidade de arrastar e soltar os componentes, como botões, campos de texto, notificações, entre outros, é possível alterar várias propriedades dos objetos. O projeto ainda permite o upload de arquivos e adição de novas telas.

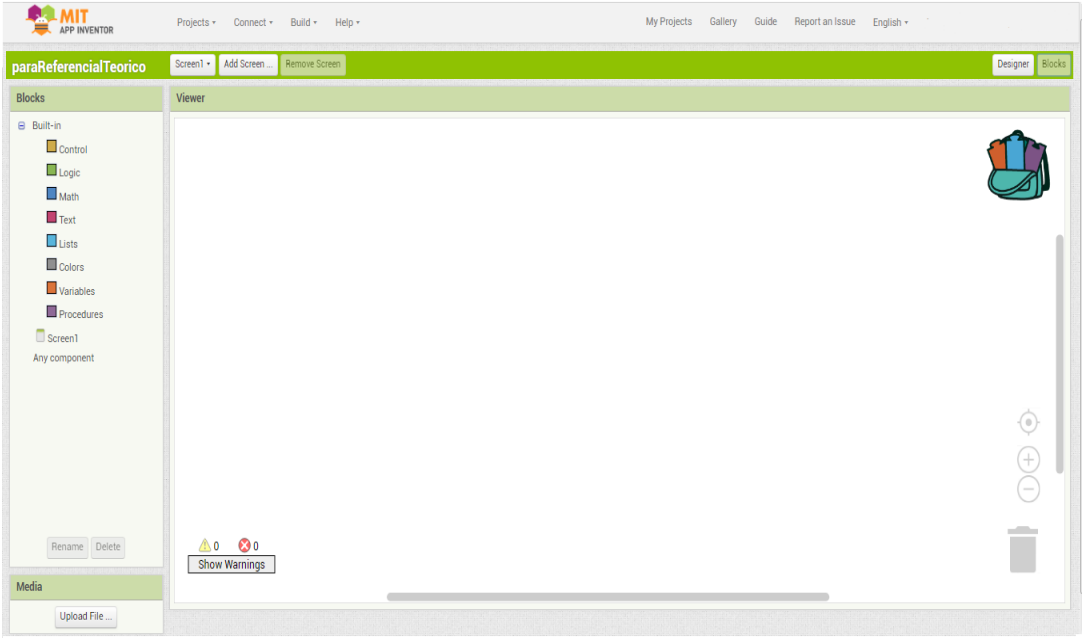
Figura 4 – Espaço de interface do App Inventor



Fonte: http://appinventor.mit.edu (2018)

O segundo espaço é para programar o comportamento do programa, onde são utilizados os blocos de programação para dar funcionalidades ao aplicativo que deseja-se criar (Figura 5).

Figura 5 – Espaço de programação do App Inventor



Fonte: http://appinventor.mit.edu (2018)

Na área de programação do App Inventor, os blocos são separados por categoria, facilitando o modo como o estudante adiciona instruções. Além disso, os componentes usados na interface, também possuem sua própria categoria de blocos, sendo possível, por exemplo, a adição de um evento quando a tela inicializar ou um botão for clicado. Com a plataforma do MIT, o estudante não precisa lembrar das instruções sintáticas para programar, ele apenas pode procurar os blocos na categoria desejada e arrastá-lo para a área de programação, permitindo-o criar as funcionalidades sem complicações, levando-o a pensar melhor na estrutura lógica para o problema (WOLBER et al., 2014). Além do mais, o ambiente possibilita a reutilização dos códigos em blocos em outras telas através da mochila, em que o estudante pode guardar procedimentos criados dentro da mochila e usá-los em outro momento.

O ambiente mostra pontos positivos em diversos aspectos quanto a trabalhar com a lógica e outros conceitos importantes, contudo, apesar das vantagens oferecidas pelo uso do App Inventor, ele não possibilita a visualização das linhas de código produzidas através dos blocos, como o Code.org. Nesse sentido, é importante que o estudante de ensino superior

possa, além de compreender os conceitos introdutórios sem complexidade, saber como a sintaxe dos blocos são produzidas, de modo a minimizar as dificuldades no primeiro contato com uma linguagem de programação.

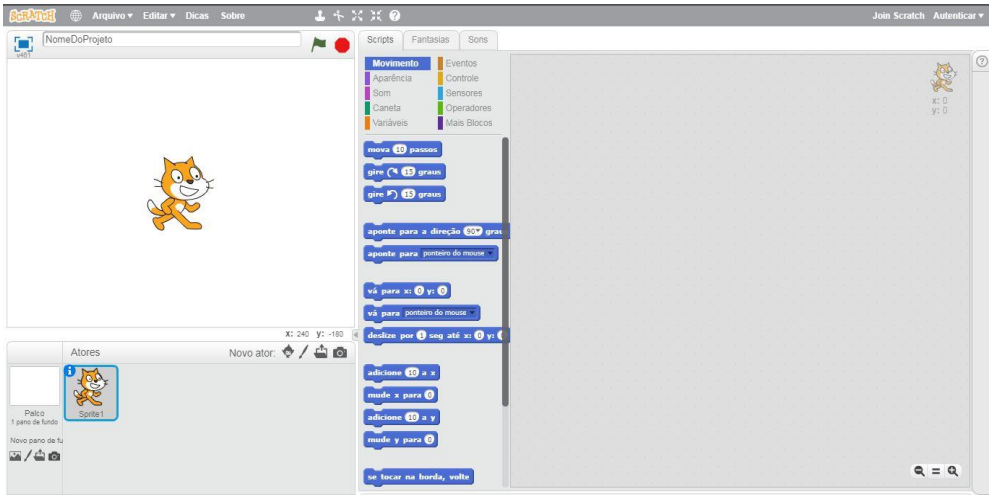
2.3.1.3 Scratch

Em 2007, Mitchel Resnick e seu grupo do MIT, lançou o Scratch, com o objetivo de despertar a criatividade e a aprendizagem das pessoas que viriam a utilizar essa linguagem de programação, principalmente as crianças (RESNICK, 2017). Esse ambiente conta com uma comunidade online, onde cada pessoa pode compartilhar seus projetos e criações (RESNICK, 2017).

Assim como o App Inventor e o Code.org, a programação do Scratch é construída a partir do arrastar e encaixar blocos de programação. Contudo, diferente das outras plataforma, o Scratch tem uma interface diferenciada, que visa a utilização da programação para produção de animações e jogos. Essa variedade de recursos, visa cumprir a proposta de desenvolver o pensamento criativo, levando estudantes, crianças, professores, pesquisadores e tantos outros a terem a experiência de criar e compartilhar o que produziu, contribuindo para o aprendizado de forma colaborativa (RESNICK, 2017).

Nesse sentido, a ferramenta tem alguns espaços para o processo de criação, que permite a visualização de diversos recursos (Figura 6).

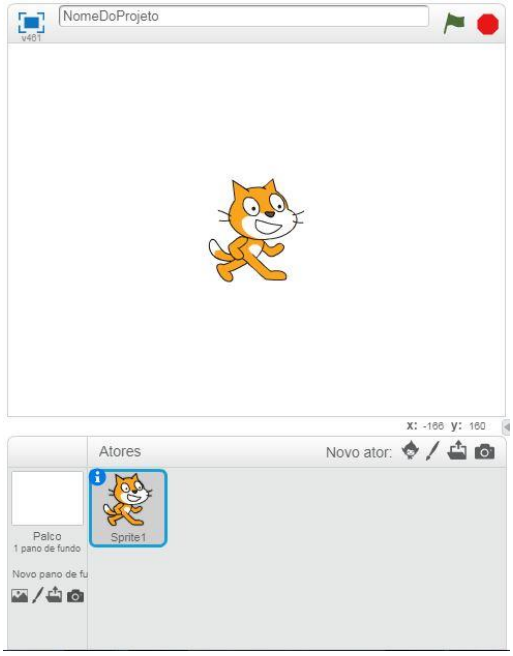
Figura 6 - Ambiente Scratch



Fonte: https://Scratch.mit.edu (2018)

O primeiro ambiente, é montado a parte gráfica, da animação ou jogo a ser produzido, nomeado pela ferramenta como Palco (Figura 7). Nele é possível inserir personagens, nomeado pela ferramenta de atores, que podem ser inseridos a partir da própria biblioteca do Scratch, de um arquivo ou câmera do computador, além do criador do projeto ter a possibilidade de desenhar algum personagem dentro do ambiente.

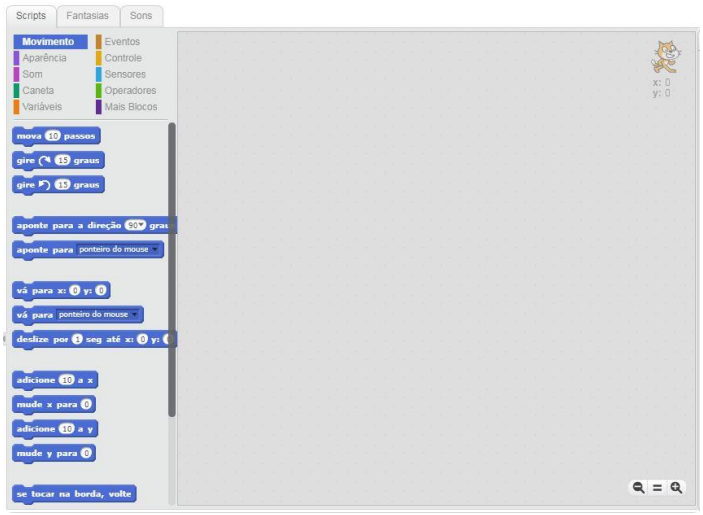
Figura 7 – Palco do Ambiente Scratch para inserção dos objetos



Fonte: https://Scratch.mit.edu (2018)

No segundo espaço, é possível a visualização de três áreas. A primeira área é dos scripts, que é constituída pelos blocos, divido por categorias, facilitando o encontro de funções de programação e a possibilidade da criação dos próprios blocos, conhecida nos conceitos introdutórios de programação como funções, ou procedimentos (Figura 8). Essa área oferece a possibilidade da utilização de operadores, movimentos, estruturas condicionais e de repetição, variáveis, dentre outras. Esses blocos formam estruturas básicas para implementação de um programa (MÉLO et al., 2011), onde podem ser usados clicando e arrastando para área de programação.

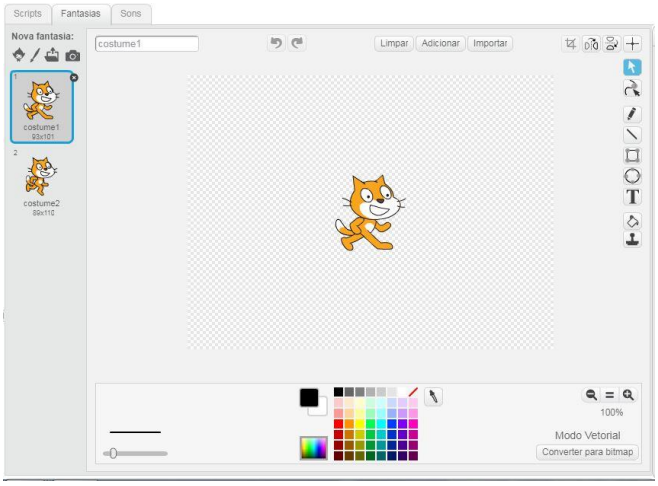
Figura 8 – Área de programação do Ambiente Scratch



Fonte: https://Scratch.mit.edu (2018)

Outra área apresentada dentro desse espaço de programação, é a de fantasias (Figura 9). Que permite a edição de personagens, planos de fundo, e inserção de novas imagens de movimentos para o objeto.

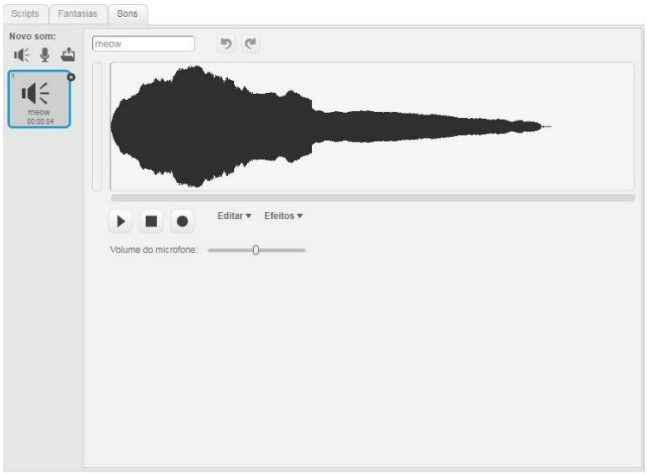
Figura 9 – Área de fantasias do Ambiente Scratch



Fonte: https://Scratch.mit.edu (2018)

A última área, possibilita a inserção de recursos de som (Figura 10), podendo ser da biblioteca do ambiente, uma música do computador ou um som capturado a partir de um dispositivo de gravação.

Figura 10 – Área de sons do Ambiente Scratch



Fonte: https://Scratch.mit.edu (2018)

Diante disso, o Scratch é um ambiente que oferece uma gama alta de recursos, atingindo o objetivo do seu desenvolvimento. Além do mais, na área de programação, é possível fazer um personagem andar, com o uso de poucos blocos. Segundo Mélo et al. (2011), os blocos possuem um sistema de orientação, indicando o seu uso de forma correta, como o uso de uma condição em um bloco de desvio condicional (Figura 11).

Figura 11 – Condição em um bloco de desvio condicional

Fonte: https://Scratch.mit.edu (2018)

Dessa forma, o Scratch é um bom ambiente de ensino de programação, que conta com diversas características que permitem a aprendizagem de programação, a criação e o desenvolvimento de diversas soluções (MÉLO at al., 2011). O ambiente ainda pode ser usado em uma versão off-line instalada no computador pessoal, onde é realizada a produção de um projeto e oferecida a possibilidade de compartilha-lo no ambiente online. Contudo, assim como o App Inventor, não oferece a possibilidade de visualização em linhas de código, o que para o aluno de graduação no curso de computação é importante, de modo a conhecer como reproduzir o que foi construído em código textual. Isso oferece a possibilidade para, em uma

etapa posterior, manter um contato com uma linguagem utilizada nas aplicações de mercado, como Java (JEKINS, 2002).

Além dessas ferramentas, o emprego de uma metodologia que envolva o aluno de forma mais ativa no seu aprendizado, pode despertar competências necessárias para programação.

2.4 METODOLOGIAS ATIVAS

As metodologias de ensino, desempenham um papel importante no ensino de conteúdos da computação. Desse modo, a metodologia tradicional, que visa a transmissão de conhecimento através de um modelo expositivo, produz pouca interação do aluno em um processo que requer um envolvimento mais ativo (NAGAI e IZEKI, 2013). Além do mais, a forma passiva que o discente se encontra na metodologia tradicional, pode gerar uma desmotivação no aprendizado dos conteúdos de programação (CHICON, QUARESMA e GARCÊS, 2018). Por isso, novas metodologias têm sido propostas como meio para diminuir as dificuldades encontradas no ensino de lógica para programação. É o caso das metodologias ativas, que propõem, segundo Diesel, Baldez e Martins (2016), colocar o aluno no centro do processo de ensino/aprendizagem. De acordo com Berbel (2011), a metodologia ativa desperta a curiosidade e faz com que os alunos busquem novos elementos não abordados na sala de aula, enriquecendo a interação entre discentes e professor.

Existem várias metodologias ativas, como por exemplo a de estudo de caso, de processo incidente, métodos de projetos, aprendizagem baseada em problema - PBL (BERBEL, 2011), dentre outras. Uma das metodologias que se destaca é a PBL, pois possibilita um maior envolvimento dos discentes com o problema, de modo a estimular a aprendizagem e leva-los a trabalharem em conjunto, desenvolvendo e adquirindo habilidades importantes (NAGAI e IZEKI, 2013), principalmente, na área de computação. Diante disso a PBL se mostra uma metodologia interessante, uma vez que o estudante tem um papel ativo na resolução dos problemas propostos, na busca do conhecimento e aprendizagem dos conteúdos.

Outra metodologia ativa, também conhecida e utilizada, é a *Peer Instruction* ou Instrução por Pares, que tem como objetivo criar uma maior interação entre os alunos nas aulas, onde o professor fala, brevemente, sobre os conteúdos, apresenta perguntas conceituais e os alunos pensam individualmente na resposta (CHICON, QUARESMA e GARCÊS, 2018). Dependendo da quantidade de acertos, o professor pode revisar os tópicos apresentados no

início, dividir a turma em pequenos grupos para elaborarem uma nova solução, ou pode passar para um novo tópico e uma nova pergunta. Dessa forma a Instrução por Pares permite aos acadêmicos trabalharem os conceitos estudados, elaborar hipóteses e solucionar os problemas juntos (CHICON, QUARESMA e GARCÊS, 2018).

Esses e outros métodos procuram trazer uma nova abordagem de ensino, onde o acadêmico se envolva, ativamente, na aprendizagem. Portanto, é necessário avaliar como e qual metodologia aplicar no ensino de programação, de forma a proporcionar ao estudante um aprendizado eficaz.

2.5 TRABALHOS RELACIONADOS

Algumas pesquisas envolvendo o ensino de lógica de programação, lançam mão de algumas ferramentas para auxiliar os estudantes em suas dificuldades. As duas principais ferramentas que ganham espaço em cursos superiores da área de computação, são o App Inventor (GUIMARÃES, ARRUDA e MARTINS, 2016; JUNQUEIRA, 2016) e o Scratch (PEREIRA, MEDEIROS e MENEZES, 2012; BATISTA et al., 2016; SANTOS et al., 2015, AMBRÓSIO e COSTA, 2010).

Uma pesquisa realizada por Guimarães, Arruda e Martins (2016), com 15 alunos do curso de Sistemas da Informação, mostrou que 83% responderam que o App Inventor contribui para aprendizagem da lógica de programação, sendo que 17% consideraram que o ambiente ajuda parcialmente no aprendizado. Com isso, a grande maioria também declarou que a interface se mostrou fácil e amigável, possibilitando a prática da programação sem complicações (GUIMARÃES, ARRUDA e MARTINS, 2016). Nesse sentido, a ferramenta do MIT, se mostrou efetiva para o ensino e aprendizagem de lógica de programação, uma vez que o aluno não precisa se preocupar na memorização de comandos, e sim, na estruturação da lógica para resolver os problema proposto e construir a aplicação (GUIMARÃES, ARRUDA e MARTINS, 2016).

Não é somente em cursos da área de computação que encontra-se dificuldades na aprendizagem dos fundamentos de computação. Junqueira (2016) afirma que nos diversos cursos de engenharia existem obstáculos que os estudantes enfrentam e cita o exemplo do curso de Engenharia de Produção, nas disciplinas de programação, que apresentam um alto nível de reprovações, desde a criação do curso até o primeiro semestre de 2015.

Diante disso, a proposta realizada por Junqueira (2016), consistiu na elaboração de um projeto, que permitisse aos alunos compreenderem como funciona a lógica de programação,

através da linguagem visual com blocos, pretendendo eliminar a dificuldade no uso de sintaxe em outras linguagens textuais. Nesse sentido, a pesquisa teve como base a utilização do App Inventor em um curso, dividido em turmas de 10 alunos, de modo que os projetos fossem desenvolvidos para auxiliar os discentes no desenvolvimento do raciocínio lógico, trabalhando os conceitos como estruturas condicionais, repetição e funções. Junqueira (2016), ainda observa, através da aplicação de um questionário de avaliação do curso, que 70% dos alunos se sentiram totalmente motivados na proposta de criar aplicativos para ajudar no aprendizado, e que 90% conseguiram fazer assimilação dos conceitos e tópicos nas disciplinas de lógica de programação. Apesar dos resultados pertinentes, é necessário uma melhor avaliação e melhoria das pesquisas realizadas, para que o ensino e aprendizagem na área de estudo, utilizando App inventor, possa ter um melhor desenvolvimento quando aplicado em salas de aula.

O Scratch também pode ser aplicado a alunos ingressantes dos cursos na área de computação, como demonstrado por Pereira, Medeiros e Menezes (2012), onde apresentam uma proposta de aplicação para alunos do ensino superior usando o ambiente Scratch. A proposta consiste em utilizar a linguagem visual em blocos da ferramenta, para levar ao estudante a pensar sobre um problema para calcular a média de alguns alunos e verificar se foram aprovados ou não (PEREIRA, MEDEIROS e MENEZES, 2012). Por meio da avaliação das entradas, dos dados a serem processados e do resultado a ser produzido, o discente da disciplina de algoritmos e programação, pode desenvolver o raciocínio na resolução do problema, sem se preocupar com detalhes sintáticos. Segundo Pereira, Medeiros e Menezes (2012), o uso do Scratch, no ensino de conceitos iniciais para o ingressante no curso, pode contribuir para o educando focar no aprendizado da lógica do programa, e depois dos fundamentos básicos tiverem sido bem absorvidos, o professor pode introduzir para uma linguagem textual de programação. Dessa forma, conforme a pesquisa, os alunos que iniciam a disciplina usando essa ferramenta, desenvolvem melhor sua compreensão sobre o problema e entendimento das estruturas básicas de programação. (PEREIRA, MEDEIROS e MENEZES, 2012).

Além disso, Batista et al. (2016), fazem um estudo da aplicação do Scratch das séries iniciais até o ensino superior, de modo a mostrar a contribuição oferecida pela ferramenta nas diferentes idades e fases de aprendizado. Com isso, Batista et al. (2016) relatam sobre uma oficina de oito horas realizada com alunos dos cursos de sistemas de informação e ciência da computação. A maioria desses estudantes eram ingressantes no curso, e alguns veteranos contribuíram nas atividades envolvendo a ferramenta (BATISTA et al, 2016). Diante disso,

segundo Batista et al. (2016), o uso dessa ferramenta possibilita aos acadêmicos interligar conceitos que serão vistos em linguagens de programação textual.

Sendo assim, Batista et al. (2016) e Pereira, Medeiros e Menezes (2012) buscam trazer, para ingressantes no ensino superior de computação, uma abordagem com programação visual. Santos et al. (2015) apresenta uma abordagem para alunos que repetiram a disciplina de lógica e algoritmo de programação, na Universidade Federal do Rio Grande do Norte , que devido as dificuldades não conseguiram passar na disciplina com a média exigida.

Como um fator que despertasse o interesse por parte dos estudantes que estavam revendo conceitos da disciplina, utilizou-se uma abordagem com programação visual em blocos, o Scratch, e uma placa micro controladora, o arduino (SANTOS et al., 2015). Nesse sentido, utilizou-se uma abordagem do Scratch para arduino, conhecido como Scratch for Arduino (S4A), desenvolvido pela equipe do Citilab, possibilitando a aprendizagem de programação através do uso da programação em blocos com o arduino (SANTOS et al., 2015). A pesquisa de Santos et al. (2015) foi realizada com alunos que estavam repetindo a disciplina, e os conceitos trabalhados envolveu estruturas de repetição, estruturas condicionais, estruturas aninhadas e outros conceitos básicos inerentes a programação, que possibilitou uma melhor assimilação com uma linguagem de programação e os blocos do S4A.

Ao final do experimento, Santos et al., (2015), apresenta os resultados através da aplicação do questionário, em que os discentes matriculados pela segunda vez fizeram boas considerações na utilização da ferramenta, afirmando que a plataforma despertou o interesse e possibilitou motivação ao verem os exercícios realizados na prática, o que para eles torna a atividade mais atraente. A pesquisa ainda demonstrou um aumento no rendimento individual dos alunos, medido a partir das notas, em comparação com o semestre inicial (SANTOS et al., 2015). Apesar das boas ponderações e de atingir o resultado de ensino em lógica de programação, segundo Santos et al. (2015), a falta de recursos para disponibilização dos componentes eletrônicos para serem trabalhados individualmente, gerou certa dificuldade, já que os alunos consideraram que as aulas práticas são importantes para aprenderem os conteúdos de forma mais fácil.

Ambrósio e Costa (2010) utilizaram o Scratch combinado a metodologia ativa PBL, com alunos do curso de Ciência da Computação da Universidade Federal de Goiás - UFG, na disciplina Programação de Computadores I. O trabalho foi aplicado com duas turmas diferentes, o que possibilitou uma pesquisa com maiores resultados. Além do mais, o Scratch foi trabalhado aliado com a linguagem de programação C, o que permitiu uma melhor

associação do que era visto na linguagem visual. Na primeira turma, em 2008, a metodologia PBL foi aplicada seguindo, a rigor, os passos definidos, levando os alunos a se preocuparem mais em seguir as etapas do que com a solução dos problemas (AMBRÓSIO e COSTA, 2010). Na segunda turma, em 2009, a perspectiva foi mudada, fazendo modificações na aplicação da metodologia e deixando o foco maior na resolução de problemas. A pesquisa apresentou resultados satisfatórios, uma vez que a maioria dos alunos das duas turmas avaliaram de forma positiva a aplicação da PBL, cerca de 73% na turma de 2008 e 65% na turma de 2009. Apesar dos bons resultados, e de uma melhor motivação dos alunos nos estudos da disciplina, alguns pontos negativos foram identificados, como a falta de um material de apoio pré-selecionado, que levou os discentes a pesquisarem materiais de forma autônoma, encontrando alguns de linguagem avançada e que produziu certo grau de dificuldade na compreensão do assunto, fazendo alguns desanimarem no processo de aprendizagem. Ambrósio e Costa (2010) concluem, mesmo identificados pontos de vista negativos, que a metodologia contribuiu para o estudante ser mais ativo no seu aprendizado, podendo trabalhar de forma colaborativa com outros grupos.

Sendo assim, é necessário desenvolver atividades que possibilitam a prática por todos os alunos, utilizando ferramentas e metodologias ativas, de forma que a compreensão e o entendimento dos fundamentos de programação possam ser bem consolidados, permitindo o acadêmico prosseguir bem no que concerne a essa área de estudo.

3 ALGORITMOS E PROGRAMAÇÃO I

A disciplina Algoritmos e Programação I - API é oferecida, anualmente, no primeiro semestre do curso de Ciência da Computação na UESB, com carga horária de 90 horas (PPC, 2011). Umas das aptidões que o curso procura desenvolver no discente, até sua formação, é o raciocínio lógico e abstrato (PPC, 2011), e a disciplina de API é uma dos componentes essenciais nesse desenvolvimento.

Esse componente curricular, visa mostrar ao aluno conceitos e propriedades de algoritmos, estratégias de resolução de problemas e implementação, o aprendizado de uma linguagem de programação imperativa, estruturas de controle e decisão, dentre outros (PPC, 2011). Sendo assim, é uma das disciplinas fundamentais para o bom progresso no curso de Ciência da Computação, devido à própria estrutura e competências exigidas do formando nessa área.

Portanto, antes de apresentar a proposta do projeto com programação em blocos e metodologia ativa, tornar-se necessário compreender como foi a forma de ensino adotada antes da aplicação do presente trabalho. Para isso, foram feitas entrevistas com dois professores e quatro alunos do curso de Ciência da Computação que haviam cursado a disciplina, possibilitando uma visão desse componente curricular que é tão importante para os cursos desta área.

3.1 ENTREVISTA COM PROFESSORES QUE MINISTRARAM A DISCIPLINA

A entrevista para os professores constitui-se de dez perguntas: 1) Como foi sua experiência ao ministrar a disciplina? 2) Quais as dificuldades que você observou nos alunos ao aprenderem programação? 3) Como foi o início da disciplina? 4) Você já usou programação em blocos? 5) Você já fez uso de uma programação mais lúdica, como a robótica, por exemplo? 6) O que você acha que poderia contribuir para um melhor aprendizado dos alunos na disciplina? 7) Quais as dificuldades que você teve ao ministrar a disciplina? 8) Você já deu aula para alunos repetentes da disciplina? 9) O que você acha de usar a programação em blocos antes de mostrar uma linguagem de programação textual? e 10) Os alunos sentem mais dificuldades na construção sintática ou na lógica de programação?

Essas perguntas tiveram como objetivo entender o que foi observado pelos professores ao ministrarem a disciplina, inclusive quais as principais dificuldades dos alunos. O modelo da entrevista completa pode ser observado no apêndice A.

Foram entrevistados dois professores, individualmente, que já haviam ministrado a disciplina. Em face à primeira pergunta, consideraram boa a experiência e tiveram desafios em comum quando ensinaram API. Em relação às dificuldades dos alunos, é possível perceber que compartilham de opiniões relacionadas, pois um destacou a construção do pensamento lógico e a base matemática, enquanto o outro destacou as estruturas inerentes a programação. Isso permite observar que a disciplina traz complexidades semelhantes para cada discente, pois conseguem compreender os conceitos, mas na hora da prática e solução de problemas apresentam alguns impasses. Uma diferença notada entre os professores entrevistados, é na abordagem inicial da disciplina. Um começa com uma linguagem textual e imperativa, que com a utilização do Python, como uma linguagem inicial, obteve um bom resultado. O outro começa com uma pseudo-linguagem, em momento recente utilizou o VisualG, embora a experiência nesse último caso não tenha sido satisfatória, pois os acadêmicos acabam apresentando certa rejeição à ferramenta e querendo aprender uma linguagem imperativa.

Nenhum dos docentes fez uso da programação em blocos ou o uso de robótica com programação na ministração da disciplina. O que mostra que a proposta desse trabalho é interessante para avaliar o uso de uma nova ferramenta no ensino de API.

Os professores apresentaram opiniões distintas quanto ao que poderia contribuir para um melhor aprendizado dos alunos. O primeiro respondeu que uma carga horária de atividades práticas poderia ter um papel importante para que os acadêmicos aplicassem os conceitos aprendidos. O segundo professor defendeu que uma dedicação maior dos estudantes e uma metodologia que pudesse trabalhar a transição do concreto para o abstrato e os estimulasse a gostar de programação, poderia contribuir melhor.

Os dois docentes disseram que fazer os alunos vencerem as complexidades nas estruturas de programação, pensamento lógico e obter resultados satisfatórios, de modo que aprendessem os conceitos e resolvessem bem os exercícios, foram desafios que enfrentaram ao ministrar o componente curricular. Com isso, também informaram que já haviam ensinado para alunos repetentes, e que esses apresentaram as mesmas dificuldades quando da primeira vez que cursaram a disciplina. Isso mostra que mesmo entendendo os conceitos, é necessário desenvolver um raciocínio lógico, e procurar motivá-los nessa busca pela aprendizagem da disciplina, unindo a boa compreensão dos conteúdos com a prática da programação.

Por fim, os professores nunca tinham utilizado a programação em blocos na metodologia da disciplina, e por isso consideraram interessante a ideia, afirmando que seria necessário fazer uma experiência para definir se essa ferramenta poderia contribuir com o aprendizado da lógica na disciplina. Na última pergunta, os dois docentes afirmaram que os

acadêmicos apresentavam maiores dificuldades na lógica de programação do que nas construções sintáticas. A partir dessas entrevistas com os professores, foi possível perceber que as dificuldades entre os alunos são comuns, e existe uma maior dificuldade em desenvolver um pensamento lógico e abstrato exigidos pela própria programação. Com isso, é importante encontrar metodologias que permitam aos discentes praticar a resolução de problemas que os façam aprimorar o raciocínio lógico, de maneira que os conceitos teóricos e a prática caminhem juntos no aprendizado da disciplina.

3.2 ENTREVISTA COM ALUNOS VETERANOS APROVADOS NA DISCIPLINA

Para compreender a visão dos discentes sobre a disciplina, foi necessário também entrevistá-los. Foram entrevistados quatro alunos, individualmente, que haviam sido aprovados na disciplina e de turmas diferentes, com o objetivo de coletar informações sobre a aplicação da disciplina e suas dificuldades.

A entrevista constitui-se de oito perguntas, 1) Como foi sua experiência na disciplina? 2) Você acha que as atividades práticas contribuíram para o seu aprendizado? 3) Você teve dificuldades na disciplina? Se sim, quais foram? 4) Você acha que deveria ter mais aulas práticas? 5) O que você achou da metodologia aplicada? 6) O que você acha que poderia melhorar na metodologia da disciplina? 7) Você conhece programação em blocos? Se sim, você acredita que introduzir a disciplina com programação em blocos o ajudaria a enxergar melhor a lógica do problema antes de passar para uma linguagem de programação textual? E 8) Você sentiu mais dificuldade nas construções sintáticas ou na lógica?

Essas perguntas serviram como base para entender a perspectiva do estudante em relação a disciplina de API, o modelo das entrevistas completas podem ser observadas no apêndice B. A partir da entrevista, pode ser compreendido como cada acadêmico enxergou a disciplina de uma forma em comum, com suas próprias dificuldades.

O relato da experiência com a disciplina, de forma geral, foi boa, mostrando que alguns sentem dificuldades no início na construção do raciocínio e outros no uso de alguns recursos da própria programação. Com isso, todos consideram que as aulas práticas são de fundamental importância e contribuição no aprendizado da disciplina, pois permite perceber os conceitos que são vistos na teoria.

No relato das dificuldades, nota-se que alguns assuntos avançados, como recursão e funções, são os conteúdos que mais se destacam, embora em uma das entrevistas, o entendimento e a compreensão do problema tenha se mostrado uma complexidade a ser

vencida nos estudos na programação. Sobre a possibilidade de ter mais aulas práticas, nota-se que metade dos entrevistados acreditam que foram suficiente quando cursaram a disciplina, enquanto a outra parte, de turmas mais recentes, destacou que poderia ter mais prática, pois contribuem com o melhor aprendizado da programação.

Em relação a metodologia aplicada, todos gostaram de como os conceitos foram trabalhos e praticados, e sugeriram melhorias, que houvesse um ensino mais dinâmico no início da disciplina de modo a ajudar na compreensão dos conceitos iniciais, que fosse realizado um acompanhamento melhor do aluno nas atividades e mostrar o porquê de estarem aprendendo programação, ou seja, casos mais práticos no cotidiano de cada um, possibilitando um maior interesse e imersão do discente nos estudos da disciplina.

É interessante observar que todos conheciam a programação em blocos e acreditam que a ela pode contribuir para o desenvolvimento do raciocínio lógico na disciplina. Dessa forma, isso demonstra que trabalhar com esse desenvolvimento do pensamento lógico é um dos pontos importantes a serem considerados no ensino de programação. Por fim, metade dos entrevistados responderam que a maior dificuldades que tiveram estava na construção sintática, enquanto a outra parte informaram que a lógica da programação é que foi um desafio maior.

A partir dessas perspectivas de acadêmicos que já haviam cursado a disciplina, puderam ser observados alguns desafios a serem vencidos e aspectos a serem melhorados no ensino, de modo que pudessem não somente praticar o uso das estruturas de programação, mas o desenvolvimento da lógica, contribuindo para uma melhor compreensão do problema e por conseguinte nas soluções dos desafios e execução dos exercícios de forma satisfatória.

3.3 APLICAÇÃO DO QUESTIONÁRIO COM OS ALUNOS REPETENTES MATRICULADOS NA DISCIPLINA

Além das entrevistas, antes de iniciar a disciplina, foi aplicado um questionário, para os alunos repetentes matriculados, com o objetivo de compreender as possíveis dificuldades que levaram à reprovação e qual era a visão da turma diante da programação, metodologia e das ferramentas que seriam usadas na primeira unidade. O questionário foi constituído de oito perguntas, as perguntas 2), 3), 4) e 5) foram perguntas abertas e as outras quatro de multiplica escolha: 1) Ano de ingresso no curso de ciência da computação. 2) O que é programação pra você? 3) Quais as dificuldades que contribuíram para sua reprovação em API? 4) O que você acha que poderia ajudá-lo a ser aprovado em API? 5) Quais conteúdos da disciplina que você

teve mais dificuldade em aprender? 6) Você sabe o que é programação em blocos? 7) Você já usou alguma ferramenta de programação em blocos? 8) Você já teve contato com alguma metodologia de aprendizagem ativa?

O questionário foi aplicado com oito alunos que matricularam na disciplina. Os resultados da primeira pergunta podem ser vistos na figura 12.

Figura 12 – Respostas da primeira pergunta do questionário: Ano de ingresso no curso de computação:

Fonte: Do autor (2019)

Percebe-se que a maior parte dos repetentes vem da turma 2018.1, anterior ao semestre onde o presente projeto foi aplicado.

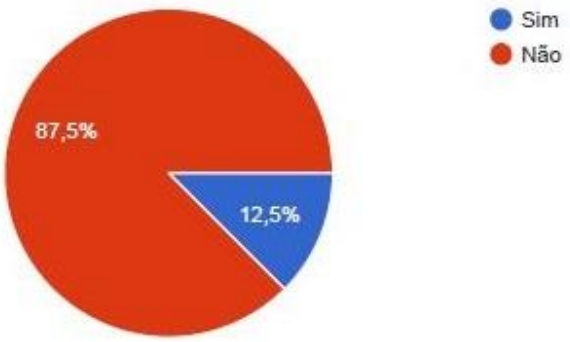
Na segunda pergunta, sete responderam e um aluno deixou a resposta em branco. As compreensões sobre o que é programação foram diversas. Alguns entendem como uma forma de dar ordens a máquina, outros como a utilização de código para inovação e facilitar a vida das pessoas e alguns entendem como uma pequena parte ligada ao mundo de desenvolvimento de sistemas. Dessa forma, nota-se que a visão que eles possuem da programação é ampla, e pode contribuir para formação de cada um deles.

As terceira pergunta, sobre as dificuldades que levaram a reprovação, foi respondida pelos oito alunos e bem variadas. Destacando a interpretação e resolução dos problemas, complexidade de alguns assuntos e a falta de prática na programação. Diante dessas respostas, a que mais se apresentou foi a dificuldade na interpretação e resolução do problema, isso mostra que é necessário, primeiramente, mostrar uma metodologia que faça com que o discente compreenda os problemas e descubra as possíveis soluções, para logo depois trabalhar os aspectos de uma linguagem de programação textual e como aplicar a solução do problema em linhas de código.

A quarta pergunta, do que poderia ajudar na aprovação, foi respondia por sete estudantes e as respostas destacaram a importância da prática e o apoio para entender o que deve ser feito no programa. Nisso percebe-se o quanto que é importante adotar uma metodologia mais dinâmica com os conteúdos de programação, e levá-los a pensar sobre o problema e buscar encontrar a solução, oferecendo uma forma mais prática de trabalhar com os assuntos da disciplina.

A pergunta sobre quais conteúdos tiveram mais dificuldades em entender foi respondida por todos os oito discentes. Arranjos, funções e ponteiro, foram os conteúdos mais citados. Isso mostra que tais conteúdos exigem um grande nível de abstração e de como implementá-los em linhas de código. Após entender, nessas cinco primeiras perguntas, o perfil dos acadêmicos e suas dificuldades, no final do questionário buscou-se verificar se eles tinham algum conhecimento prévio com a programação em blocos e a metodologia ativa. Na sexta pergunta, sobre se sabiam o que era programação em blocos, apenas um, dentre os oito discentes, respondeu que sim. Essa avaliação pode ser visualizada no gráfico da figura 13.

Figura 13 – Respostas da pergunta 6) Você sabe o que é programação em blocos?

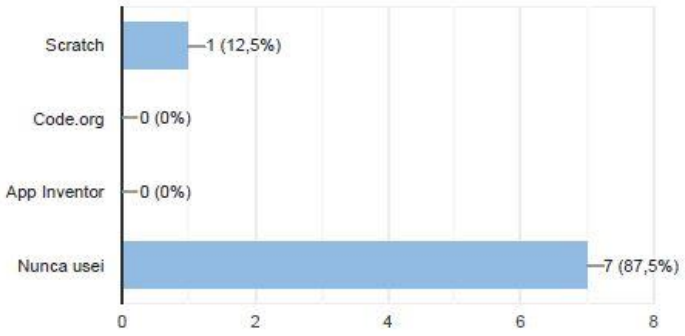


Fonte: Do autor (2019)

Nesse sentido, a abordagem proposta trouxe novidades para os acadêmicos, que até então eram acostumados a iniciar com uma linguagem de programação textual ou uma pseudo-linguagem.

Na pergunta seguinte, se tinham usado alguma ferramenta de programação em blocos, apenas um respondeu que havia usado o Scratch, enquanto o restante respondeu que nunca usaram, como pode ser observado na figura 14.

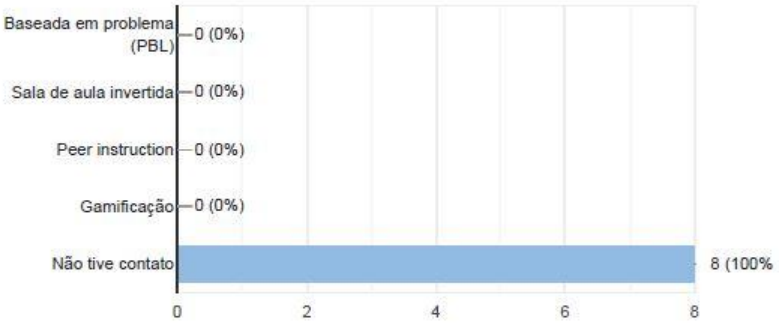
Figura 14 – Respostas da pergunta 7) Você já usou alguma ferramenta de programação em blocos?



Fonte: Do autor (2019)

Após identificar se os alunos haviam tido contato com essa nova abordagem de programação, a última pergunta do questionário abordou a metodologia ativa. A pergunta, sobre o contato com algumas metodologia ativa foi respondida pelos oito alunos, o que pode ser observado na figura 15.

Figura 15 – Respostas da pergunta 8) Você já teve contato com alguma metodologia de aprendizagem ativa?



Fonte: Do autor (2019)

Verifica-se que os alunos não conheciam ou não tiveram contato com alguma metodologia ativa. Dessa forma, como observado por Gomes, Henrique e Mendes (2008), procurar trabalhar conceitos tão dinâmicos de forma estática é um dos fatores que interfere no aprendizado da programação. A metodologia ativa oferece um dinamismo maior, colocando o aluno para atuar no seu processo de aprendizado, não apenas sendo ouvinte, como também observando na prática os conteúdos. Diante disso, usar essa aprendizagem ativa na disciplina

de API, envolvendo atividades práticas, pode contribuir para o acadêmico ter um bom aprendizado na lógica de programação e suas estruturas.

4 PROPOSTA DE ENSINO E APRENDIZAGEM EM PROGRAMAÇÃO

A busca e o uso de novas metodologias para o ensino de programação é necessário para conduzir o discente por um conteúdo que possui complexidade e, na maioria das vezes, não foi visto antes do ingresso no curso de computação ou áreas afins. Por isso, apenas o uso de apresentações projetadas, explicações verbais e textos, podem não oferecer uma boa compreensão de assuntos dinâmicos, como é o caso das disciplinas de programação (GOMES, HENRIQUES e MENDES, 2008). Sendo assim, a partir dos estudos e pesquisas realizadas, foi proposto para a primeira unidade da disciplina Algoritmos e Programação I - API, em caráter especial, o uso de uma ferramenta de programação em blocos, de uma metodologia ativa e de uma plataforma educacional para comunicação com os estudantes e acompanhamento da realização das atividades. A turma contou com dez alunos matriculados, porém, apenas oito alunos participaram e frequentaram as aulas.

4.1 FERRAMENTA DE PROGRAMAÇÃO EM BLOCOS E A PLATAFORMA EDUCACIONAL

Com a proposta inicial de levar o aluno a pensar melhor sobre o problema e como solucioná-lo, foi selecionada uma linguagem de programação em blocos, que permite ao estudante trabalhar, primeiramente, mais com a lógica da programação e focar menos em sintaxes e detalhes de uma linguagem de programação textual. Além disso, foi utilizada uma plataforma educacional para comunicação extraclasse, postagens de materiais do conteúdo da disciplina e acompanhamento das resoluções das atividades.

A ferramenta de programação em blocos escolhida foi o Scratch, pois oferece a possibilidade de desenvolver projetos offline e online, e visualização do projeto em execução de forma mais rápida. O code.org não foi utilizado devido ao nível de dificuldade dos desafios propostos, por serem considerados fáceis para estudantes repetentes da disciplina, e por oferecer somente a possibilidade de uso online. O app inventor, apesar de possuir muitos comandos e possibilidades, a visualização do projeto em execução não é imediata como no Scratch, sendo necessária a compilação do projeto e a instalação do aplicativo no smartphone para o aluno ver o resultado do seu desenvolvimento, o que torna o processo lento e demorado, principalmente, diante de alguns projetos maiores.

A versão do Scratch utilizada foi a 2.0, pela estabilidade à época da execução desse trabalho e possuir os comandos necessários para as atividades propostas na disciplina. A

plataforma educacional utilizada foi a Google *Classroom*, desenvolvida pela Google for Education, que possibilita a comunicação, compartilhamento de conteúdo e material, entrega de atividades, respostas de questionários e outras funções entre o professor e aluno (SILVA, 2016). O ambiente é considerado de fácil configuração, otimizando a comunicação e uma boa organização de arquivos, já que a ferramenta é integrada com o sistema de armazenamento nas nuvens do Google Drive e do serviço de e-mail do Gmail (SILVA, 2016). Por essas vantagens e integração a plataforma foi adotada na disciplina, o que permitiu agilidade na execução dos exercícios, postagem de materiais e um bom acompanhamento dos alunos.

4.2 METODOLOGIA ATIVA

A metodologia ativa proposta foi uma adaptação da PBL. Essa metodologia enfatiza a resolução de problemas para um melhor aprendizado sem aulas teóricas, deixando os estudos dos conceitos como responsabilidade extraclasse do aluno e o papel do professor como um instrutor (AMBRÓSIO e COSTA, 2010). Embora a PBL se caracteriza por essa formalidade, para utilizá-la na disciplina foi necessário fazer algumas adaptações, porque alguns conceitos precisavam ser firmados em sala de aula, como por exemplo, os métodos de resolução de problemas, estruturas de repetição, variáveis e a apresentação da ferramenta Scratch.

As adaptações realizadas envolveram alguns aspectos da proposta por Nuutila et al. (2005), onde é apresentado a PBL em sete passos para o ensino de programação: 1) Exame do caso; 2) Identificação do problema; 3)*Brainstorming*; 4) Definição de um modelo; 5) Definição dos objetivos de aprendizados; 6) Estudo independente; 7) Discussão sobre o aprendizado.

No primeiro passo os alunos se familiarizam com o material e descrevem o caso, e no segundo eles identificam os pontos importantes (NUUTILA et al., 2005). Na terceira etapa os alunos se reúnem e fazem as associações das ideias e experiências anteriores com aquilo que eles já conhecem. Já na quarta etapa eles modelam o problema e as ideias discutidas antes, construindo conceitos importantes (NUUTILA et al., 2005). No quinto passo os alunos identificam no modelo proposto os objetivos de aprendizagem e verificam as inconsistências que podem ocorrer, logo após isso, na sexta etapa, eles coletam mais informações sobre o problema com leituras e análise do modelo e materiais (NUUTILA et al., 2005). Na última fase, conforme Nuutila et al. (2005), ocorre a discussão dos conceitos e mecanismos aprendidos e análise dos materiais utilizados.

Embora a forma de aplicação da PBL apresentada possuir cada etapa definida para que os alunos compreendam as estruturas de um problema, no âmbito da programação isso pode levar o discente a se preocupar mais com a metodologia (AMBRÓSIO e COSTA, 2010) do que com os conceitos próprios da programação. Sendo assim, a aplicação da PBL na primeira unidade da disciplina API, do presente projeto, passou por modificações para adequar-se ao conhecimento anterior que os repetentes tinham e com às estruturas que precisavam compreender.

A primeira modificação a ser destacada, é a redução da quantidade de passos, pois alguns podem ser trabalhados em conjunto e de forma mais resumida. A modificação realizada diminuiu os sete passos para três: 1) Apresentação do problema; 2) Definição de um método para resolução do problema; 3) Discussão sobre a solução e o aprendizado.

O primeiro passo consistiu em apresentar ao discente o problema a ser solucionado, levá-lo a refletir sobre o que foi proposto e quais conteúdos poderiam ser utilizados, semelhante ao que foi mostrado na metodologia PBL apresentada por Nuutila et al. Já o segundo passo, foi formado por etapas e ao invés dos alunos focarem na criação de um modelo, foi proposto um método para compreensão e resolução de problemas. O método utilizado foi o de POLYA (1995), por ser de fácil entendimento. O terceiro e último passo foi a discussão da solução proposta por cada aluno e a verificação de como aplicaram os conteúdos.

4.2.1 Método de George Polya

Para um melhor entendimento da modificação realizada na metodologia PBL, referente ao segundo passo, é necessário conhecer as 4 etapas adotadas por Polya (1995) na resolução de um problema. A primeira fase é a de compreender o problema. Nessa fase o discente precisa responder algumas perguntas, como por exemplo, o que se pede no problema? Quais são os dados necessários? É possível dividir em partes e fazer anotações? É possível traçar figuras e desenhar algum diagrama? (POLYA, 1995). Respondendo essas perguntas, o aluno vai entender as características essenciais do problema apresentado.

Na segunda fase, o discente traça um plano para resolução, verificando se já viu algo semelhante, se nas suas experiências já o viu o problema de forma ligeiramente diferente ou conhece algo utilizado em soluções anteriores. Após essa verificação, o plano pode ir sendo traçado, utilizando situações passadas, ou até mesmo ideias e conteúdos adquiridos anteriormente.

Na terceira fase ocorre a execução do plano para solução do problema. Os passos traçados na fase anterior, devem ser verificados a cada execução. Nesse momento, o acadêmico deve verificar se é possível ver e demonstrar claramente os passos que estão corretos. Por fim, acontece a análise do resultado, observando se é possível chegar ao resultado por um caminho diferente, se os argumentos são verificáveis e se é possível utilizar o mesmo método para resolver um outro problema.

Utilizando o método de Polya, no segundo passo da metodologia adotada na disciplina, os alunos conseguiram ter uma visão de como poderiam solucionar os problemas propostos e de como executariam o plano na ferramenta de programação em blocos.

4.3 APLICAÇÃO DA PROPOSTA DE ENSINO NA SALA DE AULA

As aulas foram realizadas no turno da tarde, por conta da disciplina estar sendo oferecida em caráter especial, ou seja, fora do semestre regular. Além disso, as aulas foram distribuídas pelo colegiado do curso nas terças, possuindo quatro aulas consecutivas de cinquenta minutos, e quintas, duas aulas de cinquenta minutos, sendo ministradas no laboratório de linguagem de programação, o que permitiu envolver menos aulas expositivas e mais aulas práticas, já que os estudantes repetentes conheciam boa parte da base teórica, mas tinham dificuldade em aplicá-la na prática.

4.3.1 Primeira aula

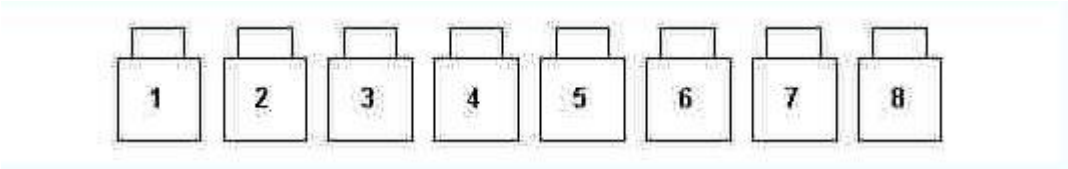
Assim que a disciplina foi iniciada, no primeiro dia, foi apresentado aos alunos o plano de curso proposto, a apresentação da plataforma educacional do Google *Classroom*, demonstração do Scratch, atividade de ordenação de números e atividades práticas, com o intuito de avaliar como eles resolveriam o problema da ordenação e que conhecessem a ferramenta de programação em blocos.

4.3.1.1 Problema de ordenação

A atividade de ordenação foi baseado no problema proposto por Setzer e Carvalheiro (1993) e permitiu observar como os discentes enxergavam o problema e como buscavam uma solução para ele. O desafio é constituído de alguns compartimentos, e cada compartimento

contém um número natural, onde os estudantes devem realocá-los em ordem crescente. Na figura 16, é possível visualizar como é o desafio.

Figura 16 – Problema de ordenação



Fonte: Setzer e Carvalheiro (1993)

Dessa forma, os alunos deveriam levantar as tiras de papel que estavam nos compartimentos, e visualizariam um número natural. Assim, era necessário seguir cinco regras: 1) É possível levantar uma tira do papel de um compartimento e analisar o conteúdo; 2) Não deve-se memorizar os números dentro do compartimento, ou fazer qualquer anotação, sendo ele considerado desconhecido; 3) Apenas no máximo duas tiras podem ser levantadas ao mesmo tempo; 4) O conteúdo das duas tiras podem ser comparados, para saber qual o menor número e 5) Duas tiras podem ser trocadas de compartimento (SETZER E CARVALHEIRO, 1993). Essa atividade permitiu trabalhar a busca de soluções de problemas, o desenvolvimento do raciocínio lógico e compreensão de algoritmos.

Assim, o desafio foi realizado em dupla e cada uma resolveu de uma forma diferente. Embora tenham compreendido o problema e ordenado os números, foi identificada uma dificuldade em escrever os passos que levaram ao resultado final. Portanto, percebeu a necessidade de levá-los a pensar e descrever as características do problema e as soluções obtidas nas atividades posteriores.

Além do desafio de ordenar os números, com o intuito de levá-los a ter um contato com a ferramenta foi proposta uma nova atividade no Scratch, onde pudessem fazer o personagem dizer "*Hello World*". Em seguida, foi pedido para receber como entrada o nome do aluno, e assim o personagem falar o nome junto com a expressão. A dificuldade inicial, foi apenas de saber onde encontrar os blocos corretos para realizar o desafio, ainda que o Scratch tenha sido apresentado nos primeiros momentos da aula, algumas dúvidas sobre a localização dos blocos persistiram. Embora tenha ocorrido esse período de adaptação, a atividade foi concluída com sucesso, e os discentes puderam compreender conceitos como entrada, processamento e saída de dados, algo importante para a aprendizagem da lógica de programação e algoritmos.

Alguns vídeos e materiais sobre conceitos de programação foram colocados no ambiente educacional, onde os estudantes poderiam ver em casa, sendo preparados para alguns conteúdos da próxima aula.

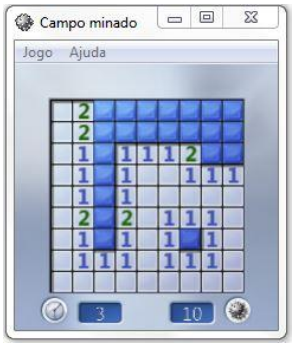
4.3.2 Segunda aula

A segunda aula do planejamento começou com os conceitos básicos de algoritmos e lógica de programação, destacando os conceitos de entrada, processamento e saída de dados, algo que eles tinham visto na aula anterior através da atividade no Scratch, e apresentando conteúdos importantes que iriam ser usados nas próximas atividades. Como os alunos eram repetentes, a parte de apresentação desses conceitos foi pequena e rápida, deixando o restante da aula para atividades, desafios e problemas a serem resolvidos, possibilitando ao discente ter um papel mais prático e ativo no aprendizado da programação. Assim, um dos primeiros problemas apresentados foi o do jogo campo minado.

4.3.2.1 Problema do campo minado

Apesar dos discentes conhecerem esse desafio, que surgiu no ano de 1989 no Windows 3.1 e desenvolvido por Robert Donner (VIENSCI, 2009), eles não compreendiam logicamente qual era a ideia do jogo, e as soluções que eles executavam era compostas por cliques aleatórios na arena do campo minado. O objetivo com essa atividade, foi de despertar e trabalhar o pensamento lógico dos alunos, já que o desafio proposto pelo jogo poderia ser resolvido seguindo passos lógicos. O desafio do campo minado, consiste em percorrer a arena sem clicar nas minas e encontrar todos os quadrados vazios. Para isso, assim que o jogador clica em um dos quadrados, é revelado um número, que informa a quantidade de minas que estão escondidas em torno dos quadrados que o cercam, figura 17.

Figura 17 – Desafio do campo minado



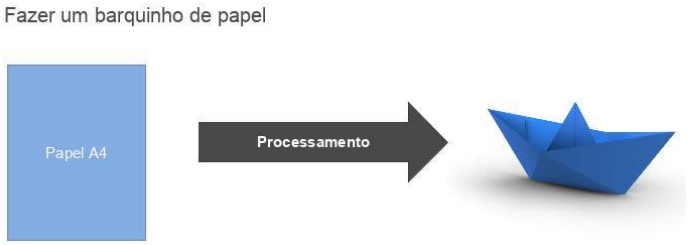
Fonte: Microsoft Windows (2019)

Apesar de parecer simples, os discentes puderam observar o jogo em uma perspectiva diferente, e tentaram fazer um desafio usando a dedução lógica. Embora tenham levado um tempo para completar o campo minado e encontrar todos os quadrados vazios sem minas, encontraram a solução sem o uso de cliques aleatórios, permitindo o desenvolvimento do pensamento lógico. Logo depois foi pensando em um possível algoritmo para o jogo do campo minado, e construída uma sequência de passos que permitiriam alguém encontrar a resolução do desafio. Nesse sentido, foi feita uma breve explicação do que seria um algoritmo e passados dicas de como descrevê-lo, depois apresentado um novo problema para os estudantes.

4.3.2.2 Problema do barco de papel

A nova atividade, que foi montar um barco de papel, teve como objetivo levar o acadêmico a fixar os conceitos de entrada, processamento e saída de dados, bem como conseguir formular uma sequência de passos lógicos e descrevê-los, formando assim um algoritmo e solucionando o problema. A primeira atitude a ser realizada foi ajudar os estudantes a identificar qual era o problema. A princípio nem todos lembravam ou sabiam montar um barco de papel, por isso antes de entenderem o que precisariam fazer, a professora levou os alunos a pensar no que seria necessário ter como entrada, o que teriam no processamento e no que queriam como saída, conforme observado na Figura 18. Foi feita uma primeira demonstração dos passos para construção do barco de papel, sem nenhuma descrição, onde todos os alunos puderam acompanhar na prática com as folhas que receberam.

Figura 18 – Exemplo de algoritmo para o barco de papel

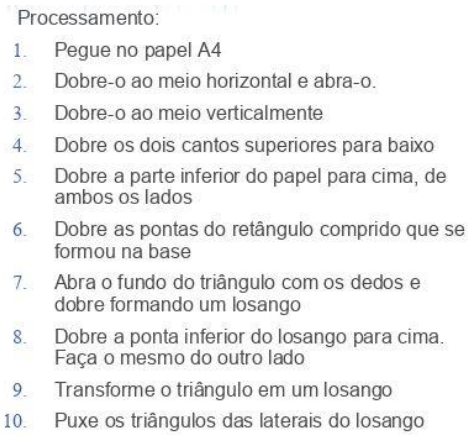


Fonte: Do autor (2019)

A folha de papel A4 foi demonstrada como entrada e o barco como a saída a ser alcançada. Assim, outra parte do desafio seria o processamento. Sendo assim, foi proposto aos discentes criarem o barco pela segunda vez sozinhos. Ao final da segunda construção, foi pedido para que eles descrevessem os passos que realizaram no processamento para formarem um possível algoritmo de resolução.

A resposta na descrição foi satisfatória, pois conseguiram entender que o processamento envolveu passos sequenciais e puderam perceber melhor o algoritmo na prática. Após a participação dos alunos na descrição dos passos, foi mostrado um algoritmo preparado previamente para aula, que também atendeu a sequência proposta pelos discentes, conforme a figura 19.

Figura 19 – Sequência de passos do processamento para solução do problema do barco de papel

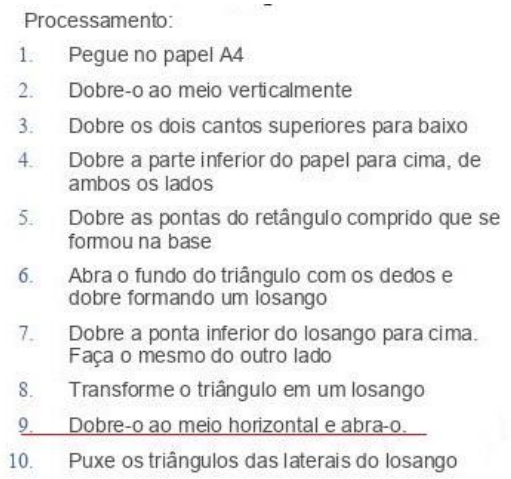


Fonte: Do autor (2019)

Logo depois, foi perguntado se uma mudança na ordem nos passos, prejudicaria o algoritmo. As respostas foram positivas, pois foi identificado no processamento realizado pelos alunos, que quando ocorreu o salto de um dos passos, ao invés do barco, se obteve um

chapéu. Então foi realizado um teste com uma mudança na sequência de passos do processamento apresentado, mostrado na figura 20.

Figura 20 – Mudança na sequência de passos do processamento para solução do problema do barco de papel.



Fonte: Do autor (2019)

Observando o passo nove em destaque, ele ocorria logo no passo dois, porém, a fim de análise e verificação, foi mudado para antes do último passo. Com isso, as dobraduras no papel não saíram corretas, o que impossibilitou a saída desejada do barco. Isso demonstrou a importância de criar algoritmo descrevendo a sequência de passos da maneira correta. Finalizando o aprendizado com o barco de papel, foi dado início a um novo exercício com o Scratch.

4.3.2.3 Problema do cálculo de salário com o Scratch

O exercício proposto para ser resolvido no Scratch, foi o cálculo de salário de um funcionário de acordo com a função exercida. A figura 21 mostra o problema apresentado para a turma.

Figura 21 – Exercício cálculo do salário

1) Calcule o salário de um funcionário de acordo com a função e as horas trabalhadas, conforme tabela a seguir (adote R$ 957,00 para o valor do salário mínimo):
As funções são divididas em: G – Gerente e P – Programador.

FUNÇÃO	VALOR HORA TRABALHADA
G	18% do salário mínimo
P	15% do salário mínimo

- Se o usuário informar uma função inexistente, mostrar uma mensagem de erro.
- Se o usuário digitar um texto em lugar da quantidade de horas trabalhadas, mostrar uma mensagem de erro.

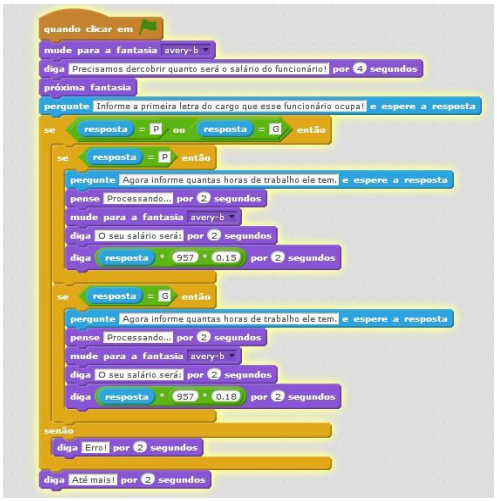
Fonte: Do autor (2019)

O problema permitiu abordar conceitos da lógica de programação, desde entrada, processamento e saída, como também as estruturas de condição, que apesar de não terem sido abordadas em uma aula anterior, permitiu os discentes retomarem o que tinham visto na primeira vez que cursaram a disciplina e praticar os conteúdos na resolução do exercício. A atividade foi passada para ser realizada em casa e enviada através do ambiente educacional *classroom*, as dúvidas posteriores seriam respondidas na próxima aula.

A princípio a atividade foi entregue por três alunos no ambiente, que usaram os recursos do Scratch para fazer o exercício com uma animação e conseguiram utilizar a programação em blocos com êxito, resolvendo o problema de forma satisfatória com um bom raciocínio lógico. A figura 22 mostra o que foi produzido por um dos alunos na atividade entregue, onde pode ser observada a tela de animação criada (Fig. 22(a)); e a estrutura do código (Fig. 22(b)).

Figura 22 – Exercício cálculo do salário resolvido por um dos discentes da disciplina

a) Tela de animação do problema



b) Código em blocos da solução do problema

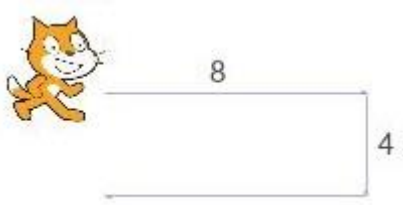
Fonte: Discente da disciplina (2018).

Percebe-se que mesmo sem ministrar a aula sobre o conteúdo de estruturas de condição, o aluno conseguiu cumprir o desafio com êxito, e conseguiu visualizar no Scratch, de forma rápida, a execução do código. Essa utilização do conceitos, mesmo sem uma prévia revisão, permitiu o discente firmar e praticar o conhecimento que possuía resolvendo o problema, tendo um papel ativo no seu aprendizado. Esse desafio permitiu demonstrar conteúdos de estruturas de condições, o uso de variáveis, de entrada e saída de dados e raciocínio lógico. A próxima aula foi utilizada para responder as dúvidas daqueles que não entregaram a atividade, demonstrar o método de Polya, propor mais problemas e resolver mais exercícios.

4.3.3 Terceira aula

A terceira aula foi iniciada com uma pequena apresentação expositiva a respeito de como resolver um problema, sendo mostrado em seguida a heurística de Polya para que fosse possível aos discentes descreveram os passos antes de resolverem qualquer problema. Isso permitiu com que pensassem no que precisariam como entrada e como seria o processamento para obter a saída desejada. Assim, seguindo as quatro etapas da heurística, poderiam solucionar os desafio apresentados de maneira mais rápida e mais eficiente. Como exemplo, foi mostrado o desafio de fazer um personagem percorrer um caminho no Scratch (figura 23).

Figura 23 – Exemplo usado para mostrar a heurística de Polya na prática



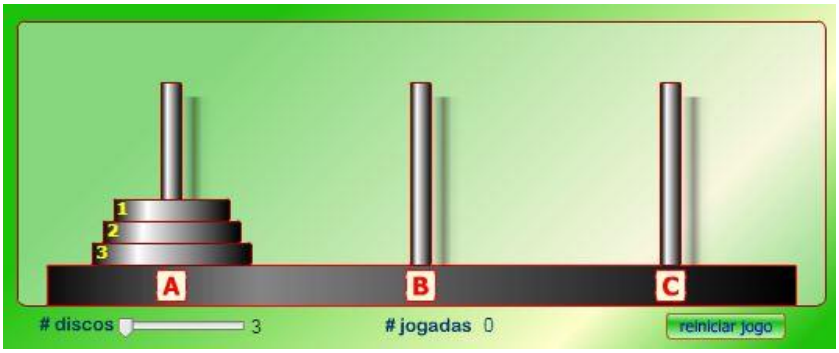
Fonte: Professora da disciplina (2019).

Apesar de ser um exercício simples, o intuito foi entender a heurística de Polya e como usá-la em qualquer problema. O método foi bem compreendido pelos alunos, que estavam acostumados em apenas executar o plano e tentar resolver os desafios. Após isso, foi apresentado um problema comum que permitiu aplicar o método de solução de problemas usando as quatro etapas estudadas.

4.3.3.1 Problema das torres de Hanói

A nova atividade apresentada foi o problema das torres de Hanói. O problema é constituído por três torres e uma quantidade variável de discos, sendo o caso mais simples constituído de dois ou mais discos, com tamanhos distintos, como pode ser visto na figura 24. O objetivo é passar todos os discos de uma torre para uma outra escolhida previamente, usando uma das torres como auxiliar para transferência e seguindo duas regras: 1) Só se pode mover um disco de cada vez para outro pino e 2) Um disco maior nunca pode ficar em cima de um disco menor.

Figura 24 – Problema da torre de Hanói



Fonte: Jogo desenvolvido pela Universidade Federal do Rio Grande do Sul (2005).[1]

[1] Disponível em: <https://www.ufrgs.br/psicoeduc/hanoi/>. Acesso em: 04 mar. 2019.

Esse desafio foi proposto para os discentes, que acessaram um jogo desenvolvido pela Universidade Federal do Rio Grande do Sul, conforme figura 21, e poderiam escolher a quantidade de discos e testar o raciocínio lógico para a solução do problema. Foi ressaltada a importância de, novamente, pensar no problema e nos passos que poderiam ser dados para chegar ao objetivo desejado, uma vez que movimentos aleatórios em nada iriam contribuir para o aprendizado. A princípio, no caso simples com três pinos, os resultados foram rápidos, à medida que aumentavam a quantidade de discos, a dificuldade crescia. Essa atividade possibilitou um aprendizado na forma de solucionar problemas, pensando sobre ele, traçando um plano e executando para chegar ao propósito final. Além disso, embora não tenha sido a finalidade do exercício apresentado, o desafio da torre de Hanói, pode ajudar a compreender o conceito de recursão (COSTA, 2008), algo tão importante para a lógica de programação.

Após a finalização com a torre de Hanói, foram iniciadas as atividades com o Scratch, durante as quais poderiam continuar pensando na estrutura lógica de resolver os problemas usando o método de Polya, sem se preocuparem com a sintaxe, já que usariam a programação em blocos.

4.3.3.2 Problema do cálculo do índice de massa corporal usando o Scratch

Antes de apresentar a lista com novos exercícios a serem resolvidos pela turma, foi iniciada uma discussão sobre o problema de cálculo do salário de um funcionário de acordo com a função, pois apenas três alunos tinham entregado a atividade no *classroom* e foi necessário falar sobre o desafio. E para ajudar aqueles que ficaram com dúvidas, foi aplicado o método de Polya que tinham usado no exercício anterior. Pode ser observado, que as dúvidas consistiam em não compreender totalmente o problema ou não saber quais estruturas que precisavam ser usadas para atingir o objetivo. Então foi proposto que cada um abrisse um bloco de notas, e começasse a escrever, seguindo as quatro etapas ensinadas, sobre o exercício. Assim houve uma demonstração da proposta de um algoritmo pela professora, que permitiu os alunos compreenderem o problema e traçarem o próprio plano. Com isso, o restante da turma conseguiu resolver o desafio e ver a saída desejada.

Um novo exercício foi apresentado para a turma, que deveria fazer uma pequena animação onde fosse possível calcular o índice de massa corporal – IMC, de uma determinada pessoa, de acordo com o problema demonstrado na figura 25.

Figura 25 – Problema para calcular o IMC

2) Filomena decidiu investir na sua saúde: está malhando e fazendo natação. Mas para verificar os resultados, ela quer saber o seu IMC – índice de massa corporal. Ajude Filomena a saber seu IMC e informe a ela o seu estado de acordo com a tabela abaixo:

PESO	MENSAGEM
menor que 17	Muito abaixo do peso
Igual a 17 ou menor que 18.5	Abaixo do peso
Igual a 18.5 ou menor que 25	Peso normal
Igual a 25 ou menor que 30	Acima do Peso
Igual a 30 ou menor que 35	Obesidade Grau I
Igual a 35 ou menor que 40	Obesidade Grau II
Igual a 40 ou maior que 45	Obesidade Grau III

Fonte: Do autor (2019)

Antes que os alunos tentassem resolver o problema no Scratch, foi proposto, novamente, que eles pensassem no problema antes, seguindo a heurística de Polya, onde traçassem um plano, utilizando um editor de texto comum e depois executaria usando a programação em blocos. De início não foi atrativo para os alunos essa descrição para traçar o plano, pois estavam acostumados a começar programando, mesmo sem compreender o problema. Contudo, foi possível observar na descrição algumas dificuldades na lógica, sendo possível auxiliar e corrigir antes de começar a programar. Na figura 26, é possível observar a descrição dos passos por um dos alunos.

Figura 26 – Descrição de um aluno do problema de IMC

1º)
a)a) Qual é o problema?
informar o imc da pessoa

b)Existem restrições? Quais?
o tipo muda dependendo do resultado

c)Tem alguma palavra ou termo desconhecido? Quais?
não

2º)
a)Já viu este problema antes ou algum similar?
já vi similar

b)Pode dividir este problema em partes? Quais?
sim, analisar o resultado até achar o tipo correto

c)Qual(is) a entrada do problema?
o peso e a altura da pessoa

d)Qual a saída esperada do problema?
o valor do imc da pessoa

e)Quais ações são necessárias para resolver o problema?
fazer o calculo do imc

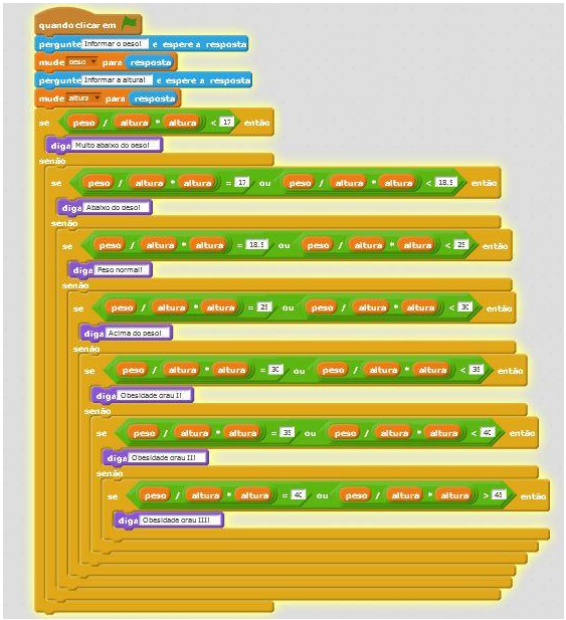
f)Escreva um algoritmo para resolver esse problema:
- pede ao usuario o peso
- pede ao usuario a altura
- calcular peso dividido pela altura x altura
- analisa o resultado
- compara na tabela
- devolve o resultado

Fonte: Discente da disciplina (2018)

O discente descreveu as duas primeiras etapas da heurística de Polya respondendo às perguntas que foram mostradas na sala de aula para auxiliar na compreensão do método e encontrar uma solução de forma mais prática. Percebe-se que mesmo com perguntas e respostas simples, é possível estabelecer uma estrutura lógica seguindo os passos descritos na figura 26. Dessa forma, com pequenas questões pode-se construir um algoritmo simples, entendendo o que vai ser preciso e qual o objetivo a ser alcançado. Após a compreensão do problema e descrição mostrada na figura 26, o aluno implementou o algoritmo no Scratch que permitiu a criação de uma animação para execução da atividade. A figura 27 mostra a tela de animação (fig. 27 (a)); e a tela de programação em blocos (fig. 27 (b)).

Figura 27 –Exercício IMC resolvido pelo discente da disciplina

a) Tela de animação do problema



b) Código em blocos da solução do problema

Fonte: Discente da disciplina (2018)

Assim, foi realizada a execução do plano, terceira etapa do método de Polya, e logo em seguida a comprovação do resultado pelo estudante. Além de colocar em prática as quatro etapas da heurística, a turma pode perceber o uso de estruturas de condições aninhadas, o uso de variáveis, operações matemáticas usando variáveis e uso de operações lógicas no exercício, mesmo não tendo nenhuma aula de revisão sobre esses assuntos. E nesse sentido, destaca-se a participação ativa do estudante no seu processo de aprendizado, que conseguiu realizar o desafio utilizando o método proposto e fazer a programação em blocos; e em que o professor

e monitores atuaram apenas como auxiliadores, sem muitos conceitos teóricos e praticando a programação. Com isso, observa-se que se adaptaram bem ao Scratch, uma vez que na ferramenta não é necessário se preocupar com a sintaxe, mas focar em toda semântica do problema e estruturar as ideias de lógica de programação, que nesse caso, seguiu um algoritmo criado e descrito pelos próprios acadêmicos. Após essas atividades, a aula foi encerrada, e como forma de exercício extra classe, deixado um exercício na plataforma educacional, para discutir as possíveis dúvidas e dificuldades na próxima aula.

4.3.4 Quarta aula

A quarta aula foi dedicada à resolução de exercícios com o Scratch e uso da heurística de Polya para resolução de problemas. A aula foi iniciada com o exercício passado no ambiente educacional, verificando quais foram as dificuldades e quem tinha feito a descrição e solucionado o problema e logo em seguida apresentada uma lista com vários problemas para os acadêmicos solucionarem e desenvolverem seu aprendizado na lógica de programação.

4.3.4.1 Problema do cálculo da média usando o Scratch

O primeiro problema consistia em ajudar uma professora com o cálculo da média do aluno e logo depois da média da turma. Sendo assim, a atividade foi dividida em duas questões. A turma conseguiu responder bem a primeira, contudo a segunda gerou algumas dúvidas e dificuldades. A atividade passada para turma é mostrada na figura 28.

Figura 28 –Problema do cálculo da média dos alunos e da turma

1) Em uma determinada escola, a professora estava com problemas para calcular a média dos alunos, por isso ela optou por usar o Scratch e pediu ajuda aos estudantes de Algoritmo e Programação I para construir um programa que possibilite informar as notas de um aluno e retorne a média dele.

Com seus conhecimentos sobre programação, crie um programa no Scratch que dê a média de um aluno. Considere que existem 4 unidades e a média é 6,0.

2) Com base no problema anterior, a professora observou que o programa pode melhorar. Além de calcular a média do aluno, ela quer saber a média geral da turma. Ajude a criar um programa que calcule a média geral da turma. (Lembre-se que a professora deve escolher entre calcular a média do aluno e calcular a média da turma).

Fonte: Do autor (2019)

Além de resolver o desafio utilizando o Scratch, foi pedido à turma para descrever os passos usando o método de solução de problemas. Como a primeira questão tinha sido respondida de forma satisfatória e os alunos tinham descrito o problema, não houve dúvidas em relação a ela, pois conseguiram realizar a programação em blocos sem dificuldades. A figura 29 mostra a descrição dos passos usando heurística de Polya apresentada (fig. 29 (a)); a tela de animação criada para o desafio (fig. 29(b)); e a tela de programação (fig. 29 (c)), criada por uma das alunas para solução da primeira a questão.

Figura 29 – Solução do problema para primeira questão

Etapa 1: Compreender o problema:
a) Qual é o problema?
Calcular a media dos alunos.

b) Existem restrições? Quais?
Tem que ter 4 unidades e a media é 6,0.

c) Tem alguma palavra ou termo desconhecido? Quais?
Não

Etapa 2- Traçar um plano
a) Já viu este problema antes ou algum similar?
Sim.

b) Pode dividir este problema em partes? Quais?
Sim. Determinar cada passo, como fazer, o que fazer.

c) Qual(is) a entrada do problema?
Notas das unidades.

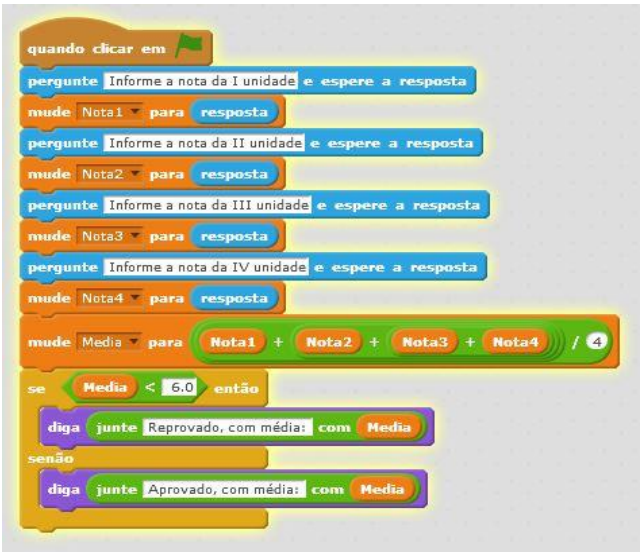
d) Qual a saída esperada do problema?
Média.

e) Quais ações são necessárias para resolver o problema?
Saber quais as notas do aluno (informados pelo usuarios),
calcular pela soma das unidades e
dividir tambem pelas unidades.

f) Escreva um algoritmo para resolver esse problema.
-Criar a variavel notas que sera informada pelo usuario de acordo com a unidade;
-Somar as notas da unidades e dividir pela quantidade de unidades;
-Colocar o resultado em media .
-Apresente a saida para o usuario.

a) Descrição dos passos e proposta de um algoritmo simples

b) Tela de animação criada para solução do problema



c) Código em blocos da solução do problema

Fonte: Discente da disciplina (2018)

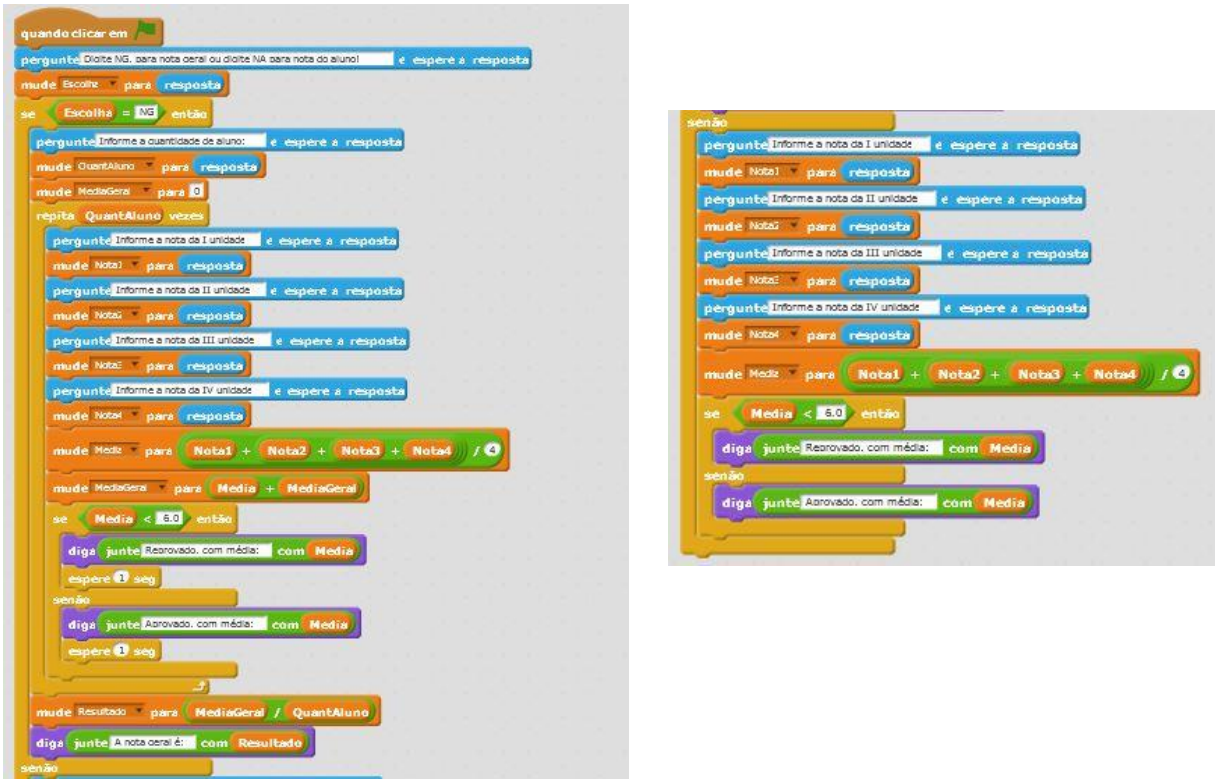
É possível perceber, que compreendendo o problema e traçando um plano, a execução acontece sem muitas dificuldades, porque além de não precisar focar nos problemas sintáticos, o desafio já foi bem entendido, dividido em partes e um algoritmo simples já foi proposto para implementação através da heurística ensinada. Com isso é importante observar que o estudante passa a ter um papel ativo e central no seu aprendizado, pois ele mesmo responde as questões de compreensão do problema, analisa o que será fornecido como entrada e saída, quais ações vão ser necessárias, propõe um algoritmo simples para solucionar o problema e implementa o plano traçado.

Ainda na primeira questão do problema, nota-se a compreensão dos conceitos e aplicação deles na prática, tanto das estruturas de condição, como também do uso de variáveis. Na figura 29 (c), pode ser observado como a aluna usou as variáveis para guardar os valores de entrada, como fez o processamento e guardou um resultado na variável média e apresentou o resultado para o usuário, incluindo até mesmo uma mensagem se o aluno foi reprovado ou não, aquém do pedido na questão, mostrando o interesse por parte da discente em implementar algo mais.

A segunda questão do desafio, de calcular a média geral da turma, é que provocou algumas dúvidas dos alunos e uma certa dificuldade no uso de estrutura de repetição. O primeiro questionamento foi quanto à descrição da questão (figura 28), de como seria realizado o cálculo geral da turma. Então, junto com os estudantes, foram esclarecidos alguns pontos: a professora deveria informar as notas dos alunos em cada unidade, seria realizado o cálculo da média de cada aluno individualmente, e, logo em seguida, fazer o cálculo da média da turma. O segundo questionamento, foi quanto ao uso das estruturas de controle, que, diferente das estruturas de condição, não foi tão simples de compreender, mesmo não tendo uma aula de revisão do conteúdo. Apesar disso, com o auxílio da professora e monitores, conseguiram relembrar e entender o que tinham visto na primeira vez que cursaram a disciplina.

Após isso, cada um, individualmente, passou a traçar o plano para a solução, como já haviam feito isso na primeira questão, foi necessário apenas algumas modificações nas respostas e no algoritmo planejado. Na figura 30 é possível visualizar a modificação do código de programação em blocos para a solução do problema pela mesma aluna citada anteriormente (fig. 29).

Figura 30 – Solução da segunda questão do problema de cálculo da média



Fonte: Discente da disciplina (2018)

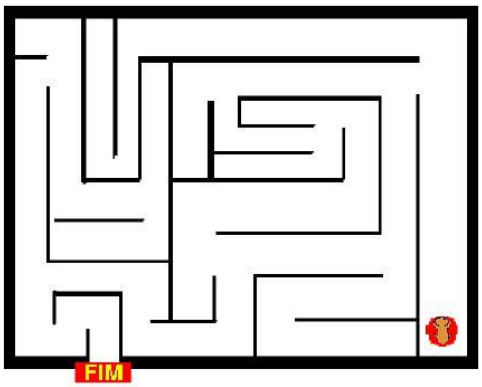
Pode-se observar que a aluna acrescentou uma estrutura de condição global, para atender a necessidade do usuário de escolher se gostaria de calcular a média de apenas um aluno ou calcular a média da turma. Com isso, uma estrutura de controle foi adicionada para o caso da escolha ser a média geral da turma, onde a professora deveria informar a quantidade de alunos e em seguida as notas das unidades de cada um deles. Após isso o programa calcularia a média de cada aluno e somaria cada uma dessas médias em uma variável, dividindo-a pela quantidade de alunos informada logo no início da aplicação. Com essa atividade, o conteúdo de estruturas de repetição, estruturas de condição, entrada e saída de dados, processamento e variáveis foi bem compreendidos pelos estudantes, que puderam praticar e ver a execução através da tela de animação do Scratch.

É importante destacar, que mesmo sem uma aula prévia de estruturas de repetição, os alunos conseguiram usar na programação em blocos, sem a preocupação com a sintaxe, compreendendo o conceito por trás do comando e sua utilização. A próxima etapa da aula foi o desafio para fazer um labirinto, onde o personagem pudesse chegar ao destino sem tocar nas paredes.

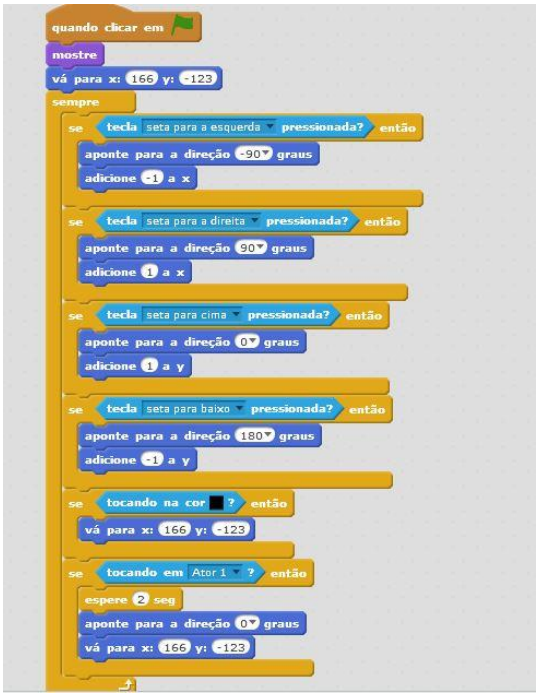
4.3.4.2 Desafio do labirinto

Esse desafio permitiu a aprendizagem das estruturas de condição e de laços, onde utilizaram as estruturas aninhadas, de modo a verificar se o personagem tocava nas paredes do labirinto ou se havia conseguido chegar ao destino. Nessa etapa não foi cobrada a entrega da descrição do problema como nas atividades anteriores, eles iriam traçar o plano, usando a heurística de solução de problemas, depois realizar a implementação e comprovar o resultado. A figura 31 mostra o resultado produzido por um dos alunos da disciplina (fig. 31 (a) mostra a tela de execução, e a fig.31(b) mostra o código realizado pelo discente).

Figura 31 – Solução do desafio do labirinto



a) Tela de execução do desafio



b) Implementação usando programação em blocos

Fonte: Discente da disciplina (2018)

Nota-se na codificação do problema que foi utilizada estruturas aninhadas, com verificações a serem atendidas. A turma, mesmo sem aulas teóricas sobre esses conceitos, conseguiu executar bem essa atividade, encontrando a solução sem dificuldades. Isso mostra que à medida em que eles estavam resolvendo os exercícios, com a compreensão do

problema, traçando um plano, implementando e visualizando os resultados, eram capacitados para desafios mais avançados, praticando a programação sem a complexidade de observar detalhes sintáticos, contribuindo para um enfoque maior no pensamento lógico e na estruturação das ideias para cumprir o que era proposto. Assim o desafio do labirinto foi cumprido de modo satisfatório pela turma.

4.3.5 Quinta aula

Na quinta aula um novo desafio foi proposto com o objetivo dos estudantes praticarem o raciocínio lógico e as estruturas de programação, principalmente entenderem o uso de condicionais, onde pudessem criar um quiz criativo com dez perguntas de assuntos diversos. O enunciado do exercício proposto pode ser observado na figura 32.

Figura 32 – Enunciado do exercício da criação de um quis

1) Faça um quiz no Scratch com no mínimo 10 perguntas. O quiz deve ter 4 alternativas para cada pergunta. O usuário deve clicar na alternativa, se a alternativa estiver errada, ela deve ficar vermelha, se estiver correta, deve ficar verde. Para cada pergunta certa o usuário pode ganhar diferentes quantidades de pontos de acordo com a dificuldade da pergunta. Por exemplo, em uma pergunta fácil, se ele acertar, ganha mais 1 ponto, em uma pergunta difícil, se ele acertar, ganha mais 2 pontos.

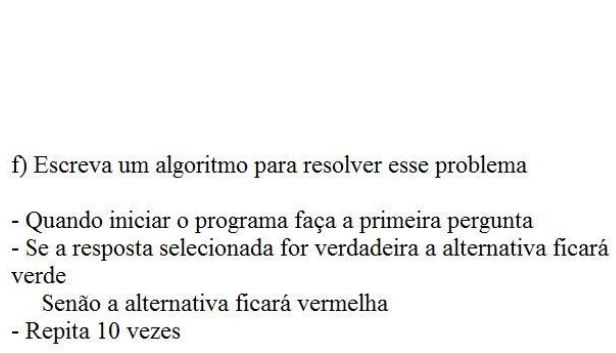
Fonte: Do autor (2018)

A princípio eles demonstraram certa dificuldade em como construir uma estrutura que solucionasse o problema, apesar de terem compreendido a atividade. Assim, depois de apresentada a atividade, usar o método de resolução de problemas contribuiu para traçar o plano apropriado e colocá-lo em execução. Criar um plano antes de começar a implementar e testar o código, possibilitou definir as perguntas que iriam ser utilizadas, os assuntos e como seria a passagem de uma pergunta para outra depois que o usuário respondesse a questão. Além disso, traçar esse plano permitiu estabelecer algumas regras: o usuário não poderia responder duas vezes a mesma pergunta, não poderia voltar a uma pergunta respondida e a informação do acerto ou do erro seria informado assim que clicasse na alternativa. Dessa forma em caso de acerto somaria um determinado número de pontos e em caso de erro diminuiria uma determinada quantidade de pontos.

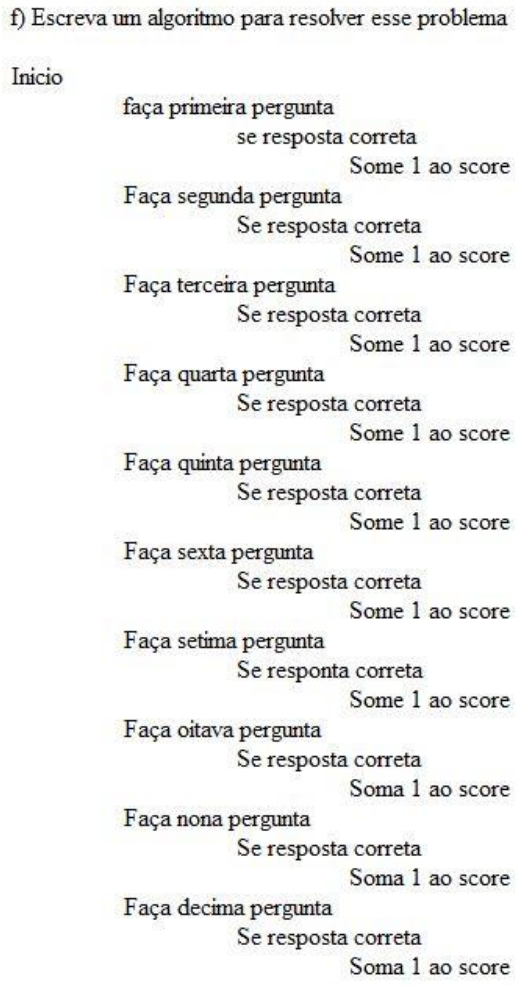
Essa identificação do que o quiz teria como entrada e o que mostraria como saída, foi um dos fatores principais para os alunos entenderem as estruturas que precisariam formar com a programação em blocos, ou seja, quais objetos seriam necessários, quais comandos deveriam ser utilizados e o que poderiam fazer para passar de uma pergunta para outra. Isso foi de fundamental importância para trabalhar o raciocínio lógico, uma vez que precisariam criar uma estrutura que respeitasse as regras e que pudesse ser utilizada para acrescentar mais perguntas sem a necessidade de modificação de todo o algoritmo.

Desse modo, cada acadêmico implementou o quiz com aspectos distintos, tanto pelas perguntas e a forma demonstrativa de animação, como também no planejamento e nas estruturas de programação usadas. A figura 33, mostra um algoritmo simples feito por dois estudantes quando descreveram o planejamento usando a heurística de Polya.

Figura 33 – Algoritmos simples criados por dois estudantes



a) Discente 1



b) Discente 2

Fonte: Discentes da disciplina (2018)

Observar-se que os passos construídos foram pensados de modo diferente, enquanto um utilizou um passo de repetição, apesar de não ter sido realizada uma aula teórica sobre o assunto (fig. 33 (a)), outro utilizou vários comandos de condição para avaliação das respostas no quiz (fig. 33 (b)). Isso mostra que as alternativas encontradas pelos discentes tomaram caminhos diferentes para a solução do problema, e mesmo com uma descrição básica eles já conseguem entender melhor o que precisam fazer na hora da implementação, o que pode contribuir para um melhor aprendizado e preparação para problemas maiores.

Após todo o plano traçado e problema compreendido os alunos foram produzindo o quiz, inclusive editando imagens, criando menus iniciais com a programação em blocos e utilizando os recursos da ferramenta combinado com a lógica produzida por eles no planejamento. O discente, que formou o algoritmo simples da figura 33 (a), não usou uma estrutura de repetição da programação em blocos, mas conseguiu criar a programação para cada objeto no Scratch e foi reutilizando em objetos que teriam a mesma função dentro do quiz, o que trouxe agilidade na hora de solucionar o problema. Assim o resultado do que a discente produziu pode ser observado na figura 34, onde consta a tela de animação (fig. 34 (a), parte do código em blocos e a lógica de programação usada (fig. 34(b)).

Figura 34 – Resultado do projeto da aluna

a) Tela inicial do projeto

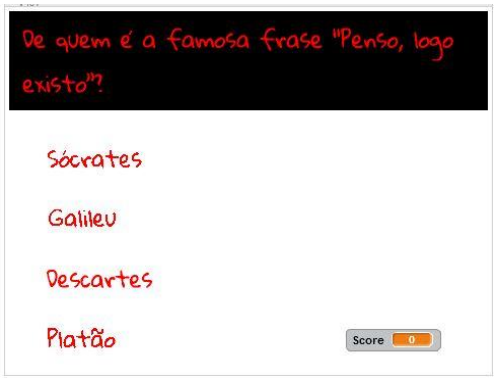
b) Código de programação em blocos da solução

Fonte: Discente da disciplina (2018)

O resultado do projeto foi bem satisfatório e percebe-se que houve uma mudança na descrição do algoritmo e na sua execução. Apesar da ideia principal ser mantida, a estudante acrescentou a quantidade de pontos ganhos e perdidos, dependendo da pergunta que iria ser respondida, e utilizou estruturas de seleção para identificar qual era a pergunta atual e assim demonstrar se a alternativa clicada era correta ou errada. Mesmo com a mudança na descrição, o problema foi bem entendido e executada, inclusive na tela de animação inicial foi mostrado as regras e a variável de pontos aos jogadores, que quando começava o quiz a variável era iniciada com zero. Além disso, a estudante programou para que quando a alternativa clicada fosse a certa a cor mudaria para verde, caso contrário para vermelho.

A rápida apresentação do resultado na tela do Scratch, permitiu que a turma progredisse nas dificuldades, um vez que, mesmo com o plano traçado enxergavam alguns erros, eles conseguiam entender o que estava acontecendo de forma ágil, e encontravam nos blocos de programação uma maneira de estruturar melhor a lógica que tinham pensando. O segundo aluno, que descreveu o algoritmo simples da figura 33 (b), criou o projeto de forma diferente utilizando uma lógica que necessitou de mais objetos no projeto, o que não influenciou no resultado e nem na execução do plano traçado. A figura 35, mostra como foi a utilização dos objetos na tela (fig. 35 (a)) e a lógica usada para passagem entre as perguntas (fig. 35 (b)).

Figura 35 – Resultado do projeto do aluno



a) Tela do projeto em execução

b) Código de programação em blocos da solução

Fonte; Discente da disciplina (2018)

Nesse projeto, nota-se que o aluno não usou uma estrutura de condição, mas com a ferramenta de programação em blocos, ele conseguiu produzir uma estrutura semelhante, caso

o objeto clicado dentro daquela pergunta fosse correta, o jogador ganharia um ponto, caso contrário, não ganharia nenhum ponto. Assim como o projeto anterior da aluna, o aluno implementou para quando o jogador clicasse em uma resposta correta a alternativa ficasse verde, caso contrário ficaria amarela. O resultado do estudante permitiu enxergar que o pensamento lógico usado para resolver o problema foi bem satisfatório, uma vez que utilizando recursos da programação em blocos conseguiu simular estruturas lógicas que determinassem quando o usuário errasse ou acertasse a pergunta.

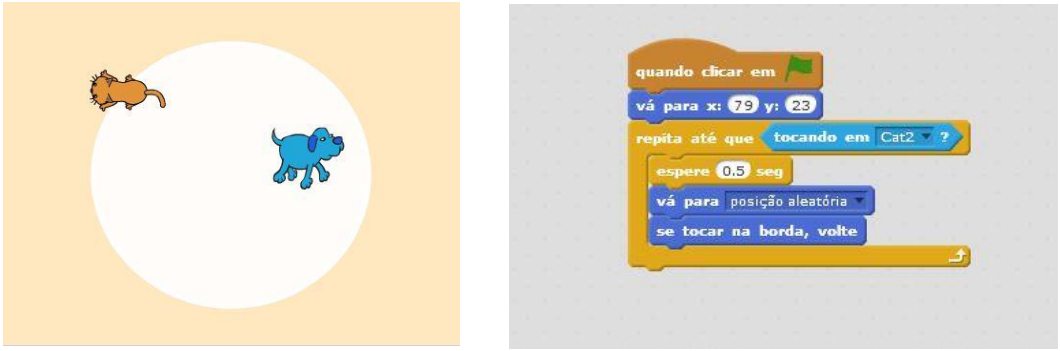
Esses dois projetos, foram apenas parte do que a turma conseguiu produzir na construção do quiz, que abrangeu o raciocínio lógico e a forma como cada um deveria estruturar as ideias.

4.3.6 Sexta aula

Percebendo o avanço da turma nos desafios e na utilização de estruturas de repetição, onde os conceitos não haviam sido trabalhados na sala, a sexta aula foi iniciada com uma breve explicação sobre o uso de laços, para lembrar aos repetentes aquilo que eles já tinham visto na primeira vez que cursaram a disciplina. Isso foi importante para conhecer qual era a compreensão dos estudantes em relação ao conteúdo. Dessa forma, foi passado dois desafios simples para que pudessem praticar o uso da repetição, mesmo que eles já tivessem utilizados em atividades anteriores, para assim observar o aprendizado de cada um e como tinham entendido o conceito dessas estruturas.

O primeiro desafio foi para que fizessem dois atores andarem pela tela até que um tocasse no outro, isso permitiu demonstrar o laço com as características do *while*, pois iriam ficar se movimentando pela tela até se encontrarem. Foi um desafio bem simples, e que a turma conseguiu finalizar sem dificuldades, pois compreenderam bem o problema e entenderam o que era necessário fazer. Mesmo não sendo exigido a digitação do planejamento para a solução, como nas demais atividades, eles foram passando por cada etapa da heurística de Polya e conseguiram apresentar um resultado satisfatório. A figura 36 mostra o resultado de um dos alunos.

Figura 36 – Resultado do desafio para prática de repetição na programação



a) Tela de animação do projeto

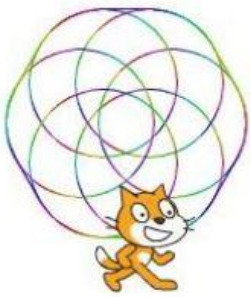
b) Código do projeto em blocos

Fonte: Discente da disciplina (2018)

O discente utilizou corretamente o bloco de repetição, de modo a movimentar os dois personagens para posições aleatórias da tela até se encontrarem. No código da figura 36 (b), é possível observar a programação do cachorro, que quando tocasse no gato sairia do laço.

O segundo problema envolveu um maior grau de complexidade, pois iriam precisar usar laços aninhados. Nele era preciso desenhar seis círculos coloridos, conforma a figura 37.

Figura 37 – Exemplo do que deveria ser produzido pela turma



Fonte: Do autor (2018)

Com esse desafio era possível auxiliar no aprendizado das construções dos laços aninhados e do pensamento lógico para desenhar os círculos na tela. Um dos acadêmicos, mesmo não sendo exigido, digitou um planejamento simples para solução do problema de modo a compreender melhor o que devia ser feito, o que contribuiu para estruturar a lógica e obter um resultado satisfatório. O código produzido pelo aluno pode ser observado na figura 38, onde junto com os lações aninhados, utilizou a lógica da mudança de ângulos para obter os seis círculos esperados.

Figura 38 – Código em blocos da solução do problema criada pelo discente

Fonte: Discente da disciplina (2018).

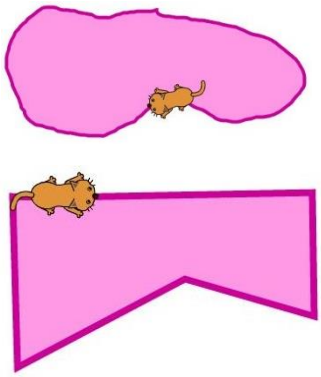
Dessa forma, a turma conseguiu progredir bem nesse desafio, pois encontrou a solução para o problema e utilizou as estruturas aninhadas. Apesar de no início apresentar algumas dificuldades na compreensão de como usariam a programação para chegar ao objetivo, por fim entenderam e organizaram bem a estrutura do código. Após as duas atividades, foi apresentada a proposta de projeto para a primeira unidade da disciplina com o objetivo de desenvolverem o raciocínio lógico e trabalharem os conceitos iniciais da programação tendo como base os problemas já resolvidos. O projeto deveria ser desenvolvido no Scratch e cada discente deveria pensar em um jogo a ser produzido, lembrando dos aprendizados de como resolver um problema através da metodologia apresentada, pesquisar material extra se fosse necessário, entregar e apresentar o projeto na última aula da primeira unidade. As próximas três aulas serviriam para mais um exercício e planejamento do projeto.

4.3.7 Sétima aula

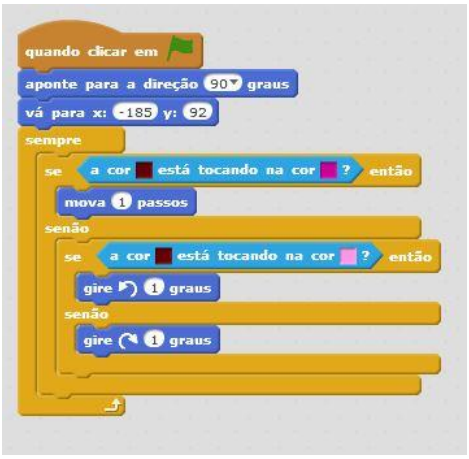
Antes dos alunos iniciarem o planejamento e execução do projeto, foi proposto mais uma atividade com o intuito de firmarem os conceitos de condições e laços em programação. A professora então lançou o desafio de fazerem um seguidor de linha, onde um personagem

poderia percorreria circuitos, de determinada cor, que mesmo alterando o formato do percurso continuaria a seguir a linha sem problemas. Embora os acadêmicos tenham levado uma boa parte do tempo pensando no problema e tentando encontrar uma estrutura adequada da lógica, não foi um problema difícil de ser resolvido, uma vez que com o auxílio da professora e discutindo entre si, conseguiram montar um algoritmo simples para resolver o problema. A figura 39 mostra o resultado de um dos discentes.

Figura 39 – Resultado do desafio do seguidor de linha



a) Percursos construídos na tela de animação



b) Código em blocos da solução do problema

Fonte: Discente da disciplina (2018)

Com isso é possível observar que mesmo com a mudança do percurso o personagem continuava a seguir a linha, pois o código produzido avaliava sempre a cor do objeto, e enquanto não encontrasse a cor continuaria a se movimentar procurando a linha do circuito que deveria seguir. No código observa-se a estrutura de repetição e condição juntas, que faziam o personagem seguir sempre a linha e avaliava condições para que não se desviasse do caminho. Isso foi importante, para enxergar como os alunos estavam trabalhando o raciocínio lógico e de que maneira encontravam soluções para os desafios com os conteúdos de programação.

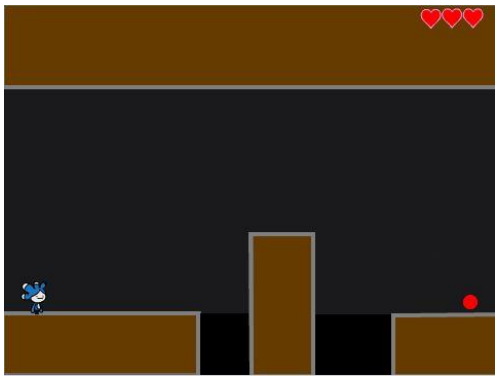
Sendo assim, a solução encontrada pela turma foi bem satisfatória, pois o código em blocos ficou bem conciso, solucionando o desafio com eficiência e eficácia. A partir do término desse exercício, a turma agora iria usar as duas aulas restantes para planejar o projeto da primeira unidade e colocar o plano em execução, restando a última aula para entrega e apresentação de cada trabalho.

4.3.8 Apresentação dos projetos

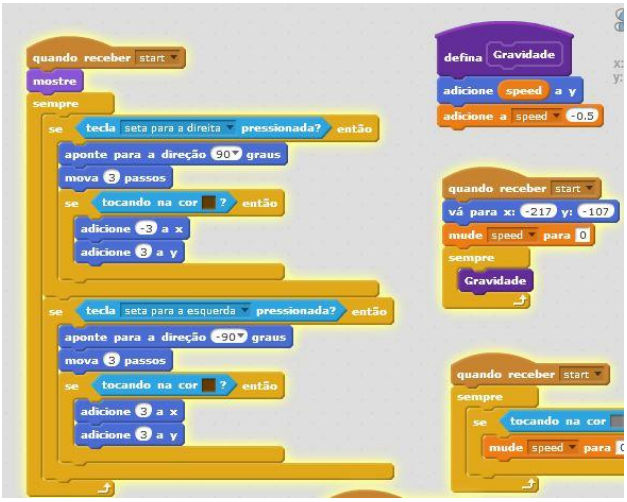
Na última aula da primeira unidade cada discente entregou e apresentou o projeto, que consistiu na produção de um jogo utilizando os conteúdos praticados em sala de aula. A turma conseguiu executar bem todo projeto, alguns alunos, inclusive, utilizaram pesquisas e material extra classe para conseguir produzir o jogo desejado, mostrando motivação no cumprimento do trabalho. Dos oito alunos da disciplina sete conseguiram entregar o projeto e o resultado foi bem interessante, porque alguns utilizaram recursos da ferramenta que não haviam sido explicados na aula, e utilizaram o raciocínio lógico de modo bem estruturado, alcançando o objetivo do que queriam fazer.

Um, dos sete projetos produzidos, foi um jogo de plataforma, onde era necessário levar o personagem a alcançar uma esfera de luz em cada fase, para isso precisava desviar de obstáculos e pular entre superfícies. A figura 40 mostra a tela de animação do projeto (fig. 40 (a)) e a parte do código em blocos (fig. 40 (b)).

Figura 40 – Projeto de jogo da primeira unidade



a) Tela do jogo em execução



b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Nesse projeto o aluno utilizou cerca de trinta objetos, algumas variáveis, operações lógicas, estruturas de condição, estruturas de repetição e função através do recurso do Scratch para a execução do projeto. As estruturas de condição, inclusive as aninhadas, foram muito

bem utilizadas por ele, assim como as estruturas de repetição, embora tenha sido redundante em certos momentos, conseguiu entregar um projeto bem estruturado.

Ao analisar o projeto foi possível observar também que o discente fez pesquisas extraclasse, o que possibilitou um maior rendimento no uso e na motivação para programar seu próprio projeto, pois conseguiu criar uma função, uma vez que esse conteúdo não tinha sido trabalhado na primeira unidade da disciplina e nem mostrado na ferramenta de programação em blocos.

As variáveis utilizadas serviram para controle da vida e de alcance do pulo do personagem. O jogo poderia ter melhorias nos movimentos do personagem e o salto entre as superfícies, mas se tratando das pesquisas e da ferramenta, ele foi bem produtivo no projeto usando recursos e conceitos de programação. Assim, com a execução desse projeto, o aluno pode se empenhar e se motivar no aprendizado da programação, desenvolvendo o raciocínio lógico.

Os outros seis projetos também foram bem produzidos e podem ser observados no anexo A. Sendo assim, o projeto foi importante para firmar os conteúdos da primeira unidade da disciplina e levarem os repetentes a desenvolverem melhor o raciocínio lógico e a participarem de forma mais ativa na construção do seu aprendizado, sem focar nas complexidades das estruturas sintáticas e possibilitar à eles a produção do próprio jogo.

Dessa forma, a primeira unidade, foco desse projeto, foi encerrada com sucesso, sendo necessário analisar se tudo o que foi trabalhado nas aulas possibilitou uma mudança na aprendizagem da disciplina.

5 ANÁLISE DOS RESULTADOS

Após o encerramento da primeira unidade, onde foi trabalhado os conceitos iniciais e básicos da programação usando a linguagem em blocos e uma metodologia ativa de ensino, foi realizado outro questionário com os alunos da disciplina no intuito de analisar e avaliar as opiniões e resultados obtidos, fazendo uma conclusão dos rendimentos, desafios que foram observados e das melhorias que podem ocorrer no futuro.

O questionário foi composto de oito afirmações que foram avaliadas de acordo com a escala Likert de 5 pontos, sendo que 1 para discordo totalmente, 2 para discordo parcialmente, 3 para não concordo nem discordo, 4 para concordo parcialmente e 5 para concordo totalmente. Além disso, houve duas perguntas abertas com o objetivo de avaliar de maneira mais detalhada a opinião sobre o uso da programação em blocos e algumas sugestões dos

discentes como melhorias a serem realizadas. Os oito discentes matriculados na disciplina responderam ao questionário.

Na primeira afirmação 1) Gostei de utilizar programação em bloco na disciplina de API, sete alunos avaliaram que concordam totalmente com a afirmação, marcando 5 pontos, e um avaliou que concorda parcialmente, marcando 4 pontos, conforme mostra a figura 41.

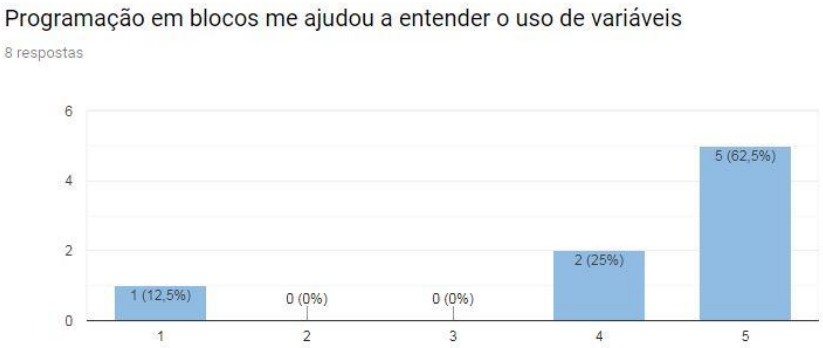
Figura 41 – Resposta do questionário da primeira afirmação

Fonte: Do autor (2018)

Isso permite entender, que mesmo já tendo tido um contato com uma linguagem textual antes, todos acharam interessante a programação em blocos e gostaram da experiência. Na segunda afirmação, 2) programação em bloco me ajudou a entender lógica de programação, foi obtido o mesmo resultado da afirmação anterior, assim, além de terem gostado, concordaram que a linguagem contribuiu para um entendimento da lógica.

Já a terceira afirmação do questionário, 3) Programação em bloco me ajudou a entender o uso de variáveis, mostrou-se opiniões mais específicas, como mostra a figura 42.

Figura 42 – Resposta do questionário da terceira afirmação

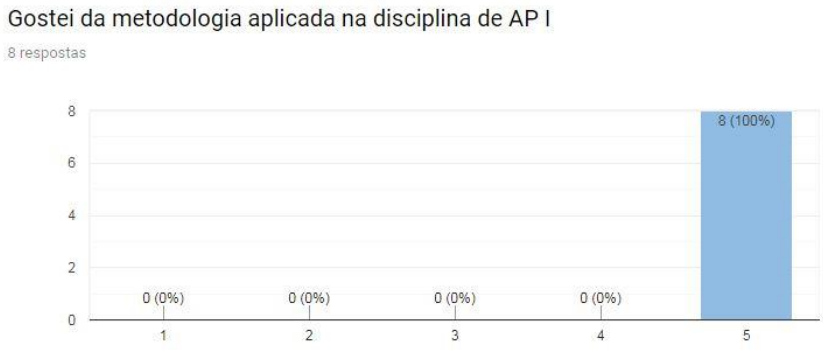


Fonte: Do autor (2018)

Como percebe-se, cinco alunos concordaram totalmente, marcando 5 pontos, que a programação em bloco ajudou no entendimento do conteúdo de variáveis, dois alunos concordaram parcialmente, marcando 4 pontos, e um aluno discordou totalmente, marcando 1 ponto. Esse resultado se repetiu nas afirmações seguintes, 4) Programação em bloco me ajudou a entender como usar comandos de seleção (se então, se então senão) e 5) Programação em bloco me ajudou a entender como usar comandos de repetição. Sendo assim, a maioria dos alunos concordaram que o uso de programação em blocos contribuiu para o entendimento desses conceitos da disciplina.

Sobre a metodologia usada, na sexta afirmação, 6) Gostei da metodologia aplicada na disciplina de API, todos os discentes marcaram 5 pontos, concordando totalmente, conforme figura 43. Isso mostra que a metodologia foi importante para o estímulo e motivação do aluno, uma vez que todas as aulas foram no laboratório e todos se envolveram no seu processo de aprendizado de maneira ativa, criando, produzindo e executando.

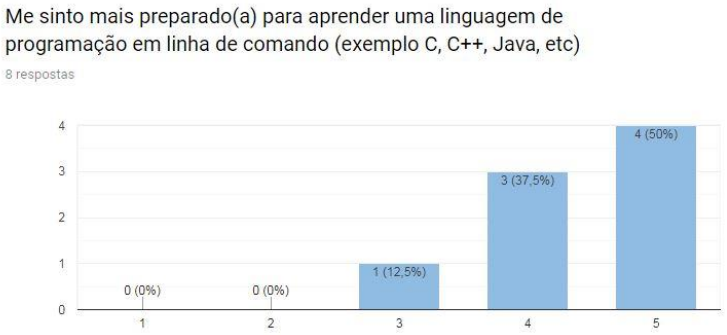
Figura 43 – Resposta do questionário da sexta afirmação



Fonte: Do autor (2018)

Sobre a sétima afirmação no questionário, 7) Me sinto mais preparado(a) para aprender uma linguagem em linha de comando (exemplo C, C++, Java, etc), quatro discentes responderam que concordavam totalmente, três que concordavam parcialmente e um que não concordava e nem discordava da afirmação, conforme figura 44. A partir disso, observar-se que a maioria dos acadêmicos, depois da experiência com a programação em bloco, se sentiam mais preparados para aprender uma linguagem textual de programação.

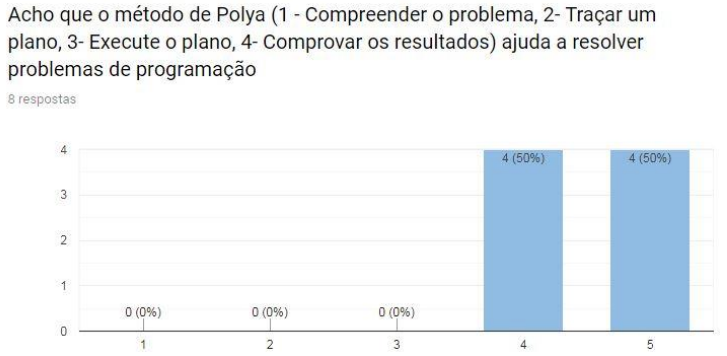
Figura 44 – Resposta do questionário da sétima afirmação



Fonte: Do autor (2018)

Na última afirmação, 8) Acho que o método de Polya (1 – Compreende o problema, 2 – Traçar um plano, 3 – Execute o plano, 4 – Comprovar os resultados) ajuda a resolver problemas de programação, 50% dos acadêmicos concordaram totalmente, e 50% concordaram parcialmente com a afirmação, conforme figura 45. Nota-se que consideraram a heurística interessante, já que na primeira vez que cursaram a disciplina, tentavam resolver o problema usando uma linguagem de programação, mas não paravam para compreender o desafio e fazer o planejamento da solução.

Figura 45 – Resposta do questionário da oitava afirmação



Fonte: Do autor (2018)

Após a resposta das oito afirmações, foram respondidas as duas perguntas abertas, procurando entender a opinião dos alunos sobre a experiência com programação em blocos, o que haviam gostado ou não, e pedido sugestões para as próximas unidades da disciplina. Todas as respostas foram positivas em relação a experiência com a programação, consideraram a linguagem em blocos mais interativa e menos cansativa quanto a programação textual, conseguiam testar as condições de modo que entendiam o que devia ser corrigido e

gostaram da interação com a ferramenta Scratch e sua simplicidade. As sugestões se destacaram por pedir mais exercícios nas próximas unidades e que continuasse a ter mais aulas práticas.

Essas opiniões demonstraram o bom resultado que foi obtido, referente a motivação e estímulo dos discentes, que apoiaram e gostaram da experiência, tanto com a metodologia quanto com a ferramenta de programação em blocos. Embora tenha sido verificado um resultado satisfatório, nas aulas práticas e na pesquisa, alguns desafios foram observados ao longo da aplicação da pesquisa.

A quantidade de estudantes foi menor do que em turmas iniciais, o que permitiu um acompanhamento melhor de cada aluno na compreensão do problema e execução das atividades práticas, facilitando um bom resultado dos alunos. Outro aspecto observado, é que, conquanto tenham usado a heurística de Polya, adotada na metodologia ativa, em certos exercícios não se sentiram motivados a traçar um plano, partindo logo para implementação e tendo dificuldades em entender certos aspectos do problema.

O Scratch se mostrou uma boa ferramenta na compreensão inicial da disciplina, mas é sugerido o desenvolvimento de uma ferramenta própria para ensino de estudantes iniciais no ensino superior de computação, uma vez que o Scratch não possui alguns recursos para o ensino de certos conteúdos, como por exemplo vetores. Outra questão importante é possuir uma ferramenta que possa fazer a conversão da programação em blocos para um código textual de uma determinada linguagem, o que auxiliaria os discentes a fazerem uma relação melhor no aprendizado prático.

Mesmo com esses desafios, os estudantes empenharam-se nas atividades e exercícios, se sentindo motivados nas pesquisas extraclasse para desenvolvimento do projeto. Além do mais, foi possível observar um estímulo maior nas atividades, e até sugeriram que pudesse ter mais exercícios práticos. A metodologia ativa, incluindo a heurística de Polya, proporcionou ao estudante uma participação maior no próprio aprendizado e no desenvolvimento do raciocínio lógico, junto com a programação em blocos que se mostrou uma ótima aliada no ensino dos conceitos iniciais, preparando o aluno para execução das soluções encontradas, sem a complexidade de estruturas sintáticas.

6 CONCLUSÕES

Diante das dificuldades encontradas no ensino de programação, tanto pelos desafios que o professor enfrenta, como também pela complexidade no aprendizado por parte dos discentes, é necessário encontrar alternativas que proporcionam uma participação mais ativa dos alunos, de modo que possam ser estimulados e motivados para a aprender programação. Logo, ter aulas mais interativas e dinâmicas, não somente com o uso de apresentações, mas envolver exercícios que permitem a prática da solução de problema e das estruturas de programação, é um fator importante para que o estudante possa desenvolver-se nessa área e avançar de maneira satisfatória nos cursos de computação.

Dessa forma, o presente trabalho utilizou uma metodologia ativa de ensino, junto com uma ferramenta de programação em blocos Scratch para o ensino na primeira unidade da disciplina Algoritmos e Programação I, da Universidade Estadual do Sudoeste da Bahia – UESB, em uma turma especial. O resultado atendeu as expectativas e, apesar do número pequeno de alunos, o trabalho foi bem aplicado e executado. Sendo assim, a experiência foi considerada proveitosa, uma vez que os alunos conseguiram compreender os conceitos e desenvolver os projetos propostos.

O uso da linguagem em blocos foi eficiente no ensino das estruturas básicas da programação, pois removeu a complexidade sintática, e os alunos puderam desenvolver melhor o pensamento lógico, deixando os desafios das construções sintáticas para a segunda unidade. É certo que mais avanços e avaliações devem ser feitas, inclusive para aplicação em turmas iniciais, pois a pesquisa foi realizada com alunos repetentes, e que, de certo modo, já haviam tido um contato com a programação, mesmo não havendo firmado os conceitos necessários para uma aprovação anterior.

Assim sendo, o uso da metodologia ativa e programação em blocos contribuiu para os estudantes repetentes firmarem o conceitos, desenvolverem habilidades na solução de problemas e executarem a solução usando as estruturas de programação, dessa forma o presente trabalho foi concluído de forma satisfatória e eficaz.

6.1 TRABALHOS FUTUROS

Destaca-se os seguintes trabalhos futuros a serem realizados:

- Desenvolvimento de uma ferramenta de programação em blocos própria para o ensino introdutório de programação, para turmas inicias do curso de computação.
- Aplicação da metodologia ativa, junto com heurística de solução de problema e linguagem de blocos na turma inicial de computação.
- Análise e avaliação da transição entre a linguagem em blocos e a linguagem de programação textual.
- Desenvolvimento de um ambiente virtual onde podem construir suas ideias e projetos, de modo a praticar ainda mais desafios, incentivar o aprendizado ativo e pesquisas extraclasse.

REFERÊNCIAS

AMBRÓSIO, Ana Paula; COSTA, Fábio Moreira. **O uso de PBL para o ensino de algoritmos e programação de computadores**. São Paulo, PBL 2010 Congresso Internacional. 2010.

AREIAS, Cristiana; MENDES, António. **ProGuide: A dialogue-based tool to support initial programming learning**. Portugal. 3rd E-Learning Conference. 2006.

BATISTA, Esteic, Esteic Janaina S. et al. **Uso do Scratch no ensino de programação em Ponta Porã: das séries iniciais ao ensino superior**. V Congresso Brasileiro de Informática na Educação. 2016.

BERBEL, Neusi Aparecida Navas. **As metodologias ativas e a promoção da autonomia de estudantes**. Londrina, Semina: Ciências Sociais e Humanas, v. 32, n. 1, p. 25-40, jan./jun. 2011.

BERSSANETTE, João Henrique; FRANCISCO, Antonio Carlos de. **Uma proposta de ensino de programação de computadores com base na aprendizagem colaborativa utilizando redes sociais**. Congresso Internacional e Educação e Tecnologias. 2018.

BILABILA, Augusto Manuel. **CompAlg - Ferramenta de ensino e aprendizagem da lógica de programação**. Portugal, Faculdade de Ciências da Universidade do Porto em Ciência de Computadores. 2017.

CAVALCANTE, Ahemenson Fernandes; COSTA, Leonardo dos Santos; ARAÚJO, Ana Liz Souto Oliveira de. **Um estudo de caso sobre competências do pensamento computacional estimuladas na programação em blocos no code.org**. V Congresso Brasileiro de Informática na Educação. 2016.

COSTA, Eli Banks Liberato de. **A história da ciência e o ensino da recursividade: as torres de Hanói**. Workshop de História da Ciência e Ensino. Campus Marquês de Paranaguá, PUCSP. 2008.

CHICON, Patrícia Mariotto M; QUARESMA, Cindia Rosa T.; GARCÊS, Solange Beatriz B. **Aplicação do método de ensino peer instruction para o ensi. no de lógica de programação com acadêmicos do curso de ciência da computação**. Passo Fundo. 5º Seminário Nacional de Inclusão Digital. 2018.

DIESEL, Aline; BALDEZ, Alda Leila S.; MARTINS, Silvana Neumann. **Os princípios das metodologias ativas de ensino: uma abordagem teórica**. Rio Grande do Sul, v. 14, n. 1, p. 268-288. 2017.

EFOPOULOS, Vassilios; DAGDILELIS, Vassilios; EVANGELIDIS, Georgios; SATRATZEMI, Maya. **WIPE: A programming environment for novices**. SIGCSE Bulletin, 37 (3), 113-117. 2005.

FRANÇA, Rozelma Soares de; AMARAL, Haroldo José Costa do. **Proposta metodológica de ensino e avaliação para o desenvolvimento do pensamento**

computacional com o uso do Scratch. Pernambuco. XIX Workshop de Informática na Escola (WIE 2013). P. 179-188. 2013.

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de Programação: a construção de algoritmos e estrutura de dados**. São Paulo: Prentice Hall, 2005.

GOMES, Anabela de Jesus. **Dificuldades de aprendizagem de programação de computadores: contributos para a sua compreensão e resolução**. Coimbra, Universidade de Coimbra. 2010.

GOMES, Anabela; MENDES, António José. **SICAS: Interactive system for algorithm development and simulation**. Coimbra. 2001.

GOMES, Anabela; AREIAS, Cristiana; HENRIQUES, Joana; MENDES, José Mendes. **Aprendizagem de programação de computadores: dificuldades e ferramentas de suporte**. Revista portuguesa de pedagogia, Coimbra, ano 42-2, p. 161-179, 2008.

GOMES, Anabela; HENRIQUES, Joana; MENDES, José Antônio. **Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores**. In Educação, Formação & Tecnologias; vol.1(1). p. 93-103. 2008. Disponível em http://eft.educom.pt. Acesso em: 10 de ago. 2018.

GOMES, Tancicleide C. S.; MELO, Jeane C. B. de. **App Inventor for Android: Uma nova possibilidade para o ensino de lógica de programação**. II Congresso Brasileiro de Informática na Educação. 2013.

GUIMARÃES, Leila Jane B. L. S; ARRUDA, Ana Paula Diniz; MARTINS, Agnaldo Lopes. **O uso do software livre app inventor no processo de ensino e aprendizagem da lógica no curso de graduação em sistemas de informação**. 2016.

JENKINS, Tony. **On the Difficulty of Learning to Program. In Proceedings of 3rd Annual LTSN_ICS Conference**. The Higher Education Academy (p. 53–58), 2002.

JUNIOR, José Augusto Teixeira de Lima; VIEIRA, Carlos Eduardo Costa; VIEIRA, Priscila de Paula. **Dificuldades no processo de aprendizagem de Algoritmos: uma análise dos resultados na disciplina de AL1 do Curso de Sistemas de Informação da FAETERJ - Campus Paracambi**. Cadernos UniFOA, Volta Redonda, n. 27, p. 5-15, abr. 2015.

JUNQUEIRA, Sirly Henrique Ferreira. **Ensino de lógica de programação com o uso de ferramenta visual voltada para dispositivos móveis com android**. XXXVI Encontro Nacional de Engenharia de Produção. 2016.

LEITE, Vanessa Matias; SENEFONTE, Helen C. M; Barbosa Cinthyan R.S.C.; SEABRA, Rodrigo Duarte. **VisualAlg: Estudo de caso e análise de compilador destinado ao ensino de programação**. Nuevas Ideas en Informática Educatica TISE. 2013.

LOPES, Maísa Soares dos Santos et al. **Web environment for programming and control of a mobile robot in a remote laboratory**. IEEE Transactions on learning techonologies, vol. 10, n. 4, out. 2017.

LOPES, Maísa Soares dos Santos. **Ambiente colaborativo para ensino aprendizagem de programação integrando laboratório remoto de robótica**. Salvador, Universidade Federal da Bahia. 2017.

MASSAM, Eduardo Garcia. **Análise ergonômica das ferramentas portugol ide, portugol viana e visual**. Bandeirantes. Universidade Estadual do Norte do Paraná. 2012.

MÉLO, Francisco Édson Nogueira de; CUNHA, Raimundo Ricardo Matos da; SCOLARO, Dyonad Renan; CAMPOS, Jhonatan Luiz. **Do Scratch ao arduino: Uma proposta para o ensino introdutório de programação para cursos superiores de tecnologia**. Santa Catarina. XXXIX Congresso Brasileiro de Educação em Engenharia. 2011.

MCCRACKEN, Michael et al. **A multi-national, multi-institutional study of assessment of programming skills of first-year CS students**. P. 125-140. 2001.

NAGAI, Walter Aoiama; IZEKI, Claudia Akemi. **Relato de experiência com metodologia ativa de aprendizagem em uma disciplina de programação básica com ingressantes dos cursos de Engenharia da Computação, Engenharia de Controle e Automação e Engenharia Elétrica**. RETEC. 2013.

NUUTILA, Esko; TÖRMÄ, Seppo; e MALMI, Lauri. **PBL and Computer Programming – The Seven Steps Method with Adaptations.** Helsinki University of Technology, Filand. Computer Science Education, 15(2):123–142, June 2005.

OLIVEIRA, Eduardo Gomes de. **Uma proposta para apoiar o ensino e aprendizagem de linguagens de programação**. Minas Gerais. 2017.

PADIR, Taskin; CHERNOVA, Sonia. **Guest editorial special issue on robotics education**. IEEE Transaction on education, vol. 56. 2013.

PEREIRA, Pryscilla de Sousa; MEDEIROS, Marcos. MENEZES, José Wally Mendonça. **Análise do Scratch como ferramenta de auxílio ao ensino de programação de computadores**. Pará. XL Congresso Brasileiro de Educação em Engenharia. 2012.

POLYA, George. **A arte de resolver problemas: um novo aspecto do método matemático.** 2 reimpr. Rio de Janeiro. 1995.

POTKONJAK, Veljko et al. Virtual laboratories for education in Science, technology, and engineering: A review. **Computers & Education**. P. 309-327. 2016.

PPC. **Projeto Pedagógico do Curso de Graduação de Ciência da Computação.** Universidade Estadual do Sudoeste da Bahia. Vitória da Conquista. 2011.

RESNICK, Mitchel. **Lifelong Kindergarten: Cultivating creativity through projects, passion, peers, and play**. MIT Media Lab. 2017. cap 1.

SANTOS, Antunes; GORGÔNIO, Arthur; LUCENA, Amarildo; GORGÔNIO, Flavius. **A importância do fator motivacional no processo ensino-aprendizagem de**

algoritmos e lógica de programação para alunos repetentes. Rio Grande do Norte - Universidade Federal do Rido Grande do Norte. 2015.

SETZER, Valdemar W.; CARVALHEIRO, Fábio H. **Algoritmo e sua análise – uma introdução didática.** São Paulo. Depto. De Ciência da Computação, Instituto de Matemática e Estatística da USP. Caderno da Revista do Professor de Matemática, Vol. 4, No. 1, 1993, pgs. 1-26. Disponível em: < https://www.ime.usp.br/~vwsetzer/alg/algoritmos.html>. Acesso em: 01 mar. 2019.

SILVA, Spartacus Sousa da. **Aplicação da ferramenta Google Classroom para melhoria de desempenho pedagógico na disciplina de informática aplicada à contabilidade.** Paraíba – Universidade Federal da Paraíba. 2016.

TRENTIN, Marco Antônio S.; PÉREZ, Carlos A. S.; SANTOS, Antônio Venicius dos. **A Utilização de Laboratórios Virtuais na Melhoria do Processo de Ensino-Aprendizagem**. Porto Alegre, RS, jan. 2002.

VIENSCI, Rafael Gustavo Fabris. **Campo minado inteligente**. Trabalho de conclusão de curso. Engenharia da Computação - Universidade Positivo. Curitiba. 2009.

WOLBER, David; ABELSON, Hal; SPERTUS, Ellen; LOONEY, Liz. **App Inventor 2: Create your own android apps**. 2. Ed. 2014.

ZANATTA, Andrei Cardozo. **Programação de computadores para crianças: Metodologia do code club Brasil**. Araranguá. Universidade Federal de Santa Catarina. 2015.

APÊNDICE A - Questionário aplicado aos professores que ministraram a disciplina Algoritmos e Programação I na Universidade Estadual do Sudoeste da Bahia - UESB.

1) Como foi sua experiência ao ministrar a disciplina?

2) Quais as dificuldades que você observou nos alunos ao aprenderem programação?

3) Como foi o início da disciplina? (Começou com uma linguagem de programação imperativa, ou passou antes por outro processo como por exemplo o portugol, visual g?)

4) Você já usou programação em blocos? Se sim, qual foram as ferramentas e como foi a experiência?

5) Você já fez uso de uma programação mais lúdica, como o uso de arduíno? Se sim qual foi a experiência?

6) O que você acha que poderia contribuir para um melhor aprendizado dos alunos na disciplina?

7) Quais as dificuldades que você teve ao ministrar a disciplina?

8) Você já deu aula para alunos repetentes da disciplina? Se sim, quais foram as principais dificuldades que você enxergou neles?

9) O que você acha de usar a programação em blocos antes de mostrar uma linguagem de programação textual para eles?

10) Os alunos sentem mais dificuldades na construção sintática ou na lógica de programação (compreensão e solução do problema)?

APÊNDICE B - Questionário aplicado aos alunos que cursaram a disciplina Algoritmos e Programação I na Universidade Estadual do Sudoeste da Bahia - UESB.

1) Como foi sua experiência na disciplina?

2) Você acha que as atividades práticas contribuíram para o seu aprendizado?

3) Você teve dificuldades na disciplina? Se sim, quais foram?

4) Você acha que deveria ter mais aulas práticas?

5) O que você achou da metodologia aplicada?

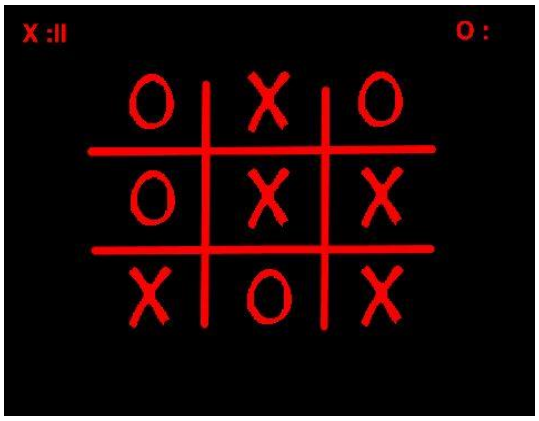
6) O que você acha que poderia melhorar na metodologia da disciplina?

7) Você conhece programação em blocos? Se sim, você acredita que introduzir a disciplina com programação em blocos o ajudaria a enxergar melhor a lógica do problema antes de passar para uma linguagem de programação textual?

8) Você sentiu mais dificuldade nas construções sintáticas ou na lógica?

ANEXO A – Projetos produzidos em programação em blocos com o Scratch na disciplina Algoritmos e Programação I na Universidade Estadual do Sudoeste da Bahia - UESB.

Figura 46 – Segundo projeto de jogo da primeira unidade (Jogo da velha)

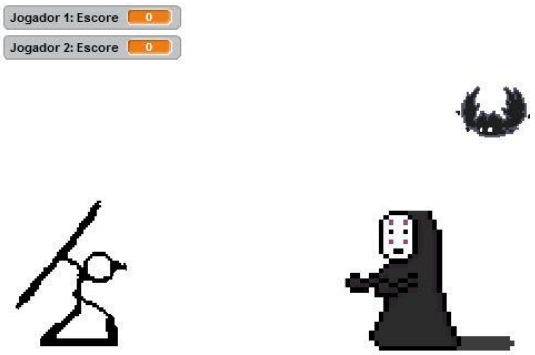


a) Tela do jogo em execução

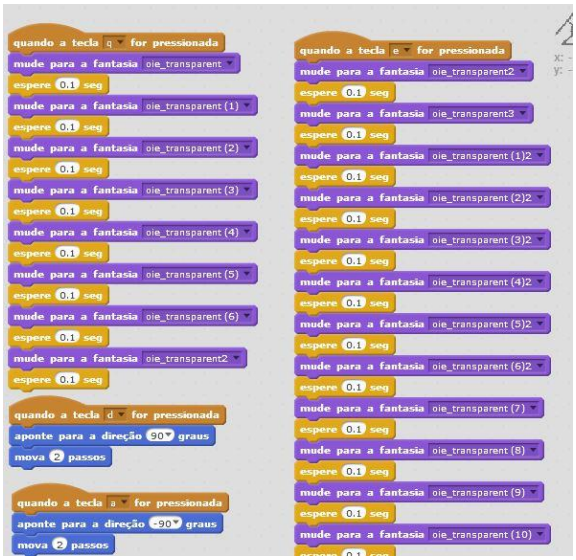
b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Figura 47 – Terceiro projeto de jogo da primeira unidade (Jogo de luta)



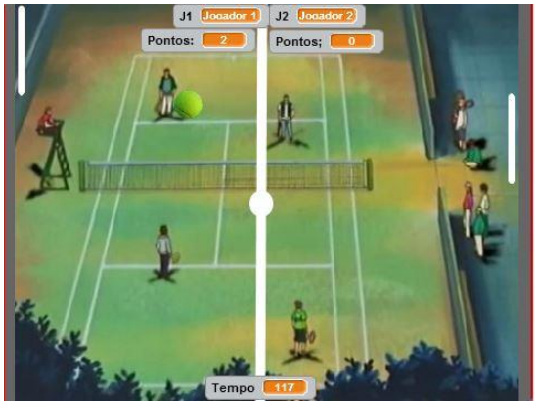
a) Tela do jogo em execução



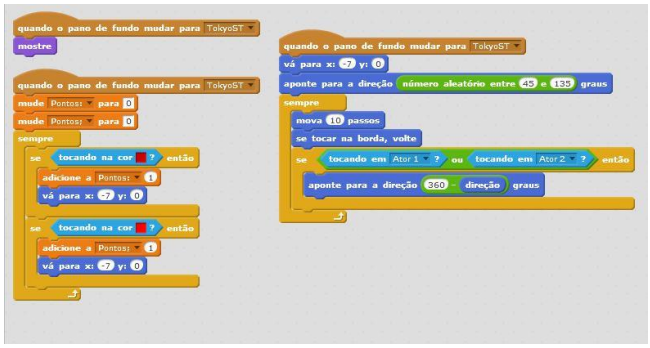
b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Figura 48 – Quarto projeto de jogo da primeira unidade (Jogo de tênis)



a) Tela do jogo em execução

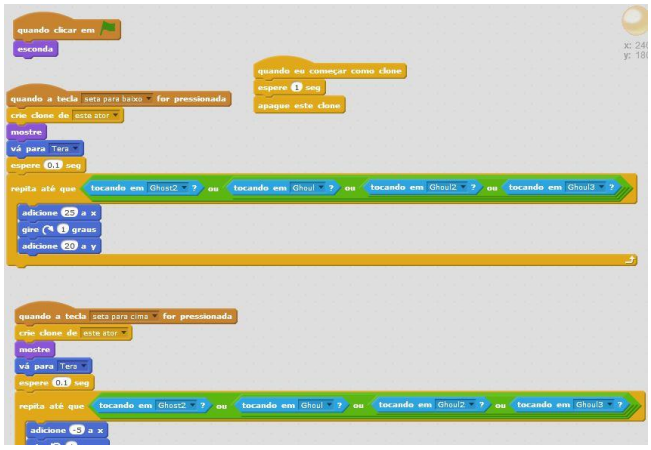


b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Figura 49 – Quinto projeto de jogo da primeira unidade (Jogo de aventura)

a) Tela do jogo em execução

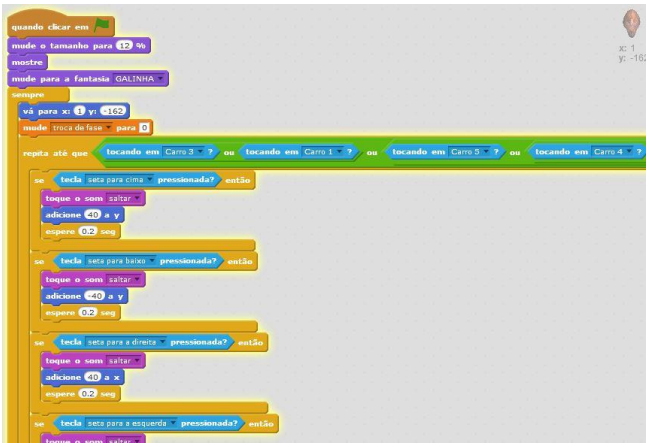


b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Figura 50 – Sexto projeto de jogo da primeira unidade (Jogo de aventura)

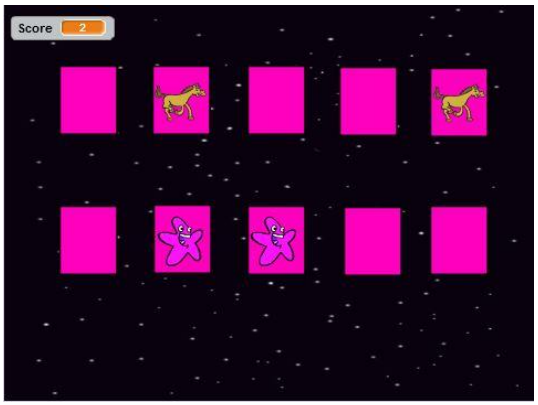
a) Tela do jogo em execução



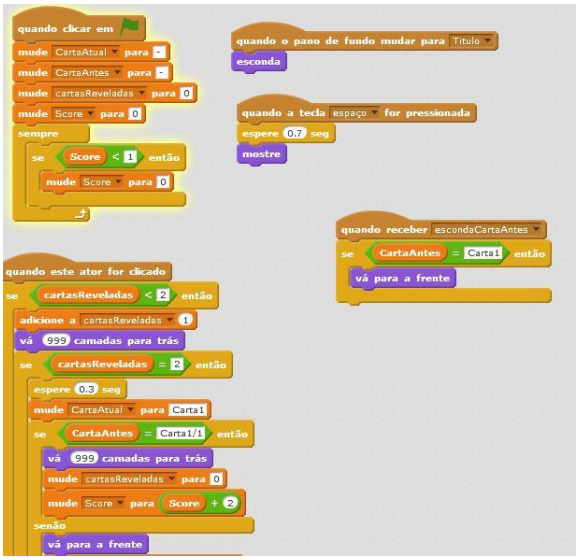
b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)

Figura 51 – Sétimo projeto de jogo da primeira unidade (Jogo da memória)



a) Tela do jogo em execução



b) Parte do código em blocos do jogo

Fonte: Discente da disciplina (2018)