UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
***Campus* Cornélio Procópio**
PROGRAMA DE PÓS-GRADUAÇÃO EM ENSINO
MESTRADO PROFISSIONAL EM ENSINO

PAULO ROBERTO ANASTACIO

O USO DO *SCRATCH* NO ENSINO DE PROGRAMAÇÃO

CORNÉLIO PROCÓPIO – PR
2020

PAULO ROBERTO ANASTACIO

O USO DO *SCRATCH* NO ENSINO DE PROGRAMAÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Ensino da Universidade Estadual do Norte do Paraná – *Campus* Cornélio Procópio, como requisito parcial à obtenção do título de Mestre em Ensino.

Orientador: Prof. Dr. Rudolph dos Santos Gomes Pereira

Coorientadora: Prof.ª Dr.ª Daniela de Freitas Guilhermino Trindade

CORNÉLIO PROCÓPIO – PR
2020

Ficha catalográfica elaborada pelo autor, através do
Programa de Geração Automática do Sistema de Bibliotecas da UENP

A
A534u

Anastacio, Paulo Roberto
O USO DO SCRATCH NO ENSINO DE PROGRAMAÇÃO / Paulo Roberto Anastacio; orientador Rudolph dos Santos Gomes Pereira; co-orientadora Daniela de Freitas Guilhermino Trindade - Cornélio Procópio, 2020.
85 p. :il.

Dissertação (Mestrado em Dissertação (Mestrado Profissional em Ensino) - Universidade Estadual do Norte do Paraná, Centro de Ciências Humanas e da Educação, Programa de Pós-Graduação em Ensino, 2020.

1. Programação. 2. Scratch. 3. Pensamento Computacional. I. dos Santos Gomes Pereira, Rudolph , orient. II. de Freitas Guilhermino Trindade, Daniela, co-orient. III. Título.

PAULO ROBERTO ANASTACIO

O USO DO *SCRATCH* NO ENSINO DE PROGRAMAÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Ensino da Universidade Estadual do Norte do Paraná – *Campus* Cornélio Procópio, como requisito parcial à obtenção do título de Mestre em Ensino.

Após realização de Defesa Pública o trabalho foi considerado:

_____

BANCA EXAMINADORA

_____
Orientador: Prof. Dr. Rudolph dos Santos Gomes Pereira
Universidade Estadual do Norte do Paraná – UENP

_____
Coorientadora: Prof.ª Dr.ª Daniela de Freitas Guilhermino Trindade
Universidade Estadual do Norte do Paraná – UENP

_____
Prof. Dr. José Augusto Fabri
Universidade Tecnológica Federal do Paraná – UTFPR

_____
Prof. Dr. João Coelho Neto
Universidade Estadual do Norte do Paraná – UENP

Cornélio Procópio, 13 de Maio de 2020.

ANASTACIO, Paulo Roberto. **O uso do *Scratch* no ensino de programação**. 2020. 86 f. Dissertação (Mestrado Profissional em Ensino) – Universidade Estadual do Norte do Paraná, Cornélio Procópio, 2020.

RESUMO

Junto às inovações tecnológicas estão presente as ferramentas educacionais que ofertam possibilidades de auxiliar, facilitar e aprimorar o ensino de determinados conteúdos. Pesquisas na área de Computação apontam que os alunos apresentam grandes dificuldades em disciplinas, como no caso da programação, a qual muitas vezes é responsável pelo alto índice de reprovação e evasão dos cursos. Pesquisadores dessa área afirmam que o uso de ferramentas educacionais como o *Scratch* tem o objetivo de motivar o aprendizado de conceitos de programação por meio de uma experiência agradável, assim, ao programar por meio da ferramenta, o aluno é levado a pensar computacionalmente, utilizando as etapas do Pensamento Computacional, como a Decomposição, o Reconhecimento de Padrões, a Abstração e o Algoritmo. O uso do Pensamento Computacional auxilia o aluno não somente na disciplina e no curso de graduação, mas também no seu cotidiano fora do âmbito educacional, onde estão sujeitos a tomada de decisões e a solução de problemas aos quais necessitam do Pensamento Computacional. No desenvolvimento da pesquisa, buscou-se elaborar um caderno de atividades para o uso da ferramenta *Scratch* no desenvolvimento do Pensamento Computacional como modo de auxiliar no ensino de programação. Foi realizado um curso introdutório para o ensino de programação, com alunos do primeiro ano do curso de Ciência da Computação de uma Universidade pública do Norte do Paraná, no qual foram abordados os conceitos de lógica de programação, estruturas de decisão e de repetição, visando minimizar as dificuldades encontradas na disciplina de programação como forma de verificar a adequação do caderno de atividades ao contexto atual. Os resultados apresentados pelos alunos no decorrer do curso foram analisados qualitativamente e indicaram a adequação do caderno de atividades no desenvolvimento do Pensamento Computacional a partir do uso da ferramenta *Scratch* no ensino de programação.

Palavras-chave: Programação. *Scratch*. raciocínio lógico. Estrutura de decisão. Estrutura de repetição. Pensamento Computacional

ANASTACIO, Paulo Roberto. **The use of Scratch in programming teaching**. 2020. 87 f. Dissertação (Mestrado Profissional em Ensino) – Universidade Estadual do Norte do Paraná, Cornélio Procópio, 2020.

ABSTRACT

Along with technological innovations, educational tools are present that offer possibilities to assist, facilitate and improve the teaching of certain contents. Research in the Computing Area shows that students have great difficulties in subjects, as in the case of programming, which is often responsible for the high failure and dropout rate and dropout rate. Researchers in this area claim that the use of educational tools such as Scratch has the objective of motivating the learning of programming concepts through a pleasant experience, thus, when programming through the tool, the student is led to think computationally, using the stages of Computational Thinking, such as Decomposition, Pattern Recognition, Abstraction and Algorithm. The use of Computational Thinking assists the student not only in the discipline and in the undergraduate course, but also in their daily life outside the educational scope, where they are subject to decision making and the solution of problems to which they need Computational Thinking. In the development of the research, it was sought to elaborate a notebook of activities for the use of the Scratch tool in the development of Computational Thinking as a way to assist in the teaching of programming. An introductory course for the teaching of programming was held, with twenty hours, for students of the first year of the Computer Science course, at the State University of Northern Paraná, at the Luiz Meneghel-Bandeirantes campus, in which the concepts of logic were addressed programming, decision and repetition structures, aiming to minimize the difficulties found in the programming discipline as a way of verifying the adequacy of the activities notebook to the current context. The results presented by the students during the course were analyzed qualitatively and indicated the adequacy of the notebook of activities in the development of Computational Thinking from the use of the Scratch tool in teaching programming.

Keywords: Programming. Scratch. logical reasoning. Decision structure. Repetition structure. Computational Thinking

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

EAD	Ensino a Distância
PCNs	Parâmetros Curriculares Nacionais
PC	Pensamento Computacional
TDIC	Tecnologias Digitais de Informação e Comunicação
TI	Tecnologia da Informação

SUMÁRIO

1. INTRODUÇÃO

As inovações tecnológicas se fazem presente no cotidiano da população, passando por atualizações e transformações a todo momento, ao ter como base a observação das inovações tecnológicas presentes no âmbito escolar, por intermédio de games, *softwares*, ambientes virtuais de aprendizagens dentre outros, nota-se que essas tecnologias oferecem possibilidades de auxiliar, facilitar e aprimorar o ensino e a aprendizagem (LIMA, 2019).

Barbosa, Baisso e Almeida (2018) alegam que estamos entrando na quarta revolução industrial na qual os recursos tecnológicos estão mudando o mundo por meio da Inteligência Artificial, de algoritimo mais eficazes de teste genéticos dentre outras inovações tecnológicas. Deste modo, essas inovações possibilitam uma maneira de planejar a vida, executar trabalhos, se relacionar com pessoas, adquirir informações, cuidar da saude entre outros.

As tecnologias estão presentes em todas as áreas de conhecimento, podendo potencializar o ensino e a aprendizagem de conhecimentos. A área das Tecnologias Digitais de Informação e Comunicação (TDIC) por exemplo, passa por várias inovações, onde softwares e hardwares são desenvolvidos e inovados a todo momento (LIMA, 2019).

As TDIC vem desempenhando um papel fundamental na organização e inovação de diversas áreas de conhecimento, visto que o ambiente de trabalho possui dependência dos recursos proporcionados pela área da computaçao pois o mercado de trabalho exige de seus funcionários habilidades como a tomada de decisão e a resolução de problemas que são apontadas como fator comum e necessário nessa área (ALBUQUERQUE; PONTES, 2016).

As habilidades da área das TIDC e suas inovações tecnológicas são requisitadas e utilizadas constantemente pela sociedade que, mesmo em tempos de crise econômica, necessita dos profissionais destas áreas no mercado de trabalho (LIMA, 2019). Essas inovações oferecem a sociedade ferramentas mais aprimoradas na busca pela melhoria da qualidade de vida das pessoas (SILVA, 2018).

Nesse sentido, Brackmann (2017) alega que a população está imersa em meio às tecnologias e necessita delas para efetuar tarefas do cotidiano, sejam elas básicas ou complexas, podendo ser utilizadas em diversas áreas.

Na área educacional, por exemplo, as inovações tecnológicas se fazem presentes por intermédio de *softwares* educativos, games, ensino a distância (EAD) e dispositivos tecnológicos, no entanto, a utilização de tais tecnologias nesse ambiente necessita ser útil para a vida do aluno, possibilitando que ele associe o conhecimento adquirido em sala de aula ao seu cotidiano, utilizando as ferramentas educacionais como auxiliadores do ensino e da aprendizagem (DIESEL; BALDEZ; MARTINS, 2017).

Ao introduzir as tecnologias em sala de aula, se faz necessário direcionar o envolvimento do aluno no processo de ensino, propiciando a aprendizagem de conteúdo, possibilitando-o a solucionar problemas os quais podem ser encontrados dentro e fora da sala de aula (LOPES; RIBEIRO, 2018).

A capacidade de solucionar problemas é uma habilidade matemática que envolve algumas habilidades como raciocínio lógico, levantamento de hipóteses, planejamento de ações e formulação de problemas, as quais tem papel fundamental na formação integral do indivíduo, preparando-o para viver em sociedade e tomar decisões (PONTE *et al*., 2007).

Dentre essas habilidades, o raciocínio lógico é fundamental no processo de ensino de programação e se faz presente em qualquer área do conhecimento, sendo desenvolvido quando o indivíduo se depara com situações problemas e necessita organizar o pensamento detalhadamente, buscando uma solução para revolvê-los (TRETIN *et al.*, 2019)

Neste aspecto, as dificuldades comumente apontadas pelos alunos como por exemplo, a dificuldade em utilizar raciocínios lógicos, estão relacionadas aos conceitos fundamentais da Computação, os quais têm o papel de promover o desenvolvimento de habilidades e competências fundamentais para auxiliar na tomada de decisão e resolução de problemas (BRACKMANN, 2017).

Segundo os dados da *Programme for International Student Assessment* (OECD, 2018), grande parte dos alunos da Educação Básica brasileira apresenta dificuldades na resolução de problemas nas diversas áreas do conhecimento, com destaque para o raciocínio lógico, a contagem, a capacidade de abstrair informações e o desenvolvimento do pensamento crítico.

A Pró-Reitoria de Graduação da Universidade Federal do Pampa, após uma pesquisa de cinco anos, afirmou que os cursos de Ciência da Computação e Engenharia da Computação apresentaram uma taxa de 66% desistência por parte

dos alunos, afirmando ainda que um dos motivos para esse alto índice de evasão, está relacionado a dificuldades encontradas nos conteúdos da disciplinas de Programação e Algoritmos, os quais exigem do aluno o desenvolvimento do raciocínio logico e abstrato, direcionados a resolução de problemas (PROGRAD, 2018).

Além das dificuldades apresentadas no ensino básico, Dias e Serrão (2014) apontam que vários países, inclusive o Brasil, apresentam altos índices de reprovação e evasão nos cursos de graduação da Área da Computação devido as dificuldades encontradas na disciplina de programação, que exige habilidades vinculadas a tomada de decisão e raciocíonio lógico.

Bittencourt *et al.* (2013) ressaltam que as dificuldades econtradas pelos alunos na disciplina de programação, se dá pela necessidade de aprender diversos conceitos em pouco tempo, desenvolver raciocinio lógico e capacidades específicas da programação como o Pensamento Computacional (PC).

Objetivando auxiliar as dificuldades encontradas nesses cursos entende-se como adequada a utilização de ferramentas que evidenciem o processo de solucionar problemas e estimulem o Pensamento Computacional por meio da programação (MESTRE *et al*., 2015). Ao compreender o ensino sob esta perspectiva, surge a preocupação em relação as estratégias adotadas a fim de auxiliar o ensino de programação, almejando minimizar as dificuldades encontradas pelos alunos ao se depararem com a disciplina de programação.

O Pensamento Computacional, é apontado por Mestre *et al*. (2015) como o domínio de habilidades diretamente relacionadas à resolução de problemas, envolvendo a capacidade de ler e interpretar textos além de compreender as situações contidas nos problemas e transcreve-las em modelos matemáticos.

Assim, para promover o Pensamento Computacional em sala de aula, é necessário incorporar estratégias no processo de ensino no qual o aluno participa da construção do processo de forma flexível, interligada e híbrida, podendo desfrutar dos benefícios de se utilizar as ferramentas educacionais como o *Scratch* (BRACKMANN, 2017).

Amaral, Silva e Pantaleão (2015) relatam em suas pesquisas que dentre os centros de estudos como a Universidade Federal de Pernambuco e Universidade de Tecnologia da Informação de Copenhagen estão utilizando ferramentas a fim de facilitar o ensino de programação, promovendo o Pensamento Computacional. Dentre as ferramentas utilizadas nas universidades está o *Scratch*, a

qual tem o objetivo auxiliar o ensino de programação e estimular o Pensamento Computacional (AMARAL; SILVA; PANTALEÃO, 2015).

Uma das vantagens apontadas por Gomes e Melo (2013) em relação a essa ferramenta é que ela não exige dos estudantes a sintaxe de uma linguagem de programação textual, uma vez que a ferramenta utiliza uma linguagem de programação visual por blocos lógicos, facilitando o ensino de programação.

Para que o aluno desenvolva e solucione um projeto utilizando o *Scratch*, é necessário que o mesmo conclua as quatro etapas do Pensamento Computacional: a decomposição, os padrões, a abstração e o algoritmo (RAABE *et al*., 2018). Tais etapas são responsáveis por garantir que o aluno seja capaz de dividir o problema em passos menores e concretizá-los por meio da utilização dos blocos de programação contidos na ferramenta (TRETIN *et al*., 2019).

Corroborando com os apontamentos feitos pelo autor supracitado, Gomes e Melo (2013) e Ramos (2014) utilizaram em seus trabalhos ferramentas como o *Scratch* e o MIT App Inventor para auxiliar o ensino de programação e promover o desenvolvimento do Pensamento Computacional.

Deste modo, pode-se perceber que a utilização do *Scratch* no âmbito educacional possibilita a troca de experiências sobre o uso da programação, estimulando o aluno a utilizar a lógica matemática, estruturas de decisão, repetição e desenvolver o Pensamento Computacional (TRETIN *et al*., 2019).

Pensando nesse contexto, o presente estudo aponta as TDIC como um recurso pedagógico, a qual irá contribuir e potencializar o ensino de programação. Logo, esse trabalho apresenta o uso da ferramenta *Scratch* como um recurso tecnológico para o ensino de programação.

A partir das contextualizações apontadas pelos pesquisadores em relação ao ensino de programação, emergiu o seguinte questionamento para iniciar o percurso investigativo: De que forma a ferramenta *Scratch* contribui para o ensino de programação e o desenvolvimento do Pensamento Computacional?

Nesse sentido, o objetivo geral dessa pesquisa consistiu em elaborar um caderno de atividades para o uso da ferramenta *Scratch* no desenvolvimento do Pensamento Computacional como modo de auxiliar no ensino de programação, que foi utilizado em um curso com alunos do primeiro ano do curso de Ciências da Computação de uma universidade pública estadual do norte do Paraná.

Apresenta, ainda, como objetivos específicos:

- Identificar os conhecimentos prévios dos alunos de Ciências da Computação sobre a ferramenta *Scratch*, Lógica de Programação, Estruturas de Decisão, Repetição e Pensamento Computacional;
- Promover o desenvolvimento de um curso relacionado à Lógica de Programação, Estruturas de Decisão e Estruturas de Repetição e o Pensamento Computacional;
- Analisar os resolução dos exercícios efetuados pelos alunos, levando em consideração os pilares do Pensamento Computacional;
- Apresentar a percepção dos alunos quanto ao curso e sua contribuição para o ensino de programação.

Posto isso, a dissertação foi organizada em 5 (cinco) capítulos: o primeiro, trata-se da introdução, assim contextualizando a temática; o segundo apresenta o aporte teórico; o terceiro é dedicado ao percurso investigativo com as seguintes seções: Ensino de Programação; *Scratch* e Pensamento Computacional; o quarto tece as análises dos dados coletados. E por fim, no quinto e último capítulo, as considerações finais, limitações e trabalhos futuros sobre a pesquisa.

2. APORTE TEÓRICO

Nesse capítulo, apresenta-se o aporte teórico para a discussão e compreensão da temática abordada a partir dos seguintes temas: Ensino de Programação, *Scratch* e Pensamento Computacional.

2.1 Ensino de Programação

Valente (2016) afirma que programar é um trabalho que objetiva fornecer instruções ao computador a fim de solucionar um conjunto de problemas tradicionais. Aprender uma linguagem de programação é uma atividade desafiadora, contudo, tornar a programação disponível para um maior número de indivíduos é algo significativo, já que pode estimular várias capacidades cognitivas como pensar logicamente e solucionar problemas (SCAICO *et al*., 2012).

Ao dominar a programação, é possivel aplicar as técnicas de modo a solucionar diversos tipos de problemas, nas mais diversas áreas por meio de *softwares* e aplicativos (SCAICO *et al*., 2012).

De acordo com Forbellone e Ebrspacher (2005) o raciocínio é a forma mais complexa do pensamento, sendo que a lógica estuda a “correção do raciocínio”, organizando o pensamento. Assim, segundo esses autores apontam a importância da lógica no dia a dia das pessoas, pois essa prática está presente no pensar, falar, escrever e agir corretamente, ordenando os pensamentos a partir do uso da lógica.

Para que ocorra a aprendizagem da programação é fundamental que o aluno tenha a capacidade de abstrair, para que assim consiga compreender, propor e solucionar problemas por meio de raciocínios lógicos (ZANETTI; OLIVEIRA, 2015). Os autores ainda propõem a utilização de ferramentas educacionais que estimulem os alunos por meio da ludicidade e usabilidade, possibilitando o encontro de subsídios para a o ensino e a aprendizagem de programação.

Zaharija, Mladenovic e Boljat (2013) defendem que o ensino de programação poderia estar incluso nas disciplinas do Ensino Fundamental e Médio, possibilitando integrar às metodologias curriculares, ferramentas auxiliadoras como *Scratch*, que objetivam facilitar a aprendizagem de programação.

Diante da demanda de profissionais relacionados as áreas da

computação, a falta de profissionais capacitados e as dificuldades encontradas no ensino de programação, surge a necessidade de motivar crianças e adolecentes a desenvolverem novas formas de pesamento como a Lógica de Programação (VENTURA, 2018). Almejando que em um futuro próximo estes alunos não sejam apenas usuários das tecnologias, mas também desenvolvedores dessas, capazes de solucionar problemas e pensar computacionalmente (VENTURA, 2018).

Em consonância a esse cenário, o ensino de programação se mostra pertinente e poderia estar presente em disciplinas do ensino básico como Biologia, Matemática, dentre outros, pois o ensino de programação possibilita desenvolver o pensamento computacional e lógico dos alunos, facilitando encontrar soluções para problemas por meio da tecnologia e suas ferramentas (KAFAI; BURKE, 2013).

O aprendizado da programação, possibilita ao aluno o desenvolvimento de habilidades, abstração, reflexão e planejamento, garantindo a este a possibilidades de compreender e solucionar problemas dentro e fora das salas de aulas (VENTURA, 2018).

Blikstein (2008) ressalta que o ensino de programação não é uma prática exclusiva dos cursos da área da Computação. De acordo com as exigências da atualidade, os profissionais de diversas áreas como por exemplo economia e ciências, devem saber utilizar ferramentas a fim de criar modelos computacionais que facilitem, aprimorem e aperfeiçoem trabalhos e atividades do cotidiano (BLIKSTEIN, 2008).

Ao analisar os cursos da área da Computação, foi possível notar que os alunos frequentemente apresentam dificuldades nas disciplinas de programação (LONG, 2007). Rocha *et al.* (2010) e Pereira *et al.* (2012) revelam em suas pesquisas que a disciplina de lógica de programação é responsável pela reprovoção ou desistências de 75% dos alunos da Área da Computação.

Dentre as matérias ofertadas pelos cursos das áreas das tecnologias de informação, uma que se destaca é a de Lógica de Programação, que é apontanda pelos alunos e pesquisadores como responsável pelo alto número de evasão de discentes, alegando dificuldades na compreensão e aprendizagem dessa matéria (VENTURA, 2018).

De acordo com Maltempi e Valente (2000), muitos alunos da área da Computação quando iniciam a faculdade apresentam dificuldades nos exercícios de raciocínio lógico e matemático, apontando o sistema educacional como responsável

por este obstáculo, pois o aprendizado ocorre pela memorizaçao de conteúdos e os alunos não desenvolvem aspectos relacionados a autonomia e criatividade.

Fassbinder, De Paulo e Araújo (2012) ressaltam dentre os motivos do elevado número de evasões dos cursos de Computação está a dificuldade na disciplina de programação, pela falta de conhecimentos do aluno em relação aos princípios básicos do curso, dificuldades na lógica mátemática, dificuldades em solucionar problemas, falta de dedicação dos alunos e a falta de interesse na linguagem de programação abordada.

A disciplina de programação é uma das responsáveis pela frequente evasão de alunos, e como forma de minimizar essa evasão se faz presente a utilização de ferramentas tecnológicas como o *Scratch*, que possui vantagens da tecnologia e ludicidade em prol do ensino, possibilitando despertar o interesse dos alunos (ARAÚJO, 2016; INÁCIO, 2016).

Atualmente a aprendizagem da Lógica de programação apresenta valor para a sociedade, pois busca profissionais com pensamentos criativos e que saibam tomar decisões a fim de solucionar problemas (VENTURA, 2018).

Albertin e Albertin (2008) afirmam que dominar a habilidade da programação é tão importante quanto foi aprender a ler no século passado, nessa pespectiva o autor almeja que o usuário não apenas utilize os aplicativos e jogos do seu celular,mas ajude a desenvolve-los.

A sociedade atualmente, passou a utilizar as tecnologias de informação no dia a dia, e para atender a essa demanda as pessoas buscam por melhorias em *softwares* e *hardwares* a fim de agilizar tarefas e aperfeiçoar desempenhos. Deste modo, é essencial incentivar jovens da área da Computação a aprender programação, já que esta é a base para o desenvolvimento de novos *softwares* que busca atender a demanda da população (VENTURA, 2018). E para que profissionais e alunos dominem e utilizem a Programação, se faz necessário o estudo da Lógica de Programação para o desenvolvimento de algoritimos coerentes e válidos (FORBELLONE; EBERSPÄCHER, 2005).

Um algoritimo pode ser estabelecido como uma sequência de passos que visam atingir um objetivo delineado, essa lógica investiga a ordenação do pensamento e do raciocínio lógico podendo ser determinada como a “arte de pensar” e corrigir o pensamento constantemente (FORBELLONE; EBERSPÄCHER, 2005).

Quanto aos os cursos da Computação, Souza (2009) aponta que

esses possuem o hábito de iniciarem o ensino de programação por meio de uma disciplina introdutória, na qual são abordados conceitos fundamentais, tais como: tipos de dados, constantes, variáveis, sintaxes de entrada e saída, operadores aritiméticos e as estruturas sequenciais. Assim, esta disciplina introdutória tem o objetivo de guiar o aluno a dar os passos inicias para que ocorra o ensino e aprendizagem da Lógica de Programação.

Macedo, Petty e Passos (2007) e Macedo (2009) apontam que a utilização de ferramentas tecnológicas educacionais auxiliam na aprendizagem de programação e caracterizam tais ferramentas como instigadoras e complexas, apesar de suas complexidades essas são consideradas eficazes e proporcionam situações de aprendizagens na perspectiva do aluno, instigando-o a pensar de várias maneiras a fim de utilizar estrátégias, concentração, reflexão, coordenação e tomada de decisões mediante as ações do outro jogador.

As formas de pensamento dos seres humanos são responsáveis por alterar suas rotinas assim como o desenvolvimento tecnológico. Deste modo, as tecnologias associadas a educação facilitam a adequação dos alunos diante das inovações tecnológicas, proporcionando conhecimento para lidarem com o desenvolvimento de habilidades fundamentais a fim de solucionar problemas. (MACEDO; PETTY; PASSOS, 2007; MACEDO, 2009).

Vislumbrando uma possibilidade de minimizar essas dificuldades no ensino da lógica de programação, Ventura (2018) aponta para a utilização atividades lúdicas, já que essas além de propiciar o aprendizado de iniciantes na disciplina de programação também despertam interesse pelo assunto (VENTURA, 2018).

Ramos (2014) aponta que o ensino associado a ferramentas educacionais, como o *Scratch*, contribui para que o exercício e o desenvolvimento dos aspectos cognitivos se tornem mais lúdicos e prazerosos.

Figueiredo, Carvalho e Milani (2017) salientam que é importante discutir sobre o uso das tecnologias digitais nas metodologias de aprendizagem, e por isso apontam

> As tecnologias digitais – como tema ou objeto de estudo – têm um papel importante na aprendizagem, pois a escola deve repensar quem são os seus alunos de hoje. Estes estão inseridos em um contexto tecnológico de rápidas mudanças nas formas de produção do conhecimento científico e de acesso a esse conhecimento, e isso se torna um desafio para a escola (p. 247).

Valente (2016) apresentam as facilidades oferecidas pelo *Scratch* para estudantes que estão no início do processo de aprendizagem da programação, afirmando que

> [...] a programação é baseada em uma linguagem de blocos visuais, projetados para facilitar a manipulação da mídia por programadores novatos. O *Scratch* substitui a digitação do código por blocos, sendo que cada um corresponde a uma ação específica que o computador realiza. O bloco pode ser escolhido, arrastado e encaixado em outros blocos para a formação de instruções para o computador. Esses blocos facilitam o processo de descrição das instruções para a máquina uma vez que a sintaxe das instruções é definida pelo encaixe dos blocos, contribuindo para minimizar esse tipo de erro, que é muito comum em linguagem de programação baseada na codificação de comandos (p. 874).

Neste contexto, quando um aluno é estimulado a solucionar um problema, busca por estratégias que permitam utilizar a criatividade a fim de encontrar diferentes maneiras de resolvê-lo, promovendo o raciocínio lógico e o pensamento computacional (COSTA; PESSÔA; GOMES, 2017).

A seguir abordou-se o contexto relacionado a ferramenta educacional Scratch.

2.2 *Scratch*

O *Scratch* é uma ferramenta educacional que utiliza linguagem de programação visual, desenvolvida pelo grupo Lifelong Kindergarten do Instituto de Tecnologia de Massachusetts MIT Media Lab. A ferramenta é um *software* gratuito que foi contruido em 2007 sob o slogan “imaginar, programar, compartilhar”, cujo objetivo é motivar pessoas de qualquer faixa etária a se tornarem produtoras do conhecimento com o auxílio do computador (REZENDE; BISPO, 2018).

A ferramenta apresenta algumas potencialidades como liberdade de criação, comunicação, criatividade e compartilhamento, aprendizagens de conceitos escolares, manipulação de mídias, troca de projetos via internet, permitindo que os usuários compartilhem projetos entre si, reutilizar e adaptar projetos já existentes, e integração de objetos do mundo físico (SCRATCH, 2017).

Dessa forma, ela se mostra promissora no ensino de programação devido a sua ludicidade, podendo ser utilizada para introduzir a lógica de programação

e desenvolver o pensamento computacional nos anos iniciais dos cursos da computação, auxiliando a aprendizagem do aluno (REZENDE; BISPO, 2018).

A linguagem de programação do *Scratch*, como já mencionado, utiliza elementos visuais e de multimídia colaborando positivamente no ensino e na aprendizagem de programação de maneira simples e eficiente por meio da programação de "blocos lógicos" que representam funcionalidades específicas da ferramenta (REZENDE; BISPO, 2018).

Este *software* possibilita que jovens aprendam a pensar de maneira criativa, fazer reflexões sistemáticas, auxiliando o trabalho colaborativo o qual é fundamental no século XXI. Además, foi desenvolvido com o objetivo de auxiliar jovens a programar e compartilhar suas criações com outros membros (BRESSAN; AMARAL, 2015).

O *Scratch* é utilizado em vários países e esta disponível em mais de 40 idiomas, inclusive em português, sendo encontrado para *download* no *site* (http://www.*Scratch*brasil.net.br/index.php/sobre-o-*Scratch*.html) compatível com os sistemas operacionais Mac, Windows e algumas versões do Linux. Sua finalidade é auxiliar ensino e aprendizagem de conceitos matemáticos, desenvolver o pensamento computacional, o pensamento criativo, o raciocínio lógico dentre outros (RIBEIRO; RODRIGUES; PEREIRA, 2014).

Essa ferramenta aborda a programação de forma lúdica, apresentando uma interface gráfica, onde se encontram blocos de códigos separados por cores de acordo com suas funcionalidades, prontos para serem arrastados para o palco onde ocorre a programação (REZENDE; BISPO, 2018). Diferentemente das linguagens de programação – Java, C, entre outras – em que o código é desenvolvido manualmente, caracter por caracter (AMBRÓSIO; COSTA, 2010).

A ferramenta, ainda se demonstra simples, interativa, e proporciona resultados satisfatórios na resolução de algoritimos os quais ocorrem por meio da programação em blocos e utilizam a programação procedural, assim como na linguagem de programação C (AMBRÓSIO; COSTA, 2010).

Assim, com o surgimento das TDIC, foi possível notar uma transformação significativa no processo de ensino dos alunos (SANTOS *et al.,* 2015). As ferramentas educacionais por exemplo, quando utilizados de maneira adequada possibilitam a motivação para o aprendizado, tornando esse processo mais agradável para o aluno (SANTOS *et al*., 2015).

Outra vantagem de se utilizar essas ferramentas no âmbito do ensino é que as atividades são mais excitantes para os alunos, levando-os a superar suas próprias barreiras no processo de aprendizagem (SANTOS *et al*., 2015).

As atividades são desenvolvidas na ferrameta *Scratch* a partir do encaixe dos blocos respeitando a lógica da programação e são divididos em 8 categorias: Movimento, Controle, Aparência, Som, Caneta, Sensores, Operadores e Variáveis (OLIVEIRA *et al.,* 2014).

Para Martins (2012) desenvolver um projeto utilizando a ferramenta *Scratch* exige que o aluno pense computacionalmente para que posteriormente ele seja capaz de dividir o problema em partes menores e concretizá-los, utilizando os blocos de programação contidos na ferramenta.

Deste modo, a ferramenta oferece ao aluno liberdade de escolhas e experimentação de comandos a fim de solucionar o problemas, no qual o aluno pode realizar inúmeras tentativas que poderão dar certo ou errado, e devido a rapidez do *feedback* é rápido é possível refletir sobre o melhor caminho e atingir seu objetivo final (TENÓRIO *et al.,* 2016).

O potencial lúdico da ferramenta *Scratch* se faz notável, pois toda a lógica e estrutura envolvida na ferramenta assemelha-se com as linguagens de programação que exigem alto índice de abstração, pois há estudos que apoiam seu uso em disciplinas de cursos superiores que envolvam a lógica de programação, estruturas de decisão e repetição (OLIVEIRA *et al.,* 2014).

Pereira *et al.* (2012) apontam em sua pesquisa que a utilização dessa ferramenta no início da disciplina de programação, faz com que o aluno obtenha melhor compreensão dos conceitos de programação, tais como lógica de programação, estruturas de decisão e repetição, operadores lógicos, variáveis e o desenvolvimento do pensamento computacional.

Uma das vantagens da sua utilização no processo do ensino de programação é o fato de ser baseada em uma linguagem de programação por blocos, como um quebra cabeça, sendo um *software* educativo que desenvolve a criatividade e a parte lógica do aluno, não exigindo conhecimento prévio de outras linguagens de programação (ARAÚJO, 2016).

O *Scratch*, além de ser uma ferramenta gratuita e rica em recursos lúdicos, apresenta algumas vantagens quando comparada a outros *softwares* como o MIT App Inventor, Game Maker, Alice entre outros com fins educativos, pois

oportuniza e da suporte para o aluno imaginar, programar e compartilhar atividades, fazendo-o não apenas o usuário, mas também o desenvolvedor das TDIC (FIGUEIREDO; CARVALHO; MILANI, 2017).

De acordo com Maloney *et al.* (2010) o *Scratch* tem o objetivo de introduzir noções de linguagem de programação em pessoas que não tem domínio desta, já que é uma ferramenta com *layout* simples*,* trabalhando apenas com uma janela, facilitando a manipulação e execução dos algoritimos e comandos (*SCRATCH*, 2017).

Ainda sobre a forma de programação, a ferramenta *Scratch* é similar a forma procedural de programação utilizada em outras linguagens de programação. Assim, quando o aluno assimila a linguagem utilizada pelo *Scratch*, é possível afirmar que ele apresenta facilidades para dominar uma linguagem de programação convencional e utilizar o Pensamento Computacional para solucionar problemas (TENÓRIO *et al.,* 2016).

A seguir descutiu-se a respeito do pensamento computacional.

2.2 Pensamento Computacional (PC)

O Pensamento Computacional é apontado por Wing (2016) como a capacidade de reformulação e resolução de problemas do mundo real. Resnick (2012) ressalta que estimular o pensamento computacional nas pessoas, propicia compreensão de informações tecnológicas, tornando-se usuários e desenvolvedores das TDIC.

O desenvolvimento do PC não é uma tarefa simples, pois exige a criatividade do aluno para que ele seja capaz de utilizar diferentes estruturas lógicas a fim de solucionar problemas (RESNICK, 2017).

Esse pensamento oferece as pessoas a possibilidade de desenvolverem o pensamento abstrato, pensamento algorítmico, pensamento lógico e pensamento dimensionável (WING, 2016). Assim, quando o PC é desenvolvido nos jovens, promove grande influência no futuro desta pessoa, pois capacita-os no desenvolvimento das TDIC utilizadas na sociedade moderna (VENTURA, 2018).

Blikstein (2008) salienta que o pensamento computacional é uma ferramenta computacional de poder cognitivo e operacional humano, que possibilita

aumentar a produtividade, inventividade e criatividade.

Wing (2016) defende a ideia de que o pensamento computacional é o agrupamento de mecanismos de raciocínio lógicos que podem ser utilizados na resolução de problemas computacionais, sendo utilizado em diferentes aplicações e podendo ultrapassar as fronteiras das áreas da Computação.

Brennan e Renick (2012) definem o pensamento computacional como um conjunto de etapas de resolução, decomposição e definição de problemas, além do pensamento lógico, a abstração, e reconhecimento de padrões.

No caso da BNCC o PC é definido pelo seguinte conceito:

> [...] pensamento computacional envolve as capacidades de compreender, analisar, definir, modelar, resolver, comparar e automatizar problemas e suas soluções, de forma metódica e sistemática, por meio do desenvolvimento de algoritmos (BRASIL, 2018, p. 474).

Para Liukas (2015, p. 110,) o PC é definido como:

> Pensar nos problemas de uma forma que permita aos computadores resolvê-los. O pensamento computacional é algo que as pessoas fazem, não os computadores. Inclui o pensamento lógico e a capacidade de reconhecer padrões, pensar com algoritmos, decompor um problema e abstrair um problema.

Já para Furber (2012, p. 29):

> Pensamento computacional é o processo de reconhecer aspectos da computação no mundo que nos rodeia, e aplicar ferramentas e técnicas da Ciência da Computação para entender e raciocinar sobre sistemas e processos tanto naturais quanto artificiais.

De maneira geral, CSTA; ISTE, 2011, p. 1, aponta em seu trabalho, as práticas e estratégias do PC:

> O pensamento computacional (PC) é um processo de solução de problemas que inclui (mas não está limitado a) as seguintes características:
>
> • Formulação de problemas de uma forma que nos permita usar um computador e outras ferramentas para ajudar a resolvê-los
>
> • Organização e análise lógica dos dados
>
> • Representação de dados através de abstrações como modelos e simulações
>
> • Automatização de soluções por meio de pensamento algorítmico

> (uma série de etapas ordenadas)
> • Identificação, análise e implementação de possíveis soluções com o objetivo de alcançar a combinação mais eficiente e eficaz de etapas e recursos
> • Generalização e transferência deste processo de resolução de problemas para uma ampla variedade de problemas
> Essas habilidades são apoiadas e realçadas por uma série de qualidades ou atitudes que são dimensões essenciais do PC. Essas qualidades ou atitudes incluem:
> • Confiança em lidar com a complexidade
> • Persistência no trabalho com problemas difíceis
> • Tolerância com ambiguidades
> • A capacidade de lidar com problemas em aberto
> • A capacidade de se comunicar e trabalhar com outras pessoas para alcançar um objetivo ou solução comum.

Associando todas as nomenclaturas e conceitos em relação ao PC, Grover; Pea, (2013, p. 39-40), apontam as etapas do PC:

> • Abstrações e generalização de padrões (incluindo modelos e simulações)
> • Processamento sistemático de informações
> • Sistemas de símbolos e representações
> • Noções algorítmicas de fluxo de controle
> • Decomposição estruturada de problemas (modularização)
> • Pensamento iterativo, recursivo e paralelo
> • Lógica condicional
> • Restrições de eficiência e performance
> • Depuração e detecção sistemática de erros.

Embora não exista um conceito concreto em relação ao PC, pesquisadores afirmam que ele se dá ao conjunto de habilidades e estratégias utilizadas na Ciência da Computação as quais são necessárias para solucionar problemas (WING, 2016).

Assim, o PC pode ser definido como uma abordagem que tem o intuito de resolver problemas utilizando processos de análises de dados, criação de modelos e construção de algoritimos que possibilitem auxiliar e facilitar o trabalho das pessoas (WING, 2006).

Raabe *et al.* (2018, p. 57) afirmam que o PC é constituído por 4 pilares

> I) decomposição: identificação e redução de um problema em partes menores para facilitar a compreensão;
> II) reconhecimento de padrões: verificar se uma solução que atende a uma parte pode ser repetida em outras situações dentro de um problema macro através de pontos que se repetem;
> III) abstração: manter o foco no que é importante e excluir o que pode ser descartado e;
> IV) algoritmo: conjunto de passos que que compõe uma solução replicável.

Blikstein (2008) aponta algumas características do pensamentos computacional: utilização do computador como ferramenta para auxiliar a resolução de problemas; análise de dados e sistematização lógica; exposição de dados por modelos e simulações; automação de soluções por meio do pensamento algorítmico; reconhecer, analisar e desenvolver possíveis soluções, generalizando as soluções para uma variedade de problemas.

Deste modo, o PC não é a prática de navegar na internet, operar softwares e enviar e-mail, mas sim de utilizar o computador buscando desenvolver capacidades cognitivas por meio de construções, produções e invenções (BLIKSTEIN, 2008).

É possível notar as iniciativas de covergência do pensamento computacional no Brasil, por meio de *notebooks*, *tablets, smartphones,* etc (PEREIRA; FRANCO, 2018). Contudo, a utilização destes dispositivos se dão apenas na reprodução de informações, não sendo explorado na maioria das vezes o potencial e os benefícios que as Tecnologias Digitais de Informação e Comunicação proporcionam para a construção do conhecimento (RODRIGUES; ALMEIDA; VALENTE, 2017).

Assim, as TDIC apresentam vantagens para pontencializar o desenvolvimento de conceitos relacionados ao PC, como por exemplo, organizar idéias, situações e solucionar problemas, estimulando o raciocínio lógico e o PC com o auxílio das ferramentas educacionais (VALENTE, 2016).

Nessa perspectiva o *Scratch* é uma das ferramentas utilizada no Brasil que possibilita trabalhar o PC, propiciando com que o aluno adquira habilidades como a lógica de programação e tomada de decisão entre outras (ARAÚJO, 2016). Deste modo, quando o aluno domina as habilidades do PC, apresenta facilidades na compreensão e interação de diferentes assuntos, encontrando soluções para lidar com a complexidade dos problemas do cotidiano (PEREIRA; FRANCO, 2018)

3. PASSOS METODOLÓGICOS

Nesse capítulo, evidenciamos todo o percurso investigativo da pesquisa, uma vez que foram utilizados diversos procedimentos para a elaboração dessa dissertação.

3.1 Caracterização da pesquisa

Para responder ao objetivo de criar um caderno de exercícios utilizando a ferramenta *Scratch e* consequentemente estimular o Pensamento Computacional nos alunos auxiliando no ensino de programação, realizou-se uma pesquisa qualitativa já que o pesquisador buscou coletar e analisar dados que permitiram explorar as dificuldades e compreender os problemas encontrados pelos alunos no curso desenvolvido nesse trabalho (CRESWELL, 2013).

A abordagem qualitativa leva em consideração a reflexão do pesquisador sobre os fatos observados e a sua comunicação com o campo de pesquisa são considerados como parte do processo (FLICK, 2009).

Para Gerhard e Silveira (2009, p. 32) ressaltam que

> [...] os pesquisadores que utilizam os métodos qualitativos buscam explicar o porquê das coisas, exprimindo o que convém ser feito, mas não quantificam os valores e as trocas simbólicas sem se submeterem a prova de fatos, pois os dados analisados são não – métricos (suscitados e de interação) e se valem de diferentes abordagens.

Por partir de aspectos que abordam a realidade e que tem o objetivo de explicar as relações sociais, a pesquisa qualitativa apresenta-se como a mais adequada na construção dessa dissertação, sendo que o trabalho iniciou do cotidiano das universidades e do uso das ferramentas educacionais para auxiliar o ensino de programação e desenvolvimento do pensamento computacional.

Esse tipo de pesquisa tem como características:

> […] objetivação do fenômeno; hierarquização das ações de descrever, compreender, explicar, precisão das relações entre o global e o local em determinado fenômeno; observância das diferenças entre o mundo social e o mundo natural; respeito ao caráter interativo entre os objetivos buscados pelos investigadores, suas orientações teóricas e seus dados empíricos; busca de resultado os mais fidedignos possíveis; oposição ao pressuposto que defende um modelo único de

> pesquisa para todas as ciências (GERTHARD; SILVEIRA, 2009, p. 32).

Para a construção desse trabalho, foram utilizados os procedimentos metodológicos da pesquisa bibliográfica de cunho qualitativo exploratório, a qual proporcionou a fundamentação teórica necessária para a elaboração do produto educacional e preocupou-se com os aspectos que envolviam a realidade do ensino de programação na Universidade Estadual do Norte do Paraná.

A primeira etapa da pesquisa bibliográfica é a escolha da temática, que de acordo com Lakatos e Marconi (2010) “as fontes para escolha do assunto podem originar-se da experiência pessoal ou profissional [...]” (p. 44). Dessa maneira, foi realizado um levantamento nos periódicos da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e no Congresso Brasileiro de Informática na Educação (CIBIE) entre os períodos de 2018 á 2020. Como critério para a seleção das pesquisas, utilizou-se as palavras-chave: “Programação”, “*Scratch*”, “Raciocínio lógico”, “Estrutura de decisão”, “Estrutura de repetição” e “Pensamento Computacional”.

Em vista disso, a temática pesquisada partiu da premissa de que a Tecnologia Digital de Informação e Comunicação oferta recursos tecnológico que possibilitam tornar o aprendizado mais atrativo para o aluno, auxiliando o processo de ensino.

Por meio do aporte teórico apresentado, notou-se que alunos e pesquisadores apontam a disciplina de programação como sendo uma das dificuldades encontradas nos cursos da área da Computação, e consequentemente ser a responsável pelo alto índice de evasão e desistência de alunos (DIAS; SERRÃO, 2014). Assim, o presente estudo subsidiou a elaboração do produto educacional, que consistiu em um caderno de atividades com o intuito de potencializar o processo de ensino de programação.

3.2 *Locus* da pesquisa

A pesquisa realizou-se na Universidade Estadual do Norte do Paraná (UENP), no *campus* Luiz Meneguel de Bandeirantes.

A UENP conta com dois cursos da área da Computação, sendo eles a Licenciatura em Computação e Ciência da Computação. Levando em consideração a disponibilidade dos alunos, os participantes da pesquisa foram os alunos do primeiro ano do curso de Ciência da Computação.

A escolha do público alvo foi definida a partir do levantamento bibliográfico, que apontou os anos iniciais dos cursos da área da Computação como sendo responssáveis pela grande evasão e desistência dos alunos, já que de acordo com Ventura (2018) é nessa etapa que os alunos se deparam com muitos obstáculos ao decorrer da disciplina, tais como, dificuldades em solucionar problemas, utilizar estruturas de condição e repetição, além da grande dificuldade em pensar computacionalmente.

3.3 Etapas e instrumentos para a coleta de dados

Para realizar a análise qualitativa utilizou-se de questionário semi-estruturado - Apêndice A, aplicado no ano de 2019 com os estudantes participantes, composto por questões dissertativas que buscavam verificar as percepções sobre o uso da ferramenta *Scratch* no ensino da Lógica de Programação.

A partir dos objetivos da pesquisa, foram definidas as etapas e instrumentos para a coleta de dados.

Na etapa 1ª foi realizado o levantamento dos estudos científicos nas bases de dados CAPES e CIBIE com o intuito de verificar as ferramentas digitais mais utilizadas para o ensino da Lógica de Programação, a qual apresentou como resultado a ferramenta *Scratch*;

A etapa 2ª contou com uma discussão junto aos alunos para a identificação de dificuldades no conteúdo das estruturas lógicas, de repetição e condição;

Já na etapa 3ª foi elaborado o curso e a lista de exercícios com o obejtivo de realizar uma intervenção com a ferramenta *Scratch* com os alunos do primeiro ano do curso de Ciência da Computação;

Na etapa 4ª realizou-se a análise dos questionários, os quais foram aplicados ao decorrer do curso, e respondidos pelos alunos de modo a compreender

as suas percepções em relação à experiência com a caderno de atividades no ensino de programação;

Na 5ª e última etapa analisou-se a resolução dos exercícios efetuados pelos alunos. A resolução de tais atividades foram gravadas do início ao fim por meio do software Atube Catcher*. Para a correção das atividades foram utilizados os pilares do Pensamento Computacional por meio de uma análise qualitativa livre.

Essas etapas permitiram acompanhamento dos alunos no decorrer do curso e assim identificar a adequação das atividades componentes do caderno para o desenvolvimento do Pensamento Computacional com o uso da ferramenta *Scratch*

3.4 O Caderno de Atividades e a estrutura do Curso

3.4.1 O caderno de Atividades

O caderno de atividades elaborado para desenvolvimento do curso busca auxiliar no ensino de programação por meio do Pensamento Computacional com o uso da ferramenta *Scratch*.

O Caderno de atividades foi composto por 15 exercícios e desenvolvido paralelamente ao curso. Os conteúdos utilizados para a construção desse produto estão diretamente relacionados ao processo de ensino de programação, e utilizou-se da ferramenta *Scratch* como ferramenta educacional auxiliadora no processo.

Para o desenvolvimento prático das atividades, foram selecionadas as estruturas de lógica de programação, decisão e repetição, apontadas por Pereira *et al.* (2012) como sendo fundamentais na disciplina de programação. Já a análise das atividades seguiu os pilares do Pensamento Computacional, o qual objetiva identificar se o aluno foi capaz de concluir todas as etapas das atividades RAABE *et al*., 2018).

* Atube Catcher 2020, Software Gratuito – diego@dsnetwb.com – Site: https://atubecatcher.com.br/

3.4.2 A estrutura do curso

A aplicação do curso na modalidade presencial, realizado em agosto e setembro de 2019, teve como objetivo auxiliar o ensino de programação com alunos do primeiro ano do curso de Ciências da Computação da Universidade Estadual do Norte do Paraná, *campus* de Bandeirantes/PR – Luiz Meneghel. Visando desenvolver um caderno de atividades para professores da área da Computação, abordando o uso da ferramenta *Scratch* no ensino de programação e o desenvolvimento do Pensamento Computacional, o curso foi elaborado e implementado em consonância com o produto educacional denominado Caderno de Atividades.

As atividades propostas no curso possibilitaram: a) acompanhar os alunos em relação ao desenvolvimento dos exercícios; b) observar a comunicação na aquisição de conhecimento e na troca de experiências; c) verificar a realização de atividades contextualizadas nos módulos em relação as estruturas; e d) analisar o Pensamento Computacional.

Assim, foi associada a ferramenta *Scratch* ao ensino de programação por meio de um curso envolvendo conteúdos teóricos, de vídeos e atividades sobre a temática. O conteúdo programático do curso foi definido junto aos professores da Área da Computação de modo a contribuir com o conhecimento teórico e prático dos alunos sobre o ensino de programação por meio do *Scratch.*

O curso obteve carga horária total de vinte horas, que foram divididas em dez encontros presencias de duas horas, em três módulos, conforme apresentado na Tabela 1.

Tabela 1 - Carga horária do curso e Módulos.

MÓDULO	ENCONTRO	DURAÇÃO
I – Lógica de Programação	1º	8 Horas
	2º	
	3º	
	4º	
II – Estrutura de Decisão	5º	6 Horas
	6º	
	7º	
III – Estrutura de Repetição	8º	6 Horas
	9º	
	10º	

Fonte: O autor (2019).

O Módulo I foi responsável por apresentar e coletar informações dos alunos em relação as dificuldades que os mesmos enfrentavam na disciplina de programação. Para isso, os alunos foram submetidos a responderem alguns questionários e solucionarem alguns exercicícios relacionados a lógica de programação. Abaixo segue o cronograma do Módulo I.

Tabela 2 - Cronograma e Atividades do Módulo I.

DATA	CONTEÚDO	ATIVIDADES
05/08	- Apresentação do curso; - Debate e discussão acerca da temática;	- Apresentação pessoal; - Apresentação do curso, cronograma das atividades, textos para leitura, atividades e questionários para avaliação. - Apresentação do Módulo 1; - Discussão a respeito das dificuldades encontradas na aprendizagem de programação “lógica de programação”;
06/08	- Debate e discussão acerca da temática; - Ambientação da ferramenta *Scratch*; - Coleta de dados. Encontro presencial	- Discussão a respeito das dificuldades encontradas na aprendizagem de programação “lógica matemática”; - Apresentação e acesso a ferramenta *Scratch*; - Avaliação diagnóstica; - Questionário 1: “Você já utilizou a ferramenta *Scratch*?”; - Questionário 2: “Você sabe o que é lógica de programação”; - Questionário 3 “Você já utilizou estruturas de decisão?”; - Questionário 4 “Você já utilizou estruturas de repetição?”.
12/08	- Discussão de textos; - Explorar a ferramenta *Scratch* e suas funcionalidades; - Desenvolvimento de Atividades.	- Discussão de textos a respeito das dificuldades da aprendizagem de programação “Lógica matemática”; - Mostrar aos alunos quais as funcionalidades das ferramentas *Scratch* e onde estão localizadas. - Aplicar exercícios básicos envolvendo lógica matemática a fim dos alunos familiarizarem com a ferramenta.
13/08	- Discussão de textos; - Correção de atividades; - Desenvolvimento de Atividades; - Avaliação.	Correção de exercícios; - Discussão de textos a respeito das dificuldades da aprendizagem de programação; - Aplicar exercícios envolvendo lógica matemática; - Avaliação.

Fonte: O autor (2019).

O Módulo II teve o objetivo de explorar exercícios de programação que envolvessem estruturas de condição, sendo necesssário utilizar a lógica de programação explorada no Módulo I. Para um melhor detalhamento das atividades e objetivos do Módulo II, foi desenvolvido um cronograma, o qual é exibido pela Tabela 3.

Tabela 3 - Cronograma e Atividades do Módulo II.

DATA	CONTEÚDO	ATIVIDADES
19/08	- Debate e discussão acerca da temática; - Correção de atividades; - Desenvolvimento de atividades.	- Correção de Exercícios; - Apresentação do Módulo 2, cronograma das atividades, textos para leitura, atividades para avaliação; - Discussão a respeito das dificuldades encontradas na aprendizagem de programação “estruturas de condição”; - Aplicar exercícios básicos envolvendo estrutura de repetição;
20/08	- Correção de exercícios; - Discussão de textos; - Desenvolvimento de Atividades.	- Correção de Exercícios; - Discussão de textos a respeito das dificuldades da aprendizagem de programação “Estrutura de condição”; - Aplicar exercícios básicos envolvendo estrutura de condição.
26/08	- Discussão de textos; - Correção de atividades; - Desenvolvimento de Atividades; - Avaliação.	Correção de exercícios; - Discussão de textos a respeito das dificuldades da aprendizagem de programação “estrutura de condição”; - Aplicar exercícios envolvendo estrutura de condição; - Avaliação; Ex. 2

Fonte: Elaboração própria (2019).

Por fim, o Módulo III objetivou explorar a utilização das estruturas de repetição para realizar atividades de programação no *Scratch*. Ressalta-se que os conhecimentos utilizados anteriormente nas atividades do Módulo I e II, tambem foram necessárias para solucionar as atividades do Módulo III, assim a os módulos se complementaram. A seguir foi representado a Tabela 4, contendo o cronograma das atividades do Módulo III.

Tabela 4 - Cronograma e Atividades do Módulo III.

DATA	CONTEÚDO	ATIVIDADES
27/08	- Debate e discussão acerca da temática; - Correção de atividades; - Desenvolvimento de atividades.	- Correção de exercícios; - Apresentação do Módulo 2, cronograma das atividades, textos para leitura, atividades para avaliação; - Discussão a respeito das dificuldades encontradas na aprendizagem de programação “estruturas de condição”; - Aplicar exercícios básicos envolvendo estrutura de repetição;
02/09	- Discussão de textos; - Correção de exercícios; - Desenvolvimento de Atividades.	- Discussão de textos a respeito das dificuldades da aprendizagem de programação “Estrutura de condição”; - Aplicar exercícios básicos envolvendo estrutura de condição.
03/09	- Discussão de textos; - Correção de atividades; - Desenvolvimento de Atividades; - Avaliação; - Avaliação/Questionário.	Correção de exercícios; - Discussão de textos a respeito das dificuldades da aprendizagem de programação “estrutura de condição”; - Aplicar exercícios envolvendo estrutura de condição; - Avaliação; Ex. 3 - Questionário; Opinião sobre o *Scratch.*

Fonte: O autor (2019).

As atividades desenvolvidas foram propostas e contextualizadas levando em consideração as dificuldades apontadas por alunos da área de computação e pesquisadores que apontaram as estruturas lógicas e a lógica matemática como sendo responssáveis por tais dificuldades Pereira *et al.* (2012). A fim de avaliar a adequação das atividades ao curso foram selecionados e desenvolvidos três exercícios, os quais são expostos na Tabela 5.

Tabela 5 - Atividades avaliativas por Módulo.

Módulo	Ativ.	Descrição
I	1	Desenvolva uma calculadora por meio de um algoritmo que o usuário escolha qual operação deve ser realizada: Adição, Subtração, Divisão e Multiplicação, em seguida são inseridos dois valores pelo usuário, na sequência a operação é realizada e o resultado é exibido.
II	2	Desenvolva um algoritmo que calcule a média de três notas as quais serão inseridas pelo usuário. Nota 1; Nota 2; Nota 3.

		Calcule, analise a média total do usuário e aponte se ele está aprovado ou de exame, considerando que a média é 7. Caso o aluno fique de exame, analise se o mesmo tem o direito de efetuar uma nova prova, caso seja possível, o usuário insere a nota do exame e o programa executa um novo cálculo a fim de saber se ele será aprovado ou não. Considerando que a média mínima do exame é 4 e a mínima para passar no exame é 5. Ao final, o algoritmo exibe ao usuário se ele está aprovado, aprovado com exame ou reprovado.
III	3	Desenvolva um algoritmo no qual o usuário digite diversos números inteiros e ao digitar -1, o programa exibe na tela qual foi o maior e o menor número digitado.

Fonte: O autor (2019).

As atividades apresentadas na Tabela 5 envolvem situações problemáticas que exigem do aluno habilidades de análise, interpretação, cálculo, lógica matemática e generalização da informação, além da necessidade de utilizar estrutururas lógicas de programação por meio do *Scratch* e o Pensamento Computacional.

3.5 Descrição das Atividades

Abaixo apresentou-se as atividades utilizadas para avaliar o desempenho dos alunos em relação a resoluçao dos problemas por meio de algoritimos desenvolvidos no *Scratch.*

- Estrutura 1: Lógica de Programação

Atividade 1 - Desenvolva uma calculadora por meio de um algoritmo que o usuário escolha qual operação deve ser realizada: Adição, Subtração, Divisão e Multiplicação, em seguida são inseridos dois valores pelo usuário, na sequência a operação é realizada e o resultado é exibido.

O objetivo dessa atividade foi avaliar se os alunos conseguiam pensar computacionalmente e utilizar estruturas lógicas a fim de desenvolverem um algoritmo para efetuar operações simples com dois dígitos. Aparentemente parece um simples problema, porém o aluno necessita usar raciocínios lógicos e estruturas específicas

para decompor o problema em problemas menores visto que estamos trabalhando com 4 operações distintas.

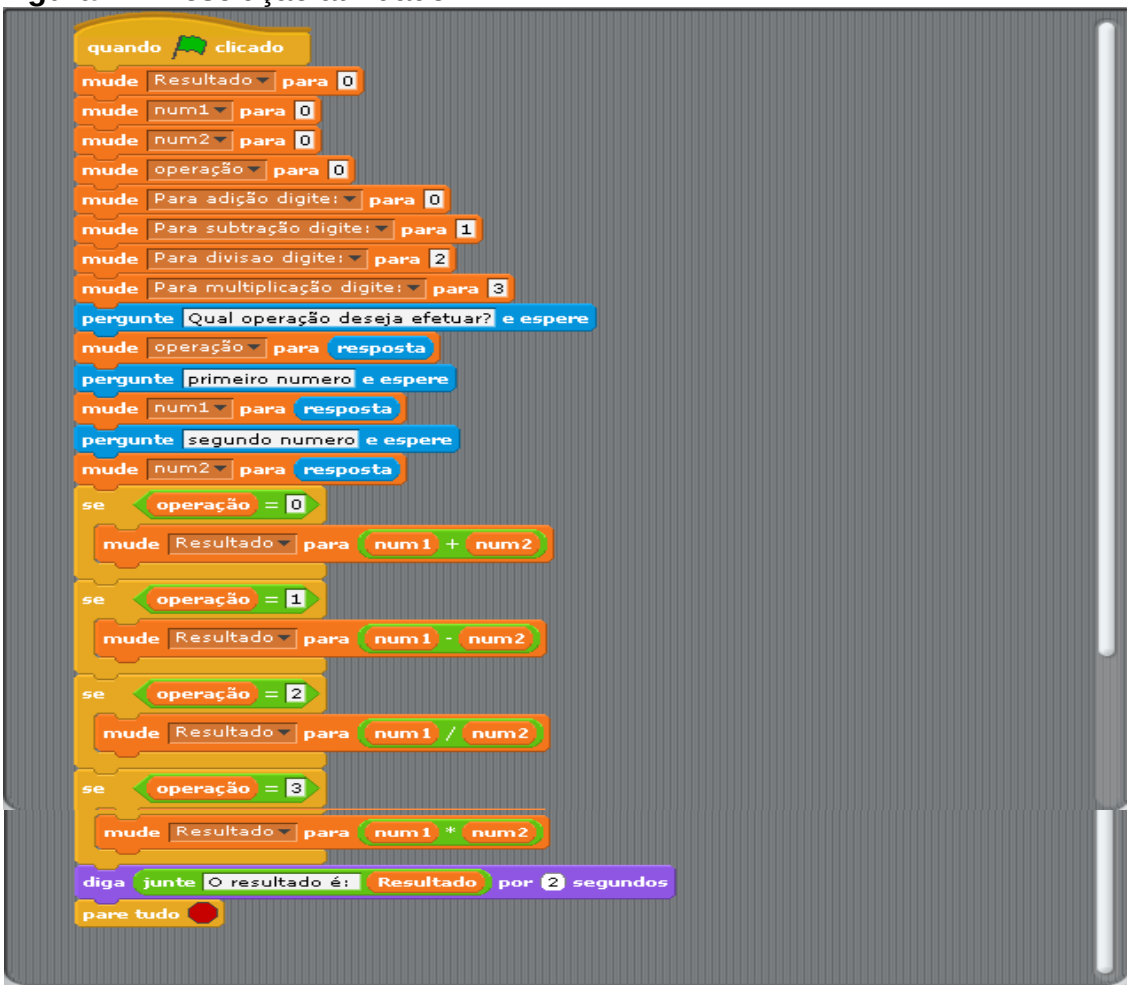
Tabela 6 – Etapas da Atividade 1.

Decomposição - O que precisa ser feito para desenvolver uma calculadora?
Primeiramente é necessário que os alunos definam e criem variáveis as quais serão responsáveis por armazenar os números inseridos pelos usuários além de outras variáveis as quais serão responsáveis por armazenar os resultados das operações e as operações que serão efetuadas. Em seguida, fazer a associação dos numerais e organizar como serão inseridos no algoritmo.
Padrões - O que as operações têm em comum?
Nessa parte, o aluno deve pensar como fazer os agrupamentos de numerais, como denominá-los e também os distinguir por operação, seja ela uma adição, subtração, multiplicação ou divisão.
Abstração - O que diferencia as operações?
Agora é a hora do aluno utilizar raciocínios lógicos e pensar como utilizar os numerais, os grupos, como e quando exibir os resultados, estruturando logicamente como cada função será executada.
Algoritmo - Como você pode reunir todas essas informações para criar uma série de instruções que podem ser seguidas por seus colegas de classe?
Essa etapa se constitui na elaboração do código, na qual o aluno precisa utilizar uma linguagem de programação a fim de fazer a comunicação entre o usuário e a máquina.

Fonte: O autor (2019).

A seguir foi representado por meio da Figura 1 uma alternativa para a resolução da atividade 1.

Figura 1 – Resolução atividade 1.



Fonte: O autor (2019).

- Estrutura 2: Condição

Atividade 2 - Desenvolva um algoritmo que calcule a média de três notas as quais serão inseridas pelo usuário.

Nota 1; Nota 2; Nota 3. Calcule, analise a média total do usuário e aponte se ele está aprovado ou de exame, considerando que a média é 7.

Caso o aluno fique de exame, analise se o mesmo tem o direito de efetuar uma nova prova, caso seja possível, o usuário insere a nota do exame e o programa executa um novo cálculo a fim de saber se ele será aprovado ou não. Considerando que a média mínima do exame é 4 e a mínima para passar no exame é 5. Ao final, o algoritmo exibe ao usuário se ele está aprovado, aprovado com exame ou reprovado.

Essa atividade foi desenvolvida com o objetivo de analisar se os alunos conseguem utilizar estruturas de condição a fim de solucionar problemas. Para

isso foi desenvolvido um problema decorrente do cotidiano dos universitários, que é o cálculo de notas a fim de saber se o aluno conseguiu atingir a média necessária para passar de ano.

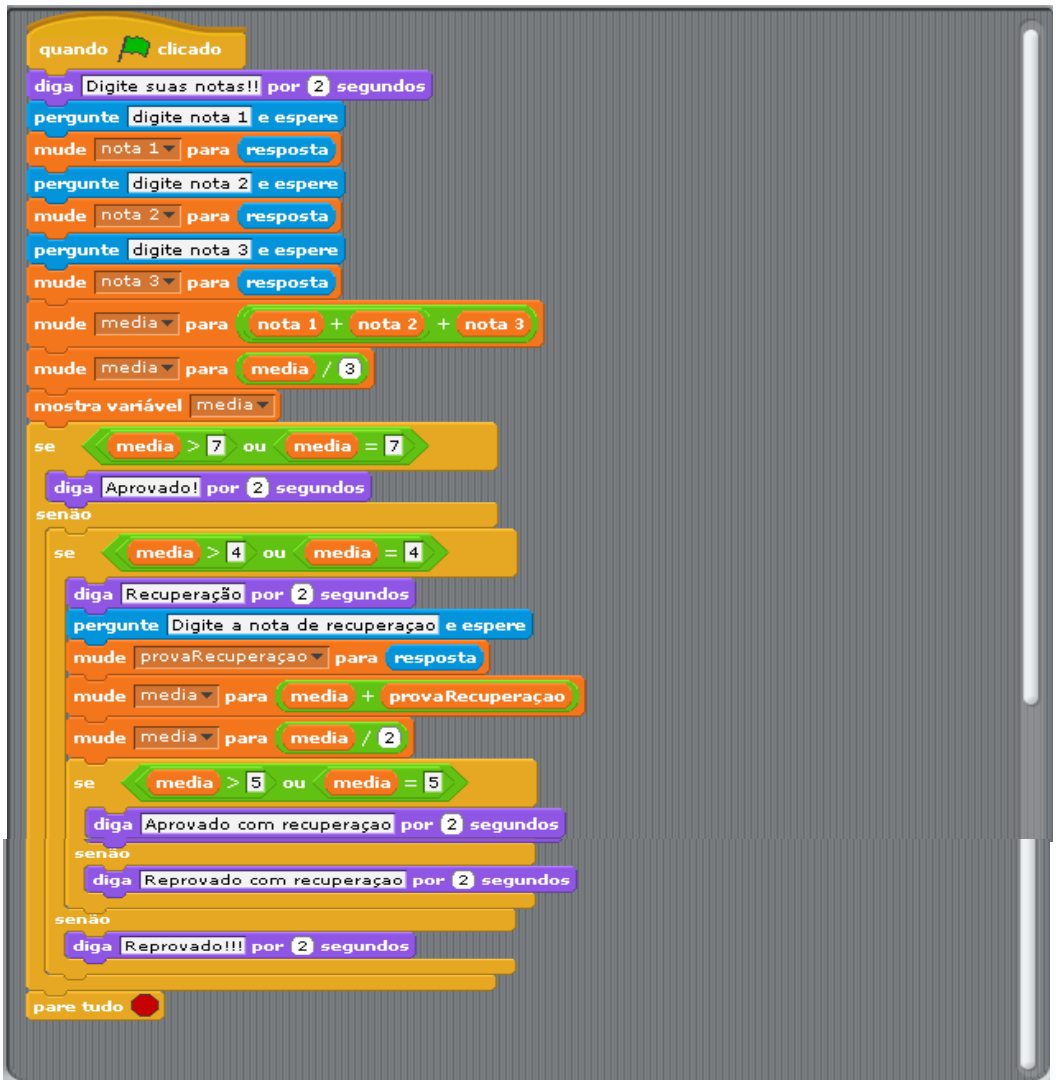
Tabela 7 – Etapas Atividade 2.

Decomposição - O que precisa ser feito para calcular médias?
Nessa primeira fase do pensamento computacional, o aluno necessita buscar em seus conhecimentos como efetuar o cálculo para tirar a média das notas, por mais que seja um exercício rotineiro, o mesmo apresenta um grau relevante de dificuldade pois, além das médias é necessário utilizar as estruturas de condição e fazer comparações de valores a fim de chegar a um esquema computacional.
Padrões - O que as operações tem em comum?
Já na etapa dos padrões, o aluno necessita de um plano mental o qual o auxiliará a encontrar dados semelhantes no problema, isso por que é necessário efetuar agrupamento de padrões semelhantes como soma das notas, média das notas, diferença entre números e comparações.
Abstração - O que diferencia as operações?
Esse exercício exige do aluno domínio de estruturas lógicas, visto que o mesmo necessita fazer comparações e utilizar conectivos lógicos a fim de solucionar o problema, nesse caso, quais estruturas irá utilizar, quais métodos de entrada e saída, variáveis, dentre outros.
Algoritmo - Como você pode reunir todas essas informações para desenvolver um algoritmo que solucione os cálculos necessários?
A partir das fases anteriores espera-se que os alunos consigam estruturar o código e solucionar o problema, utilizando principalmente as estruturas de condições, as quais são fundamentais para a solucionar a atividade. Isso depende de testes nos quais o usuário utiliza a linguagem de programação para que o computador execute as operações.

Fonte: O autor (2019).

A seguir foi representado por meio da Figura 2 uma alternativa para a resolução da atividade 2.

Figura 2– Resolução atividade 2.



Fonte: O autor (2019).

- Estrutura 3: Condição

Atividade 3 - Desenvolva um algoritmo no qual o usuário digite diversos números inteiros e ao digitar -1, o programa exibe na tela qual foi o maior e o menor número digitado.

Buscando avaliar a capacidade dos alunos de utilizarem estruturas de repetição. Essa atividade foi desenvolvida com o intuito de ele consiga criar uma estrutura lógica para selecionar o maior e o menor número dentre os números inseridos pelo usuário. Assim, é necessário que o aluno efetue cálculos, além de utilizar estruturas lógicas, estruturas de condição e estruturas de repetição.

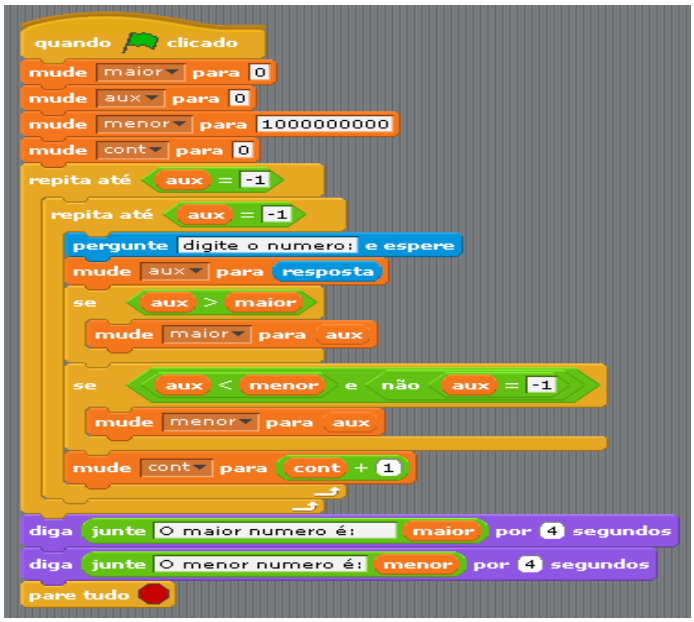
Tabela 8 – Etapas Atividade 3.

Decomposição - O que precisa ser feito para selecionar os números inseridos?
Nessa primeira parte espera-se que aluno divida o problema em partes menores, por exemplo, distinguir os números menores dos maiores e pensar como ele fará a seleção dos numerais.
Padrões - Quais são as semelhanças encontradas na atividade?
Agora é o momento em que os alunos farão os agrupamentos, no caso da inserção de números, quais os padrões que o aluno irá criar para armazenar tais informações, e como elas serão tratadas dentro do algoritmo. Qual o padrão utilizado para que o programa finalize a repetição e exiba os resultados.
Abstração - Quais as diferenças entre os algoritmos digitados?
Na terceira etapa o aluno distingui os algoritmos por meio de comparações com os números inseridos anteriormente, e para que isso ocorra é necessário que o mesmo desenvolva uma estrutura que envolva estruturas lógicas e de repetição.
Algoritmo - Como estruturar um código a fim de resolver essa atividade?
Nessa etapa o aluno deve organizar e estruturar o código responsável por efetuar os cálculos e comparações dos valores inseridos no programa, e para que isso ocorra o mesmo necessita utilizar e organizar estruturas lógicas por meio de algoritmos que seguem uma ordem lógica.

Fonte: O autor (2019).

A seguir foi representado por meio da Figura 3 uma alternativa para a resolução da atividade 3.

Figura 3 – Resolução atividade 3.



Fonte: O autor (2019).

As atividades acima foram responsáveis por avaliar a utilizaçao das estruturas lógicas necessárias para solucionar os problemas, assim o intuito dessas foi observar qual a lógica que os alunos utilizaram para desenvolver os algoritimos e como eles organizam os dados.

A seguir, no próximo capítulo, apresentou-se e analisou-se os os algoritimos desenvolvidos pelos alunos.

4. DESCRIÇÃO E ANÁLISE DOS DADOS

Neste capítulo, apresenta-se os dados coletados dos alunos durante o curso, bem como as análises dos exercícios. Para a coleta desses dados utilizou-se o Scratch e o *software* Atube®, o qual foi responsável por gravar a resolução e o desenvolvimentos dos exercícios do início ao fim, auxiliando e enrriquecendo a análise das atividades.

Os dados sistematizados representam a compreensão dos alunos sobre a utilização do *Scratch* na disciplina de programação e foram replicados abaixo por meio de figuras. Assim, visando organizar os conteúdos obtidos, optou-se por particionar esse capítulo em quatro categorias.

O curso contou com a presença de 16 alunos do 1º ano do curso de Ciência da Computação da Universidade Estadual do Norte do Paraná - UENP. Em função do considerável número de inscritos e a complexidade em analisar os exercícios, foi necessário estabelecer critérios de inclusão e exclusão para a análise de dados, que foi definido da seguinte forma: dos 16 alunos participantes, apenas 15 concluíram o curso com no mínimo 80% de frequência, destes o total de 6 alunos concluíram o curso com 100% de frequência. Desse modo, os 6 alunos que obtiveram 100% de frequência foram selecionados para a análise dos dados.

Os alunos foram codificados com a vogal A em um número ordinal: A1, A2, A3 ... A6. Os excertos foram descritos na íntegra, uma vez que não houve a correção gramatical.

Posto isso, a Categoria 1 para a análise, foi associada ao questionário e a análise dos exercícios relacionados as **Estruturas Lógicas de Programação**, o qual foi aplicado no (Módulo I) do curso, contemplando as unidades dos conceitos iniciais utilizados na disciplina de programação e o Pensamento Computacional para solucionar problemas**.**

Na Categoria 2 verificou-se o questionário e a análise dos exercícios relacionados às **Estruturas de Condição** que foi aplicado no Módulo II do curso. Desse modo, as unidades analisadas permitiram investigar a habilidade dos alunos em utilizar estruturas de condição para solucionarem problemas.

Na Categoria 3 analisou-se os exercícios relacionados as **Estruturas de Repetição**, as quais foram exploradas no (Módulo III) do curso.

Já a Categoria 4 refere-se aos questionários que foram respondidos ao decorrer do curso, buscando evidenciar o conhecimento dos alunos sobre o ***Scratch*** e as possíveis dificuldades encontradas na resolução das atividades**.** Tendo isso em vista, as unidades consistiram em investigar a utilização do *Scratch* em sala de aula para auxiliar o ensino de programação por meio do desenvolvimento do **Pensamento Computacional**.

4.1 Categoria 1 – Lógica de Programação.

O raciocínio é a forma mais complexa do pensamento, sendo que a lógica de programação estuda a “correção do raciocínio”, organizando e ordenando os pensamentos a partir do uso da lógica (FORBELLONE; EBRSPACHER, 2005).

Nesse contexto, essa categoria encarregou-se de analisar as informações relacionadas as lógicas de programação abordadas no Módulo I, as quais foram coletadas por meio de atividades e questionários. Para isso, os alunos participaram de aulas que envolveram conceitos inicias de programação como por exemplo utilizar e criar variáveis, utilizar operadores lógicos como Maior, Menor, Igual e Diferente, efetuar operações utilizando os operadores de Soma, Subtração, Divisão e Multiplicação, inserção e saída de informações entre outras operações que são fundamentais para iniciar um algoritmo por meio de linhas de códigos.

Nessa categoria os alunos efetuaram diversas atividades a fim de se familiarizarem com a ferramenta *Scratch*. Entretanto, para a análise dos dados foi utilizado uma atividade em específica, a qual teve o objetivo de avaliar a habilidade dos alunos em utilizar a lógica de programação e os 4 pilares do Pensamento Computacional na solução da atividade.

A atividade utilizada para avaliar o Módulo I está representada no Quadro 1.

Quadro 1 – Atividade avaliativa do (Módulo I).

Desenvolva uma calculadora por meio de um algoritmo que o usuário escolhe qual operação deve ser realizada: Adição, Subtração, Divisão ou Multiplicação, em seguida são inseridos dois valores pelo usuário, na sequência a operação é realizada e o resultado é exibido.

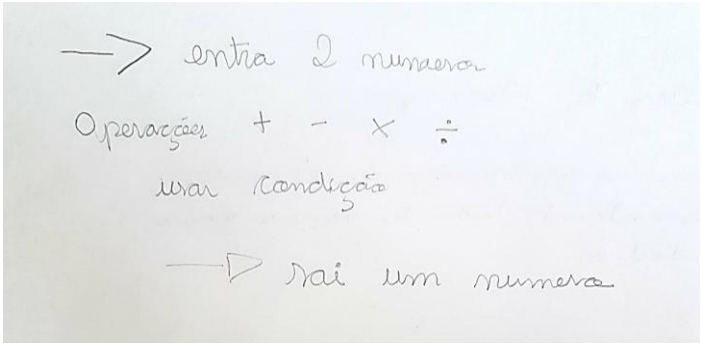
Fonte: O autor (2019).

Seguindo as etapas do Pensamentos Computacional, nota-se que alguns alunos utilizaram rascunho para esboçar a Decomposição da atividade, as quais foram replicadas abaixo por meio das Figuras 4, 5 e 6.

A etapa de Decomposição é apontada por RAABE *et al*., (2018) como sendo responsável por dividir o problema em problemas menores, ou seja, elaborar e particionar o problema complexo em partes menores, menos complexas, facilitando o desenvolvimento e o entendimento da atividade.

Assim, observou-se que os alunos A2, A3 e A5 utilizaram o rascunho para efetuar a decomposição do problema.

Figura 4 – Esboço A2.

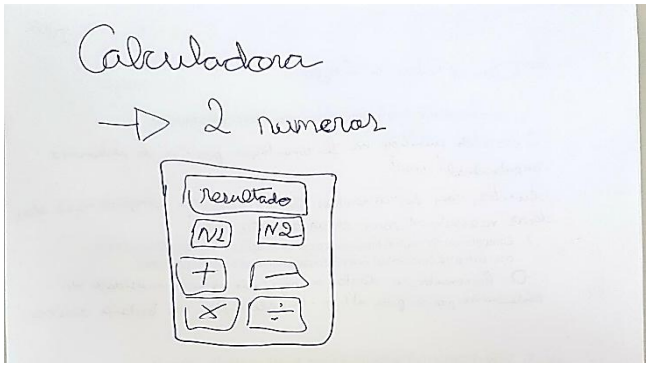


Fonte: A2 (2019).

De acordo com a Figura 4, ficou claro que o aluno esboçou o mecanismo de uma calculadora por meio de setas e escritas, assim, supõe-se que o mesmo consiga utilizar o pensamento computacional para solucionar o problema por meio da programação no *Scratch,* isso pois, o aluno por mais que tenha utilizado um rascunho, fica claro que o mesmo seguiu uma ordem lógica, organizando e agrupando os dados, além de direcionar as informações que seriam inseridas pelo usuário e as que seriam exibidas para o usuário.

Já na Figura 5, notou-se que A3 utilizou desenhos para demonstrar a quantidade de números que serão digitados, os operadores que serão utilizados e a quantidade de informações que o problema irá gerar.

Figura 5 – Esboço A3.

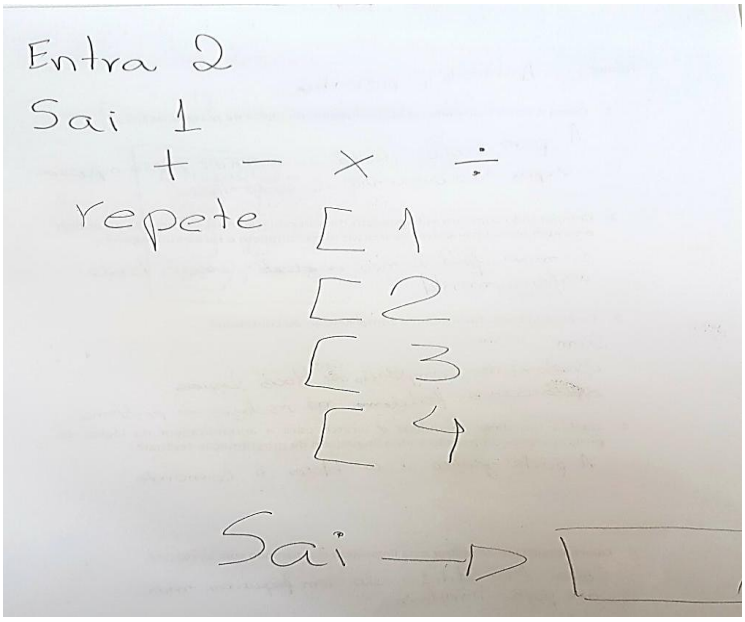


Fonte: A3 (2019).

Ainda explorando a Figura 5, é possível notar que A4 segue uma ordem lógica em seu desenho, na qual ele utiliza padrões para agrupar os dados que serão inseridos pelo usuário, processados pelo programa e os dados que o programa irá exibir ao usuário.

Já A5 desenvolveu um esboço mais detalhado, o qual apontou até mesmo uma estrutura de repetição, a qual ele poderia utilizar para desenvolver a atividade.

Figura 6 – Esboço A5.

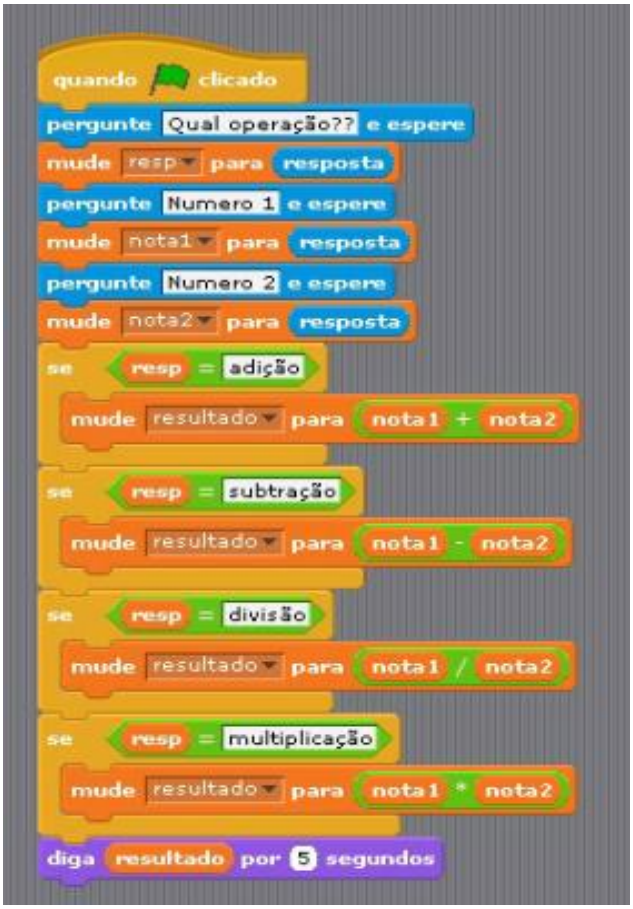


Fonte: A5 (2019).

As figuras 4, 5 e 6 ainda evidenciam à segunda etapa do Pensamento Computacional denominada Reconhecimento de Padrões as quais são apontadas por RAABE *et al*., (2018). Na figura 4 e 6 por exemplo, os alunos A3 e A5 fizeram o agrupamento dos operadores de Soma, Subtração, Multiplicação e Divisão, agrupando-os na mesma linha, propondo assim uma ordem ou padrão.

Para melhor detalhamento e categorização das etapas decorrentes do Pensamento Computacional foram exibidos por meio das Figuras 7, 8 e 9 os respectivos algoritmos desenvolvidos pelos alunos utilizando o *Scratch*.

Figura 7– Algoritmo dos alunos A1, A2 e A3 atividade 1.



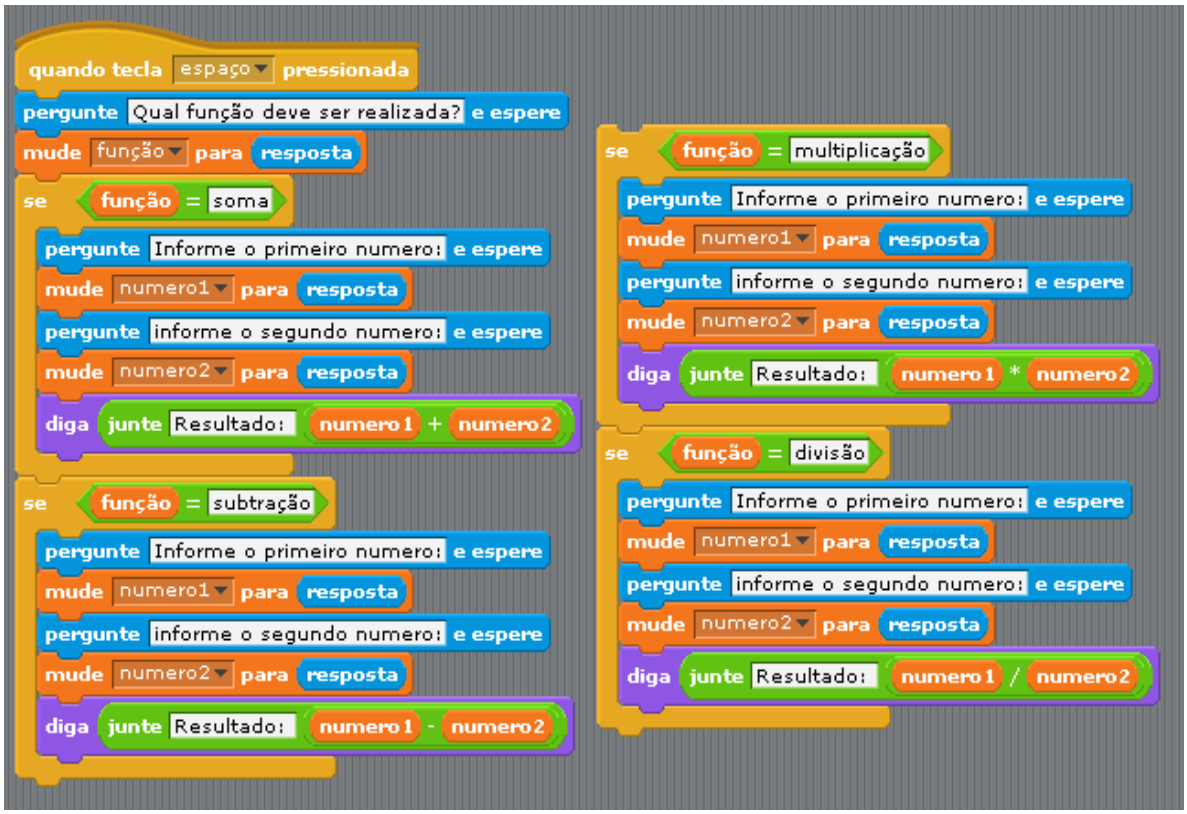
Fonte: A1 (2019).

De acordo com a Figura 7 nota-se que os alunos A1, A2, A3 conseguiram organizar e utilizar os padrões necessários para solucionar a atividade.

Ao analisar a figura 8 ressaltou-se que o padrão utilizado por A4 e A6 distinguiu dos demais, uma vez que os mesmos utilizaram o padrão para selecionar a operação desejada pelo usuário e ao decorrer do código, junto aos padrões para

receber os números inseridos pelos usuários. Já os outros alunos optaram por coletar as informações no início do algoritmo A1, A2 e A3

Figura 8 – Algoritmo A4 e A6, atividade 1.



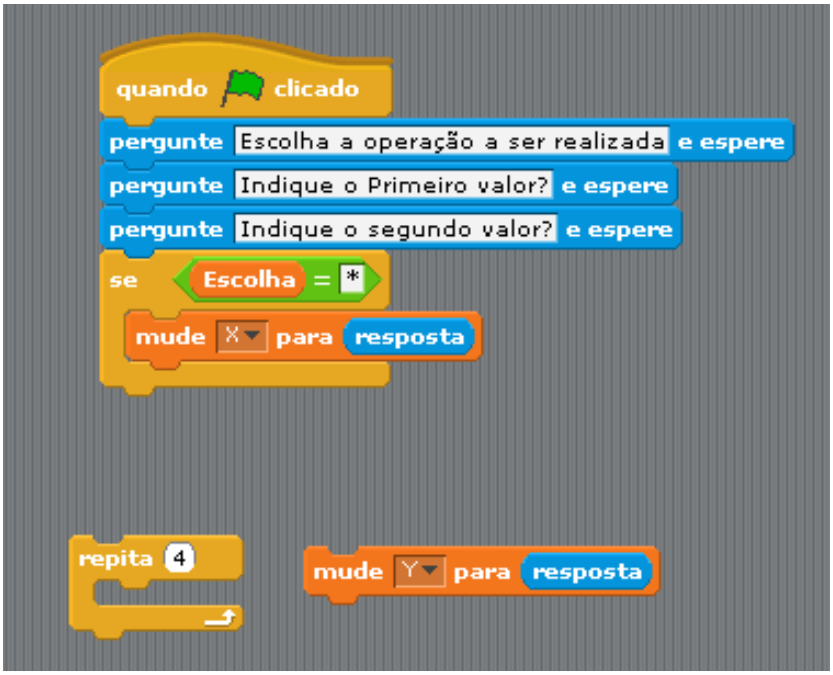
Fonte: A4 e A6 (2019).

Entretanto, as soluções desenvolvidas e apresentadas por A1, A2, A3, A4 e A6 atingiram os objetivos propostos pela atividade 1.

Já o A5 não conseguiu criar padrões utilizando a ferramenta *Scratch*, todavia, a Figura 6 exibe o esboço efetuado pelo mesmo aluno, no qual é possível notar que ele conseguiu criar padrões utilizando a folha de papel. O aluno ainda deixou observações no rascunho: “Não consegui fazer o programa executar correto; não consegui chegar ao objetivo final que era exibir o resultado da operação” (A5).

Assim, de acordo com a análise dos dados desse aluno conclui-se que o mesmo tinha conhecimento de como solucionar o problema por meio de desenhos e numerais, utilizando uma folha de papel, mas não apresentou habilidades necessárias para desenvolver um algoritmo por meio do *Scratch* a fim de solucionar atividade proposta. A tentativa de resolução efetuado pelo aluno é apresentada pela Figura 9.

Figura 9 – Algoritmo A5, atividade 1.



Fonte: A5 (2019).

Desse modo, ressalta-se que o aluno A5 apresentou dificuldades em utilizar o Pensamento Computacional, ou seja, ele sabe como solucionar o exercício por meio de papel e caneta, mas não sabe transcrever a resolução dessa atividade utilizando uma linguagem de programação, a qual é o meio de comunicação entre o homem e o computador (WING, 2016).

Com exceção do A5, todos os outros conseguiram efetuar a terceira etapa do PC, visto que conseguiram utilizar estruturas que filtraram e classificaram o problema. De modo geral, esses alunos utilizaram estruturas de condições para filtrar os problemas de maneira separada. Raabe *et al.* (2018) apontam que a terceira etapa, denominada Abstração, é responsável por filtrar e classificar os dados, por meio estruturas que permitam separar apenas os elementos essenciais em determinado problema, deste modo, os alunos com exceção de A5, conseguiram utilizar tal etapa.

Ao averiguar as estruturas desenvolvidas pelos alunos, percebeu-se que A1, A2 e A4 utilizaram a estrutura de condição (SE) o qual tornou a execução do programa mais rápida e eficiente se comparado a uma suporta utilização de estrutura de condição (SE) e (SENÃO) a qual foi considerada nessa atividade como uma estrutura de execução mais lenta e com maior probabilidade de gerar problemas futuros, uma vez que para que algoritmo efetue uma subtração é necessário que o programa execute cada estrutura anterior e análise se é a operação escolhida pelo

usuário ou não. Porém, o algoritmo desenvolvido por ele também funcionou de maneira correta, sanando os objetivos propostos pela atividade.

A última etapa denominada Algoritmo reúne a estratégia ou o conjunto de instruções claras e necessárias, ordenadas para a solução de um problema por meio de uma linguagem de programação. Essa etapa foi concluída por A1, A2, A3, A4 e A6 posto que desenvolveram algoritmos eficazes por meio da linguagem de programação do *Scratch*, utilizando todas as etapas do PC. Vale ressaltar ainda que a ferramenta proporcionou diversidade de códigos os quais utilizaram caminhos diferentes para concluir a atividade.

Antes de finalizar o Módulo I, os alunos responderam a um questionário com a seguinte indagação: Qual sua opinião em relação ao *Scratch* ao utilizá-lo para trabalhar com estruturas lógicas de programação? As respostas foram replicadas por meio da Tabela 9.

Tabela 9 – Respostas dos alunos.

A1	“Ajudou a entender os comandos melhor porque na ferramenta os comandos em blocos ficam mais fácil de utilizar porque é só arrastar”.
A2	“Demonstra tudo de maneira gráfica e intuitiva”.
A3	“Como já tive a disciplina de programação no primeiro semestre, não me auxiliou muito”.
A4	“Auxilia no entendimento do que está sendo feito, devido a sua simplicidade é fácil de entender o que se deve fazer para resolver o problema proposto”.
A5	“Linguagem mais clara, com várias opções, ferramentas para construir o código”.
A6	“Ajuda a desenvolver o pensamento computacional através de ações simples para resolver problemas”.

Fonte: O autor (2019).

As informações descritas pelos alunos se fazem pertinente já que o *Scratch* possui o objetivo de auxiliar o processo inicial de programação, devido às suas características de usabilidade e à sua linguagem de programação gráfica que ocorre por meio de blocos lógicos, os quais são arrastados e encaixados dando forma ao algoritmo (REZENDE; BISPO, 2018). Esse processo de arrastar os blocos lógicos foi apontado pelo aluno A1 como um facilitador no processo de entendimento da lógica de programação, indicando que o *Scratch* auxilia o processo de ensino já que é classificada pela maioria dos alunos como sendo intuitiva, simples e ajuda a resolver problemas desenvolvendo o pensamento computacional.

Levando em consideração os dados obtidos no Módulo I. Concluiu-se que o *Scratch* apresenta vantagens ao ser trabalhado em sala de aula para o ensino

de programação. Essa ferramenta é apontada pelos autores contidos no aporte teórico como sendo benéfica para o ensino introdutório de programação, por isso o A3 afirmou que a ferramenta não o auxiliou, já que ele havia concluído o primeiro semestre da disciplina de programação. Percebe-se pelas atividades efetuadas em sala e pela avaliação (Figura 6) que o aluno já tem uma boa base de programação e sabe utilizar o pensamento computacional para resolver problemas.

A categoria a seguir demonstra a análise dos dados referentes ao Módulo II.

4.2 Categoria 2 – Estruturas Lógicas de Condição.

Nessa categoria foram analisados os dados referentes às estruturas lógicas de condição que foram coletadas no Módulo II. Tais estruturas se fazem presente na maioria dos algoritmos de programação, sendo apontado por Pereira *et al.* (2012) como uma das estruturas fundamentais para o desenvolvimento de um algoritmo.

Para avaliar a capacidade dos alunos em utilizarem estruturas de condição, os mesmos foram encarregados de solucionar a atividade do Quadro 2.

Quadro 2 – Atividade avaliativa do Módulo II.

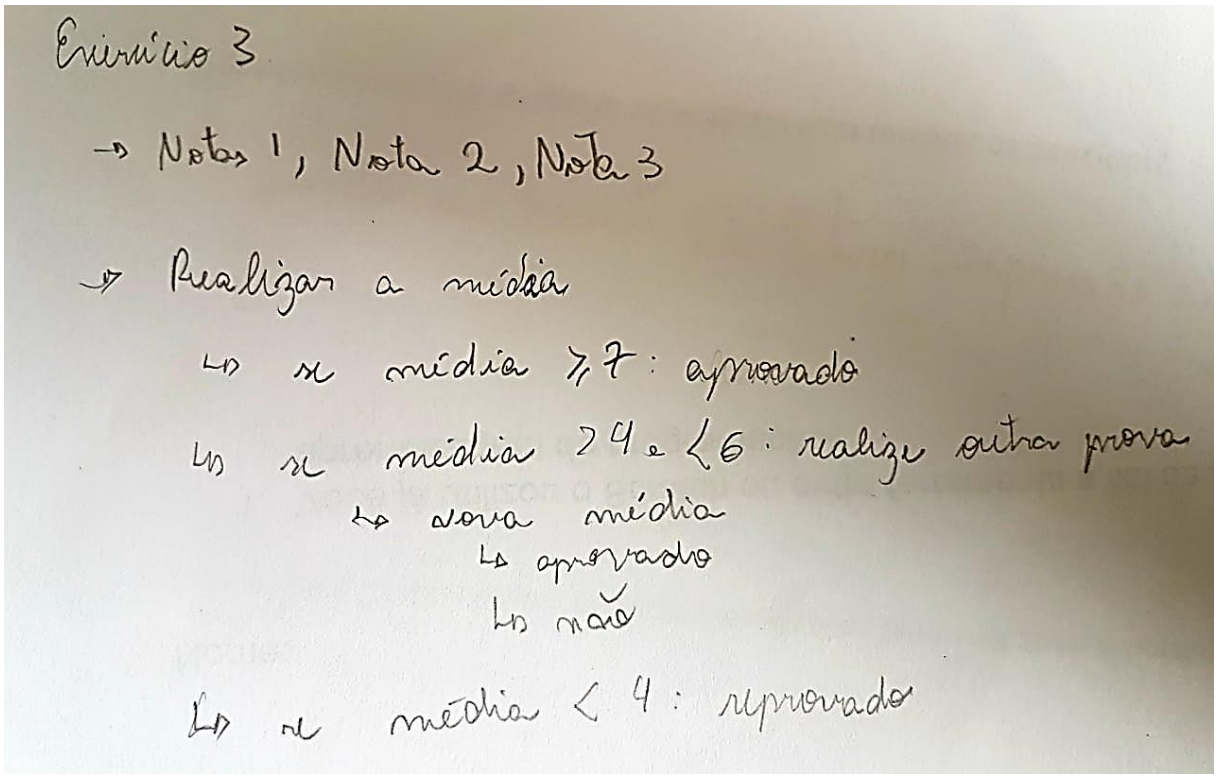
Desenvolva um algoritmo que calcule a média de três notas as quais serão inseridas pelo usuário. Nota 1; Nota 2; Nota 3. Calcule, analise a média total do usuário e aponte se ele está aprovado ou de exame, considerando que a média é 7. Caso o aluno fique de exame, analise se o mesmo tem o direito de efetuar uma nova prova, caso seja possível, o usuário insere a nota do exame e o programa executa um novo cálculo a fim de saber se ele será aprovado ou não. Considerando que a média mínima do exame é 4 e a mínima para passar no exame é 5. Ao final, o algoritmo exibe ao usuário se ele está aprovado, aprovado com exame ou reprovado.

Fonte: O autor (2019).

Seguindo as etapas do Pensamento Computacional com o objetivo de avaliar as atividades, foram replicados por meio das Figuras 12; 13; 14; e 15 os

rascunhos desenvolvidos pelos alunos A2, A3, A5 e A6. Os alunos A1 e A4 não utilizaram rascunho.

Figura 10 – Esboço A2 e A3.

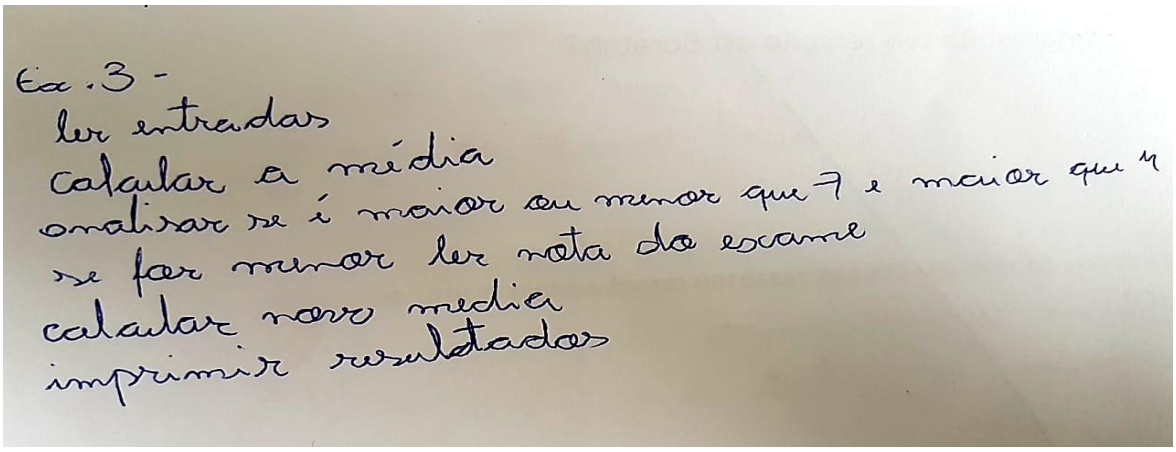


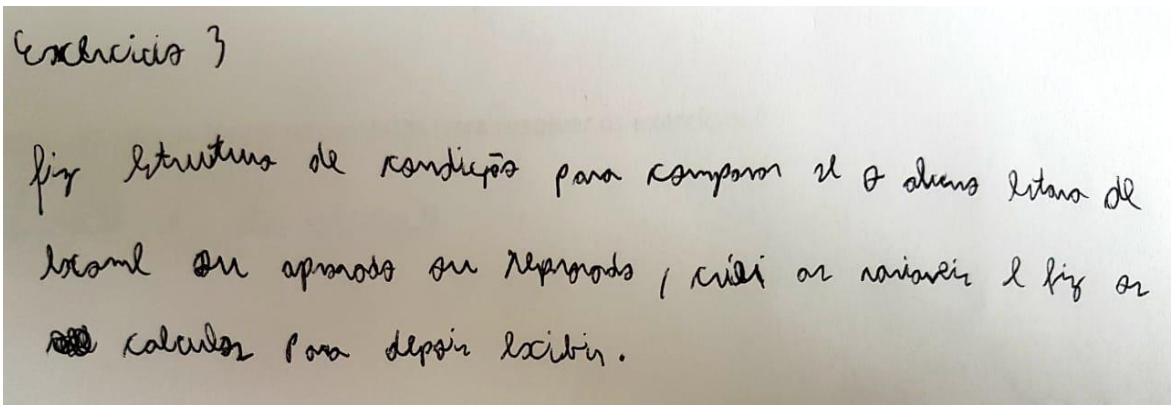
Fonte: A2 (2019).

Ao explorar os esboços apresentados, evidenciou-se as duas primeiras etapas do PC – a Decomposição e o Reconhecimento por Padrões. Os alunos A2 e A3 praticamente seguiram o mesmo raciocínio, visto que primeiramente dividiram as variáveis de entrada, ou seja, as notas que o usuário irá inserir no programa, posteriormente fizeram um esquema de critérios utilizando numerais e operadores lógicos a fim de classificar a média do usuário e por fim classificaram o aluno como aprovado, aprovado com exame e reprovado. Nota-se ainda que seguiram uma ordem lógica por padrões a fim de dividir o problema em partes menores.

Já A5 e A6 desenvolveram um esquema mais textual, a fim de esboçar a problemática proposta pela atividade.

Quadro 3 – Esboço A5 e A6





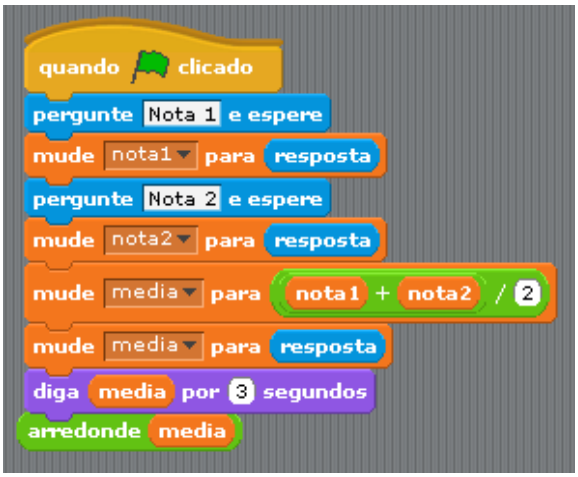
Fonte: A5 e A6 (2019).

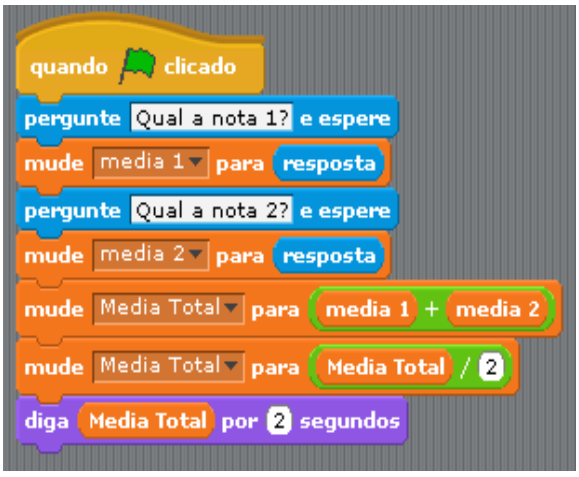
Porém, ainda fica claro que os mesmos estipularam padrões e seguiram uma ordem lógica, dividindo o problema em partes menores, representando a Decomposição e o Reconhecimento por Padrões, assim como afirma (RESNICK, 2017).

Para sondar as etapas da Abstração e do Algoritmo foram replicados abaixo por meio do Quadro 4 e 5 e as Figuras 11 e 12 os códigos desenvolvidos respectivamente por A1, A2, A3, A4, A5 e A6.

A partir dos algoritmos exibidos no Quadro 4, percebe-se que A1 e A5 não atingiram os objetivos exigidos pela atividade. Porém, vale apontar que os mesmos conseguiram efetuar uma parte da atividade, calculando a média e exibindo um resultado, importantes para o processo de programar.

Quadro 4 – Algoritmo A1 e A5, atividade 2.





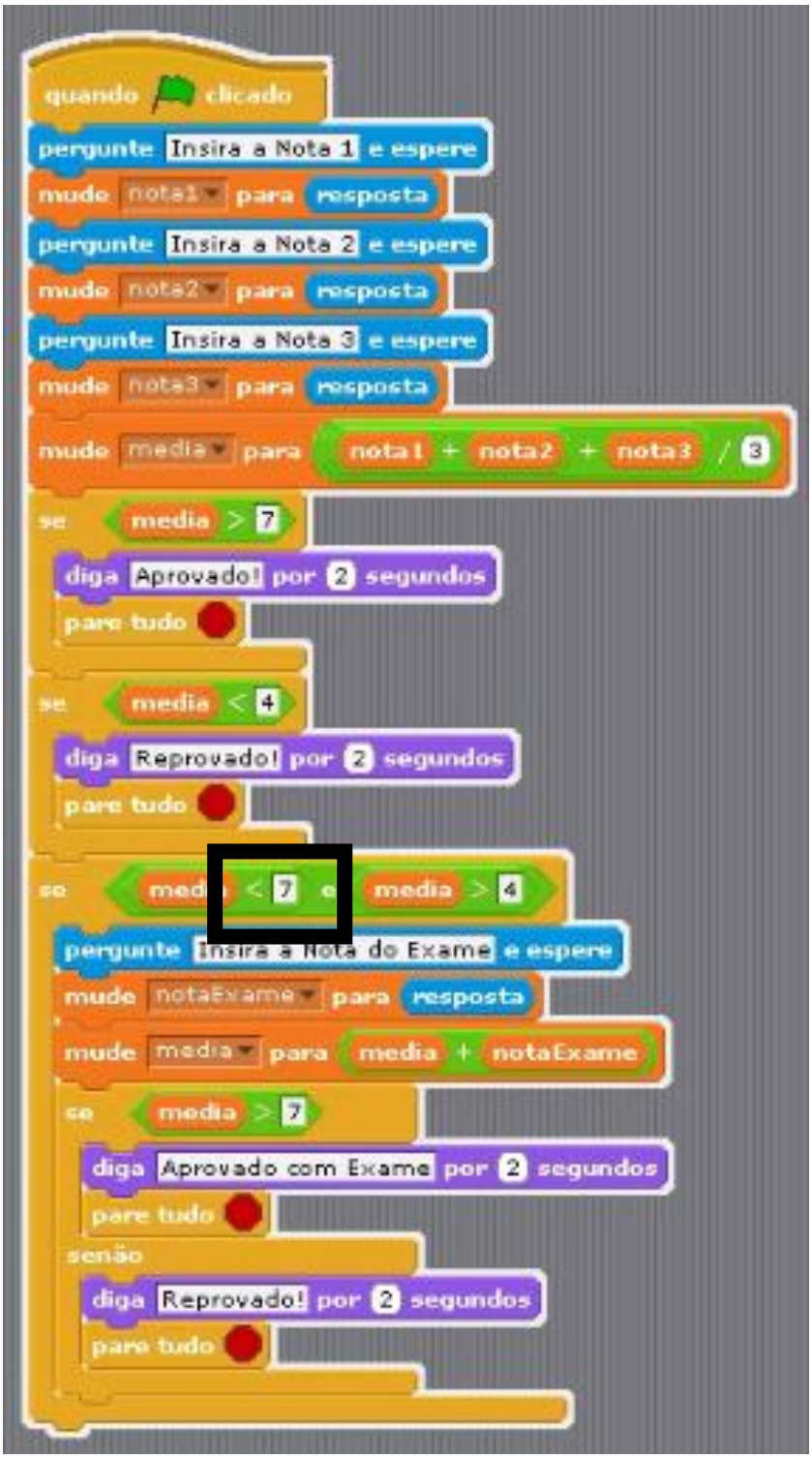
Fonte: A1 e A5 (2019).

O algoritmo desenvolvido por eles funciona perfeitamente, ficou claro que as etapas do pensamento computacional estão contidas no algoritmo, visto que o mesmo funciona. Porém, o objetivo dessa atividade era avaliar se os alunos conseguiriam utilizar estruturas de condição, as quais eram fundamentais para solucionar o problema proposto. Assim, o algoritmo desenvolvido e apresentado por A1 e A5 é funcional, utiliza os passos do PC, mas não satisfaz o objetivo proposto pela atividade devido a não utilização de estruturas de condição para tratar a média das notas do usuário.

Nesse contexto, A1 e A5 apresentaram fragilidades em utilizar estruturas de condição. Tais dificuldades, segundo Resnick (2017), podem estar associadas a falta de criatividade, falta de estímulo do Pensamento Computacional, incapacidade de utilizar e estruturar algoritmos por meio da lógica de programação, deficiência no processo de aprendizagem dessas estruturas, entre outras, as quais podem ser minimizadas com a utilização do *Scratch*.

Ao explorar os códigos desenvolvidos por A2 e A4 notou-se uma problemática semelhante as quais foram destacadas por meio de retângulos com bordas pretas, nas respetivas figuras.

Figura 11 – Algoritmo A2, atividade 2.



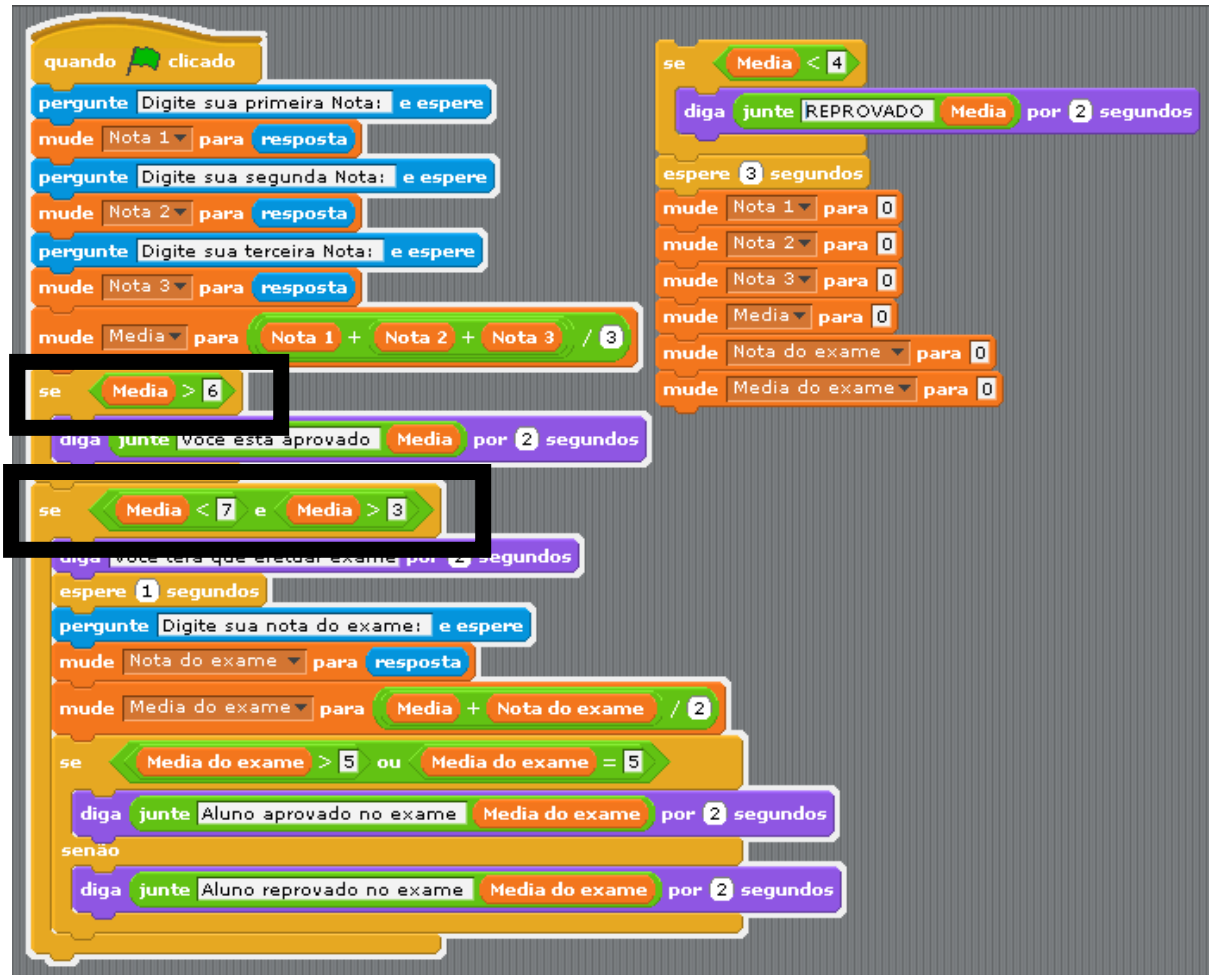
Fonte: A2 (2019).

No caso do A2, ao tratar da nota do exame, o mesmo utilizou a seguinte condição **(SE Média > 7)**, validando que se a nota referente ao exame inserida pelo usuário estivesse entre 5 e 6.9, apontaria que o usuário está REPROVADO, contudo a nota mínima exigida pela atividade para que o usuário seja

APROVADO no exame é 5. Destarte, para solucionar o problema, o A2 poderia ter utilizado as seguintes condições: **SE (Média < 5)**, ou poderia ter utilizado: **SE ((Média = 5) ou (Média > 5))**, deste modo o algoritmo teria funcionado perfeitamente, atendendo aos requisitos propostos pela atividade 2.

Já o A4 cometeu dois erros, os quais estão destacados por meio de retângulos com bordas pretas em destaque, apresentados na Figura 12.

Figura 12 – Algoritmo A4, atividade 2.



Fonte: A4 (2019).

O primeiro, ao categorizar se o usuário pegaria exame ou não, visto que A4 utilizou a seguinte condição para fazer a seleção: **SE (Média > 6)**. Considerando que a média necessária para que o usuário seja APROVADO diretamente é de no mínimo 7, com a condição desenvolvida por A4, o usuário seria aprovado com notas maiores que seis, descumprindo um objetivo determinado pela atividade 2. A estrutura correta a fim de selecionar o aluno seria: **SE ((Média = 7) ou**

(Média > 7)), assim o usuário só seria aprovado se obtivesse média maior ou igual a sete.

O segundo erro cometido por A4, está localizado na próxima estrutura de condição, onde o mesmo não soube utilizar a lógica de programação a fim de selecionar o usuário. A condição elaborada por A4 para selecionar se o usuário tinha direito de efetuar o exame ou não necessitava seguir a seguinte condição: Se a média do usuário for maior ou a quatro, ele tem o direito de efetuar o exame, mas se a média do usuário for menor que quatro, o mesmo não tem direito de efetuar o exame, sendo reprovado direto. A condição elaborada por A4 para fazer essa seleção foi a seguinte: **SE ((Média < 7) e (Média >3)),** assim sendo, se o usuário obtivesse a média maior que três, poderia efetuar o exame, o que não era permitido, levando em consideração que a média mínima é 4. A forma correta de utilizar tal condição a fim de seguir os objetivos propostos pela atividade seria: **SE ((Média < 7) e ((Média = 4) ou (Média > 4))**.

Os algoritmos desenvolvidos por A3 e A6, foram os únicos que atingiram todos os objetivos propostos pela Atividade 2. Ambos alunos souberam lidar com os pilares do PC, efetuando cada uma delas de maneira satisfatória, garantindo a solução da atividade.

Quadro 5 – Algoritmo A3 e A6, atividade 2.

Fonte: A3 e A6 (2019).

De acordo com Wing (2016), o aluno que domina a prática do PC consegue desenvolver o pensamento abstrato, pensamento algorítmico, pensamento lógico e o pensamento dimensionável. Assim, ele consegue decompor, abstrair, desenvolver padrões e elaborar algoritmos por meio de diferentes estruturas lógicas que possibilitam solucionar problemas (RAABE *et al.,* 2018).

De maneira geral, todos os alunos utilizaram as etapas do PC, até mesmo A1 e A5 que fizeram apenas uma parte da atividade. O problema foi que os mesmos não apresentaram conhecimento suficiente a fim de utilizar cada etapa de maneira lógica e eficaz, devido a falta do desenvolvimento e estímulo do PC, essa habilidade abrange inúmeras processos e etapas a serem seguidos, dentre elas, tomada de decisões, organização de informações até o desenvolvimento do algoritmo (WING, 2016).

Levando em consideração as etapas do PC, nota-se que A1 e A5, apresenta dificuldades em todas as etapas, visto que apenas iniciaram o algoritmo. A2 e A4 apresentaram falhas ao desenvolverem Padrões e ao efetuarem a Abstração dos dados. Já A3 e A6 conseguiram utilizar todas as etapas satisfatoriamente.

A seguir apresentou-se a análise dos dados coletados na Atividade 3.

4.3 Categoria 3 – Estruturas Lógicas de Repetição.

Nessa categoria, foram analisados os dados referentes as estruturas lógicas de repetição contidas no Módulo III. Para avaliar a capacidade dos alunos em utilizarem estruturas de repetição, os mesmos foram encarregados de solucionar a atividade exibida no Quadro 3.

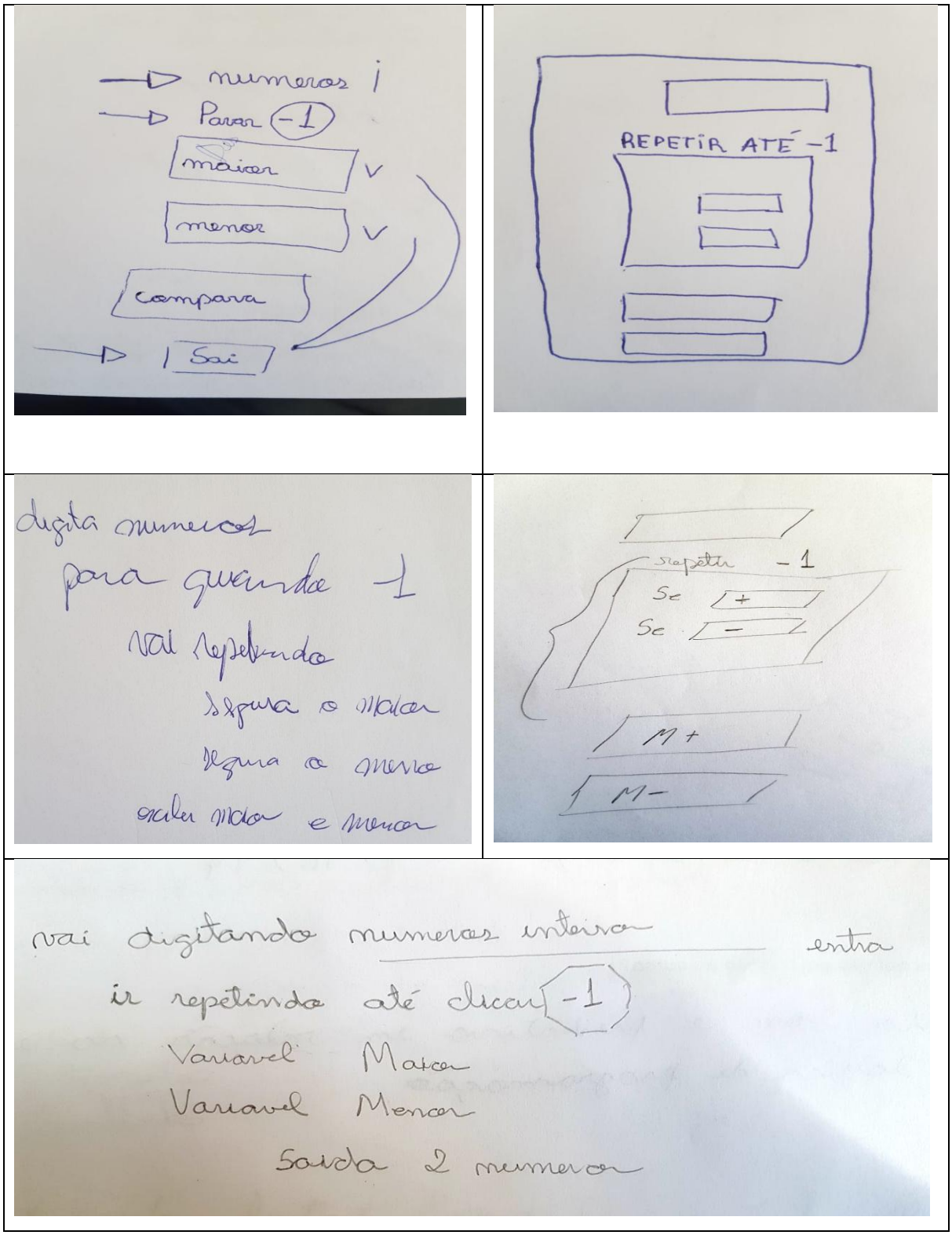
Quadro 6 - Atividade avaliativa do Módulo III.

Desenvolva um algoritmo no qual o usuário digite inúmeros números inteiros e ao digitar -1, o programa exibe na tela qual foi o maior e o menor número digitado.

Fonte: O autor (2019).

Ao explorar os dados gerados por meio da Atividade 3, constatou-se que alguns alunos utilizaram rascunho para esboçar a Decomposição da atividade, as quais foram replicadas por meio do Quadro 7.

Quadro 7 - Esboço de A1, A2, A3 A4 e A5 da atividade 3.



Fonte: A1, A2, A3, A4 e 5A (2019).

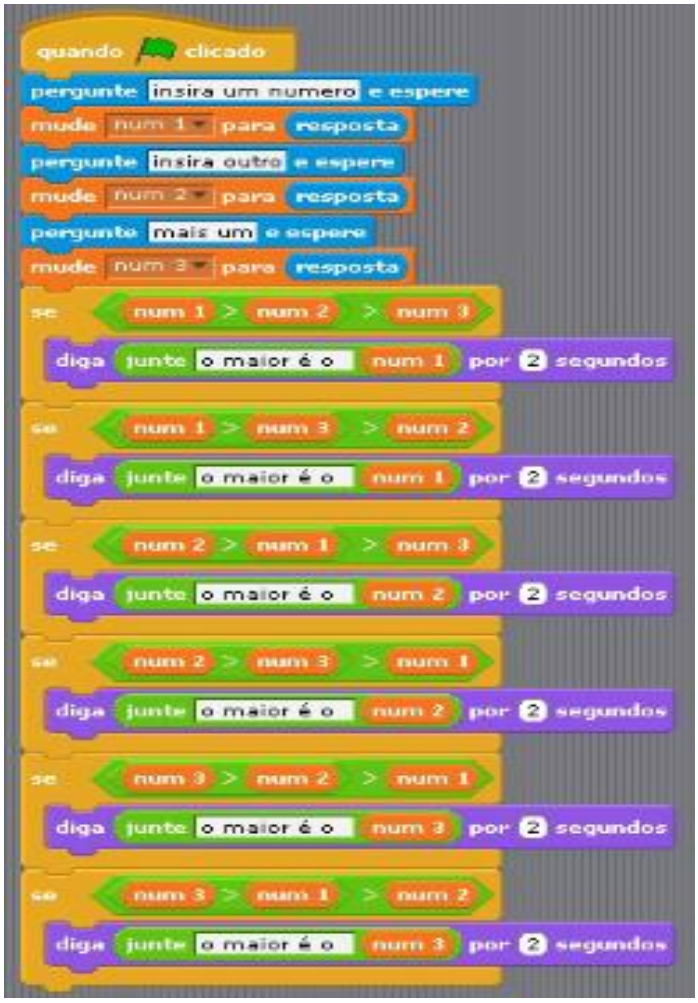
As imagens exibidas pelo Quadro 7 possibilitam afirmar que os alunos, com exceção do A6, já iniciaram os processos contidos no PC. Esses esboços

representados pelo Quadro 7 são fundamentais para que eles consigam criar um plano mental inicial, o qual possibilita auxiliar a transcrição do plano mental do papel, para o algoritmo utilizado a linguagem de programação por meio do *Scratch* (WING, 2016).

Desse modo, os esboços são uma forma de demonstrar tais planos de uma maneira superficial, menos detalhada, porém válida para organizar e particionar o problema. Uma das técnicas utilizadas pelos alunos para representarem o plano mental desenvolvido foi a utilização de retângulos para decompor, setas e linhas indicando a ordem lógica a ser seguida para o desenvolvimento de Padrões. Por isso, é nítida a utilização da estrutura de repetição, além do parâmetro utilizado para fazer a abstração da atividade.

A seguir foram exibidos os algoritmos desenvolvidos pelos alunos.

Figura 13 – Algoritmo A1, atividade 3.



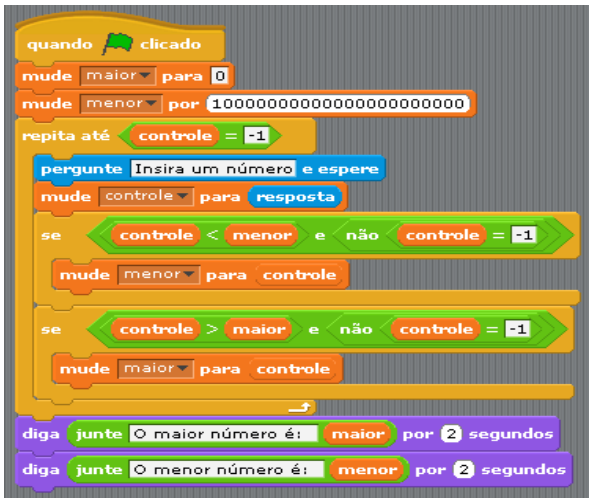
Fonte: A1 (2019).

O algoritmo desenvolvido por A1, representado pela Figura 13, apontou que o mesmo encontrou dificuldades e não conseguiu solucionar a atividade. Ao percorrer o código notou-se que o aluno não conseguiu utilizar uma estrutura de repetição, não conseguindo assim desenvolver padrões que trabalhassem com os dados, não conseguiu fazer a abstração dos dados a fim de eliminar informações desnecessárias.

O código apresentou uma tentativa de desenvolver um algoritmo utilizando estruturas de condição, porém, apenas a utilização dessas, não satisfazem a problemática proposta pela atividade. Outra fragilidade apresentada por A1 foi que ele não conseguiu desenvolver uma ordem lógica para trabalhar com as informações, ou seja, apresentando problemas já na etapa de decomposição dos dados. Vale ressaltar, que o esboço desenvolvido pelo aluno, representado no Quadro 7, aponta que o mesmo conseguiu efetuar um plano mental, conseguiu perceber que seria necessário utilizar uma estrutura de repetição, conseguiu esboçar alguns padrões e até mesmo uma ordem a ser seguida. Porém, não conseguiu transcrever o plano mental por meio de um algoritmo válido, o que possibilitou afirmar que o mesmo apresenta dificuldades em utilizar as habilidades do PC.

O código exibido pela Figura 14 desenvolvido por A2 também apresentou fragilidades, contudo, o mesmo conseguiu desenvolver a maior parte do algoritmo. O erro cometido por ele está relacionado a parte da lógica de programação, não sabendo organizar as estruturas lógicas de repetição e condição para solucionar o problema de acordo com os objetivos propostos pela atividade. O algoritmo apresentou falhas no tratamento das informações, ou seja, nos padrões estipulados, possibilitando que informações inseridas pelo usuário entrassem em variáveis errôneas, assim, exibindo valores indevidos.

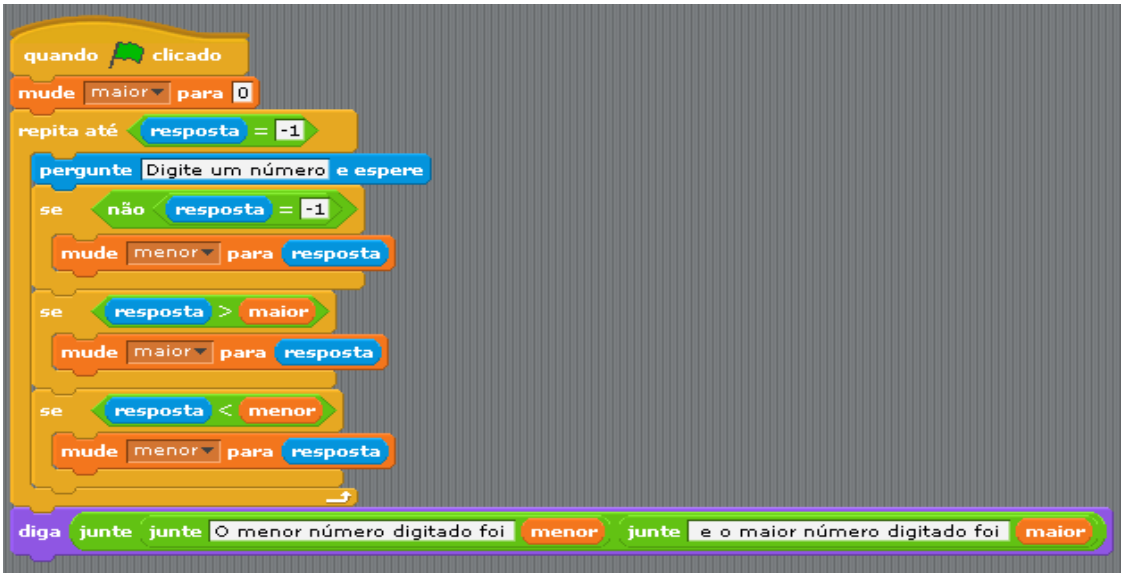
Figura 14 – Algoritmo A2, atividade 3.



Fonte: A2 (2019).

Os algoritmos desenvolvidos pelos alunos A3, A4, A5 e A6 (Figura 15 e 16) estão de acordo com os objetivos exigidos pela Atividade 3. Os mesmos solucionaram a atividade de forma correta, desenvolvendo o código utilizando as estratégias apontas pelo PC. Notou-se ainda, que os 4 alunos desenvolveram códigos semelhantes, mas utilizaram estruturas diferentes.

Figura 15 – Algoritmo A3 e A5, atividade 3.



Fonte: A3 e A5 (2019).

Esses alunos desenvolveram um algoritmo utilizando uma estrutura de repetição que continha 3 estruturas de condição. No caso de A4, exibido pela Figura 16.

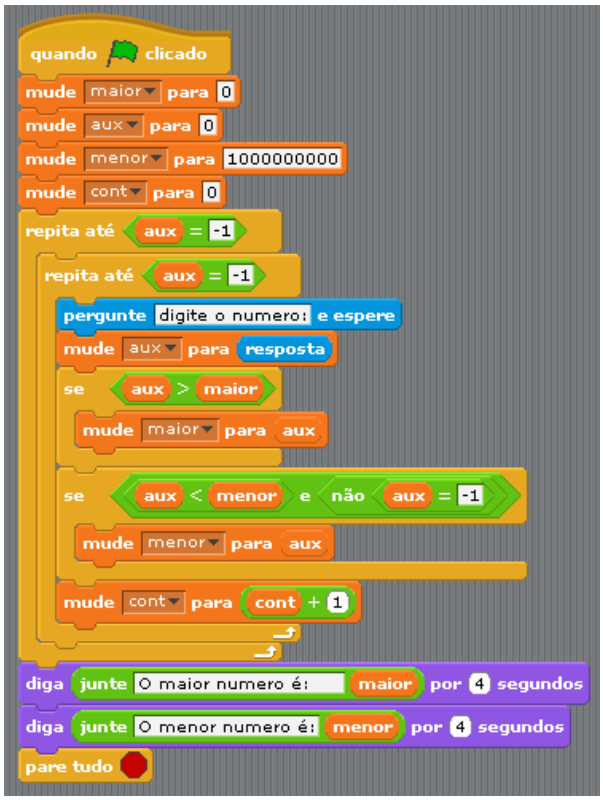
Figura 16 – Algoritmo A4, atividade 3.

Fonte: A4 (2019).

Utilizou-se de duas estruturas de repetição, uma de condição, e ainda utilizou uma lista, estrutura responsável por armazenar dados, o que o destacou dos demais. Visto que, ele utilizou uma estrutura que não foi trabalhada no curso, podendo afirmar que o mesmo explorou as funcionalidades e ferramentas que o *Scratch* oferece para solucionar problemas.

Já o algoritmo desenvolvido por A6 (Figura 17) utilizou duas estruturas de repetição, uma de condição e uma variável auxiliar a fim de armazenar informações temporárias.

Figura 17 – Algoritmo A6, atividade 3.



Fonte: A6 (2019).

Levando em consideração esse cenário, a ferramenta proporcionou liberdade de escolhas aos alunos, possibilitando que eles usufruíssem de diferentes caminhos lógicos para solucionar a problemática.

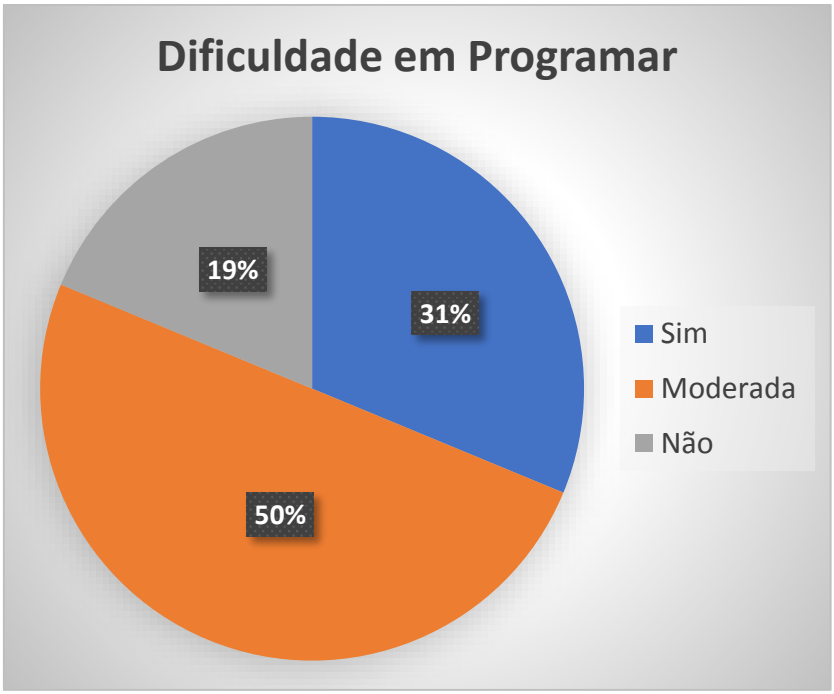
Diante dessas variedades de caminhos utilizados pelos alunos para solucionaram a atividade, a ferramenta ofereceu aos alunos a liberdade de escolhas e experimentação de comandos, podendo realizar inúmeras tentativas que poderão dar certo ou errado refletindo sobre o melhor caminho para solucionar o problema, assim, estimulando e desenvolvemento o Pensamento Computacional (TENÓRIO *et al.*, 2016).

4.4 Categoria 4 – Síntese Analítica.

Nessa categoria efetuou-se uma síntese analítica a fim de explorar os dados coletados dos alunos ao decorrer do curso por meio de questionários e as resoluções das atividades. Para enfatizar as dificuldades apontadas pelos alunos em

relação a disciplina de programação gerou-se um gráfico levando em consideração questionários respondidos por todos os alunos que participaram do curso.

Gráfico 1: Questionário relacionado as dificuldades encontradas na disciplina de programação.



Fonte: O autor (2019).

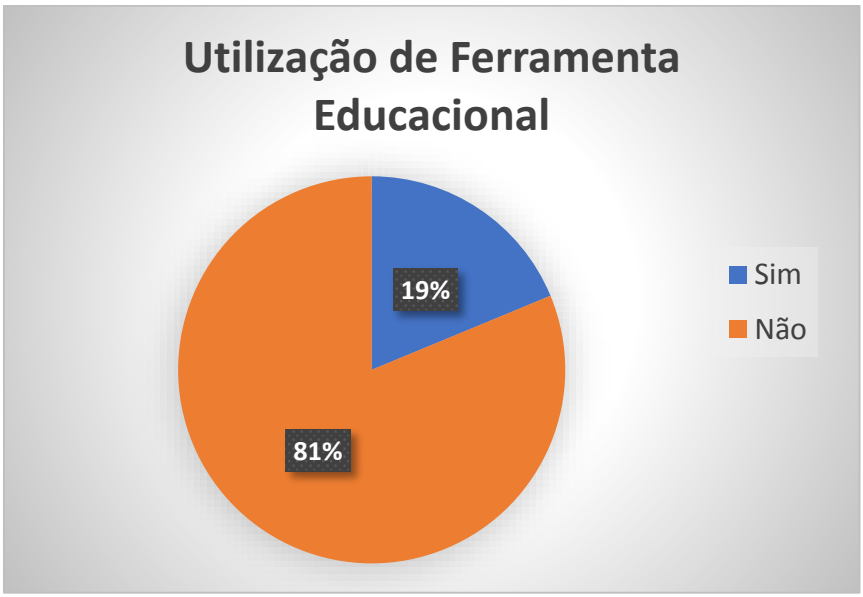
Levando em consideração o questionário aplicado, o gráfico aponta que apenas 19% dos alunos (3 alunos) não possuem dificuldades em programar, 50% (8 alunos) possuem dificuldades moderadas e encontram problemas ao desenvolverem algoritmos, por fim, 31% (5 alunos) afirmam não saber programar. Diante desse cenário fica nítida a dificuldade dos alunos ao se depararem com a disciplina de programação, visto que 81% (13 alunos) apresentam dificuldades em utilizarem a programação a fim de solucionarem problemas por meio de algoritmos.

Em consonância aos dados apresentados no Gráfico 1, Ventura (2018) aponta que a disciplina de Programação presente nos cursos da Área da Computação é uma das grandes dificuldades apontadas por alunos e pesquisadores. Sendo essa uma das disciplinas responsável pelo índice de até 75% de reprovação e desistências dos cursos da Área da Computação (ROCHA *et al*., 2010; PEREIRA *et al*., 2012). Nessa perspectiva se faz necessária a atenção e os cuidados necessários para introduzir a disciplina de programação no início do curso, almejando ensinar a

programação por meio de metodologias mais fáceis e intuitivas como no caso da utilização do *Scratch*.

Ainda levando em consideração o total de 16 alunos que iniciaram o curso desenvolvido por meio dessa pesquisa, efetuou-se um questionário com a seguinte pergunta: Você já utilizou a ferramenta *Scratch* ou outra ferramenta educacional que possibilite auxiliar a disciplina de programação? Sim ( ), Não ( ).

Gráfico 2: Questionário relacionado a utilização de ferramentas que auxiliem a disciplina de programação.



Fonte: o autor (2019).

De acordo com os dados exibidos no Gráfico 2, 19% (3 alunos) utilizaram o *Scratch* ou outra ferramenta que pode auxiliar o ensino de programação, e 81% (13 alunos) nunca utilizaram ferramentas educacionais com esse propósito.

A ausência de ferramentas educacionais em sala de aula pode ser entendida como desfavorável para o ensino de programação, já que tais ferramentas como o *Scratch*, possuem o objetivo de auxiliar o ensino de programação, oferecendo uma nova maneira de aprender, possibilitando minimizar as dificuldades encontradas na disciplina de programação (VENTURA, 2018).

Após os alunos conhecerem e utilizarem a ferramenta *Scratch* para desenvolver algumas atividades introdutórias como somar de números e criar variáveis, coletou-se algumas informações por meio de um questionário cuja pergunta foi: Qual seu ponto de vista em relação ao *Scratch*?

O Tabela 9, representa os excertos dos alunos, ressaltando que os mesmos foram descritos na íntegra, uma vez que não houve a correção gramatical.

Tabela 9 – Primeiras impressões dos alunos em relação a ferramenta.

A1	“Facilita a lógica, é intuitivo, dinâmico”.
A2	“Facilita maior entendimento de como é o pensamento computacional, com a possibilidade de visualizar o conteúdo das variáveis facilita o entendimento do processo e a resolução dos problemas”.
A3	“A ferramenta é bem didática, possui diversas opções para construir algoritimos. A interação também faz parte para melhor entendimento das operações”.
A4	“O *Scratch* é de muito mais fácil entendimento que uma linguagem de programação convencional. Permite trabalhar mais facilmente com imagens”.
A5	“A ferramenta *Scratch* me ajudou na programação com *Arduino* já que a ferramenta do *Arduino* na Wele é muito parecida com *Scratch*. Com o entendimento e as montagens de blocos em aula consegui desenvolver meus projetos com *Arduino*. Por ser mais intuitiva também o *Scratch* deixa a programação mais divertida”.
A6	Não respondeu.
A7	“Ela é uma ótima ferramenta introdutória para iniciantes em programação. É simples e ajuda a desenvolver o pensamento computacional”.
A8	“Auxilia no pensamento computacional, forma simples de entender a lógica das variáveis”.
A9	“Mais simples de programar; o programa é bem divertido; a programação com utilização de blocos é de grande auxilio no estudo da lógica”.
A10	“Bloco de código, português, comandos simples”.
A11	“É simples é intuitivo, de fácil entendimento para quem não está acostumado com a lógica de programação”.
A12	“Visualmente mais bonito. É de grande ajuda p/ quem está começando a programar. É simples de usar. É uma ferramenta bem lúdica, mostrando em blocos o que seria um código complicado e pouco explicativo”
A13	“Ferramenta fácil, e mais clara, e também de mais fácil compreensão”.
A14	“Ajuda a desenvolver a lógica de programação para aqueles que ainda não tem um grande conhecimento nos assuntos básicos”.
A15	Não respondeu.
A16	“Ferramenta funcional”.

Fonte: O autor (2019).

De modo geral, nota-se que os alunos tiveram uma visão positiva da ferramenta, apontando a mesma como funcional, intuitiva, lúdica e facilitadora no processo do entendimento da programação. Alem disso, A7 e A12 apontaram a ferramenta como auxiliadora na introdução do processo de programação, ou seja, podendo ser utilizada como ferramenta introdutória para o ensino de programação.

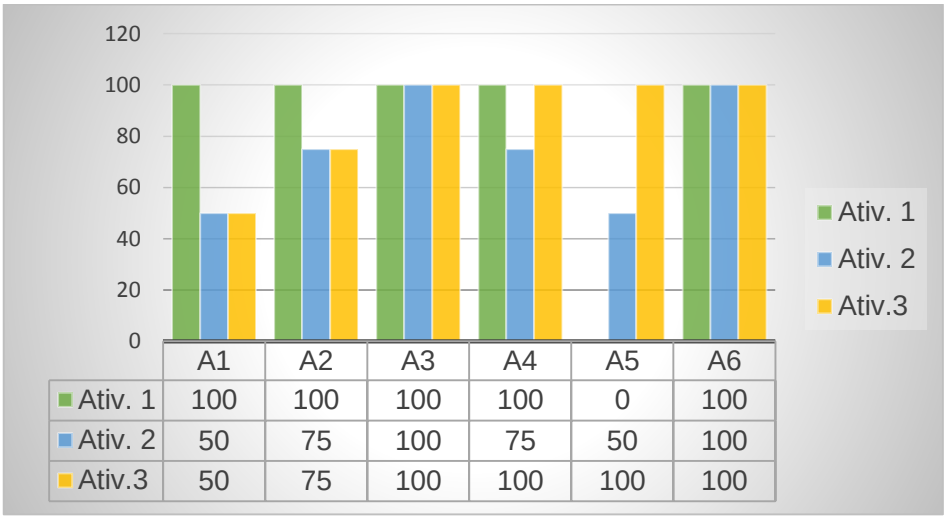
Nessa perspecitva, Valente (2016) relata que o *Scratch* é uma ferramenta que auxilia o processo de ensino de programação, visto que é uma ferramenta que aborda conceitos fundamentas para a aprendizagem de programação

inicial, podendo ser utilizada como uma forma alternativa para introduzir o ensino de programação em sala de aula.

Com base nas categorias elaboradas *a priori*, a seguir serão apresentadas as análises dos excertos dos participantes. Cabe ressaltar que, a partir desse momento apenas os 6 participantes selecionados irão passar pelo pressuposto da Análise de dados, levando em consideração os critérios de inclusão.

Ao considerar os resultados obtidos por meio das atividades 1, 2 e 3 foi possível notar, de modo geral, que os alunos demonstraram melhorias ao decorrer do curso. Para expor esses resultados foi gerado um gráfico indicando o desempenho dos alunos durante o curso, o qual foi apresentado abaixo.

Gráfico 3 – Desempenho individual dos alunos em relação às atividades avaliativas.



	A1	A2	A3	A4	A5	A6
Ativ. 1	100	100	100	100	0	100
Ativ. 2	50	75	100	75	50	100
Ativ.3	50	75	100	100	100	100

Fonte: O autor (2020).

O Gráfico 3 foi gerado por meio da verificação dos algoritmos desenvolvidos pelos alunos utilizando o *Scratch* e utilizou-se das seguintes pontuações: 0; 25; 50, 75 e 100 para avaliar e classificar as etapas concluídas de cada atividade. A pontuação 0 indica que o aluno não conseguiu desenvolver nenhuma linha de código válida para a atividade proposta. A pontuação 25 aponta que o aluno conseguiu apenas declarar variáveis e iniciar um esquema estrutural lógico. A pontuação 50 garante que o aluno conseguiu atribuir variáveis, utilizar estruturas lógicas, exibir resultados, entretanto não estavam satisfazendo os objetivos da atividade. No caso da pontuação 75, o aluno conseguiu desenvolver a maioria do algoritmo de forma válida, porém, apresentou alguns erros de raciocínio lógico e

utilização de parâmetros errados. Já a pontuação 100, indicou que o aluno conseguiu solucionar a problemática proposta pela atividade.

De acordo com os dados gerados pelo gráfico, nota-se que A1 regrediu em relação a primeira atividade. Entretanto, ao acompanhar o desenvolvimento dos algoritmos desenvolvidos por ele, nota-se que ele obteve melhoras em relação ao desenvolvimento dos algoritmos, nos quais ele conseguiu declarar variáveis, conseguiu utilizar estruturas de condição funcionais, criar padrões e efetuar abstrações, ainda sim, apresentando dificuldades em todas as etapas do PC.

No caso do A2 e A4, a regressão em relação a primeira atividade e as demais, demonstrou ser um caso de falta de atenção visto que os alunos apresentam o domínio de como desenvolver e estruturar o algoritmo a fim de solucionar problemas. Todavia, os mesmos não se atentaram aos objetivos propostos pela atividade, o que fez com que eles utilizassem dados errôneos, não fazendo uso correto do processo da abstração de dados, apontado por Raabe *et al*., (2018) como responsável por eliminar erros por meio das estruturas lógicas e padrões predefinidos.

O A5 apresentou uma melhora gradativa e extremamente satisfatória em relação ao desenvolvimento das atividades. No início do curso o aluno A5 apresentou muita dificuldade ao desenvolver algoritmos, mas seus esboços demonstraram que ele tinha conhecimento de como solucionar o problema utilizando lápis e folha de papel, faltando os saberes necessários para transformar tal pensamento em um algoritmo utilizando a linguagem de programação. Vale ressaltar que ele apresentava dificuldades até mesmo em declarar variáveis, o que é uma das operações mais simples e fundamentais para desenvolver um algoritmo (VENTURA, 2018). Contudo, ao decorrer do curso o mesmo conseguiu progredir desenvolvendo algoritmos válidos os quais seguiam os princípios da lógica de programação, assim, auxiliando no desenvolvendo do pensamento computacional por meio da ferramenta *Scratch*, a qual permitiu com que ele adquirisse conhecimentos de programação e solucionasse problemas por meio de algoritmos, como é possível notar na resolução da sua Atividade 3, representada pela Figura 15.

Já os dados apontados por A6, demonstraram que o mesmo não apresenta dificuldade em pensar computacionalmente e utilizar a lógica de programação para solucionar problemas. Ele atingiu o objetivo de todas as atividades, fazendo jus a utilização de cada etapa do PC de maneira satisfatória.

Para finalizar essa categoria. A seguir, por meio da Tabela 10 foram exibidas as respostas coletadas dos alunos, em relação a opinião dos mesmos a respeito do curso. Nesse caso, devido à variedade e importância das opiniões, optou-se por replicar a opinião de todos os alunos que concluíram o curso com o mínimo de 80% de frequência. Entretanto, a ordem dos alunos continuou a mesma, sendo assim, as opiniões de A1 á A6 são respectivamente dos alunos cujos os quais passaram pela análise e discussão de dados acima.

Tabela 10 – Opinião do curso.

A1	“O curso me ajudou muito em relação a lógica de programação consegui entender melhor os comandos”.
A2	“Muito bom e prestativo em relação ao entendimento e ensinamento da lógica de programação”.
A3	Não respondeu.
A4	“Foi muito bom para desenvolver atividades com a lógica de programação, *Scratch* é uma ótima ferramenta para quem está começando a programar, eu comecei fazendo pequenos jogos com ele, mas não sabia que muitas outras coisas poderiam ser feitas com esta ferramenta“.
A5	“Antes de entrar no curso eu não sabia nada de programação, e usar o *Scratch* como forma de ensinar e aprender a programar me ajudou muito”.
A6	“O curso ajuda a desenvolver o pensamento computacional para a programação para aqueles que ainda não tem o básico”.
A7	“O curso foi bem didático e objetivo, porém não estou satisfeito com a minha experiência, pois nos dois últimos exercícios passados em aula, não consegui encontra/ aplicar minha lógica na ferramenta”.
A8	“Tem sido um bom curso. Atende ao que eu esperava do curso”.
A9	“A curso foi muito bom no auxílio de programação, se tivesse desde o começo do ano aposto que teria ajudado muitos alunos com o raciocínio lógico. Curso muito divertido e interessante, é uma pena já ter acabado”.
A10	“Foi muito bom, a ferramenta é uma boa forma de aprender e ensinar a lógica de programação e essas aulas ajudaram bastante na disciplina de Prog1 pela sua maneira simples e fácil de se fazer o código”.
A11	“Muito bom, esclareceu muitas dúvidas minhas em relação a programação. “
A12	“Foi um curso simples, mas efetivo em relação ao aprendizado da ferramenta *Scratch*. As aulas e exercícios foram bem estruturados. Até pude desenvolver o pensamento computacional um pouco mais, graças a simplicidade, tive maior oportunidade de desenvolver soluções mais criativas para resolver os problemas”.
A13	“Como eu já estava avançado no curso de programação achei simples, mas se eu tivesse usado antes talvez ajudaria entender o conceito de variáveis e programação em geral. Gostei do curso e acho que ajudou o pensamento computacional”.
A14	“O curso é uma maneira mais fácil e de maneira mais prático o pensamento computacional. Utilizando a ferramenta *Scratch*, o aluno pode visualizar os processos devido ao mecanismo de blocos lógicos e dessa maneira entender melhor o conteúdo ensinado”.
A15	“Em geral o curso foi ótimo e sanou diversas duvidas de programação dentro da sala de aula, não só as minhas, mas a de outros colegas. Portanto creio que o professor concluiu o objetivo do curso e também creio que a implementação do

	mesmo nas aulas de Prog ajudaria no entendimento de novos alunos, ajudando assim a diminuir a evasão nos primeiros anos do curso”.

Fonte: O autor (2019).

As opiniões descritas acima contribuíram para a afirmação de que o presente trabalho atingiu seus objetivos, visto que de maneira geral, os alunos apontaram o curso como uma forma válida para ensinar programação por meio do *Scratch*, e também a escolha de tal ferramenta propicia o desenvolvimento do Pensamento Computacional e também a aprendizagem de programação, como afirma A5 em sua opinião: “Antes de entrar no curso eu não sabia nada de programação, e usar o *Scratch* como forma de ensinar e aprender a programar me ajudou muito”.

Nessa perspectiva, a utilização de tal ferramenta possibilita facilitar o entendimento da disciplina de programação, reduzir as dificuldades encontradas pelos alunos ao iniciarem a disciplina e consequentemente reduzir a grande evasão de alunos, apontada por A15.

A seguir apresenta-se as considerações finais em relação à pesquisa efetuada.

5. CONSIDERAÇÕES FINAIS

A sociedade atual utiliza das vantagens oferecidas pelas tecnologias para facilitar a execução de tarefas do cotidiano, seja para trabalhar, pesquisar, se divertir, estudar entre outras. Nessa perspectiva, as inovações tecnológicas ofertam melhorias em todas as áreas do conhecimento, bem como nas salas de aula, todavia, cabe ao professor utilizá-las a favor do ensino almejando potencializar esse processo.

Dessa forma, é imprescindível que os professores busquem novos conhecimentos e utilizem novas metodologias de ensino, como os softwares educacionais, os quais possibilitam auxiliar o processo de ensino, minimizando as dificuldades encontradas por alunos nas disciplinas complexas, como no caso de Programação.

Cabe ressaltar que a programação está presente nos cursos da área da Computação, e que seu ensino ocorre por níveis de complexidade, envolvendo cálculos matemáticos, estruturas lógicas como a de condição e de repetição. Desse modo é fundamental que o ensino de programação nos anos iniciais seja ofertado para os alunos da melhor forma possível, isso pois, o mesmo irá utilizá-la não somente na disciplina de programação e sim na maioria das outras disciplinas que compõem os cursos da Área da Computação.

Sobre essa questão, a presente pesquisa buscou investigar as dificuldades apontadas por alunos e pesquisadores em relação a disciplina de programação, sendo essa muitas vezes responsável pelo alto índice de evasão dos alunos dos cursos da Área da Computação.

Almejando minimizar tais dificuldades esse trabalho teve como finalidade utilizar a ferramenta *Scratch* como uma estratégia no desenvolvimento do Pensamento Computacional, e auxiliador para o processo de ensino de programação. Usufruindo das vantagens desse software educacional, a proposta foi de desenvolver um curso e implementar um caderno de atividades para que professores utilizem esse material para ensinar programação nos anos iniciais.

Por meio da pesquisa bibliográfica constatou-se uma carência de propostas voltadas a essa temática, assim o objetivo geral da pesquisa foi desenvolver um caderno de atividades para os professores que atuam nos anos iniciais do curso de Ciência da Computação da Universidade Estadual do Norte do Paraná – *Campus* de Bandeirantes, para o uso da ferramenta *Scratch* em sala de aula.

Nessa perspectiva, a pesquisa preocupou-se em identificar o conhecimento e as dificuldades apontadas pelos alunos do primeiro ano do curso de Ciência da Computação, em relação a disciplina de programação.

Ao decorrer do curso, verificou-se que os alunos apresentavam dificuldades de raciocínio lógico e em tomadas de decisões, ou seja, apresentavam dificuldades em pensar computacionalmente. Assim, os estudos que subsidiam essa pesquisa, apontam que a introdução da disciplina de programação faz-se fundamental para que o aluno consiga entender a complexidade de solucionar problemas por meio de linhas de códigos.

A ferramenta Scratch contribui com o ensino de programação, visto que é lúdica e simples de se utilizar, a qual apresenta apenas uma tela onde ocorre a programação Além do mais, a linguagem de programação por blocos facilita o desenvolvimento do pensamento computacional e do algoritmo, visto que o aluno tem acesso aos esquemas lógicos pré-definidos pela ferramenta, podendo organizá-los por cores, o que auxilia na organização da lógica de programação e a organizar o pensamento.

Nesse contexto, a pesquisa apresentou aos alunos a vantagem de utilizar o *Scratch*, possibilitando com que os mesmos elaborem, desenvolvam e solucionem atividades por meio da linguagem de programação por blocos lógicos que se encaixam. Desse modo, o desenvolvimento do curso, junto a implementação e execução das atividades, possibilitou mostrar uma nova alternativa para o ensino de programação.

Se tratando da análise dos dados obtidos por meio das atividades executadas pelos alunos, notou-se que a maioria não apresentou dificuldade ao elaborar um esquema mental para solucionar a atividade, visto que foram realizados por meio de desenhos e rascunhos, esquematizando uma proposta de solução para a atividade. Contudo, pode-se observar que ao transcrever esse esquema para um algoritmo que exige pensar computacionalmente, os participantes demonstraram dificuldades.

A análise dos resultados, junto a gravação da tela em tempo real, no momento em que os alunos estavam solucionando a atividade, evidenciaram que os mesmos apresentaram dificuldades em seguir os passos do Pensamento Computacional. Ao desenvolverem algoritmos, muitos apresentaram falhas principalmente no que tange a Abstração, e ou o Reconhecimento de Padrões, e ou a

Decomposição, e ou o Algoritmo. Assim, junto a essas etapas do Pensamento Computacional é que ocorrem os erros de lógica de Programação, estruturas de Condição e Repetição, junto a erros de cálculos entre outros. Em suma, os resultados da pesquisa revelam uma análise positiva para curso que foi desenvolvido acerca da temática, visto que as participantes deixam claro as dificuldades encontradas ao programarem.

No decorrem do curso, e ao analisar o desempenho individual de cada aluno, notou-se uma melhora significativa em todos os participantes, enfatizando os alunos que apresentaram dificuldades em solucionarem atividades no início do curso. Concluiu-se que os mesmos obtiveram melhora significativa na resolução das atividades, as quais possibilitaram afirmar que alunos que não dominavam a programação ao decorrer do curso conseguiram adquirir os conhecimentos necessários para solucionarem atividades utilizando o Pensamento Computacional, junto as estruturas lógicas exigidas pelos exercícios.

De maneira geral, o *Scratch* auxiliou os alunos que já dominavam a programação e principalmente aqueles que ainda não tinham o domínio da disciplina. Nessa perspectiva, alguns alunos relataram que a utilização da ferramenta *Scratch* como auxiliadora caso fosse trabalhada no início da disciplina de programação, podendo minimizar as dificuldades de aprender a programar e até mesmo minimizar a evasão do curso. Corroborando com esse relato, um dos participantes afirmou que o curso possibilitou com que ele aprendesse a programar, coisa que ele não havia aprendido antes do curso com o *Scratch*.

Fica evidente alguns fatores de limitação ao decorrer do curso, já que ferramenta *Scratch* é considerada introdutória para ensinar programação não contando com alguns comandos ou opções para agilizar o processo do desenvolvimento do código. Assim, os alunos que já possuíam o domínio de programação, apontaram a ferramenta como desatualizada. Pode-se ressaltar também, que alguns relataram dificuldades em encontrar a ferramenta que precisava para dar continuidade ao algoritmo. No mais, a ferramentas se mostrou promissora proporcionando o ensino de programação mais de maneira mais fácil e lúdica,

Tendo em vista, a importância da utilização do *Scratch* para o ensino introdutório de programação nos anos iniciais do curso de Ciência da Computação, como desdobramento da pesquisa no que tange o desenvolvimento do Pensamento Computacional, pode-se utilizar o *Scratch* para elaborar atividades de acordo com a

temática abordada na disciplina. Vale ressaltar que nesta pesquisa o *Scratch* viabilizou a elaboração do Caderno de Atividades voltados para a Lógica de Programação, Estruturas de Condição e Estruturas de Repetição, mas pode ser utilizado para abordar outros conteúdos introduzidos na disciplina, como por exemplo vetores, listas, matriz entre outros.

Em relação ao *Scratch* e seu potencial em desenvolver o Pensamento Computacional, a ferramenta mostra-se uma metodologia de ensino que pode auxiliar tanto o professor a ensinar quanto ao aluno aprender, mas para que isso ocorra é necessário que o professor inclua as ferramentas educacionais em sua metodologia de ensino.

Vislumbrando auxiliar o ensino de programação inicial, essa pesquisa desenvolveu o produto que pode ser utilizado por professores em qualquer curso que contenha a disciplina de programação, possibilitando auxiliar o processo de ensino dessa disciplina.

Cabe ressaltar que, a presente pesquisa contribuiu de forma efetiva para o aprofundar o meu conhecimento como pesquisador em relação ao desenvolvimento da Pensamento Computacional por meio da utilização do *Scratch* para ensinar programação. Ainda possibilitou compartilhar estudos científicos, atividades e experiências com professores e alunos do curso de Ciência da Computação.

Somado a isso, um quesito a ser lembrado é que ao compreender o Pensamento Computacional e utilizá-lo para programar, o aluno adquire habilidades como tomadas de decisão, elaboração e solução de problemas, não somente os que envolvem algoritmos, mas também aqueles encontrados no cotidiano dentro ou fora da faculdade, contribuindo não apenas para a conclusão do curso e sim para a execução e de tarefas fora da sala de aula.

Destarte, esta pesquisa buscou-se apresentar *Scratch* como uma nova estratégia de desenvolver o Pensamento Computacional, e ensinar programação visando minimizar as dificuldades apontadas pelos alunos ao se depararem com a disciplina de programação.

REFERÊNCIAS

ALBERTIN, A. L.; ALBERTIN, R. M. de M. Benefícios do uso de tecnologia de informação para o desempenho empresarial. **Revista de Administração Pública-RAP**, Rio de Janeiro, v. 42, n. 2, p. 275-302, 2008.

ALBUQUERQUE, A. B.; PONTES, L. B. Continuity and availability management: Case study: A hybrid model applied in databases services of a suplementary healt operator. *In:* IBERIAN CONFERENCE ON INFORMATION SYSTEMS AND TECHNOLOGIES, 11., 2016, Las Palmas. **Anais** [...]. Las palmas:IEEE, 2016, v.1, p. 1-4.

AMARAL, L; SILVA, G. B. E; PANTALEÃO, E. Plataforma Robocode como Ferramenta Lúdica de Ensino de Programação de Computadores - Pesquisa e Extensão Universitária em Escolas Públicas de Minas Gerais. *In:* SIMPÓSIO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 26, 2015, Maceió. **Anais** [...]. Maceió: SBC, 2015, v. 1, p. 200-208.

AMBRÓSIO, A. P.; COSTA, F. M. O uso de PBL para o Ensino de Algoritmos e Programação de Computadores. *In:* CONGRESSO INTERNACIONAL PBL, 6, 2010, São Paulo. **Anais** [...]. São Paulo: USP/PAN-ABP, 2010. v. 1, p. 2-11.

ARAÚJO, R. G. **Uma Investigação do uso da Ferramenta *SCRATCH* para o Ensino de Lógica de Programação no Ensino Médio**. 2016. 147f. Dissertação (Mestrado em Informática Aplicada) – Universidade de Fortaleza, Fortaleza, 2016.

BARBOSA, M. T. J.; BAISSO, M.; ALMEIDA, M. T. Surge uma nova Sociedade. *In:* SILVA, E. B. *et al*. **Automação & Sociedade:** Quarta Revolução Industrial, um olhar para o Brasil. Rio de Janeiro: Brasport, 2018, p. 3-11.

BITTENCOURT, R. A. *et al.* Aprendizagem de Programação Através de Ambientes Lúdicos em um Curso de Engenharia de Computação: Uma Primeira Incursão. *In:* CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 23, 2013, Maceió. **Anais**[…]. Maceió: SBC, 2013, v.1, p. 749–758.

BLIKSTEIN, P. **O pensamento computacional e a reinvenção do computador na educação**. [S. l.] 2008. Disponível em: http://www.blikstein.com/paulo/documents/online/ol_pensamento_computacional.html. Acesso em: 28 jan. 2018.

BRACKMANN, C. P. **Desenvolvimento do pensamento computacional através de atividades desplugadas na educação básica.** 2017. 226f. Tese (Doutorado em Informática da Educação) – Universidade Federal do Rio Grande do Sul, Porto Alegre, 2017.

BRENNAN, K.; RESNCIK, M.. New frameworks for studying and assessing the development of computational thinking. *In:* ANNUAL MEETING OF THE AMERICAN EDUCATIONAL RESEARCH ASSOCIATION, 2012, Vancouver. **Proceedings** […]. Vancouver: AERA, 2012. Disponível em: https://web.media.mit.edu/~kbrennan/files/Brennan_Resnick_AERA2012_CT.pdf.

Acesso em: 23 Mai 2020.

BRESSAN, M.; AMARAL**,** M. Avaliando a contribuição do *Scratch* para a aprendizagem pela solução de problemas e o desenvolvimento do pensamento criativo. **Revista Intersaberes***,* Curitiba, v. 10, n. 21, p. 509-526, 2015.

COSTA, S. S. F.; PESSÔA, M. T. R.; GOMES, A. J. Pensamento Computacional e educação: uma experiência com *Scratch* no primeiro ciclo do ensino básico. *In:* YAEGASHI, S. F. R. *et al.* (Org.). **Novas Tecnologias Digitais**: Reflexões, sobre mediação, aprendizagem e desenvolvimento. Curitiba: CRV, 2017. p. 271-286.

CRESWELL, J. **Qualitative inquiry and research design**: Choosing among five approaches. 3. ed. Lincoln: SAGE, 2013. 448p.

DIAS, K. L.; SERRAO, M. S. A Linguagem *Scratch* no Ensino de Programação: Um Relato de Experiência com Alunos Iniciantes do Curso de Licenciatura em Computação. *In:* CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 22, 2014, Brasília. **Anais** [...]. Brasília: SBC, 2014, p. 1475-1484.

DIESEL, A.; BALDEZ, A. L. S; MARTINS, S. N. Os princípios das metodologias ativas de ensino: uma abordagem teórica. **Revista Thema**, Lajeado, v. 14, n. 1, p. 268-288, 2017.

FASSBINDER, A. G. O.; DE PAULA, L. C.; ARAÚJO, J. C. D. Experiências no estímulo à prática de Programação através do desenvolvimento de atividades extracurriculares relacionadas com as competições de conhecimentos. *In:* CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 22, 2012, Curitiba. **Anais** [...]. Curitiba: SBC, 2012, p. 1- 4.

FIGUEIREDO, H. R. S.; CARVALHO, D. F.; MILANI, M. L. C. Os jogos digitais no desenvolvimento de habilidades matemáticas: Minecraft, Pokémon Go e softwares. *In:* YAEGASHI, S. F. R. *et al.* (Org.). **Novas Tecnologias Digitais**: Reflexões, sobre mediação, aprendizagem e desenvolvimento. Curitiba: CRV, 2017, p. 235-249.

FLICK, U. **Introdução à pesquisa qualitativa**. 3 ed. Porto Alegre: Artmed, 2009. Disponível em: https://www.ets.ufpb.br/pdf/2013/2%20Metodos%20quantitat%20e%20qualitat%20-%20IFES/Bauman,%20Bourdieu,%20Elias/Livros%20de%20Metodologia/Flick%20-%20Introducao%20%C3%A0%20Metodologia%20da%20Pesquisa.pdf. Acesso: 10 jun. 2019

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. **Lógica de programação**: a construção de algoritmos e estruturas de dados. 3. ed. São Paulo: Pearson, 2005, 232 p.

FURBER, S. **Shut down or restart? The way forward for computing in UK schools**. Londres: The Royal Society, 2012.

GERHARD, T. E.; SILVEIRA, D. T. **Método de Pesquisa**. Coordenado pela Universidade Aberta do Brasil – UAB/UFRGS e pelo curso de Graduação

Tecnológica – Planejamento e Gestão para o Desenvolvimento Rural da SEAD/UFRGS – Porto Alegre: Editora da UFRGS, 2009, p. 43-64. Disponível em: http://www.ufrgs.br/cursopgdr/downloadsSerie/derad005.pdf. Acesso em: 15 maio 2019

GOMES T. C. S; MELO J. C. B. O Pensamento Computacional no Ensino Médio: Uma Abordagem Blended Learning. *In:* CONGRESSO DA SOCIEDADE BRASILEIRA DE COMPUTAÇÃO, 23, 2013, Maceió. **Anais** […]. Maceió: SBC,2013. p. 651- 660.

GROVER, S.; PEA, R. Computational thinking in k–12: A review of the state of the field. **Educational researcher**, Stanford, v. 42, n. 1, p. 38–43, 2013.

INÁCIO, F. A. J. **Ensino de Algoritmos e Lógica de Programação**: modelo Construtivista auxiliado pelo *Scratch*. 2016. 114 f. Dissertação (Mestrado Profissional em Educação Tecnológica) – Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro, Uberaba, 2016.

ISTE (International Society for Techonology in Education). **Operational definition od computational thinking for K-12 education.** 2011. Disponível em: www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf. Acesso em: 25 Mai 2020.

KAFAI, Y. B. BURKE, Q. Computer Programming Goes Back to School. **Education Week**, California, v. 95, n. 1, p.61-65, 2013.

LAKATOS, E. M.; MARCONI, M. de A. **Técnicas de pesquisa**. 5. ed. São Paulo: Atlas. 2010. 278p.

LIMA, J. A. **Níveis de apropriação das TDIC pelos professores**. 2019. 146f. Dissertação (Mestrado em Educação) – Pontifícia Universidade Católica de São Paulo, São Paulo, 2019.

LIUKAS, L. **Hello Ruby**: adventures in coding. Crawfordsville: Feiwl & Friends , 2015.

LONG, J. Just for fun: using programming games in software programming training and education – a field study of ibm robocode community. **Journal of Information Technology Education**, San Marcos, v. 6, p-279-290, 2007.

LOPES, L. M. M.; RIBEIRO, V. S. O Estudante como protagonista da aprendizagem em ambientes inovadores de ensino*. In:* CONGRESSO INTERNACIONAL DE EDUCAÇÃO E TECNOLOGIAS e ENCONTRO DE PESQUISADORES EM EDUCAÇÃO Á DISTÂNCIA, 4, 2018, São Carlos. **Anais** [...]. São Carlos: UFSCAR, 2018. p. 1-7.

MACEDO, L. (Org.). **Jogos, psicologia e educação**: teoria e pesquisas. São Paulo: Casa do Psicólogo, 2009. 270p.

MACEDO, L.; PETTY, A. L. S.; PASSOS, N. C. **Os Jogos e o Lúdico na Aprendizagem Escolar**. Porto Alegre: Artmed, 2007. 110p.

MALONEY, J. *et al.* The *Scratch* programming language and environment. **ACM Transactions on Computing Education**, New York, v. 10, n. 4, p.1-15.

MALTEMPI, M. V.; VALENTE, J. A. Melhorando e Diversificando a Aprendizagem via Programação de Computadores. *In:* INTERNATIONAL CONFERENCE ON ENGINEERING AND COMPUTER EDUCATION, 2, 2000, São Paulo. **Anais** [...]. São Paulo: COPEC, 2000. (CD Room). Disponível em: <http://www.rc.unesp.br/igce/demac/maltempi/Publicacao/Maltempi-Valente-icece.pdf>. Acesso em: 19 jul. 2018.

MARTINS, A. R. Q. **Usando o *Scratch* para potencializar o pensamento criativo em crianças do ensino fundamental**. 2012. 113f. Dissertação (Mestrado em Educação) – Universidade de Passo Fundo, Passo Fundo, 2012.

MESTRE, P. *et al.* Pensamento Computacional: Um estudo empírico sobre as questões de matemática do PISA. *In:* CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 4, 2015, Maceió. **Anais** [...]. Maceió: SBC, 2015. p. 1281- 1289.

OECD. **PISA (Programme for International Student Assessment)**. 2018. Disponível em: http://www.oecd.org/pisa/. Acesso em: 20 nov. 2018.

OLIVEIRA, M. L. S. *et al*. Ensino de lógica de programação no ensino fundamental utilizando o *Scratch*: um relato de experiência. *In:* Congresso da Sociedade Brasileira de Computação, 34, 2014, Brasília. **Anais** [...]. Brasília: SBC, 2014. p. 1525-1534.

PEREIRA, A. C.; FRANCO, M. E. Desenvolvendo o pensamento computacional no ensino fundamental com Arduino e *Scratch*. *In:* ENCONTRO NACIONAL DE COMPUTAÇÃO DOS INSTITUTOS FEDERAIS, 5, 2018, Natal. **Anais** [..]. Natal: SBC, 2018, p. 1-4.

PEREIRA, P. S. *et al.* Análise do *SCRATCH* como ferramenta de auxílio ao ensino de programação de computadores. *In:* CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, 40, 2012, Belém**. Anais** [...]. Belém: ABENGE, 2012. p.1-9.

PONTE, J. P. *et al.* **Programa de matemática do ensino básico**. Lisboa: DGIDC, 2007.

RAABE, A.; BRACKMANN, C.; CAMPOS, F. **Currículo de Referência em Tecnologia e Computação**: da Educação Infantil ao Ensino Fundamental. São Paulo: CIEB, 2018. E-book. Disponível em: http://curriculo.cieb.net.br/assets/docs/Curriculo_de_Referencia_em_Tecnologia_e_Computacao.pdf. Acesso em: 20 Nov 2018.

RAMOS, H. A. **Pensamento Computacional na Educação Básica**: uma proposta de aplicação pedagógica para alunos do quinto ano do Ensino Fundamental do

Distrito Federal. 2014. 62f. Trabalho de Conclusão de Curso (Licenciatura em Computação) - Universidade de Brasília, Brasília, 2014.

RESNICK, M. Fulfilling Papert's Dream: "Computational Fluency for All". *In*: Technical Symposium on Computer Science Education, 5, 2017, Seattle. **Anais** […]. Seattle: SIGCSE, 2017

RESNICK, M. Reviving Papert's Dream. **Educational Technology**, Londres, v. 52, n. 4, p. 42-46, 2012.

REZENDE, C. M. C.; BISPO, E. L. Comparação entre o Uso de Pseudocódigo e a Programação Visual no Ensino de Programação: Uma avaliação a partir da ferramenta *Scratch*. *In:* Conferência Ibérica de Sistemas e Tecnologias de informação, 13, 2018, Caceres. **Anais** [...]. Caceres: CISTI, 2018, p. 1-5.

RIBEIRO, A. S. M.; RODRIGUES, F. B. V.; PEREIRA, V. M. S. Conhecendo o *Scratch* e suas potencialidades pedagógicas. *In:* SEMINÁRIO INTERNACIONAL DE INCLUSÃO ESCOLA: práticas em diálogo, 1, 2014, Rio de Janeiro. **Anais** [...]. Rio de Janeiro: CAp-UERJ, 2014. p. 1-6.

ROCHA, P. S. *et al.* Ensino e aprendizagem de programação: Análise da aplicação de proposta metodológica baseada no Sistema Personalizado de Ensino. **RENOTE**, Porto Alegre, v. 8, n. 3, p. 1-11, 2010.

RODRIGUES, A.; DE ALMEIDA, M. E. B.; VALENTE, J. A. Currículo, narrativas digitais e formação de professores: Experiências da pós-graduação à escola. **Revista Portuguesa de Educação**, Portugal, v. 30, n. 1, p. 61, 2017.

SANTOS, C. S. *et al.* Aprendendo Programação Orientada a Objetos com uma Abordagem Lúdica Baseada em Greenfoot e Robocode*. In:* CONGRESSO BRASILEIRO DE EDUCAÇÃO EM ENGENHARIA, 43, 2015, São Bernardo do Campo. **Anais** [...]. São Bernardo do Campo: ABENGE, 2015. p. 1-10.

SCAICO, P. D. *et al.* Programação no ensino médio: uma abordagem de ensino orientado ao design com *Scratch*. *In*: CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 23, 2012, Rio de Janeiro. **Anais** [...]. Rio de Janeiro: SBC, 2012, p. 1-10.

SCRATCH. **Imagine, program, share**. 2017. Disponível em: http://*Scratch*.mit.edu. Aceso em: 11 Jun 2019.

SILVA, J. M. **Estimulando o pensamento computacional com jogos digitais: uma abordagem utilizando *Scratch***. 2018. 129f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Software) – Universidade Federal do Rio Grande do Norte, Natal, 2018.

SOUZA, C. M. VisuAlg - Ferramenta de apoio ao ensino de programação. **Revista Eletrônica TECCEN**, Vassouras, v. 2, n. 2, p. 1-9, 2009.

TENÓRIO, M. M. *et al.* Conteúdos Matemáticos: Propostas com a Aplicação do *Scratch*. **Conexões-Ciência e Tecnologia**, Fortaleza, v. 10, n. 4, p. 60-70, 2016.

TRENTIN, M. A. S.; SHITSUKA, R; TEIXEIRA, A. C. Programação de computadores como uma alternativa ao modelo metodológico padrão da apropriação da informática em processos educativos. **Revista Espaço Pedagógico**, Passo Fundo, v. 26, n. 2 p. 395-409. 2019.

UNIPAMPA. **Percentual anual de evasão dos cursos**. GURI - Gestão Unificada de Recursos Institucionais/Módulo: Portal do Aluno. Pampa: PROGRAD, 2018. Disponível em: https://sites.unipampa.edu.br/prograd/files/2019/09/percentual-anual-de-evasao-dos-cursos-2018.pdf. Acesso em 28 Jan. 2020

VALENTE, J. A. Integração do Pensamento Computacional no currículo da Educação Básica: diferentes estratégias usadas e questões de formação de professores e avaliação do aluno. **e-Curriculum**, São Paulo, v. 14, n. 3, p. 864-897, 2016.

VENTURA, L. M. **A lógica de programação e os jogos digitais**: uma experiência com a ferramenta *SCRATCH*. 2018. 109f. Dissertação (Mestrado em Metodologias para o Ensino de Linguagens e suas Tecnologias) – Universidade do Norte do Paraná, Londrina, 2018.

WING, J. M. Computational thinking. **Communications of the ACM**, New York, v. 49, n. 3, p. 33-35, 2006.

WING, J. Pensamento Computacional: Um conjunto de atitudes e habilidades que todos, não só cientistas da computação, ficaram ansiosos para aprender e usar. **Revista Brasileira de Ensino de Ciência e Tecnologia**, Ponta Grossa, v. 9, n. 2 , p. 1-10, 2016.

ZAHARIJA, G.; MLADENOVIC, S.; BOLJAT, I. Introducing basic Programming Concepts to Elementary School Children. **Procedia - Social and Behavioral Sciences**, Turkey, v. 106, p. 1576-1584, 2013.

ZANETTI, H. A. P.; OLIVEIRA, C. L. V. Prática de ensino de Programação de computadores com Robótica Pedagógica e aplicação de Pensamento Computacional. *In:* CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, 4, 2015, Maceió. **Anais** [...]. Maceió: SBC, 2015. p. 1236-1245.

APÊNDICE A

UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
***Campus* Bandeirantes**

PROGRAMA DE PÓS-GRADUAÇÃO EM ENSINO
MESTRADO PROFISSIONAL EM ENSINO

Mestrando: Paulo Roberto Anastacio Data: ___/___/___

Orientador: Rudolph dos Santos Gomes Pereira

Participante: _____

Questionário.

1) Você já utilizou o Scratch ou outra ferramenta a fim de auxiliar a aprendizagem de programação??

_____
_____
_____

2) Qual sua primeira impressão em relação ao Scratch?

_____
_____

3) Qual sua percepção sobre o pensamento computacional no Scratch?

_____
_____

4) O que é Pensamento Computacional?

_____
_____
_____

5) A ferramenta Scratch oferece subsídios necessários para solucionar problemas?

_____
_____
_____
_____
_____
_____

6) Qual sua primeira impressão em relação ao Scratch?

_____
_____
_____

7) Qual sua dificuldade na disciplina de programação?

_____
_____
_____
_____

8) Você já fez alguma formação continuada que com a temática de Tecnologia Digital? Comente.

_____
_____
_____
_____

9) Qual seu ponto de vista em relação ao Scratch? Pontos positivos e negativos?

_____
_____
_____
_____

10) A utilização do Scratch contribuiu em relação aos conteúdos de lógica de programação, Estruturas de Decisão e repetição? Justifique

_____
_____
_____
_____

APÊNDICE B

UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ
***Campus* Bandeirantes**

PROGRAMA DE PÓS-GRADUAÇÃO EM ENSINO
MESTRADO PROFISSIONAL EM ENSINO

IDENTIFICAÇÃO E TERMO DE CONSENTIMENTO

1- Dados pessoais do entrevistado (Não serão divulgados. Servirão apenas para o esclarecimento de eventuais dúvidas por parte do pesquisador).	
Nome:	
Endereço:	
Idade:	Série:
Telefone:	

2- Consentimento

Prezado (a) aluno (a), você está sendo convidado a participar da pesquisa: "**O USO DO SCRATCH NA APRENDIZAGEM DE PROGRAMAÇÃO**" que tem por objetivo **AUXILIAR A APRENDIZAGEM DE PROGRAMAÇÃO**.

Essa pesquisa será realizada apenas **ALUNOS DO PRIMEIRO ANO DO CURSO DE CIÊNCIA DA COMPUTAÇÃO DA UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ – UENP, CAMPUS DE BANDEIRANTES/PR**

Sua participação no estudo consistirá em **RESPONDER ALGUMAS QUESTÕES, PARTICIPAR DE PESQUISA DE OPINIÃO E DESENVOLVER EXERCÍCIOS** SOBRE **A FERRAMENTA SCRATCH E MATÉRIAS VOLTADOS PARA LÓGICA DE PROGRAMAÇÃO, ESTRUTURAS LÓGICAS DE CONDIÇÃO E REPETIÇÃO.** O curso terá uma duração de 20 horas sendo dividida em 10 dias de 2 horas/aulas.

Você tem a liberdade de não participar da pesquisa ou retirar seu consentimento a qualquer momento, mesmo após o início do curso, sem qualquer prejuízo. Está assegurada a garantia do sigilo das suas informações. Você não terá nenhuma despesa e não há compensação financeira relacionada à sua participação na pesquisa.

Caso tenha alguma dúvida sobre a pesquisa, poderá entrar em contato com o coordenador responsável pelo estudo: **PAULO ROBERTO ANASTACIO.** Sua participação é importante e voluntária e vai gerar informações que serão úteis para **AUXILIAR ALUNOS COM DIFICULDADES NA APRENDIZAGEM DE PROGRAMAÇÃO**.

Acredito ter sido suficientemente informado a respeito do que li ou foi lido para mim, sobre a pesquisa: "O USO DO SCRATCH NA APRENDIZAGEM DE PROGRAMAÇÃO ". Discuti com o pesquisador PAULO ROBERTO ANASTACIO, responsável pela pesquisa, sobre minha decisão em participar do estudo. Ficaram claros para mim os propósitos do estudo, os procedimentos, garantias de sigilo, de esclarecimentos permanentes e isenção de despesas. Concordo voluntariamente em participar deste estudo.

_____ ____/____/____

Assinatura do aluno

Declaro que obtive de forma apropriada e voluntária o Termo de Consentimento Livre e Esclarecido deste entrevistado para a sua participação neste estudo.

_____ ____/____/____

Assinatura do responsável pela pesquisa.