

UNIVERSIDADE ESTADUAL DO PARANÁ - UNESPAR

PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO

UM ESTUDO SOBRE MOBILIZAÇÃO DE
PROCESSOS DE PENSAMENTO
COMPUTACIONAL A PARTIR DE
QUESTÕES DO ENEM

NOELI TERESINHA VALÉRIO DE ALMEIDA

Programa de Pós-Graduação em Educação Matemática
PRPGEM

União da Vitória
2026



UNIVERSIDADE ESTADUAL DO PARANÁ - UNESPAR
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM EDUCAÇÃO MATEMÁTICA

UM ESTUDO SOBRE MOBILIZAÇÃO DE PROCESSOS DE
PENSAMENTO COMPUTACIONAL A PARTIR DE QUESTÕES DO
ENEM

Noeli Teresinha Valério de Almeida

Orientador: Sérgio Carrazedo Dantas

Dissertação apresentada ao Curso de Mestrado do Programa de Pós-Graduação em Educação Matemática da Universidade Estadual do Paraná - Unespar, linha de pesquisa Tecnologia, diversidade e cultura em educação matemática, como parte dos requisitos necessários à obtenção do título de Mestre em Educação Matemática.

Noeli Teresinha Valério de Almeida

UM ESTUDO SOBRE MOBILIZAÇÃO DE PROCESSOS DE PENSAMENTO
COMPUTACIONAL A PARTIR DE QUESTÕES DO ENEM

Comissão Examinadora:

Documento assinado digitalmente
 **SERGIO CARRAZEDO DANTAS**
Data: 29/06/2026 11:34:33-0300
Verifique em <https://validar.iti.gov.br>

Sérgio Carrazedo Dantas
Presidente da Comissão Examinadora
Universidade Estadual do Paraná - Unespar

Documento assinado digitalmente
 **ADRIANA HELENA BORSOI**
Data: 29/06/2026 18:52:08-0300
Verifique em <https://validar.iti.gov.br>

Adriana Helena Borssoi – Membro da banca
Universidade Tecnológica Federal do Paraná - UTFPR

Documento assinado digitalmente
 **PAULO HENRIQUE RODRIGUES**
Data: 29/06/2026 18:20:10-0300
Verifique em <https://validar.iti.gov.br>

Paulo Henrique Rodrigues – Membro da banca
Universidade Estadual do Paraná - Unespar

Resultado: APROVADA

União da Vitória
Abril de 2026

Dedico ao meu esposo, Vilmar (in memoriam).

AGRADECIMENTOS

Agradeço, primeiramente, a Deus, pela força e perseverança concedidas ao longo desta jornada.

Ao meu esposo, Vilmar (in memorian), cujo amor, incentivo e presença continuam sendo fonte de força e inspiração em minha vida. Sua memória permanece viva em cada conquista, inclusive nesta.

À minha família, pelo apoio incondicional, incentivo constante e compreensão nos momentos de ausência. Sem vocês, este caminho teria sido muito mais difícil. Ao meu orientador, professor Dr. Sérgio Carrazedo Dantas, pela orientação segura, paciência, dedicação e valiosas contribuições ao longo de todo o desenvolvimento desta pesquisa.

Aos membros da banca, Professora Doutora Adriana Helena Borssoi, pelas contribuições; ao Dr. Paulo Henrique Rodrigues, pelas valiosas contribuições na qualificação e na defesa e por aceitarem compor a banca, enriquecendo esta pesquisa com suas sugestões.

Aos integrantes do grupo de pesquisa Autômato, pela parceria, trocas de conhecimento e apoio mútuo durante essa trajetória.

Ao meu primeiro amigo do mestrado, Luan Padilha, pelo incentivo constante. A minha amiga, Me. Suzana Ap. do Prado, por sua generosa contribuição no processo de construção e revisão deste trabalho.

Por fim, agradeço a todos que, direta ou indiretamente, contribuíram para a realização deste trabalho.

RESUMO

Esta pesquisa investiga a mobilização dos processos constituintes do Pensamento Computacional na resolução de questões de Matemática do Exame Nacional do Ensino Médio (ENEM) e tem por objetivo de demonstrar como os processos de formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração; podem ser mobilizados na resolução dessas questões por meio do Scratch. O estudo desenvolve-se a partir da resolução de questões selecionadas em dois momentos distintos: inicialmente, com base em estratégias fundamentadas no conhecimento matemático em seguida, mediante a construção de modelos computacionais no Scratch, com a explicitação sistemática dos processos envolvidos. O corpus de análise é composto pelos arquivos produzidos no Scratch, que materializam as resoluções em estruturas algorítmicas, e por registros em vídeos do processo de elaboração, os quais possibilitam identificar indícios dos processos mobilizados. A análise contempla a identificação, a descrição e a distinção desses processos, buscando compreender suas funções e articulações em diferentes situações problema. Os resultados indicam que a modelagem computacional contribui para a explicitação das estruturas de resolução, amplia a organização estratégica do raciocínio e torna visíveis etapas do processo de construção da solução. Conclui-se que a integração do Pensamento Computacional ao trabalhar com questões do ENEM apresenta potencial pedagógico para a Educação Matemática na Educação Básica.

Palavras-chave: Pensamento Computacional; Educação Matemática; Resolução de Problemas; ENEM; Scratch.

ABSTRACT

This research investigates the mobilization of the constituent processes of Computational Thinking in solving Mathematics questions from the National High School Exam (ENEM) and aims to demonstrate how the processes problem formulation, decomposition, pattern recognition, abstraction, algorithm production, and debugging can be mobilized in solving these questions using Scratch. The study develops from the resolution of selected questions in two distinct moments: initially, based on strategies grounded in mathematical knowledge; subsequently, through the construction of computational models in Scratch, with the systematic explication of the processes involved. The corpus of analysis is composed of the files produced in Scratch, which materialize the solutions in algorithmic structures, and video recordings of the elaboration process, which make it possible to identify evidence of the mobilized processes. The analysis includes the identification, description, and distinction of these processes, seeking to understand their functions and articulations in different problem situations. The results show that computational modeling contributes to the explication of the resolution structures, expands the strategic organization of reasoning, and makes visible the stages of the solution construction process. It is concluded that integrating Computational Thinking when working with ENEM questions presents pedagogical potential for Mathematics Education in Basic Education.

Keywords: Computational Thinking; Mathematics Education; Problem Solving; ENEM; Scratch.

LISTA DE SIGLAS

BNCC	Base Nacional Comum Curricular
CIEB	Centro de Inovação para a Educação Brasileira
CNE	Conselho Nacional de Educação
CSTA	Computer Science Teachers Association
ENEM	Exame Nacional do Ensino Médio
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
ISTE	International Society for Technology in Education
MEC	Ministério da Educação
MIT	Instituto de Tecnologia de Massachusetts
NSF	National Science Foundation
SBC	Sociedade Brasileira de Computação
SiSU	Sistema de Seleção Unificada

SUMÁRIO

INTRODUÇÃO	12
1 PENSAMENTO COMPUTACIONAL	16
1.1 Contexto Histórico do Pensamento Computacional	16
1.2 O Pensamento Computacional na Educação Básica	22
2 PENSAMENTO COMPUTACIONAL NO SCRATCH	25
2.1 Scratch e seus recursos	25
2.2 O Scratch e a Linguagem de Programação	27
2.3 O uso do Scratch como recurso educacional	29
2.4 O Pensamento Computacional com o uso do Scratch	31
2.4.1 Sequências	33
2.4.2 Loop (laço)	35
2.4.3 Eventos	39
2.4.4 Condicionais	41
3 DESENVOLVIMENTO METODOLÓGICO	45
3.1 Definição de Pesquisa	46
3.2 O Exame Nacional do Ensino Médio	47
3.3 Escolha das questões	48
3.4 Como as questões foram resolvidas no Scratch?	50
4 RESOLUÇÃO E ANÁLISE DAS QUESTÕES	52
4.1 Questão 1	52
4.1.1 Primeira resolução utilizando conhecimento matemático	52
4.1.2 Segunda resolução utilizando conhecimento matemático	53
4.1.3 Resolução utilizando de Pensamento Computacional	53
4.1.4 Análise dos processos de Pensamento Computacional utilizados na resolução da questão 1	56
4.2 Questão 2	60
4.2.1 Resolução utilizando conhecimento matemático	60
4.2.2 Resolução utilizando de Pensamento Computacional	61
4.2.3 Resolução utilizando de Pensamento Computacional: composição a partir do problema inicial	62
4.2.4 Análise dos processos de Pensamento Computacional utilizados na resolução da questão 2	66
4.3 Questão 3	71
4.3.1 Resolução utilizando conhecimento matemático	71

4.3.2	Resolução utilizando de Pensamento Computacional	71
4.3.3	Análise dos processos de Pensamento Computacional utilizados na resolução da questão 3	73
4.4	Questão 4	79
4.4.1	Resolução utilizando conhecimento matemático	79
4.4.2	Resolução utilizando de Pensamento Computacional	80
4.4.3	Análise dos processos de Pensamento Computacional utilizados na resolução da questão	81
5	CONSIDERAÇÕES FINAIS	86
	Referências	90

INTRODUÇÃO

No que se refere à minha trajetória acadêmica e profissional, iniciei a graduação em Matemática no ano de 2005. Posteriormente, em 2009, ingressei na docência no Ensino Fundamental, experiência que contribuiu para o desenvolvimento de reflexões acerca dos processos de ensino e aprendizagem da Matemática no contexto escolar. A prática docente possibilitou observar diferentes dificuldades enfrentadas pelos estudantes na resolução de problemas matemáticos, sobretudo em situações que exigem interpretação, organização de informações e elaboração de estratégias de resolução.

Ao longo desse percurso formativo, realizei também uma segunda graduação na área de Física, o que ampliou minha aproximação com diferentes perspectivas científicas e educacionais. Nesse processo, fortaleceu-se o interesse por abordagens que favorecessem a compreensão de conceitos matemáticos de maneira mais investigativa e estruturada, bem como pelo uso de tecnologias digitais no contexto educacional.

A experiência docente, associada ao interesse crescente pelas tecnologias digitais, conduziu-me à busca por referenciais que possibilitassem compreender de que maneira recursos computacionais poderiam contribuir para a organização do pensamento na resolução de problemas matemáticos. Nesse contexto, o contato com discussões relacionadas ao Pensamento Computacional e com ambientes de programação visual, como o Scratch, mostrou-se particularmente relevante, na medida em que tais ferramentas possibilitam estruturar procedimentos, identificar padrões e elaborar estratégias de resolução de forma sistemática.

Em 2021, submeti um projeto de pesquisa ao programa de pós-graduação, com o intuito de aprofundar essas reflexões no âmbito da Educação Matemática. Embora naquele momento o projeto não tenha sido aprovado, mantive minha aproximação com o programa ao participar, como aluna não regular, da disciplina **Tecnologia na Educação Matemática**, ministrada pelo professor Sérgio. A participação nessa disciplina possibilitou ampliar o contato com estudos que articulam tecnologias digitais e processos de ensino e aprendizagem, além de aprofundar discussões relacionadas ao uso de recursos computacionais no contexto educacional.

Posteriormente, em 2022, ingressei no programa de pós-graduação e iniciei o desenvolvimento desta pesquisa sob orientação do professor Sérgio. A partir desse momento, tornou-se possível articular de maneira mais sistemática minha experiência docente, minha formação acadêmica e as discussões teóricas relacionadas ao Pensamento Computacional e à Educação Matemática.

Nesse contexto, esta pesquisa tem como propósito investigar de que modo os processos do Pensamento Computacional podem ser mobilizados na resolução de questões de Matemática do ENEM. Em particular, busca-se explorar o uso do ambiente de progra-

mação visual Scratch como recurso para a construção de procedimentos que permitam não apenas resolver situações específicas, mas também elaborar estratégias passíveis de generalização para problemas semelhantes.

Dessa forma, a investigação desenvolvida neste trabalho insere-se no campo da Educação Matemática e procura contribuir para as discussões acerca do uso de tecnologias digitais como suporte para a organização de estratégias de resolução e para o desenvolvimento de formas de pensamento que possam favorecer a compreensão e a sistematização de processos matemáticos.

A presente pesquisa insere-se no campo das investigações que articulam o Pensamento Computacional e a Educação Matemática, tomando como foco a resolução de problemas matemáticos por meio da construção de modelos computacionais. Parte-se do pressuposto de que o Pensamento Computacional, ao mobilizar processos como formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração, pode contribuir para a elaboração de estratégias de resolução que ultrapassem casos particulares, favorecendo a generalização e a reutilização de soluções em situações matemáticas semelhantes.

Assim a pesquisa foi desenvolvida no âmbito do Grupo de Estudos e Pesquisas em Educação Matemática da Universidade Estadual do Paraná (UNESPAR), Campus de União da Vitória. Como pesquisadora, integro o Grupo de Pesquisa Autômato, coordenado pelo Prof. Dr. Sérgio Carrazedo Dantas, vinculado à linha de pesquisa Tecnologia para o ensino e aprendizagem de Matemática. Desde 2021, o grupo tem se dedicado ao estudo do Pensamento Computacional, investigando suas concepções, possibilidades e desafios no contexto da Educação Matemática.

Esse percurso investigativo tem resultado em diferentes produções acadêmicas, tais como: *Uma análise das interações entre discentes de uma disciplina de mestrado na construção de um jogo utilizando o Scratch* (Baldicera; Teixeira; Dantas, 2023); *Concepções acerca do Pensamento Computacional presentes na BNCC e do referencial curricular do Paraná no Ensino Médio* (Plewka; Dantas, 2023); *Pensamento Computacional: a construção do jogo Pac-man no Scratch* (Padilha et al., 2023); *Contradições da BNCC acerca do desenvolvimento e uso das TIDC e do Pensamento Computacional* (Lirio; Prado, 2023); *Pensando e resolvendo problemas com o GeoGebra* (Dantas, 2023); *Análise de uma coleção de livros didáticos quanto à perspectiva de Pensamento Computacional* (Plewka; Dantas, 2025); *O Pensamento Computacional na construção de animações no Scratch* (Teixeira; Zanatta; Dantas, 2025); e *Relação dos Pensamentos Computacional e Matemático na resolução de uma questão do ENEM* (Prado; Almeida; Dantas, 2025).

Nesse viés, o Grupo de Pesquisa Autômato tem se empenhado em desenvolvimentos teóricos quanto ao Pensamento Computacional no contexto da Educação Matemática e em contextos de resolução de problemas e de aprendizagem de Matemática. Para isso, o grupo atua em diferentes eixos de investigação, como a produção e análise de jogos

digitais, a resolução de problemas matemáticos e o estudo de experiências desenvolvidas em outros países e as produções decorrentes desses estudos são mobilizadas ao longo deste trabalho.

No contexto educacional mais amplo, observa-se um crescimento significativo das discussões acerca do Pensamento Computacional e de sua inserção no ensino de Matemática. Apesar disso, ainda se fazem presentes desafios relacionados à compreensão de como seus processos podem ser operacionalizados de forma sistemática na resolução de problemas matemáticos, especialmente quando se considera a explicitação desses processos por meio da construção de modelos computacionais.

Tais desafios tornam-se particularmente relevantes considerando os conhecimentos explorados no Exame Nacional do Ensino Médio (ENEM), cujas questões demandam do estudante a organização de estratégias de resolução que envolvem análise, abstração e generalização. Emerge daí a necessidade de investigar de que modo os processos do Pensamento Computacional podem ser mobilizados na resolução de questões de Matemática do ENEM, indo além da obtenção de respostas pontuais e evidenciando estruturas de resolução passíveis de serem aplicadas a problemas semelhantes.

Diante desse contexto, esta pesquisa justifica-se pela busca em compreender e explicitar como os processos do Pensamento Computacional se manifestam durante a resolução de questões de Matemática do ENEM, a partir da construção de modelos computacionais no Scratch que é o objetivo geral dessa pesquisa. Assume-se que a elaboração desses modelos possibilita tornar visíveis os processos envolvidos na resolução dos problemas, contribuindo para a análise de suas potencialidades no âmbito da Educação Matemática. A questão de pesquisa que orienta este estudo é formulada nos seguintes termos:

Como os processos do Pensamento Computacional: formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração: são mobilizados durante a resolução de questões de Matemática do ENEM a partir da construção de modelos computacionais no Scratch?

No primeiro capítulo deste trabalho, são abordados os aspectos históricos do Pensamento Computacional, bem como suas relações com a Educação Matemática e sua inserção no contexto educacional brasileiro. O segundo capítulo apresenta o programa Scratch, destacando recursos úteis para a construção de modelos computacionais voltados à resolução de problemas, com ênfase na elaboração de algoritmos passíveis de generalização.

O terceiro capítulo descreve o percurso metodológico da pesquisa, explicitando a organização do estudo e apresentando uma breve contextualização do ENEM. Nesse capítulo, detalham-se os critérios adotados para a seleção das questões analisadas, assim como o método utilizado para sua resolução e análise à luz dos processos do Pensamento Computacional.

No quarto capítulo, são apresentadas as resoluções das questões de Matemática do Exame Nacional do Ensino Médio (ENEM) desenvolvidas no Scratch, bem como a análise dessas resoluções, buscando compreender de que forma os processos do Pensamento Computacional são mobilizados ao longo da construção dos modelos computacionais no Scratch.

Por fim, no quinto capítulo, apresentam-se as considerações finais da pesquisa, nas quais são retomados os objetivos propostos, a questão de pesquisa e as principais contribuições do estudo.

Adiantamos que a pergunta norteadora é respondida durante a realização da pesquisa e é relatada na seção de análise e discutida nas considerações finais. Assim, concluímos que:

- Os processos do Pensamento Computacional — formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração — manifestam-se de forma inter-relacionada durante a resolução de questões de Matemática do ENEM, demonstram que tais processos podem constituir uma lente analítica pertinente para compreender e explicitar as estratégias mobilizadas na resolução de problemas matemáticos;
- A construção de modelos computacionais no Scratch favorece a explicitação e a organização dos processos envolvidos na resolução das questões, permitindo tornar visíveis etapas do raciocínio que, em resoluções exclusivamente algébricas ou aritméticas, tendem a permanecer implícitas, além de possibilitar a elaboração de estruturas algorítmicas passíveis de generalização para situações matemáticas análogas;
- A articulação entre Pensamento Computacional e a Educação Matemática promovem potencialidades para a compreensão da resolução de problemas como um processo estruturado, no qual a construção de algoritmos e a análise de padrões podem contribuir para a elaboração de estratégias generalizáveis, ampliando as possibilidades de análise didática de tarefas matemáticas presentes em avaliações de larga escala, como o ENEM.

Espera-se que as discussões apresentadas neste trabalho possam contribuir para o avanço das reflexões acerca das relações entre Pensamento Computacional e Educação Matemática, bem como suscitar novas investigações que ampliem o debate sobre o tema.

1 PENSAMENTO COMPUTACIONAL

O objetivo deste capítulo é realizar uma revisão sistemática a respeito do que é chamado na literatura de Pensamento Computacional. Para isso, abordamos as perspectivas de Seymour Papert, Jeannette Wing, Christian Puhlmann Brackmann, da Sociedade Brasileira de Computação (SBC) e do Grupo de Pesquisa Autômato, descrevendo o contexto histórico não linear desse conceito.

Na segunda parte do texto, discutimos o Pensamento Computacional e sua relação com a Educação Básica, destacando a integração com a Base Nacional Comum Curricular (BNCC), bem como as críticas apresentadas por Dantas (2023) e pela SBC (2017, 2018) acerca da falta de clareza conceitual do documento em relação ao tema ao longo de todo o percurso formativo.

1.1 Contexto Histórico do Pensamento Computacional

A partir do ano de 1960 iniciou-se uma discussão a respeito de uma conceituação para Pensamento Computacional no meio acadêmico. Neste texto discorreremos sobre o tema atentos a participação de Seymour Papert e de Jeannette M. Wing nesse movimento histórico. Porém, mais recentemente, outros pesquisadores, tanto no Brasil quanto em outros países, deram suas contribuições nas discussões sobre o tema (Christian Puhlmann Brackmann, SBC; Grupo de Pesquisa Autômato), os quais contemplaremos ao longo desse capítulo.

Seymour Papert é considerado o precursor das discussões sobre Pensamento Computacional. Em seu livro *Mindstorms* de 1980 ele afirma que a programação utilizando a linguagem *Logo* poderia promover o que ele chamou de *powerful - ideias poderosas e procedural knowledge* - conhecimento processual. Segundo o autor, os computadores deveriam ser utilizados para que as pessoas pudessem “pensar com” as máquinas e “pensar sobre” o próprio pensar. Papert (1980) mencionava a importância dessa atividade para o processo de construção de conhecimento e para o desenvolvimento do pensamento. Ele argumentou que a computação pode ter “um impacto profundo por concretizar e elucidar muitos conceitos anteriormente sutis em psicologia, linguística, biologia e os fundamentos da lógica e da matemática” (Papert, 1980, p. 2).

Papert (1980) aponta que o aprendiz deve construir o seu próprio conhecimento com o auxílio do computador e, sobretudo, valorizando o saber fazer. Segundo ele, essa construção é de interesse do próprio aprendiz, o que produz motivação e, posteriormente, uma aprendizagem significativa. Daí a definição:

Assim, o Construcionismo, minha reconstrução pessoal do Construtivismo,

apresenta como principal característica o fato que examina mais de perto do que os outros ismos educacionais a ideia da construção mental. Ele atribui especial importância ao papel das construções no mundo como um apoio para o que ocorreu na cabeça, tornando-se, desse modo, menos uma doutrina puramente mentalista. Também leva mais a sério a ideia de construir na cabeça reconhecendo mais um tipo de construção (algumas delas tão afastadas de construções simples como cultivar um jardim) e formulando perguntas a respeito dos métodos e materiais usados (Papert, 1980, p. 127-128).

Em sua teoria Construcionista, a construção do conhecimento é favorecida quando há a criação de *micromundos* – ambientes intelectuais em que o falso ou verdadeiro, certo ou errado, não são decisivos - tendo a participação do educador como mediador das instruções e orientador na construção de objetos relevantes, que possibilitem ao aprendiz testar suas ideias, confirmar suas conclusões, ou mesmo verificar seus erros, em um trabalho coletivo, que fomente a socialização e a cooperação em prol da construção de algo de comum interesse (Papert, 1980).

A partir desses princípios, Papert desenvolveu a linguagem de programação *Logo* que é comparada a uma linguagem profissional, pois apresenta as mesmas ferramentas dessa última e as mesmas dificuldades a serem superadas em relação à lógica de programação (Papert, 1980). Ao usar o *Logo*, o aprendiz assume o papel de ensinar à máquina o que ela deve fazer. O objetivo de Papert era a independência do aprendiz:

[...] as crianças farão melhor descobrindo ('pescando') por si mesmas o conhecimento específico de que precisam; a educação organizada ou informal poderá ajudar mais se certificar de que elas estarão sendo apoiadas moral, psicológica, material e intelectualmente em seus esforços (Papert, 1980, p. 135).

De acordo com sua teoria, quando a criança constrói o conhecimento, este é melhor interiorizado por ela.

Anos depois dessas contribuições de Papert, a expressão Pensamento Computacional recebeu maior enfoque com a produção da pesquisadora Jeannette M. Wing, em seu texto *Computational thinking* (2006), no qual a autora cunhou esta expressão que, por sua vez, desempenhou um papel fundamental na popularização do termo e sua inclusão no currículo educacional. Wing (2006) afirmou que o Pensamento Computacional envolve resolver problemas, projetar sistemas e entender o comportamento humano.

O pensamento computacional é uma habilidade fundamental para qualquer um, não apenas para cientistas da computação. Juntamente com a leitura, a escrita e a aritmética, devíamos adicionar o pensamento computacional na habilidade analítica da criança (Wing, 2006, p. 33).

Resumidamente, para Wing (2006), o Pensamento Computacional envolve habilidades de resolução de problemas, abstração e criação de algoritmos. Porém, em 2011, um novo artigo foi publicado por ela e teve como objetivo estabelecer novos fundamentos para a expressão Pensamento Computacional. Nesse texto consta que o Pensamento Computacional

[...] é o processo de pensamento envolvido na formulação de problemas e suas soluções, de modo que as soluções sejam representadas de uma forma que possa ser efetivamente executada por um agente de processamento de informações [...] Assim, o pensamento computacional se sobrepõe ao pensamento lógico e ao pensamento sistêmico. Inclui o pensamento algorítmico e o pensamento paralelo, que por sua vez envolvem outros tipos de processos de pensamento, como raciocínio composicional, correspondência de padrões, pensamento procedural e pensamento recursivo. O pensamento computacional é usado no projeto e na análise de problemas e suas soluções, interpretados de forma ampla [...] O processo de pensamento mais importante e de alto nível no pensamento computacional é o processo de abstração (Wing, 2011, p. 20-21).

Em 2014, Wing define Pensamento Computacional como “o processo de pensamento envolvido na formulação de um problema e na sua solução de tal forma que o computador possa executá-lo efetivamente” (Wing, 2014, s/p).

A influência de Wing na National Science Foundation (NSF)¹, corroborou de forma efetiva para a consolidação da expressão Pensamento Computacional, mesmo sem uma definição suficientemente clara para a interpretação da expressão.

O grupo International Society for Technology in Education (ISTE) e a Computer Science Teachers Association (CSTA) elaborou uma definição operacional para o Pensamento Computacional como um processo de resolução de problema, com as seguintes características: formulação de problemas de uma forma que permita usar um computador e outras ferramentas para ajudar a resolvê-los; organização lógica e análise de dados; representação de dados através de abstrações como modelos e simulações; automação de soluções através do pensamento algorítmico (série de passos ordenados); identificação, análise e implementação de soluções possíveis com o objetivo de alcançar a mais eficiente e efetiva combinação de etapas e recursos; e generalização e transferência desse processo de resolução de problemas para uma ampla variedade de problemas. Finalmente, observam que essas características são “apoiadas e reforçadas por uma série de disposições ou atitudes que são dimensões essenciais do Pensamento Computacional”, tais como: “confiança em lidar com a complexidade, persistência em trabalhar com problemas difíceis, tolerância para a ambiguidade e capacidade de lidar com problemas abertos” (Iste; Csta, 2011, p. 7).

Diante dessa discussão, no Brasil, o Conselho Nacional de Educação (CNE), a SBC e o Centro de Inovação para a Educação Brasileira (CIEB), preocupados com a inserção do Pensamento Computacional engajaram-se efetivamente na introdução de Pensamento Computacional, tecnologia digital, e cultural digitais no currículo da Educação Básica. Para a SBC

O Pensamento Computacional se refere à capacidade de sistematizar, representar, analisar e resolver problemas. Apesar de ser um termo recente, vem sendo

¹A NSF é uma agência independente do governo dos Estados Unidos que financia pesquisas científicas e educacionais em diversas áreas, incluindo ciências naturais, engenharia, matemática e ciência da computação. Desde sua criação em 1950, a NSF tem desempenhado um papel crucial no avanço do conhecimento científico e tecnológico, especialmente ao apoiar projetos inovadores e programas educacionais.

considerado como um dos pilares fundamentais do intelecto humano, junto com leitura, escrita e aritmética, pois como estes, serve para descrever, explicar e modelar o universo e seus processos complexos (Sbc, 2017, p. 3).

No final de 2018, com o lançamento da BNCC, inclui-se o Pensamento Computacional como uma habilidade necessária a ser desenvolvida pelo aluno da Educação Básica. Nesse contexto, surgem no Brasil pesquisadores que discutem o termo Pensamento Computacional, um deles foi Christian Puhlmann Brackmann.

Para Brackmann (2017) o Pensamento Computacional é uma competência fundamental que toda pessoa deveria adquirir, independentemente de sua área de conhecimento ou atividade. O autor destaca que quando aprendemos a ler, em seguida podemos ler para escrever, e com essas duas habilidades adquiridas podemos aprender novas coisas. Analogamente, segundo o autor, quando os estudantes aprendem a programar, programam para aprender e em seguida estão prontos para aprender e criar coisas novas.

Em sua tese, a partir das definições da SBC, Brackmann define o Pensamento Computacional a partir de quatro pilares: 1) decomposição; 2) reconhecimento de padrões; 3) abstração; e 4) algoritmos. Para ele, a decomposição envolve identificar um problema complexo e quebrá-lo em partes menores e mais fáceis de gerenciar. No reconhecimento de padrões cada um desses problemas menores pode ser analisado individualmente com maior profundidade, identificando problemas parecidos que já foram solucionados anteriormente. No que se refere à abstração, foca apenas nos detalhes que são importantes, deixando de lado informações irrelevantes. Por fim, os algoritmos são regras ou passos utilizados para resolver problemas ou subproblemas encontrados (Brackmann, 2017).

Com a finalização de sua pesquisa de doutorado, Brackmann analisou o desenvolvimento do Pensamento Computacional com a proposta de atividades desplugadas (sem o uso do computador). A partir de uma abrangente pesquisa sobre o tema, o autor sugere que:

O Pensamento Computacional é uma distinta capacidade criativa, crítica e estratégica humana de saber utilizar os fundamentos da Computação, nas mais diversas áreas do conhecimento, com a finalidade de identificar e resolver problemas, de maneira individual ou colaborativa, através de passos claros, de tal forma que uma pessoa ou uma máquina possam executá-los eficazmente (Brackmann, 2017, p. 29).

Brackmann sustenta que o Pensamento Computacional não se resume simplesmente a utilização de dispositivos ou aplicativos eletrônicos, tampouco está associado a uma abordagem mecânica de raciocínio. Sendo assim, está intrinsecamente ligado ao desenvolvimento de capacidade crítica e criativa, representando uma forma de pensamento estruturado que articula o trabalho colaborativo. Neste viés, surgem os argumentos em torno dos benefícios do Pensamento Computacional: compreensão do mundo, transversalidade em diferentes áreas, alfabetização digital, produtividade, programação que colabora no aprendizado de outras áreas, inclusão de minorias, diminuição das limitações físicas e

trabalho em equipe (Brackmann, 2017).

A última perspectiva de Pensamento Computacional abordada neste texto foi desenvolvida pelo Grupo de pesquisa Autômato. Segundo as produções acadêmicas do grupo, o Pensamento Computacional

[...] ocupa-se do tratamento de entes abstratos em interface com o pensamento matemático, buscando a resolução de problemas por meio de uma série de processos que possam ser executados por um sujeito cognitivo, utilizando ou não de dispositivos digitais (Prado, 2024, p. 8).

Neste sentido, destacamos detalhes em relação aos termos utilizados quanto a forma de tratar o Pensamento Computacional sob a ótica dos diferentes autores citados até este momento: Wing referencia habilidades do Pensamento Computacional; Brackmann aponta pilares; o ISTE/CSTA elenca características, porém, nós a partir dos escritos do Grupo de Pesquisa Autômato, vamos nos referir aos processos mentais constituintes do Pensamento Computacional, que são um a um abordados nos próximos parágrafos.

Formulação do problema: etapa em que o problema é elaborado em termos de necessidade, objeto, pergunta e é, a partir daí, que se conjectura um possível modo de resolução. Nessa etapa se questiona o que considerar, quais são as variáveis, que ações executar, e também, um ou mais planos de ação tendo como pano de fundo técnicas e repertórios.

Decomposição: consiste em dividir os problemas em partes menores a partir de um problema maior ou mais complexo. Com isso, é possível concentrar a atenção na resolução de partes específicas do problema, obtendo ao fim a solução. Para orientar a prática da decomposição, podemos nos questionar:

- Como podemos usar os detalhes para identificar partes do problema, desafio ou tarefa?
- Que partes são familiares? Que partes são desconhecidas?
- Quais são as diferentes formas de resolver o problema, desafio ou tarefa?
- É possível decompor as partes em partes menores?
- Como a decomposição do problema pode servir para resolvê-lo ou compreendê-lo? (Espadeiro, 2021, p. 6).

Reconhecimento de padrões: processo que pode surgir a partir da decomposição, quando problemas menores podem ser solucionados com base em experiências anteriores ou via repertórios matemáticos/computacionais, ou ainda, acontece ao perceber o que é constante ou variável nos dados ou formas associadas ao problema. Para identificar o reconhecimento de padrões, podemos nos questionar:

- Que semelhanças ou padrões encontramos no problema? Por exemplo, quantos objetos existem? Que cores são identificadas? Que repetições são identificadas?
- Como podemos utilizar os detalhes para identificar partes do problema? Que relações existem entre as partes?
- Como podemos descrever os padrões?

- Como se pode utilizar o padrão para fazer previsões ou tirar conclusões? (Espadeiro, 2021, p. 6).

Abstração: é o processo de concentrar a atenção no que é necessário e suficiente para a resolução de um problema ou subproblemas, desconsiderando dados ou informações irrelevantes. Podemos ter em mente algumas questões a responder para estabelecer nosso processo de abstração, por exemplo:

- Como podemos simplificar este problema/tarefa?
- Qual é a informação relevante para resolver este problema/tarefa?
- Como podemos representar claramente a informação importante?
- De que forma podemos relacionar a informação importante tendo em vista o ponto de partida e o resultado a alcançar (dar resposta para o problema ou resolver a tarefa)? (Espadeiro, 2021, p. 6).

Produção de algoritmos: é o processo de obtenção de passos ou regras de ação desenvolvidos e efetivados durante a resolução de subproblemas ou problemas. Esses algoritmos podem ser (d)escritos em códigos da língua materna, códigos matemáticos, códigos de computação/programação, entre outros. Algumas questões orientadoras na produção de algoritmos:

- Quais são as etapas necessárias para a resolução do problema?
- Qual é a informação necessária para a concretização de cada uma das etapas?
- Como estruturar todos os passos necessários para a resolução do problema ou tarefa? (Espadeiro, 2021, p. 6).

Depuração: é o processo de procura e correção de erros. Além disso, poderá assumir, de igual modo, ações de testagem, verificação, refinamento e otimização da resolução apresentada. Na depuração as ações mentais em busca da solução podem ser afirmadas, revisadas, reformuladas ou abandonadas, o que leva a rever outras ações ou a conclusão do trabalho. Durante a depuração são produtivas perguntas como:

- Como podemos garantir que o nosso plano, modelo, ou solução funcionou, ou não?
- O resultado corresponde ao que esperávamos?
- Como podemos modificar a abordagem para corrigir falhas ou imprecisões?
- Como sabemos se conseguimos corrigir o erro? (Espadeiro, 2021, p. 6).

Nas discussões do grupo Autômato existe a compreensão da relação entre o Pensamento Computacional e o Pensamento Matemático, visto que eles possuem uma estreita relação nos processos de resolução de problemas, ou seja, ambos recorrem à utilização do raciocínio lógico, à decomposição e à análise dos problemas em partes menores e à identificação de padrões; daí a nossa opção de escolha por tal abordagem.

1.2 O Pensamento Computacional na Educação Básica

Observa-se, nos últimos anos, um crescimento expressivo de propostas e programas que visam à introdução do Pensamento Computacional no cenário escolar, utilizando-se de múltiplas tecnologias, linguagens e concepções, inclusive no Brasil.

Prova disso, a BNCC, particularmente a área da Matemática, salienta a utilização das tecnologias na perspectiva da resolução de problemas e busca contribuir para o desenvolvimento de competências relativas ao Pensamento Computacional. Entre as dez competências que devem ser desenvolvidas nas etapas da Educação Básica, segundo o texto da BNCC, a quinta evidência aspectos que devem estar inseridos na prática dos professores quanto ao uso das tecnologias digitais:

[...] compreender, utilizar e criar tecnologias digitais de informação e comunicação de forma crítica, significativa e ética nas diversas práticas sociais (incluindo as escolares) para se comunicar, acessar e disseminar informações, produzir conhecimentos, resolver problemas e exercer protagonismo e autoria na vida pessoal e coletiva (Brasil, 2018, p. 9).

O entendimento dos aspectos destacados nessa competência é de suma importância para que o uso das tecnologias não seja simplesmente aparente, ou seja, com a intenção de modernizar práticas que não favorecem o protagonismo e a autonomia dos estudantes diante da aprendizagem.

Segundo Dantas (2023), ao abordar o uso das tecnologias digitais, a BNCC busca integrar um diálogo que ocorre em diversos países, nos quais se levantam questionamentos e iniciativas propostas para a inclusão de recursos tecnológicos que visam auxiliar os processos educativos, que preparem os estudantes para participação na sociedade. Neste viés, a BNCC faz referência a tecnologias digitais em quatro das dez competências gerais destinadas à Educação Básica, destacando a importância de sua integração para o desenvolvimento educacional.

No documento, ainda na área de Matemática, há recomendações sobre o termo Pensamento Computacional, que estão diretamente ligadas ao uso de fluxogramas. A falta de uma definição precisa sobre o Pensamento Computacional na BNCC, e a ênfase excessiva na elaboração de algoritmos e fluxogramas, levou a uma ampla crítica feita pela SBC, através de uma “Nota técnica”² (Sbc, 2018, p. 2).

Ainda destacando o que traz a SBC, referente a definição de Pensamento Computacional como a “Habilidade de compreender, definir, modelar, comparar, solucionar, automatizar e analisar problemas (e soluções) de forma metódica e sistemática” (Sbc, 2018, p. 2). Consoante a SBC, o Pensamento Computacional é uma habilidade que en-

²Nota Técnica da SBC sobre a BNCC-EF e a BNCC-EM, de 2018. A SBC apresenta críticas e contribuições específicas para diferentes pontos da proposta, como apresentado a seguir, tanto para a BNCC-EF, como para a BNCC EM. Ainda, segundo a SBC, o documento foi elaborado visando demonstrar sua inadequação e solicitar providências para uma correspondente revisão que, se não realizada, pode prejudicar o ensino na Educação Básica no Brasil.

volve a sistematização da atividade de resolução de problemas, representação e análise das soluções utilizando algoritmos. E propõe uma abordagem que seja estimulada desde os anos iniciais do Ensino Fundamental, permitindo que o sujeito trabalhe com abstrações e problemas tanto de forma, plugada quanto desplugada.

Embora a BNCC já trouxesse competências gerais relacionadas ao uso de tecnologias digitais e expusesse o Pensamento Computacional, o documento não detalhava o que ensinar e nem indicava estratégias para abordá-lo. Assim, em 17 de fevereiro de 2022 foi aprovado, por unanimidade, o texto que trata das Normas sobre Computação Educação Básica – Complemento à Base Nacional Comum Curricular, homologado em 03 de outubro de 2022, que apresenta a Computação como área de conhecimento estruturada em três eixos: Pensamento Computacional, Mundo Digital e Cultura Digital, sendo o primeiro aquele que mais diretamente se relaciona com a resolução de problemas.

O Pensamento Computacional é compreendido como um eixo estruturante que perpassa toda a Educação Básica, propondo habilidades que possibilitam aos estudantes desenvolver competências relacionadas à abstração, ao reconhecimento de padrões, à criação de algoritmos, à lógica e à decomposição de problemas (Brasil, 2018). Por mais que este documento seja voltado à computação na Educação Básica, devido às similaridades apontadas enquanto habilidades e que tratamos de processos constituintes do Pensamento Computacional, conforme já descrito anteriormente, optamos em discorrer sobre esse texto.

Na Educação Infantil, por exemplo, o documento destaca que a Computação deve ser vivenciada de maneira lúdica, a partir da exploração de padrões, da decomposição de problemas e da criação de algoritmos simples. Segundo o texto, o eixo Pensamento Computacional envolve “reconhecer padrão de repetição em sequência de sons, movimentos, desenhos; expressar as etapas para a realização de uma tarefa de forma clara e ordenada; experienciar a execução de algoritmos brincando com objetos (des)plugados” (Brasil, 2018, p. 2). Nessa fase, enfatiza-se o desenvolvimento inicial do raciocínio algorítmico, associado à noção de lógica binária, ao propor que as crianças sejam capazes de “criar e representar algoritmos para resolver problemas; comparar soluções algorítmicas para resolver um mesmo problema; compreender decisões em dois estados (verdadeiro ou falso)” (Brasil, 2018, p. 4).

No Ensino Fundamental, Anos Iniciais (1º ao 5º ano), as habilidades se tornam mais estruturadas, articulando o Pensamento Computacional com conceitos de representação da informação e de estruturas de dados. O documento propõe que os estudantes sejam capazes de organizar informações, criar e simular algoritmos utilizando sequências, repetições e seleções condicionais, bem como aplicar estratégias de decomposição de problemas. Em outras palavras, trata-se de identificar as principais formas de organizar e representar a informação de maneira estruturada (matrizes, registros, listas e grafos) ou não estruturada (números, palavras, valores verdade).

Nos Anos Finais do Ensino Fundamental (6º ao 9º ano), o documento amplia as expectativas, introduzindo a utilização de estruturas de dados mais complexas e o desenvolvimento de programas com controle de fluxo, repetição e condições. O desenvolvimento de soluções para problemas computacionais requer que o estudante empregue diferentes estratégias da Computação, como decomposição, generalização e reúso, de modo a organizar, simplificar e reaproveitar etapas no processo de resolução. Tal competência está articulada ao objeto de conhecimento denominado “estratégia de solução de problema”, que se subdivide em decomposição e generalização, permitindo ao aluno estruturar o pensamento de forma lógica e estruturada. Conforme o Complemento à BNCC, essa habilidade é fundamental para que os estudantes “construam e analisem soluções computacionais em diferentes áreas do conhecimento, individualmente ou em colaboração, selecionando estruturas de dados adequadas e articulando saberes escolares” (Brasil, 2018, p. 38).

Notamos, mais uma vez, que embora o documento seja voltado à computação na Educação Básica, os conceitos abordados estão estreitamente relacionados aos processos que apresentamos para a resolução de problemas utilizando o Pensamento Computacional e que tais processos não são restritos apenas a uma área de conhecimento.

No Ensino Médio, as habilidades ligadas ao Pensamento Computacional são aprofundadas, incorporando noções de desempenho de algoritmos e limites da computação. O documento propõe que os estudantes desenvolvam uma compreensão crítica sobre as potencialidades e restrições da resolução de problemas, incluindo discussões sobre a resolução algorítmica e problemas sem solução, como o clássico Problema da Parada. Conforme o texto, “os alunos devem compreender noções relacionadas à eficiência de algoritmos, à teoria da computabilidade, incluindo limites do que pode ou não ser resolvido computacionalmente” (Brasil, 2018, p. 62).

Dessa forma, o Complemento à BNCC propõe uma progressão contínua do Pensamento Computacional, iniciando na Educação Infantil, com atividades de reconhecimento de padrões e sequências, até o Ensino Médio, com análises de eficiência e limites teóricos da computação. O documento reforça que o Pensamento Computacional não se reduz à programação, mas constitui um conjunto de práticas cognitivas essenciais para a resolução de problemas em diferentes áreas, favorecendo o desenvolvimento de uma postura crítica e criativa diante das tecnologias digitais presentes na sociedade (Brasil, 2018).

2 PENSAMENTO COMPUTACIONAL NO SCRATCH

Neste capítulo apresentamos a plataforma Scratch e como ela pode ser utilizada na produção de animações, jogos e histórias interativas. Para isso, nos remetemos ao site oficial do programa e inserimos algumas animações no corpo do texto para apresentar a dinamicidade das produções que podem ser realizadas. De acordo com Brennan e Resnick (2012), o trabalho com o Scratch favorece a mobilização de processos do Pensamento Computacional. Desta maneira, ao criar e ajustar projetos na plataforma, o estudante desenvolve competências que integram o raciocínio lógico.

2.1 Scratch e seus recursos

O Scratch é um ambiente em que o usuário pode utilizar blocos de comandos para criar animações, histórias e jogos utilizando imagens, músicas, cenários do próprio programa ou externos a ele. O programa foi criado pelo Instituto de Tecnologia de Massachusetts (MIT) em 2007 e o responsável pela equipe de desenvolvimento foi Michel Resnick.

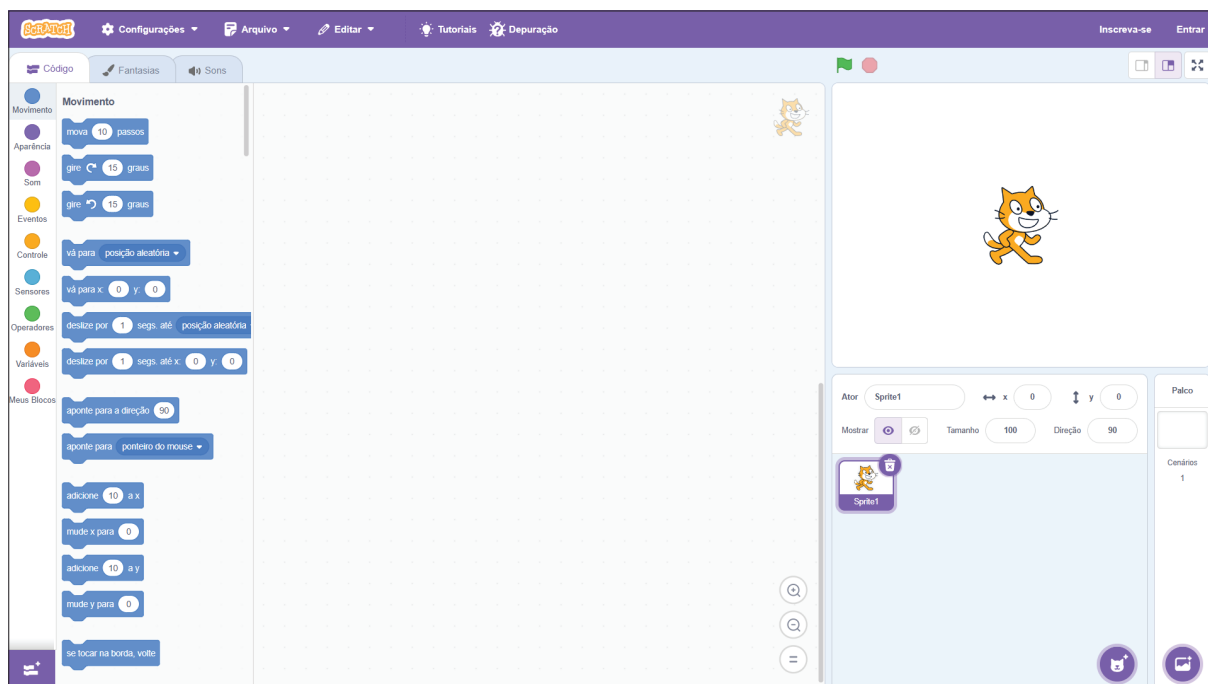
A plataforma pode ser utilizada diretamente no navegador sem a necessidade de instalação. Para isso, em sua página principal basta clicar em criar e acessar o programa online¹. Caso o usuário deseje utilizar o Scratch diretamente em seu computador, pode instalá-lo fazendo o download do programa em seu site oficial. Isso permite sua utilização mesmo que o computador não tenha acesso à internet.

O Scratch permite conexões com dispositivos *Bluetooth*. Nesse caso, uma conexão com a internet é necessária para usar o Scratch Link. Diversos dispositivos de *hardware* podem ser conectados ao Scratch, incluindo LEGO WeDo 2.0, Micro bit, Makey Makey, EV3, Lego Boost e Raspberry Pi². Além disso, Scratch permite interação por meio de um microfone ou de uma webcam usando um microfone ou uma webcam. Na Figura 2.1 apresentamos a interface inicial do Scratch.

¹O endereço online é <https://scratch.mit.edu>.

²Os dispositivos LEGO WeDo 2.0, BBC micro:bit, Makey Makey, LEGO EV3, LEGO Boost e Raspberry Pi são recursos voltados à educação em computação e robótica. O LEGO WeDo 2.0 e o LEGO Boost são kits da LEGO Education que permitem a construção de modelos físicos com motores e sensores, sendo voltados ao ensino fundamental e programáveis via Scratch Link. O LEGO EV3 (Mindstorms EV3), mais robusto, é utilizado no ensino médio e possui sensores de toque, cor e ultrassom, além de uma “brick” programável. O micro:bit é uma placa compacta com matriz de LEDs, sensores e botões, ideal para projetos interativos de codificação física. O Makey Makey transforma objetos condutores em entradas de teclado, possibilitando a criação de interfaces criativas com o Scratch. Já o Raspberry Pi é um computador de placa única que permite controlar sensores e atuadores físicos via Scratch ou Python, sendo utilizado no ensino mais avançado.

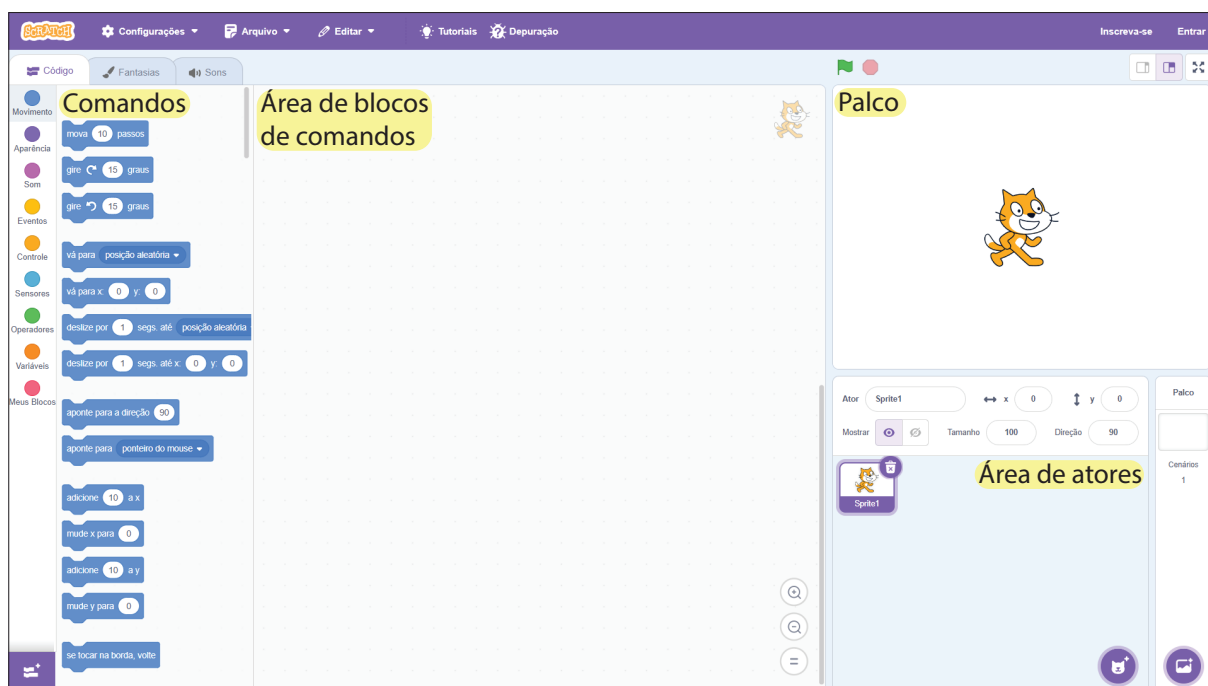
Figura 2.1: Interface do Scratch.



Fonte: Elaborada pela autora (2026).

Na Figura 2.2 apresentamos a tela do Scratch em que são destacadas as áreas da plataforma com suas respectivas denominações, que utilizaremos a partir de agora.

Figura 2.2: Interface do Scratch com áreas nomeadas.



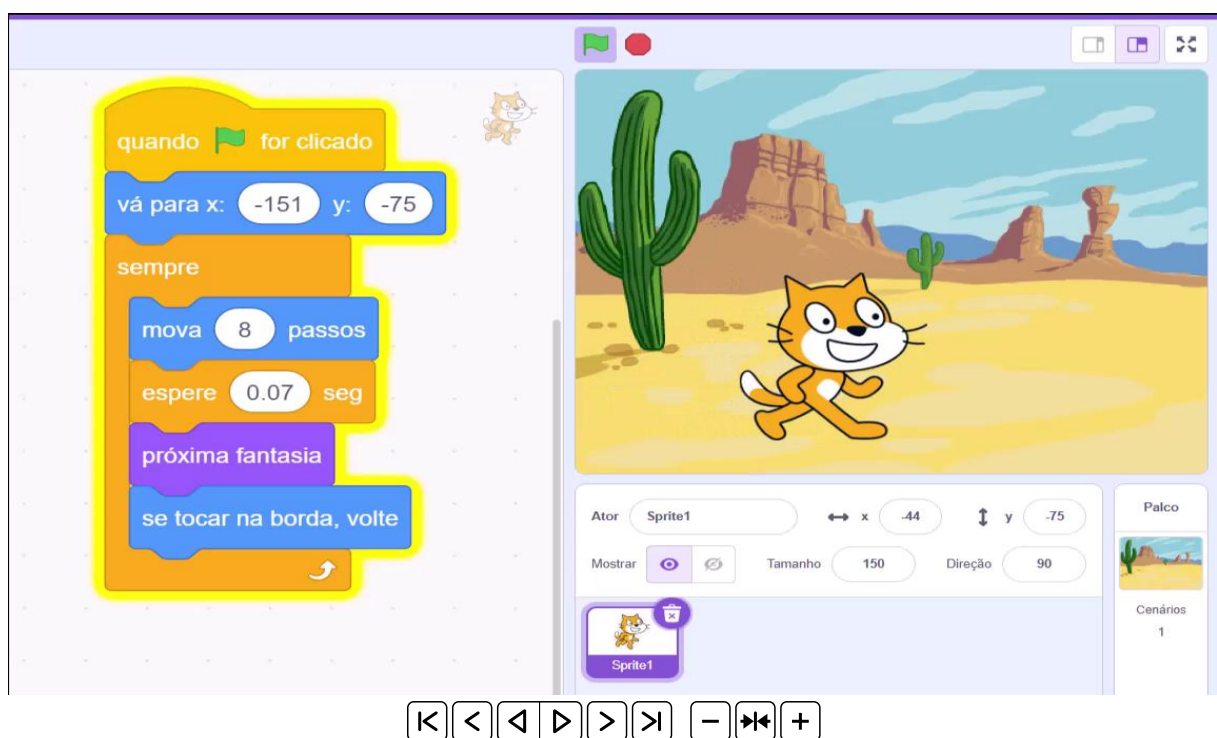
Fonte: Elaborada pela autora (2026).

Fazendo uma breve descrição de cada uma dessas áreas, temos Comando, espaço

onde os comandos estão separados por categorias (movimento, aparência, som, eventos, controle, sensores, operadores e variáveis) e cores; área de Blocos de comandos: é o espaço no qual os blocos de programação são combinados para formar algoritmos; palco: constitui o ambiente visual no qual todas as ações programadas são executadas. Trata-se do plano de visualização e validação do comportamento programado, em que os personagens interagem e no qual se observa, em tempo real, os efeitos das instruções algorítmicas; área de atores: reúne todos os atores disponíveis no projeto, permitindo a seleção, criação, edição e organização dos personagens.

Por fim, a Figura 2.3 exibe um ator no palco cujo fundo lembra uma paisagem com alguns elementos. Na Área de Blocos de Comandos temos a construção de um algoritmo com um conjunto de blocos organizados de forma sequencial, constituindo a estrutura responsável por controlar e orientar este ator, assegurando a execução coordenada de seus movimentos, fazendo com que o personagem se desloque de uma lateral a outra, trocando suas fantasias, de modo que seja exibida a animação de uma caminhada sem interrupções.

Figura 2.3: Animação no Scratch



Fonte: Elaborada pela autora (2026)

2.2 O Scratch e a Linguagem de Programação

A linguagem de programação Scratch é baseada em blocos que se encaixam, empregados pelos usuários para expressar as suas ideias. Esta linguagem foi criada em 2007 para ser usada inicialmente por crianças e adolescentes de 8 a 16 anos, e atualmente vem

sendo utilizada em todos os níveis e modalidades de ensino é gratuita e está disponível no site <http://scratch.mit.edu>, voltando-se para produção de projetos que envolvam a construção de histórias animadas, jogos, situação problemas, desafios, tutoriais e o que se for capaz de imaginar (Resnick 2014).

Em seu livro *Jardim de Infância para a vida toda*, Resnick (2020) argumenta que o Scratch foi projetado para ser uma plataforma de aprendizagem em que quatro princípios da aprendizagem criativa se entrelaçam: projeto, paixão, pares e pensar brincando. Segundo ele, esses princípios visam a constituição de sujeitos criativos e colaborativos.

Para argumentar sobre o Scratch com seus blocos de programação, Resnick (2020) remete o leitor para os princípios de Froebel (1782-1852) em sua concepção sobre o jardim da infância.

Frederick Froebel abriu o primeiro jardim de infância do mundo na Alemanha em 1837. Segundo Resnick (2017), Froebel tinha a convicção de que poderia facilitar o desenvolvimento das crianças como pensadoras criativas, ao propor uma abordagem diferente do que a mera transmissão de conhecimento, pois, o professor apresenta os conceitos e os alunos apenas os copiam. Porém, essa abordagem não se adequava às crianças pequenas.

Froebel percebeu que as crianças menores aprendem melhor se interagem com objetos que façam parte do seu cotidiano. Uma de suas criações, que ficou conhecida como “presentes” de Froebel, era um conjunto de 20 brinquedos como blocos geométricos, papéis coloridos, palitinhos, ervilhas, e outros com os quais as crianças imaginavam e criavam castelos, mosaicos, dobraduras, personagens e contavam histórias sobre eles. Froebel acreditava que brincar seria um dos fundamentos da aprendizagem.

Segundo ele, as crianças vão crescendo e as escolas abandonam os princípios aplicados no jardim de infância. Resnick (2017), argumenta que à medida que as crianças crescem, é importante expandir suas experiências além dos simples blocos utilizados no jardim de infância e assim estimular o seu pensamento criativo.

É neste viés que o grupo de pesquisa de Resnick, *Lifelong Kindergarten*, - que significa jardim de infância ao longo da vida - do *Media Lab Do Massachusetts Of Technology (MIT)*, cria atividades e estratégias tecnológicas para envolver os alunos em experiências de aprendizagem criativas (Resnick, 2020, p. 15).

A partir daí surgem os quatro elementos fundamentais da Aprendizagem criativa que Resnick (2020) denominou como “Os quatro Ps da aprendizagem criativa”, representados por projetos, pares, paixão e pensar brincando, conforme detalhamos a seguir.

Projeto: as pessoas trabalham ativamente se estiverem envolvidas em projetos significativos; pares: ocorre quando existe o compartilhamento de ideias fundamentais para promover a aprendizagem naturalmente; paixão: o engajamento emocional ajuda a impulsionar o aprendizado. Pessoas apaixonadas tendem a se engajar efetivamente em desafios e aprendem mais nesse processo. Pensar brincando: envolver experiências

divertidas, como: testar coisas novas, assumir riscos e repetir tarefas até alcançar os objetivos.

Os 4 Ps da aprendizagem criativa estão alinhados à abordagem construcionista de Papert (1980), pois “ênfatiza o valor da criação de projetos que sejam significativos para os alunos, de maneira divertida e em colaboração com os colegas” (Resnick, 2014, p. 2), ao interagir com o objeto computador por meio da linguagem Scratch. Sendo assim, o Scratch não apenas promove o desenvolvimento de habilidades, mas também incentiva uma abordagem lúdica e colaborativa na resolução de problemas, conectando os alunos a um universo de possibilidades criativas. Através dos 4 Ps da aprendizagem criativa: projetos, paixão, pares e pensar brincando, o Scratch não apenas facilita a expressão de ideias, mas também fortalece a aprendizagem.

2.3 O uso do Scratch como recurso educacional

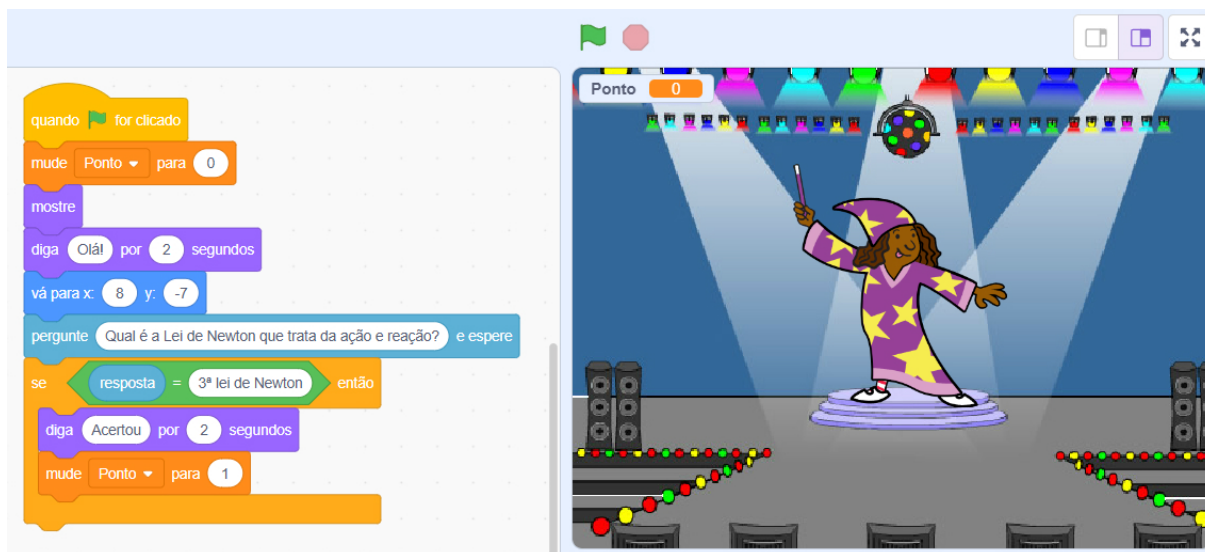
Desde a implantação das tecnologias digitais em ambientes escolares, muitos desafios vêm surgindo e o Scratch tem sido tomado como uma ferramenta pedagógica de apoio para professores a fim de minimizar alguns impactos no processo de ensino e aprendizagem. Dessa forma, Araújo, Souza e Silva (2014), afirmam que a adoção do Scratch na sala de aula oferece benefícios significativos para o processo de ensino e aprendizagem, visto que

O Scratch é uma linguagem visual de seu funcionamento ocorre por meio de blocos de estruturas de programação, sem a necessidade de se digitar nenhum código para programar e resolver os problemas propostos, a saber: a criação de histórias interativas, animações, simulações, jogo, músicas, apresentações animadas, entre outros (Araújo; Souza; Silva, 2014, p. 131).

Ainda segundo esses autores, ao criar projetos no Scratch, os estudantes são incentivados a expressar sua criatividade, a trabalhar em equipe e a desenvolver competências relacionadas à lógica de programação e à tomada de decisões. Por utilizar uma linguagem de programação por blocos, o Scratch proporciona uma abordagem prática e interativa, estimulando o Pensamento Computacional, a resolução de problemas e o desenvolvimento de habilidades cognitivas.

Além disso, o uso do Scratch na educação favorece a integração de diferentes áreas, possibilitando abordagem de outras disciplinas como, Ciências, Matemática, Geografia, Arte e Física. Os alunos são incentivados a explorar conceitos de forma prática e contextualizada, desenvolvendo um entendimento mais amplo e crítico. A Figura 2.4 é um exemplo desenvolvido no Scratch, em que são trabalhados conhecimentos de diferentes disciplinas de forma integrada. No roteiro programado, o personagem interage com o usuário por meio de perguntas que envolvem a Física (como a Lei de Newton da ação e reação).

Figura 2.4: Exemplo de projeto envolvendo conhecimentos de mais de uma disciplina.



Fonte: Elaborada pela autora (2026).

Pesquisas têm apresentado os impactos positivos do uso do Scratch em sala de aula (Resnick, 2009). Além do desenvolvimento de habilidades técnicas, os estudantes também se aprimoram em competências socioemocionais, como colaboração, resiliência e pensamento crítico.

O Scratch, vem sendo amplamente utilizado em diferentes contextos educacionais ao redor do mundo, abrangendo desde atividades em escolas, bibliotecas, programas extracurriculares, até cursos universitários (Brennan; Resnick, 2012). Essa diversidade de usos indicam o potencial do Scratch não apenas como ferramenta de ensino de programação, mas também como suporte para a aprendizagem criativa e para o desenvolvimento do Pensamento Computacional em diferentes níveis de formação.

A interação com a comunidade online do Scratch permite que os alunos compartilhem seus projetos, obtenham feedback e se inspirem em criações de outros colegas, criando um ambiente de aprendizagem colaborativo.

Além disso, a interface é intuitiva e o manuseio de suas ferramentas não requer comandos complexos. Cabe salientar ainda que possui a opção da linguagem em português e todos os ambientes criados nessa linguagem são em duas dimensões (2D).

O Scratch apresenta características que o tornam acessível e de fácil utilização, contribuindo para o engajamento dos estudantes no desenvolvimento da criatividade e da concentração voltada à resolução de problemas. Sua interface intuitiva, aliada ao suporte disponível em português, potencializa o domínio da ferramenta e contribui para o Pensamento Computacional de maneira prática e atrativa. Como ressaltam Brennan e Resnick (2012), o Scratch “permite que jovens desenvolvam habilidades de Pensamento Computacional de forma divertida e significativa” (p. 2).

2.4 O Pensamento Computacional com o uso do Scratch

O texto publicado em por Brennan e Resnick 2012, intitulado *New frameworks for studying and assessing the development of computational thinking*, apresentaram sua perspectiva acerca do Pensamento Computacional e discutiram como ele pode ser explorado por meio do uso do Scratch. Os autores analisaram produções de crianças de 8 a 12 anos compartilhadas na plataforma on-line do Scratch e, posteriormente, realizaram entrevistas com parte delas para a coleta de dados. A partir desse processo, elaboraram uma compreensão sobre o Pensamento Computacional fundamentada em três perspectivas-chave. Segundo os autores,

[...] ao estudar a atividade na comunidade on-line do Scratch e nos workshops do Scratch, desenvolvemos uma definição de pensamento computacional que envolve três dimensões principais: conceitos computacionais (os conceitos que os designers empregam enquanto programam), práticas computacionais (as práticas que os designers desenvolvem enquanto programam) e perspectivas computacionais (as perspectivas que os designers formam sobre o mundo ao seu redor e sobre si mesmos) (Brennan; Resnick, 2012, p. 3, tradução nossa).

Os conceitos computacionais referem-se aos processos envolvidos na construção e no refinamento de projetos interativos. Em Brennan e Resnick (2012), esses conceitos são identificados como elementos fundamentais mobilizados durante o desenvolvimento de projetos no Scratch, sendo apresentados da seguinte forma:

Incrementalidade e Iteração: o desenvolvimento de projetos raramente segue um caminho linear. Programadores iniciantes e experientes costumam trabalhar de forma incremental, testando pequenas mudanças e ajustando suas soluções continuamente.

Teste e Depuração: os erros são uma parte natural do processo de programação. Testar o código, identificar falhas e corrigi-las é essencial para aprimorar soluções e desenvolver um pensamento lógico refinado.

Reutilização e Remixagem: a programação não precisa partir do zero. A possibilidade de reutilizar e adaptar trechos de código de outras fontes incentiva a colaboração e a aprendizagem por meio da modificação de projetos existentes.

Abstração e Modularização: para lidar com problemas complexos, é necessário decompor grandes tarefas em partes menores e reutilizáveis. A modularização permite organizar o código de forma mais eficiente, facilitando sua compreensão e manutenção.

As perspectivas computacionais são relativas às mudanças na forma como os indivíduos veem a tecnologia, a si mesmos e o mundo ao seu redor. Em Brennan e Resnick (2012) identifica-se três perspectivas computacionais:

Expressividade: os aprendizes começam a perceber a programação como uma ferramenta para a expressão criativa, não apenas como um conjunto de regras a serem seguidas. A programação se torna um meio para contar histórias, criar jogos e visualizar ideias.

Conexão: o Pensamento Computacional não é uma atividade isolada. O envolvi-

mento em comunidades online, como a do Scratch, permite que os alunos compartilhem seus projetos, aprendam com os outros e colaborem na criação de novos conteúdos.

Questionamento: ao desenvolver habilidades computacionais, os aprendizes passam a enxergar a tecnologia de forma mais crítica, entendendo que os sistemas digitais são criados e podem ser modificados. Eles começam a questionar como os dispositivos funcionam e como podem usá-los para resolver problemas e inovar.

Os conceitos computacionais envolvem os principais recursos utilizados em programação e no desenvolvimento de projetos interativos. Em Brennan e Resnick (2012) identifica-se os seguintes conceitos computacionais:

Sequência: um conjunto de instruções organizadas em uma ordem específica para que o computador as execute corretamente, semelhante a um roteiro ou receita.

Laço (loop): a repetição de um conjunto de instruções, permitindo que ações sejam realizadas múltiplas vezes sem a necessidade de escrever comandos repetitivos.

Paralelismo: a capacidade de executar múltiplas sequências de instruções simultaneamente, essencial para criar interações complexas em ambientes digitais.

Evento: a lógica de causa e efeito em programas, em que certas ações são desencadeadas por eventos específicos, como o clique do usuário ou a colisão entre objetos.

Condiciona: estruturas que permitem a tomada de decisões dentro de um programa, determinando diferentes caminhos de execução com base em condições estabelecidas.

Operadores: blocos que realizam operações matemáticas e lógicas, possibilitando a manipulação de números, variáveis e expressões dentro de um código.

Dados: O uso de variáveis e listas para armazenar, recuperar e modificar informações, permitindo o desenvolvimento de programas dinâmicos e personalizados.

A perspectiva de Pensamento Computacional em Brennan e Resnick (2012) vai além do ensino de programação tradicional, ou seja, vai além ensinar a codificar em uma linguagem de programação específica. Seu modelo enfatiza não apenas os conhecimentos técnicos, mas também as práticas e as perspectivas que tornam o Pensamento Computacional uma ferramenta importante para a aprendizagem e para a resolução de problemas em diversas áreas. Ao integrar essas três dimensões, Brennan e Resnick (2012) argumenta que os alunos se tornam não apenas consumidores de tecnologia, mas também criadores ativos e críticos do mundo digital.

A partir desse ponto do texto nos debruçamos em analisar cada um dos conceitos computacionais em paralelo a nossa perspectiva de Pensamento Computacional, descrita anteriormente na seção sobre o tema.

2.4.1 Sequências

Entre os conceitos do Pensamento Computacional, destaca-se a noção de sequência, entendida como a organização ordenada de ações necessárias para a realização de uma tarefa.

Para Brennan e Resnick,

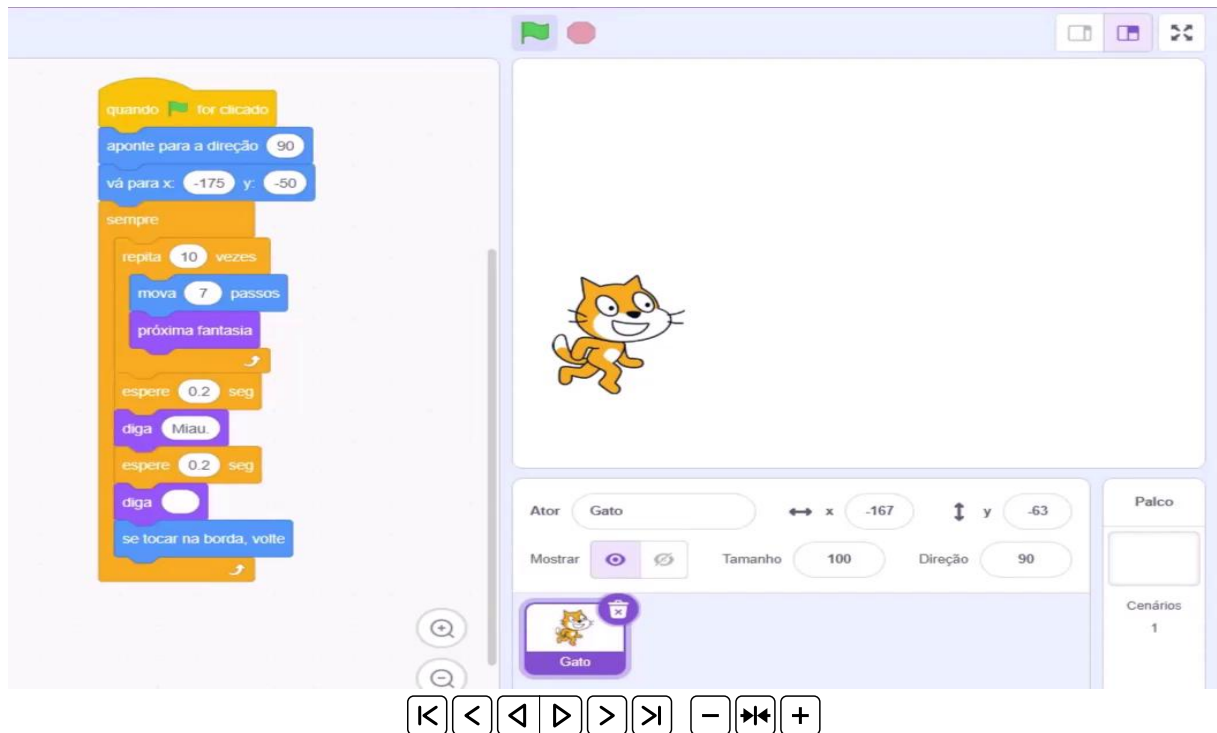
Um conceito fundamental na programação é que uma atividade ou tarefa específica é expressa como uma série de etapas ou instruções individuais que podem ser executadas pelo computador. Como uma receita, uma sequência de instruções de programação específica o comportamento ou a ação que deve ser produzida. (Brennan; Resnick, 2012, p. 03).

Considere a criação, no Scratch, de uma animação com repetição infinita em que um gato se desloca pelo palco, interrompe o movimento por alguns segundos, emite um miado e retoma a caminhada, reiniciando continuamente esse ciclo de ações. Sempre que o personagem encostar em uma das bordas verticais do palco, ele deve se virar para o lado contrário e continuar executando a mesma sequência. Para organizar esse conjunto de ações encadeadas e sucessivas, utiliza-se o conceito de sequência, que permite estruturar as etapas de forma integrada e ordenada.

Nessa elaboração, isto é, ao descrever em linguagem materna uma determinada situação, já é possível identificar a mobilização do conceito de sequência, uma vez que, ao expressar em palavras o que deverá acontecer no Scratch, tornam-se explícitos processos que precisam ocorrer em uma ordem específica. Desse modo, o conceito de sequência participa da própria **formulação do problema**, assim como o ato de formular um problema pode demandar a compreensão e o uso dessa noção de sequência.

Na Figura 2.5 observa-se a disposição dos blocos no Scratch, organizados de modo a possibilitar a resolução do problema proposto.

Figura 2.5: Animação no Scratch



Fonte: Elaborada pela autora (2026)

No Scratch, criou-se um ator e organizou-se um conjunto de comandos que determinam o estado inicial, estruturam um ciclo contínuo de repetição (Loop), combinam deslocamentos com alterações de aparência, pausas temporais e emissão de sons. A disposição desses blocos não ocorreu ao acaso, mas foi cuidadosamente planejada para atender ao que estava estabelecido na formulação do problema.

Os blocos iniciais, “alterar a direção para 90 graus” e “vá para x – 175 e y – 59”, têm a função de posicionar o gato no ponto de partida e orientá-lo na direção em que realizará sua caminhada. Em seguida, é iniciado um laço infinito (“sempre”), isto é, um conjunto de blocos de comandos que repetirá as ações internas enquanto o Scratch estiver em execução.

Os blocos inseridos no interior desse controle realizam o deslocamento do ator e, simultaneamente, efetuam a “troca de fantasias”, gerando o efeito visual de caminhada. Além disso, há blocos que estabelecem um intervalo de tempo, suspendendo brevemente a execução do programa e, na sequência, emitindo o som de miado.

A construção no Scratch, um segundo processo de Pensamento Computacional foi utilizado na sequência de blocos que acabamos de descrever, ou seja, a decomposição. Observa-se que as ações realizadas pelo ator no palco consistem em deslocar-se, alternar fantasias, interromper momentaneamente o movimento e emitir som. Para essas ações, foram atribuídos comandos particulares, organizados de modo sequencial, mostram que quem codificou realizou a decomposição de uma ação mais complexa em partes meno-

res. Em outros termos, a decomposição consistiu em estruturar um conjunto de ações elementares que, ao serem executadas em determinada ordem, resultam na ação de miar enquanto caminha.

A ação de caminhar é estruturada pela articulação de dois blocos: “mova 7 passos” e “próxima fantasia”. O primeiro comando realiza o deslocamento do ator na direção para a qual está orientado (90 graus). Em seguida, o bloco “próxima fantasia” altera sua aparência, apresentando uma nova posição das pernas. Essa alternância, imediatamente posterior ao movimento, cria a representação do ato de caminhar.

A integração entre deslocamento e mudança de fantasia constitui, portanto, uma abstração desse movimento. Ao acionar o comando “repetir”, essa abstração passa a ser executada sucessivamente, gerando o efeito visual de uma caminhada contínua.

A produção do algoritmo manifesta-se pela automação dessa abstração. Desse modo, a combinação dos blocos e o controle do fluxo por meio de estruturas de repetição resultam na solução proposta, configurando a codificação, ou seja, o algoritmo que produz a caminhada intercalada por miados do ator.

2.4.2 Loop (laço)

Segundo Brennan e Resnick (2012, p. 03), “*Loops* são mecanismos para executar uma mesma sequência múltiplas vezes”. Isso significa que, em determinadas situações, torna-se necessário repetir parte de um conjunto de blocos a fim de alcançar um efeito visual específico, como a emissão de um som que demanda diferentes entradas de dados. Nessas circunstâncias, em vez de duplicar blocos idênticos, recorre-se a uma estrutura de controle de repetição.

Na ação do gato que se desloca enquanto mia, foram empregados dois blocos de repetição. O primeiro, conforme já discutido, gera o movimento ao repetir dez vezes os comandos “mova 7 passos” e “próxima fantasia”. O segundo assegura a repetição contínua de um conjunto mais amplo de blocos enquanto o “programa” estiver em funcionamento, por meio do controle “sempre”. Em ambas as situações, verifica-se a automatização de abstrações previamente estabelecidas durante o processo de resolução.

No exemplo apresentado na Figura 2.6 o ator solicita ao usuário do “programa” digite um número que seja, simultaneamente, múltiplo de 3 e de 4.

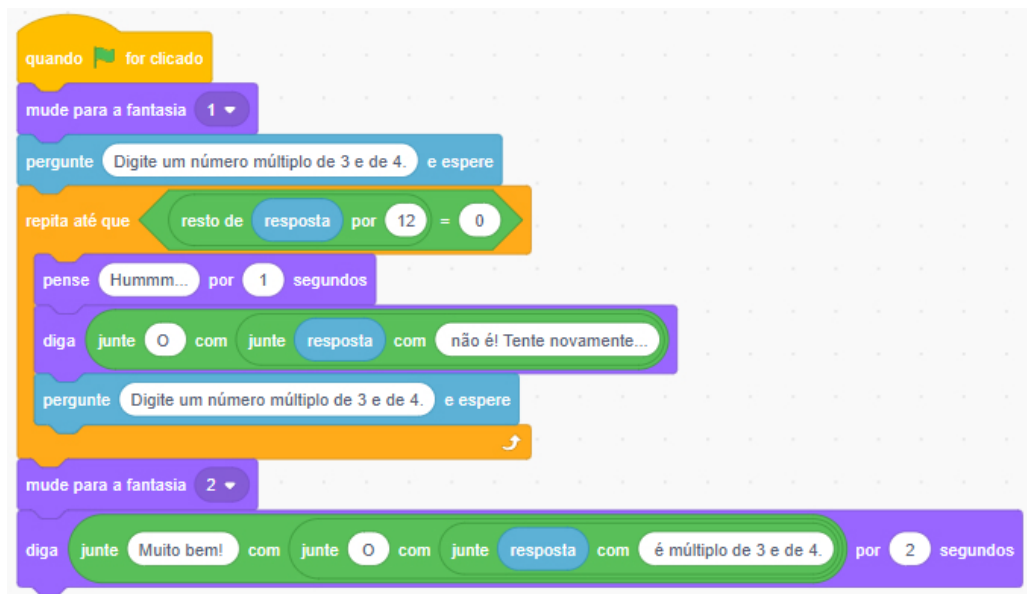
Figura 2.6: Interface de um projeto no Scratch.



Fonte: Elaborada pela autora (2026).

Enquanto essa condição não for satisfeita, o ator continuará realizando solicitações sucessivas e não avançará para o conjunto de códigos posicionado após o bloco de repetição. Trata-se, portanto, de uma estrutura cujo número de execuções não é previamente determinado, pois está condicionado à interação do usuário com o objeto.

Figura 2.7: Bloco de comandos empregando o conceito computacional loop.



Fonte: Elaborada pela autora (2026).

A interatividade incorporada ao objeto, tal como prevista pelo programador, revela a atenção a uma ação específica do usuário, assim como a antecipação de uma possível dificuldade e a definição de um encaminhamento. Essa elaboração torna-se possível a partir da decomposição de um problema em partes menores, que podem ser analisados e

tratados de forma articulada.

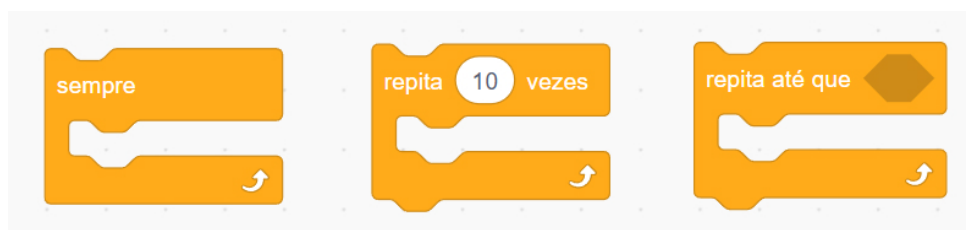
Figura 2.8: Resposta do programa ao concluir a execução do bloco de comandos.



Fonte: Elaborada pela autora (2026).

O Scratch apresenta três possibilidades de controles de repetições em comandos com funções específicas.

Figura 2.9: Exemplo de conceitos computacionais.



Fonte: Elaborada pela autora (2026).

Cada um desses blocos cumpre uma função específica no controle das repetições de comandos ou de conjuntos de comandos. A opção de escolher determinado bloco está relacionada ao efeito que se deseja alcançar, manter a execução contínua sem interrupção, repetir uma ação por um número previamente definido de vezes ou reiterá-la até que condição sejam satisfeitas.

Assim, a seleção de cada estrutura de repetição decorre da decomposição do problema e da elaboração do algoritmo orientado ao resultado pretendido.

Os *loops* podem ser empregados tanto na exploração de padrões quanto nos processos de depuração. No caso da exploração de padrões, o usuário do Scratch constrói uma sequência de comandos que represente um comportamento regular, como, por exemplo, a produção de uma sequência numérica.

No que se refere à identificação de padrões, a análise de um caso específico muitas vezes pode revelar que o conjunto de comandos realiza a ação pretendida. Entretanto, isso não assegura que a sequência produzirá o resultado esperado de forma reiterada.

Nesse contexto, o uso de um *loop* pode contribuir tanto para a identificação quanto para a construção de um padrão.

No que diz respeito à depuração, um *loop* pode ser empregado para reiterar um conjunto de comandos enquanto determinadas variáveis são alteradas. Desse modo, torna-se possível observar os efeitos dos valores assumidos sobre o resultado visual pretendido.

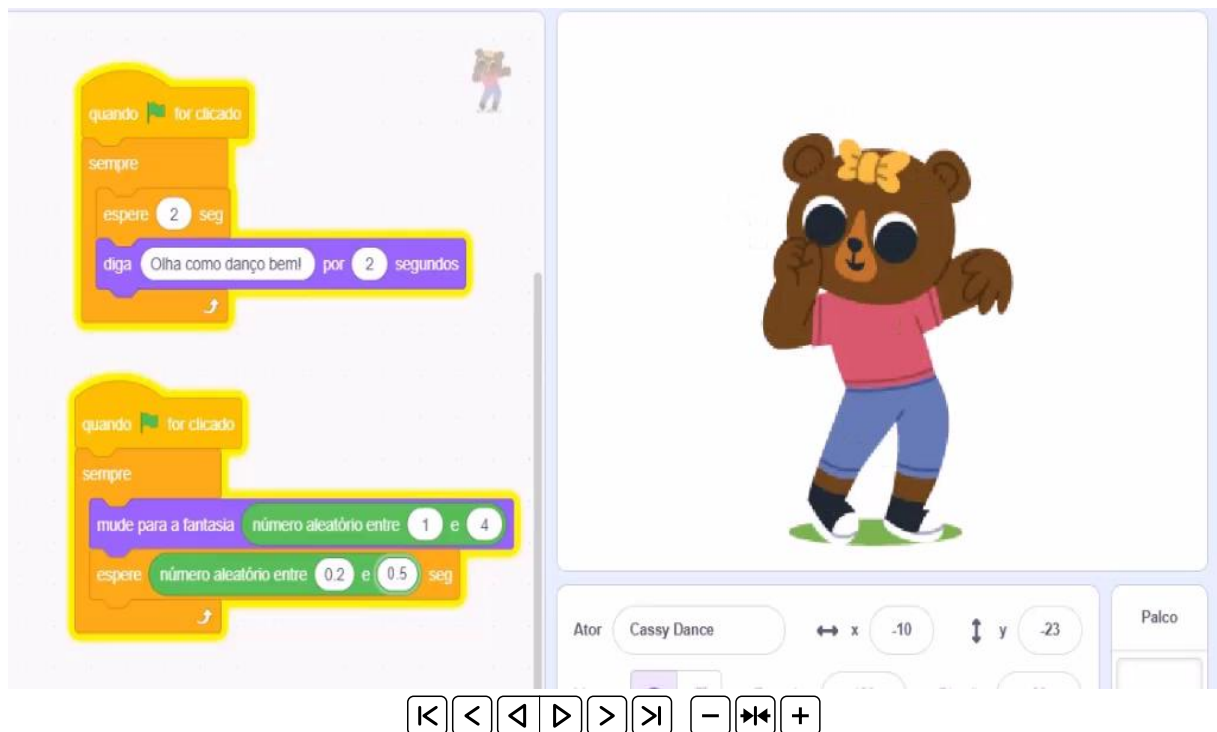
Paralelismo

De acordo com Brennan e Resnick (2012, p. 4), “A maioria das linguagens de computador modernas suporta paralelismo – sequências de instruções acontecendo ao mesmo tempo”.

A ideia de paralelismo envolve o acontecimento de diferentes eventos simultaneamente, e um acontecimento não interfere no outro. Segundo Silva (2019, p. 19), paralelismo se define em “organizar recursos com o fim de realizar tarefas simultaneamente com o intuito de alcançar um objetivo comum”.

Considere a necessidade de construir um arquivo no Scratch que apresente a animação de uma bailarina que dança sem parar ao mesmo tempo que diz em alguns momentos: “Veja como eu danço bem!”.

Figura 2.10: Animação no Scratch



Fonte: Elaborada pela autora (2026)

No que diz respeito à formulação do problema, o dançarino precisa executar dois comportamentos simultaneamente (dançar e falar). Isso envolve identificar quais ações são necessárias e quais blocos do Scratch permitem essa execução. Assim, na identificação de

tais necessidades e no estabelecimento de objetivos se manifesta a formulação do problema via a compreensão do paralelismo.

O comportamento do dançarino foi separado em dois blocos distintos: um para a animação da dança (mudança de fantasia e pausas aleatórias) e outro para a fala (esperar e exibir mensagem). Isso se traduz em uma decomposição, o que facilita a organização do código e a compreensão das ações.

Nos blocos no Scratch, observa-se o uso de repetições padronizadas. A dança segue um ciclo contínuo de mudança de fantasias e tempos de espera, enquanto a fala ocorre em intervalos regulares. Esse reconhecimento de padrões ajuda a criar um fluxo lógico no programa.

A abstração aparece, nas ações de quem programou, certamente, ao se concentrar nos elementos essenciais da dança e da fala, sem se preocupar com detalhes desnecessários. Daí, foram criadas duas sequências de instruções independentes e simultâneas: uma para a animação e outra para a fala, o que corresponde à produção de algoritmos.

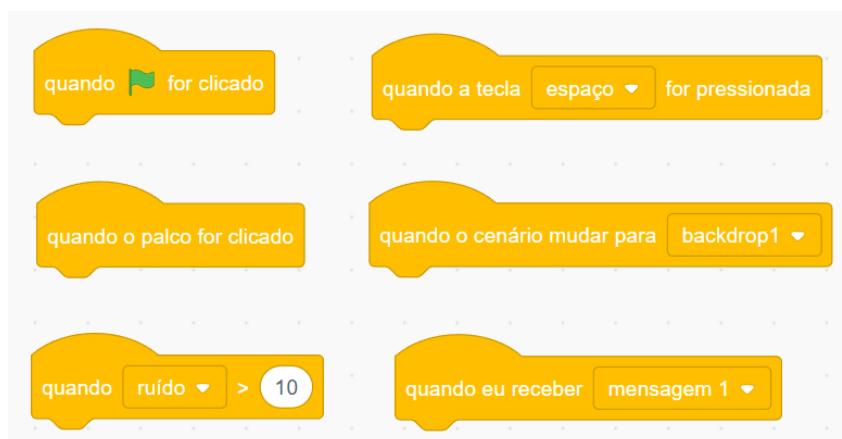
Por último, a depuração, embora não aparente na configuração final dos blocos no Scratch, acontece durante a construção do projeto. Foi necessário testar e ajustar os tempos de espera e os comandos para que os movimentos e falas do dançarino ficassem naturais.

2.4.3 Eventos

Um evento é uma ação ou uma condição que dispara a execução de um comando ou um bloco de comandos. Em outras palavras, ele pode ser interpretado como um gatilho que faz com que algo programado aconteça.

Na Figura 2.11 são exibidos alguns “topos” para comandos que são úteis para tratar eventos de usuário ou de objetos em um projeto no Scratch.

Figura 2.11: Cabeçalhos de eventos no Scratch.

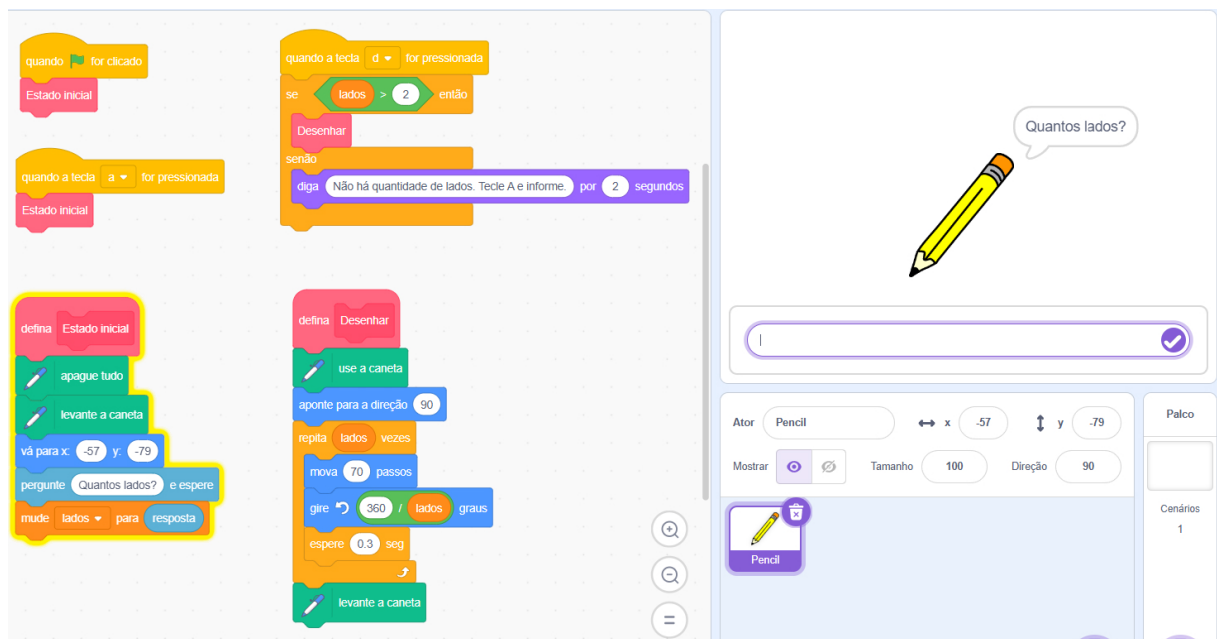


Fonte: Elaborada pela autora (2026).

A identificação dos eventos presentes em um projeto no Scratch e o tratamento desses eventos por meio de conjuntos de comandos possibilitam a produção de interatividade na construção, como ocorre, por exemplo, na interação entre um jogo e o usuário.

Considere um projeto no Scratch que constrói polígonos regulares a partir de comandos do usuário. Ao pressionar a tecla A, apaga-se o que estiver em tela e solicita-se a quantidade de lados do polígono a ser construído. Ao pressionar a tecla D o polígono é construído com os parâmetros informados ou dispara-se uma mensagem sobre a impossibilidade da construção.

Figura 2.12: Projeto no Scratch para construção de polígonos regulares.



Fonte: Elaborada pela autora (2026).

Quanto à formulação do problema, o objetivo central do projeto é criar um sistema interativo onde o usuário pode escolher o número de lados e, com base nisso, o Scratch desenha um polígono regular correspondente. Para isso, devem ser definidos os eventos que iniciam a coleta de dados e o processo de desenho.

O projeto é estruturado em diferentes partes:

- Estado inicial: define a posição do lápis e apaga qualquer desenho anterior.
- Entrada de dados: o usuário informa a quantidade de lados do polígono.
- Decisão condicional: se o número de lados for válido (maior ou igual a 2), o desenho ocorre.
- Execução do desenho: o Scratch desenha o polígono com base nos lados informados.

Essa segmentação foi utilizada na construção do projeto e caracteriza o que em Dantas (2023) chama de decomposição. Cada parte do código tem um papel bem definido e opera de forma independente e dão características modulares ao projeto e são distribuídas em eventos.

Quanto ao reconhecimento de padrões, como exemplo, o projeto aproveita um padrão geométrico importante: em todos os polígonos convexos a medida do ângulo externo corresponde a 360 graus divididos pela quantidade de lados. Tal cálculo é realizado no bloco “Desenhar” que é chamado ao se pressionar a tecla D (desenhar), ou seja, no interior de um evento disparado pelo usuário.

Ao solicitar que um polígono seja construído tecendo D, o projeto no Scratch realiza uma testagem: “Há valores de entrada maiores que 2?”. Caso haja, o evento disparado chama-se o bloco “Desenhar” que realiza uma construção genérica, ou seja, não é a lógica do desenho de um polígono específico, mas sim de um modelo genérico de polígono regular. Isso significa que o código abstrai a estrutura geométrica e pode ser usado para desenhar um triângulo, um quadrado, um pentágono, etc., apenas alterando o número de lados.

Quanto a produção de algoritmos, o tratamento de eventos usado no bloco acionado ao se pressionar a tecla D, por exemplo, inicia um algoritmo de repetição:

- o lápis se posiciona corretamente;
- ele desenha o polígono movendo-se um número fixo de passos;
- a cada passo, o lápis gira o ângulo calculado para formar o polígono;
- após completar o ciclo, o lápis é levantado para evitar traços indesejados.

O código inclui uma verificação para garantir que o usuário insira um valor válido para o número de lados. Caso contrário, uma mensagem aparece alertando: "Não há quantidade de lados. Tecele A e informe". Esse processo de depuração evita erros e melhora a experiência do usuário, garantindo que apenas valores corretos sejam utilizados.

Portanto, o tratamento dos eventos se torna um norte para a operacionalização dos processos mentais constituintes do Pensamento Computacional, segundo Dantas (2023).

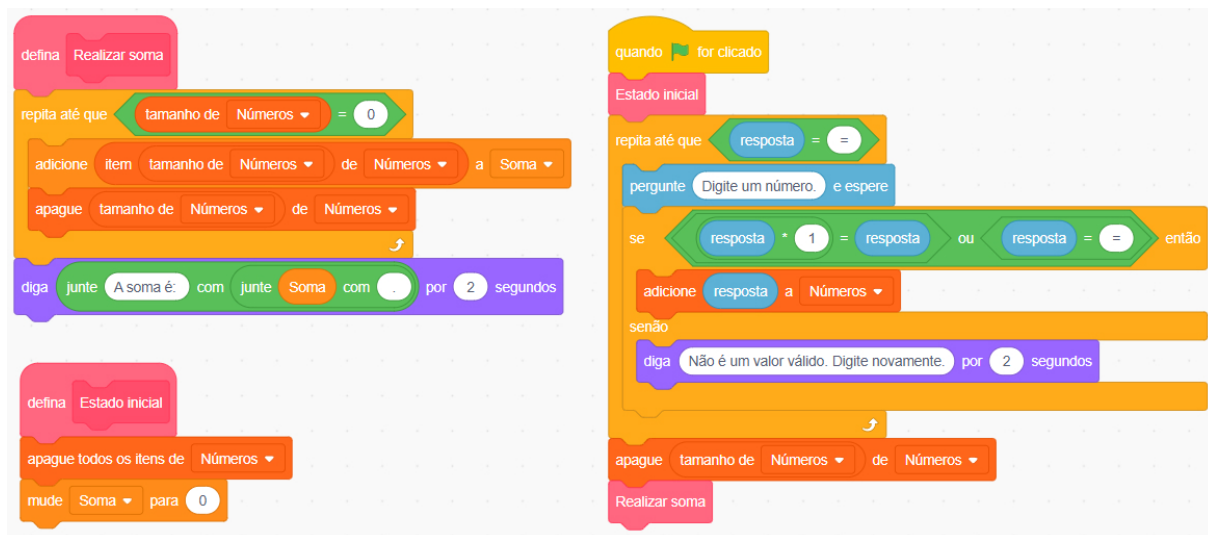
2.4.4 Condicionais

Segundo Brennan e Resnick (2012), os condicionais correspondem a um conceito computacional que permite a tomada de decisões e as condições podem ser baseadas em expressões matemáticas, expressões lógicas ou operações de textos.

Considere a construção de um projeto no Scratch que calcule a soma de uma quantidade indeterminada de números. Em outras palavras, um usuário inicia o programa e passa a digitar valores. Ao concluir sua digitação solicita a soma desses valores.

Uma possível resolução para esse problema pode ser um projeto composto por uma personagem e uma lista. A personagem interage com o usuário solicitando a digitação de novos números e a lista armazena os valores numéricos. Esse projeto pode ser constituído pelos seguintes blocos de comandos.

Figura 2.13: Cabeçalhos de eventos no Scratch.



Fonte: Elaborada pela autora (2026).

O projeto apresentado na Figura 2.13 implementa um sistema interativo para a soma de números inseridos pelo usuário. Dentro dessa lógica, estruturas condicionais são utilizadas para validar entradas e controlar o fluxo do programa. Há três funções básicas: validação da entrada do usuário, repetição até solicitação da soma pelo usuário e cálculo da soma.

Para validação das entradas do usuário há um teste, em um condicional, que verifica se o número não contém letras. Contendo letras o usuário recebe a mensagem "Não é um valor válido. Digite novamente". Caso seja um valor válido, o programa adiciona o valor a uma lista (Números) e solicita o próximo valor, ou seja, entra em um processo de repetição. Isso controla a interação com o usuário, permitindo que ele adicione quantos números desejar antes de obter a soma e finalizar o programa.

Para o cálculo da soma utiliza-se um controle de fluxo "Repita até que". Embora esse bloco represente um repetidor de ações, ele só o fará se condições forem satisfeitas. No caso específico do projeto no Scratch, ele soma valores da lista enquanto eles existirem, pois à medida que adiciona um valor a variável "Soma" o bloco de comandos deleta esse valor da lista "Números". Essa mesma estratégia de controle de fluxo, "Repita até que", foi utilizada na entrada dos valores para compor a lista a ser somada, ou seja, os valores numéricos são adicionados à lista "Números" até que o usuário tecle "=". Ao fazer isso, o fluxo do programa vai para o bloco "Realizar soma" e o personagem diz "A soma é ##."

Quanto à formulação do problema, o projeto no Scratch para soma de números,

consiste na identificação dos objetivos do programa e das etapas necessárias para sua resolução. No projeto em análise, o desafio proposto é criar um sistema que: solicite números ao usuário até que um critério de parada seja acionado; valide se as entradas são valores numéricos, impedindo a inserção de caracteres não numéricos; some os valores informados e exiba o resultado final na tela. A integração de condicionais na formulação do problema permite separar cada etapa necessária à sua resolução.

Quanto à decomposição, o projeto analisado apresenta uma estrutura modular organizada em três principais etapas:

- inicialização: definição do estado inicial, em que a variável que armazenará a soma é zerada e a lista de números inseridos é esvaziada.
- entrada de dados: um loop solicita ao usuário a inserção de números, validando cada entrada com condicionais para evitar erros.
- processamento e saída: após o término da entrada de dados, os valores são somados e o resultado é exibido na tela.

O reconhecimento de padrões auxilia na identificação de regularidades e na criação de soluções que possam ser generalizadas. No projeto, esse processo ocorre nas seguintes instâncias:

- o critério de parada do loop de entrada de dados é um padrão: o programa encerra as solicitações quando o usuário digita "=", caracterizando um comportamento previsível.
- a verificação de caracteres inválidos (letras) na entrada do usuário segue um padrão lógico que evita falhas no cálculo da soma.
- o uso da lista de números permite que os valores sejam armazenados e manipulados de forma homogênea, facilitando a soma ao final.

Quanto à abstração, no projeto, esse processo se manifesta em estratégias envolvendo condicionais, tais como:

- a aplicação de um bloco condicional que verifica a validade da entrada, sem precisar conferir cada número individualmente.
- a implementação de uma estrutura genérica de repetição que adiciona novos números à lista, independentemente da quantidade de valores inseridos pelo usuário.

A fase de produção de algoritmos e depuração envolve a implementação do código e a correção de erros para garantir seu funcionamento adequado. No projeto analisado, observa-se que:

- o uso de condicionais impede a entrada de caracteres inválidos, reduzindo erros na soma final.
- a estrutura do programa inclui um mecanismo de exibição de mensagens de erro quando o usuário insere um valor não numérico, permitindo um feedback instantâneo.

Portanto, segundo Dantas (2023), a utilização de condicionais na resolução de problemas computacionais pode-se caracterizar como um conceito computacional relevante para trazer à tona processos mentais constituintes do Pensamento Computacional.

No capítulo subsequente, serão analisadas as questões do ENEM resolvidas tanto por meio de procedimentos matemáticos quanto com o uso do Scratch, com o objetivo de evidenciar os processos constituintes do Pensamento Computacional mobilizados nessas resoluções.

3 DESENVOLVIMENTO METODOLÓGICO

Neste capítulo, apresentamos o método de análise adotado na pesquisa, cujo objetivo é mobilizar os processos constituintes do Pensamento Computacional mobilizados na resolução de questões do ENEM. Para isso, a análise fundamentou-se em duas produções elaboradas pela pesquisadora em conjunto com seu orientador: o objeto construído no Scratch e o vídeo que registrou integralmente o processo de resolução. Esses materiais permitiram analisar tanto o produto quanto o percurso de construção da solução.

O objeto desenvolvido no Scratch corresponde à materialização da solução proposta, sintetizando as decisões e ações realizadas ao longo da resolução do problema. Já o vídeo documentou o processo em sua dimensão dinâmica, registrando interações, discussões, ajustes e reformulações ocorridas durante a elaboração da solução, o que possibilitou uma compreensão detalhada do raciocínio envolvido.

A análise foi conduzida a partir dos seis processos do Pensamento Computacional, utilizados como referência teórica: formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração. Embora apresentados de forma analítica e distinta, esses processos apresentam-se, na prática, de maneira articulada e simultânea, o que impõe desafios à identificação e categorização das ações observadas.

Dessa forma, o procedimento analítico consiste em dissecar o processo de resolução, buscando identificar momentos em que cada um desses processos é mobilizado, mesmo quando eles se encontram sobrepostos. O uso combinado do objeto final e do registro em vídeo possibilitou essa leitura complementar, uma vez que a construção realizada no Scratch, por si só, não revela integralmente as decisões, impasses e correções que emergiram durante a resolução.

No que se refere à formulação do problema, a análise busca identificar momentos em que o enunciado é interpretado e reinterpretado, explicitando o que se deseja resolver, quais informações são relevantes e quais objetivos orientam a construção da solução. A decomposição é analisada a partir da divisão do problema em partes menores, evidenciada tanto na organização do código quanto nas discussões que fragmentam a tarefa em subtarefas mais manejáveis. O reconhecimento de padrões manifesta-se na identificação de regularidades, semelhanças entre situações ou estruturas recorrentes, frequentemente utilizadas para generalizar estratégias ou simplificar a resolução. A abstração é observada quando há seleção de informações essenciais e descarte de detalhes considerados irrelevantes, seja na representação do problema no Scratch, seja nas decisões registradas no vídeo. A produção de algoritmos corresponde à elaboração de sequências de comandos, regras ou procedimentos sistemáticos que orientam a resolução, tornando explícito o passo a passo adotado. Por fim, a depuração é identificada nos momentos de reformulação ou reorga-

nização das ideias, o que se traduz em teste, identificação de erros, ajustes e correções realizadas ao longo do processo, tanto no código e seus arranjos quanto nas estratégias matemáticas mobilizadas.

Reconhece-se que o caráter não linear do Pensamento Computacional dificulta sua análise segmentada. Ainda assim, essa sistematização é necessária para responder à questão central da pesquisa: "Como os processos do Pensamento Computacional: formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração: são mobilizados durante a resolução de questões de Matemática do ENEM a partir da construção de modelos computacionais no Scratch?"

A identificação desses processos foi realizada a posteriori, com base na análise conjunta do objeto construído no Scratch e do vídeo que registra o processo completo de resolução matemática. Espera-se, com isso, contribuir com subsídios teóricos e metodológicos para educadores e pesquisadores interessados em compreender e promover o Pensamento Computacional no âmbito da Educação Matemática.

3.1 Definição de Pesquisa

A pesquisa científica pode ser caracterizada como um método estruturado, que segue diversos procedimentos bem organizados os quais servem como base para alcançar o objetivo; tal maneira auxilia na compreensão acerca do(s) objeto(s) investigado(s) durante todo processo. Para Gil, ela pode ser compreendida como:

[...] procedimento racional e sistemático que tem como objetivo proporcionar respostas aos problemas que são propostos. A pesquisa é requerida quando não se dispõe de informação suficiente para responder ao problema, ou então quando a informação disponível se encontra em tal estado de desordem que não possa ser adequadamente relacionada ao problema (Gil, 2002, p. 17).

Dessa maneira, pesquisar é explorar perguntas que surgem em relação a um evento específico, buscando compreender logicamente o significado dessas ocorrências. Utilizando-se de métodos científicos para alcançar os resultados pretendidos.

Para alcançar o objetivo da pesquisa, realizamos uma análise minuciosa das questões de Matemática do ENEM, selecionando e resolvendo aquelas mais adequadas ao escopo do estudo. Em seguida, examinamos criticamente nossas próprias resoluções, buscando identificar de que maneira os processos constituintes do Pensamento Computacional foram mobilizados. Essa reflexão analítica permitiu compreender como tais processos - formulação do problema, decomposição, reconhecimento de padrões, abstração, construção de algoritmos e depuração - se manifestam nas práticas de resolução, contribuindo para caracterizar os processos que definem o Pensamento Computacional no contexto deste trabalho.

3.2 O Exame Nacional do Ensino Médio

O ENEM é uma avaliação de âmbito nacional criado em 1998 pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP) com o propósito de anualmente avaliar o desempenho de alunos concluintes e egressos do Ensino Médio (Brasil, 2002). O documento ainda complementa que esse exame pretende verificar as competências fundamentais que estão sendo desenvolvidas na Educação Básica e que são essenciais para o exercício integral da cidadania, buscando valorizar os conhecimentos do participante para resolver um problema de modo que não haja necessidade de memorizar uma fórmula ou conceito, sendo considerado, então, interdisciplinar e com problemas contextualizados. Essa prova é considerada uma avaliação de larga escala, de acordo com Carlos e Melo (2016) por se interessar em melhorar a qualidade e conhecer as características da educação brasileira através de questões padronizadas, obtendo assim uma série de informações que podem dar suporte à elaboração de políticas públicas.

O Documento Básico do ENEM (MEC, 2002) indica que a prova do ENEM era constituída por uma parte objetiva com 63 questões e uma proposta de redação, ambas as partes eram avaliadas em uma escala de pontos que variava de 0 a 100, o que Broietti (2013) aponta que durou até o ano de 2008, sendo a nota da prova objetiva o resultado da média aritmética de cada uma das cinco competências que eram avaliadas. No entanto, em 2009 a avaliação sofreu uma mudança que ficou conhecida como Novo ENEM segundo Xavier (2016) e Inep (2019), o exame passou a ser dividido em 4 áreas do conhecimento, sendo elas: Ciências da Natureza e suas Tecnologias; Linguagens, Códigos e suas Tecnologias; Matemática e suas Tecnologias; e Ciências Humanas e suas Tecnologias. A partir dessa divisão, o Enem passou a ser realizado em dois dias.

Um dos motivos para essa reformulação foi a criação do Sistema de Seleção Unificada (SiSU), que é “[...] o sistema informatizado, gerenciado pelo Ministério da Educação (MEC), pelo qual instituições públicas de educação superior oferecem vagas a candidatos participantes do Exame Nacional do Ensino Médio (ENEM)” (SISU, 2019, p. 1), de modo que as universidades públicas e federais possam utilizar a nota do exame como critério para seleção. A prova tem como base em cada área as Matrizes de Referência, segundo a qual, a área de Matemática e suas Tecnologias tem 7 competências e 30 habilidades. Entre as várias disciplinas, a prova de Matemática tem um papel fundamental na avaliação das habilidades de raciocínio lógico dos estudantes.

As provas de Matemática do Enem são compostas por 45 (quarenta e cinco) questões de múltipla escolha, e cinco alternativas (A, B, C, D, E); as questões possuem algumas características de contextualização, interpretação e raciocínio lógico e tópicos. Dentre os conteúdos abordados na prova do Enem de Matemática e suas Tecnologias podemos destacar como mais recorrentes os seguintes assuntos: geometria, escala, razão e proporção, funções, porcentagem, estatística, análise combinatória e probabilidade.

À luz das informações apresentadas e em consonância com os objetivos desta pesquisa, a próxima seção descreve de maneira pormenorizada o processo de seleção das questões do Enem que fundamentarão as análises subsequentes e orientarão o desenvolvimento do estudo.

3.3 Escolha das questões

Em um primeiro momento, decidimos examinar as questões do ENEM, buscando identificar aquelas cuja estrutura problematizadora possibilitasse a mobilização de um ou mais dos seis processos constituintes do Pensamento Computacional, conforme delineados por Dantas (2023), de modo a orientar a elaboração de possíveis estratégias de resolução.

Optamos por questões do ENEM e não de livros didáticos, por exemplo, por terem alcance tanto nas escolas públicas como privadas. Ademais, elas são exploradas em livros didáticos de Ensino Médio e, tanto professores quanto alunos possuem certa familiaridade com essas questões.

A partir dessa escolha realizamos *download* das provas do ENEM dos anos de 2010 a 2022. Passamos, em seguida, a analisar essas provas e selecionar questões que considerávamos possíveis de serem resolvidas no Scratch e utilizando os processos do Pensamento Computacional. Selecionamos as questões apresentadas na Tabela 3.1.

Tabela 3.1: Questões e processos do pensamento computacional

Ano	Número da questão	Caderno	Conteúdo mático	Mate-	Processo do pensamento com- putacional
2017	141	Amarelo	Análise Combinatória		Reconhecimento de padrões, abstração e algoritmo.
2020	176	Cinza	Progressão Geométrica	Geomé-	Reconhecimento de padrões, abstração e algoritmo.
2021	137	Amarelo	Sistemas de numeração, Conversão de base		Decomposição, reconhecimento de padrões, abstração e algoritmo.
2021	139	Amarelo	Análise combinatória		Decomposição, reconhecimento de padrões, abstração e algoritmo.
2021	142	Amarelo	As quatro operações fundamentais e expressões		Decomposição, reconhecimento de padrões, abstração e algoritmo.

Continua na próxima página

Ano	Número da questão	Caderno	Conteúdo matemático	Mate-	Processo do pensamento computacional
2021	143	Amarelo	Porcentagem		Decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	147	Amarelo	Geometria Espacial		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	152	Amarelo	Geometria Espacial		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	155	Amarelo	Função do segundo grau		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	156	Amarelo	Matrizes		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	157	Amarelo	Porcentagem e fração		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	159	Amarelo	Porcentagem e proporção		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	162	Amarelo	Função do segundo e polinômio		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	169	Azul	Probabilidade e Estatística		Formulação do problema, decomposição, reconhecimento de padrões, abstração, algoritmo e depuração.
2021	178	Amarelo	Contagem e agrupamento e combinatória		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2021	179	Amarelo	Proporcionalidade, sequências e progressões		Formulação do problema, decomposição, reconhecimento de padrões, abstração, algoritmo, depuração.
2022	141	Cinza	Geometria e sequência lógica		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.

Continua na próxima página

Ano	Número da questão	Caderno	Conteúdo Matemático	Mate-	Processo do pensamento computacional
2022	146	Amarelo	Probabilidade e Estatística		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2022	176	Cinza	Sequências e progressões		Formulação do problema, decomposição, reconhecimento de padrões, abstração e algoritmo, depuração.
2022	177	Amarelo	Sequências		Decomposição, reconhecimento de padrões e abstração.

Fonte: Elaborada pela autora (2024)

Os números das questões apresentadas na Tabela 3.1 funcionam como hiperlinks que direcionam aos respectivos cadernos de prova do Enem, nos quais as questões podem ser consultadas na íntegra.

3.4 Como as questões foram resolvidas no Scratch?

Esta seção apresenta a descrição detalhada da resolução de quatro questões listadas na Tabela 3.1. Cabe destacar que nem todas as questões selecionadas foram resolvidas; a escolha daquelas que compõem este capítulo decorre de sua pertinência aos objetivos da pesquisa. Assim, priorizamos questões cuja abordagem pudesse ser efetivamente desenvolvida no Scratch, evidenciando os processos do Pensamento Computacional.

Inicialmente, nas reuniões de orientação realizadas via Meet, procedemos à leitura e à análise minuciosa de cada uma das questões elencadas. Em seguida, resolvemos cada questão por meio de abordagens matemáticas. Em alguns casos, exploramos mais de uma forma de resolução, visando evidenciar um processo matemático distinto daquele que seria empregado na resolução fundamentada no Pensamento Computacional. Essas escolhas foram orientadas por nossas conjecturas de que a construção de um modelo computacional para a resolução de um problema, a partir dos processos constituintes do Pensamento Computacional, exige a explicitação de estratégias que não se reduzem aos procedimentos matemáticos tradicionalmente mobilizados.

Posteriormente, pensávamos na resolução do problema com os recursos do Scratch, propondo-nos a identificar os processos do Pensamento Computacional mobilizados por nós durante a construção do projeto.

Em nossos processos de resolução amparada nos processos do Pensamento Computacional, buscamos resolver um problema particular constituído a partir do enunciado lido, porém, em algumas situações foi possível compor uma solução cuja situação correspondesse a uma generalização do problema apresentado no enunciado do Enem.

Explicitaremos tal raciocínio no Capítulo 4.

Em outras palavras, nós constituímos um problema a partir dos enunciados das provas do ENEM cuja problemática poderia ser a própria do ENEM ou uma ampliação dessa problemática para casos gerais.

Segundo as autoras Ribeiro, Foss e Cavaleiro

[...] A matemática não nos ajuda a resolver esse tipo de problema, pois não provê as abstrações necessárias para descrever a solução. Além disso, não é objeto da matemática investigar como construímos um algoritmo. A ênfase do raciocínio, ou pensamento, computacional não está apenas nos produtos em si (provas ou algoritmos), e sim no processo de construção desses produtos. Ou seja, além das abstrações necessárias para descrever algoritmos, o pensamento computacional engloba também técnicas para a construção de algoritmos, que na realidade são técnicas de solução de problemas, as quais podem ser aplicadas em diferentes contextos (Ribeiro; Foss; Cavaleiro, 2020, p. 17).

Assim, nossa preocupação era que nossas resoluções no Scratch resolvessem aquele problema específico apresentado no enunciado de uma prova do ENEM, mas que também servisse como um modelo generalizado para resolver situações semelhantes.

Por último, em nossas resoluções nós compúnhamos um cenário interativo com o usuário em que o “programa” dialogasse com o utilizador solicitando dados do problema e fornecendo respostas.

4 RESOLUÇÃO E ANÁLISE DAS QUESTÕES

Neste capítulo apresentamos as resoluções das questões escolhidas nas provas de Matemática do ENEM dos anos de 2010 a 2022.

Para isso, traremos o enunciado da questão, uma ou mais resoluções utilizando conhecimento matemático¹, a resolução utilizando de Pensamento Computacional, bem como a análise dos processos constituídos à luz de nossos pressupostos teóricos sobre Pensamento Computacional.

4.1 Questão 1

A questão 1 foi retirada da prova do ENEM de 2017 e corresponde a questão de número 141 do caderno amarelo. Segue seu enunciado:

Figura 4.1: Questão 141 do Caderno Amarelo do ENEM de 2017.

Como não são adeptos da prática de esportes, um grupo de amigos resolveu fazer um torneio de futebol utilizando *videogame*. Decidiram que cada jogador joga uma única vez com cada um dos outros jogadores. O campeão será aquele que conseguir o maior número de pontos. Observaram que o número de partidas jogadas depende do número de jogadores, como mostra o quadro:

Quantidade de jogadores	2	3	4	5	6	7
Número de partidas	1	3	6	10	15	21

Se a quantidade de jogadores for 8, quantas partidas serão realizadas?

- A** 64
- B** 56
- C** 49
- D** 36
- E** 28

Fonte: ENEM - 2017.

4.1.1 Primeira resolução utilizando conhecimento matemático

A primeira resolução é a observação de um padrão aritmético no quadro. Notamos que o número de partidas aumenta em 2, 3, 4, 5, ou seja, o acréscimo possui um padrão.

¹Chamamos de conhecimento matemático o conhecimento constituído ao longo da formação na Educação Básica, especialmente na área de Matemática.

Figura 4.2: Quadro de cálculo do número de partidas.

Quantidade de jogadores	2	3	4	5	6	7
Números de partidas	1	3	6	10	15	21



Fonte: Elaborada pela autora (2024).

Assim, para 8 jogadores serão $21+7=28$ partidas.

4.1.2 Segunda resolução utilizando conhecimento matemático

A segunda resolução consiste em identificar que os jogadores quando combinados dois a dois para se enfrentarem constituem agrupamentos em que a ordem não modifica a natureza do agrupamento, ou seja, trata-se de uma combinação de oito elementos dois a dois, logo, a expressão $C_{n,p}$ indica quantas maneiras diferentes podemos escolher p elementos em um conjunto que possui n elementos, quando a ordem não importa.

- n é o total de elementos disponíveis;
- p é a quantidade de elementos que queremos selecionar;
- $C_{n,p}$ é o número total de combinações possíveis.

Dessa forma:

$$C_{n,p} = \frac{n!}{(n-p)!p!}$$

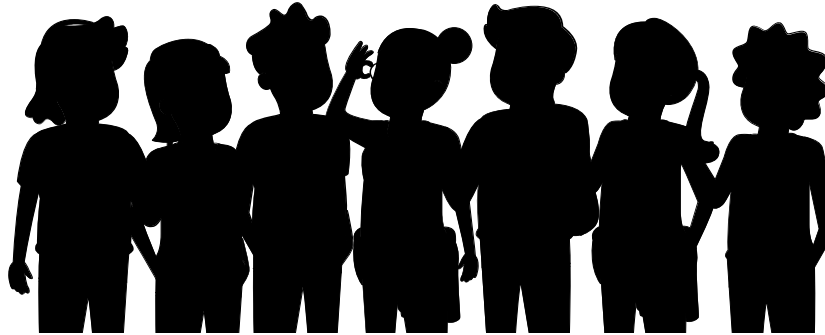
$$C_{8,2} = \frac{8!}{(8-2)!2!} = \frac{8 \cdot 7 \cdot 6!}{6!2!} = 28$$

Portanto, para 8 jogadores serão 28 partidas.

4.1.3 Resolução utilizando de Pensamento Computacional

Primeiramente, gostaríamos de destacar que para a resolução por meio do Pensamento Computacional, nosso intuito foi generalizar uma solução não apenas para o problema proposto, mas para qualquer quantidade de jogadores digitada pelo usuário. Assim, pensamos em um exemplo análogo: o encontro de 20 pessoas em que todas cumprimentaram umas às outras. A pergunta da questão seria quantos cumprimentos ocorreram nessa situação. Correlacionando esta situação com a questão 141 do ENEM de 2017, ao invés de resolver o problema para essas quantidades específicas, consideramos um encontro de n pessoas e resolvemos o problema a partir de qualquer quantidade de pessoas.

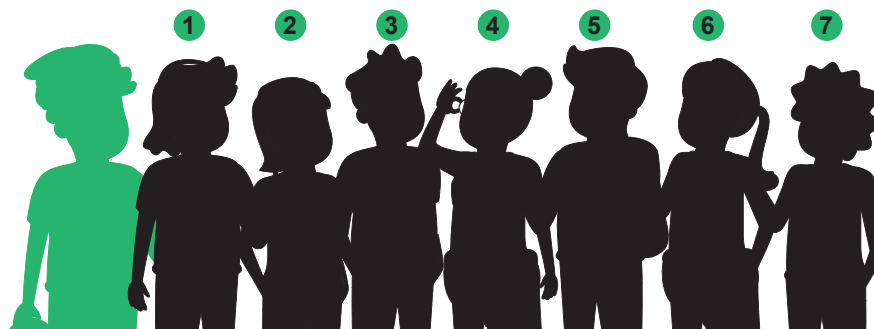
Figura 4.3: Abstração dos jogadores em fila.



Fonte: Adaptado de <https://www.freepik.com>.

Para realizar a contagem de partidas disputadas pelo grupo nos concentramos na contagem das partidas do primeiro jogador com os demais, ou seja, como temos 8 jogadores, o primeiro jogador da fila vai disputar 7 partidas. Armazenamos esse valor em uma variável e passamos para o próximo jogador.

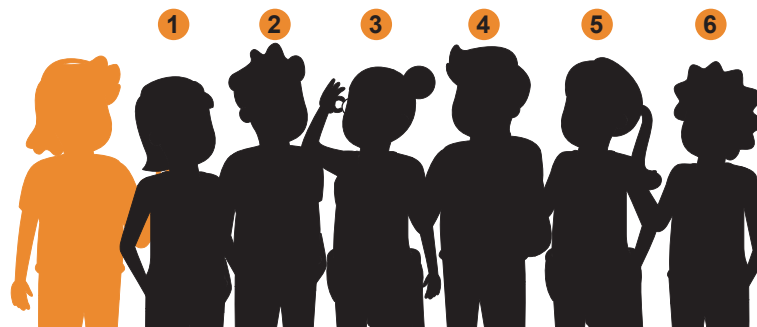
Figura 4.4: Contagem das partidas do primeiro jogador.



Fonte: Adaptado de <https://www.freepik.com>.

O segundo jogador disputará 6 partidas com os demais jogadores que estão posicionados na sequência da fila.

Figura 4.5: Contagem das partidas do segundo jogador com os seguintes.



Fonte: Adaptado de <https://www.freepik.com>.

O valor relativo a essa quantidade de partidas (6) é adicionado ao valor do cálculo anterior (7) e armazenado em uma variável “soma”. A soma final ao realizar esse processo reiteradas vezes até o penúltimo jogador corresponde à quantidade de partidas disputadas.

Essa foi a abstração que idealizamos para a resolução do problema. Porém, no Scratch, a resolução foi generalizada para uma quantidade inteira positiva qualquer de jogadores. Para isso, o primeiro jogador da fila pergunta de início “Quantas pessoas estão no torneio?”.

Figura 4.6: Interface da resolução no Scratch.

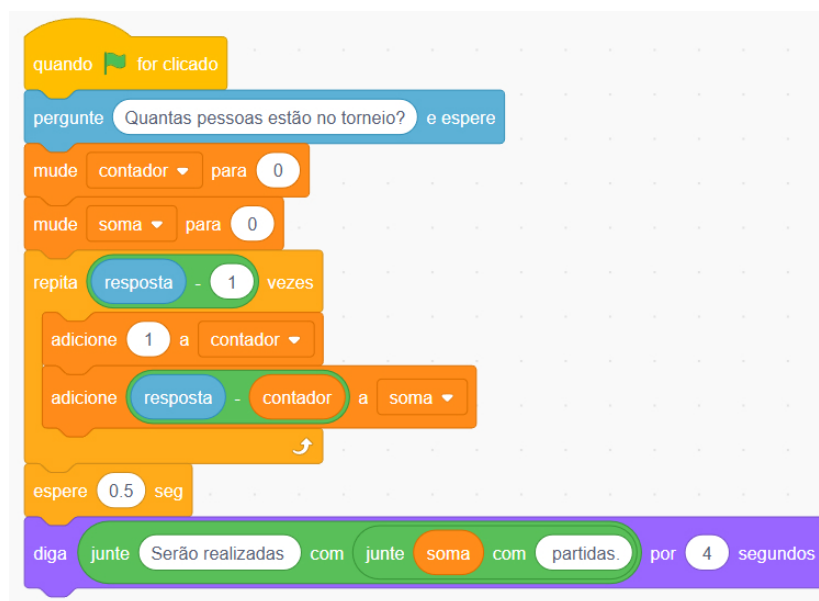


Fonte: Elaborada pela autora (2024).

Após digitar uma resposta numérica, o algoritmo zera duas variáveis (contador e soma). Em seguida, repete “resposta – 1” vezes um trecho de cálculo e apresenta a soma que corresponde a quantidade de partidas disputadas. “Resposta” corresponde ao valor digitado pelo usuário ao responder à pergunta sobre a quantidade de pessoas no torneio.

O algoritmo produzido é o seguinte:

Figura 4.7: Blocos de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

4.1.4 Análise dos processos de Pensamento Computacional utilizados na resolução da questão 1

A primeira questão trata de uma competição em que oito amigos realizam um campeonato de videogame em que disputam partidas se enfrentando mutuamente. No enunciado original do ENEM deseja-se saber a quantidade de partidas que serão disputadas no total quando todos se confrontarem.

Formulação do problema

Em nossa resolução no Scratch concebemos o problema de encontrar a quantidade de partidas nas mesmas condições do enunciado do ENEM, porém, com a possibilidade de o usuário de nosso arquivo no Scratch informar quantas pessoas estão na disputa. Assim, nossa resolução consiste na efetivação de um modelo computacional que oferece ao usuário a possibilidade de escolha de uma quantidade de jogadores e sobre ela realiza o cálculo de partidas realizadas.

Em termos de necessidade, objeto e pergunta, nosso objetivo é realizar um cálculo de partidas que se adapte a qualquer entrada fornecida pelo usuário, ou seja, a nossa pergunta a ser respondida não é uma quantidade fixa de partidas para um número fixo de jogadores, mas compor um modelo computacional sujeito às entradas do usuário.

Decomposição

A decomposição útil para a resolução da questão 1 se deu em quatro partes: entrada das informações, armazenamento das informações, processamento ou realização dos

cálculos necessários e composição de um texto para responder ao usuário.

Como se trata de uma resolução em que o usuário escolhe a quantidade de jogadores, foi necessário desenvolver uma forma de obter tal informação. Nossa escolha foi replicar um cenário de um grupo de pessoas perguntando sobre a quantidade de jogadores participantes na disputa.

Para armazenar esses dados e utilizá-los posteriormente nos cálculos, fizemos uso de três variáveis: *resposta*, *contador* e *soma*. A variável *resposta* guarda a escolha do usuário sobre a quantidade de jogadores. A variável *contador* é útil no controle de ciclos de repetição durante o processo de produção do algoritmo. Por fim, a variável *soma* funciona como um acumulador da quantidade de partidas.

Para o processamento ou cálculo da quantidade de partidas utilizamos um *loop* que realiza cálculos iterados, os quais serão detalhados no processo de produção de algoritmo.

Por fim, a última parte da decomposição obtém os dados calculados e apresenta a resposta ao usuário em uma mensagem textual.

Reconhecimento de padrões e abstração

Conforme abordamos anteriormente, as resoluções matemáticas utilizaram dois padrões diferentes. No primeiro, observou-se um padrão aritmético nos números do quadro do enunciado. No segundo, o agrupamento descrito no enunciado se encaixa em um padrão de contagem chamado combinação e se aplica um algoritmo clássico retirando a quantidade de elementos (8) e a quantidade de elementos dos agrupamentos (2).

Na resolução utilizando de Pensamento Computacional primeiramente abstraímos o grupo de jogadores como compondo uma fila de pessoas. Assim, o primeiro jogador jogaria com todos que estivessem após ele na fila. Em seguida, o segundo jogador faria o mesmo com os demais após ele na fila e assim sucessivamente até o penúltimo jogador.

O reconhecimento de padrões, nesse caso, consiste em conceber filas com 2, 3, 4, 5, ou mais pessoas. Desse modo,

Figura 4.8: Esquema do cálculo do número de partidas.

Pessoas	Fila	Partidas
2	1 – 2	1
3	1 – 2 – 3	2 + 1
4	1 – 2 – 3 – 4	3 + 2 + 1
5	1 – 2 – 3 – 4 – 5	4 + 3 + 2 + 1

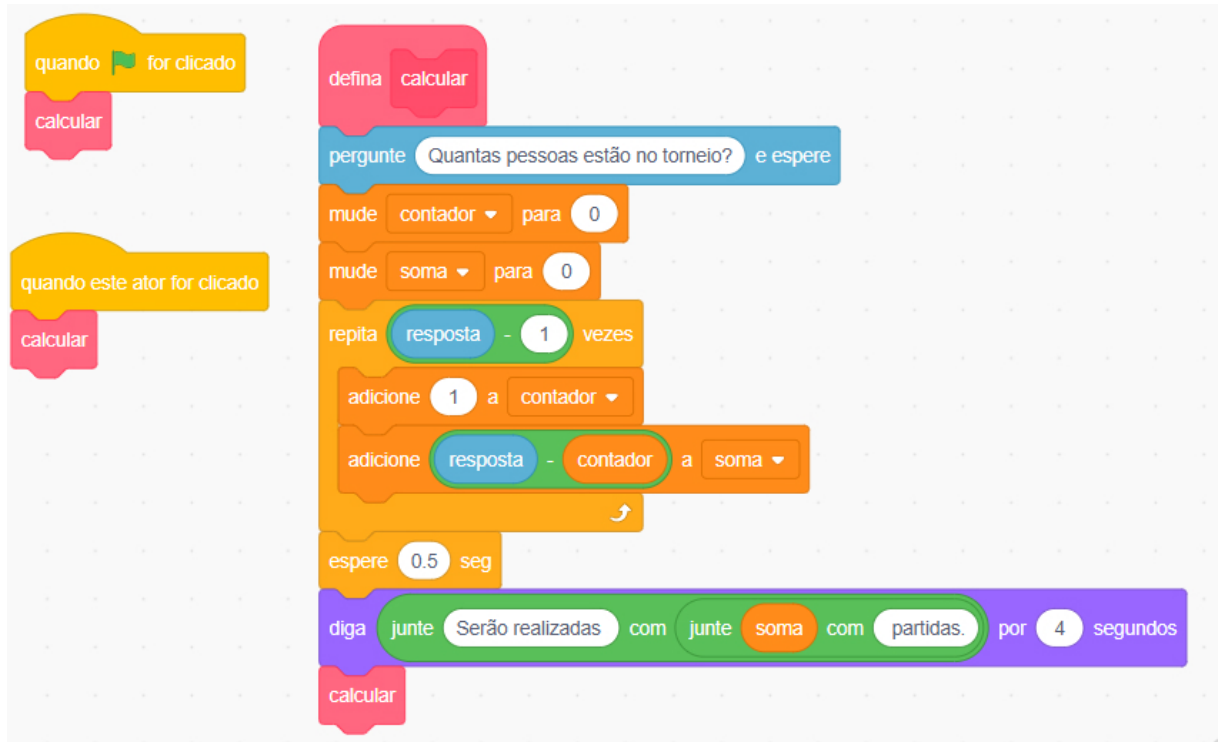
Fonte: Elaborada pela autora (2024).

Assim, o reconhecimento do padrão nos permitiu compor um algoritmo para realizar somas de forma iterada, o que será abordado no texto a seguir.

Produção de algoritmo

Tendo percebido o padrão descrito acima e também de posse da constituição oriunda da decomposição compomos o seguinte algoritmo no Scratch.

Figura 4.9: Blocos de comandos no Scratch.

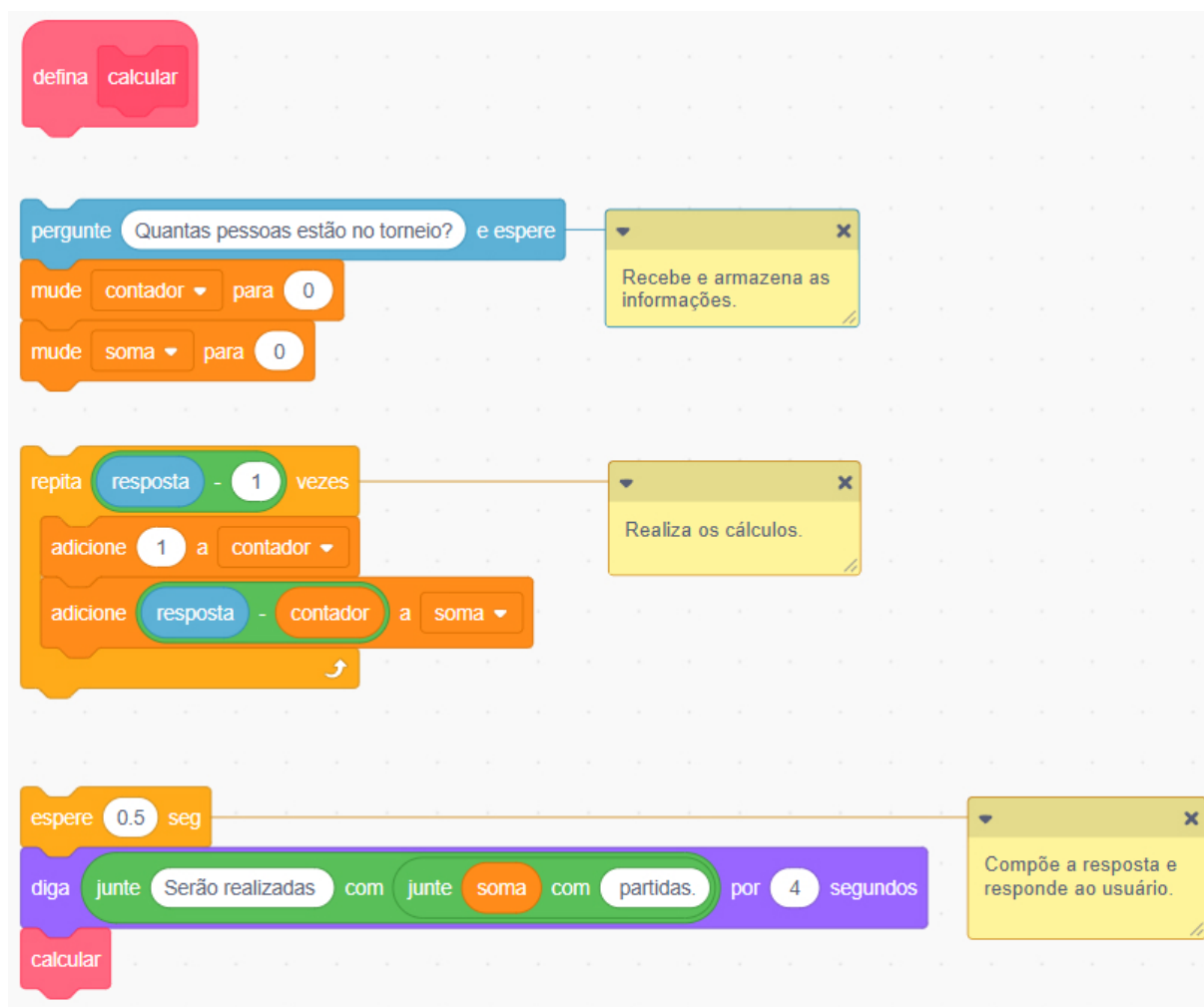


Fonte: Elaborada pela autora (2024).

O algoritmo está dividido em 3 blocos. Os dois blocos menores, *Quando a bandeira verde for clicada* e *Quando este ator for clicado* apenas disparam a execução do bloco de título *Calcular*. Essa escolha faz com que o programa seja executado ao ser iniciado ou ainda, ao clicarmos na fila (imagem) – a qualquer momento em que o programa estiver aberto.

O bloco *Calcular* realiza o que foi previsto na decomposição, ou seja, recebe as informações, as armazena, realiza os cálculos e compõe a resposta, devolvendo-a ao usuário.

Figura 4.10: Blocos de comandos no Scratch com comentários.



Fonte: Elaborada pela autora (2024).

Quanto à realização dos cálculos, o programa repete os cálculos conforme a resposta do usuário, ou seja, $\text{resposta} - 1$. Por exemplo, se o usuário desejar conhecer o número de partidas disputadas por cinco jogadores, o ciclo se repetirá quatro vezes. No primeiro ciclo a soma acumula 4. No segundo, $4 + 3$. No terceiro, $4 + 3 + 2$. E, por último, $4 + 3 + 2 + 1$, resultando em 10 partidas.

Depuração

Para entender o processo de depuração, assistimos ao vídeo que gravamos durante a resolução do problema. A solução final não retém a depuração, pois o código final é limpo, sem erros ou resíduos, representando o processo final do Pensamento Computacional. No entanto, o processo de depuração se perde ao longo do tempo.

Ao iniciarmos a resolução, percebemos que o código apresentava erros. Decidimos recomençar, reorganizando os blocos e aplicando o processo de depuração. Fizemos diversas conjecturas sobre o código para alcançar a solução correta, mas novos erros continuaram

a surgir.

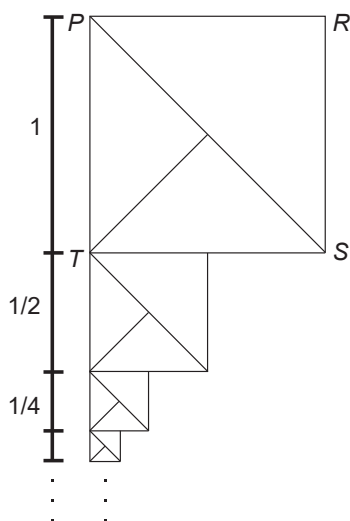
A cada execução do programa fornecendo diferentes valores de entrada e obtendo a saída do programa era realizado um ajuste necessário para que o algoritmo representasse o modelo que constituímos, fazendo uso da decomposição, do reconhecimento de padrões e da abstração.

4.2 Questão 2

A questão 2 foi retirada da prova do ENEM de 2020 e corresponde a questão de número 176 do caderno Cinza. Segue seu enunciado:

Figura 4.11: Questão 176 do Caderno Cinza do ENEM de 2020.

O artista gráfico holandês Maurits Cornelius Escher criou belíssimas obras nas quais as imagens se repetiam, com diferentes tamanhos, induzindo ao raciocínio de repetição infinita das imagens. Inspirado por ele, um artista fez um rascunho de uma obra na qual propunha a ideia de construção de uma sequência de infinitos quadrados, cada vez menores, uns sob os outros, conforme indicado na figura.



O quadrado $PRST$, com lado de medida 1, é o ponto de partida. O segundo quadrado é construído sob ele tomando-se o ponto médio da base do quadrado anterior e criando-se um novo quadrado, cujo lado corresponde à metade dessa base. Essa sequência de construção se repete recursivamente.

Qual é a medida do lado do centésimo quadrado construído de acordo com esse padrão?

- A $\left(\frac{1}{2}\right)^{100}$
- B $\left(\frac{1}{2}\right)^{99}$
- C $\left(\frac{1}{2}\right)^{97}$
- D $\left(\frac{1}{2}\right)^{-98}$
- E $\left(\frac{1}{2}\right)^{-99}$

Fonte: ENEM - 2020.

4.2.1 Resolução utilizando conhecimento matemático

Conforme o enunciado, a construção segue uma sequência de infinitos quadrados cada vez menores, em que cada lado é a metade do lado do quadrado anterior. Logo, temos a formação de uma progressão geométrica (PG) de razão $q = \frac{1}{2}$ e $a_1 = 1$.

O que se deseja é obter a_{100} :

$$a_{100} = a_1 \cdot q^{99} = 1 \cdot \left(\frac{1}{2}\right)^{99} = \left(\frac{1}{2}\right)^{99}$$

4.2.2 Resolução utilizando de Pensamento Computacional

Realizamos duas resoluções utilizando o Pensamento Computacional no Scratch. Na primeira, nosso objetivo foi obter uma generalização para o problema. Em outras palavras, o problema é resolvido para uma medida inicial do lado do quadrado e da razão incremental à escolha do usuário. Assim, uma personagem pergunta por essas medidas e o valor é armazenado em variáveis internas (lado e razão):

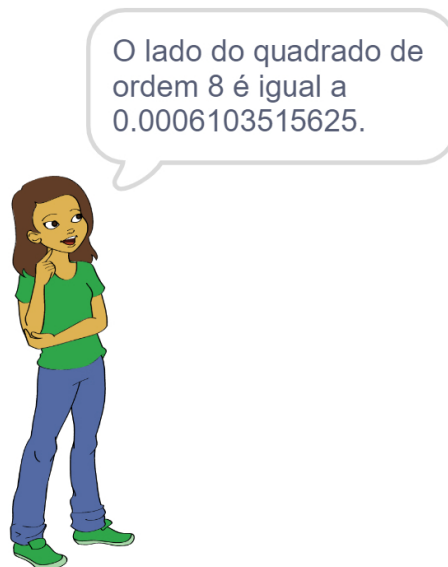
Figura 4.12: Interface da construção no Scratch.



Fonte: Elaborada pela autora (2024).

Em seguida, é solicitado que seja digitada a ordem do quadrado do qual se deseja conhecer a medida do lado. Por exemplo, ao responder 10 para o lado inicial, 0,25 para a razão e 8 para o quadrado que se deseja conhecer a medida do lado é retornado o seguinte:

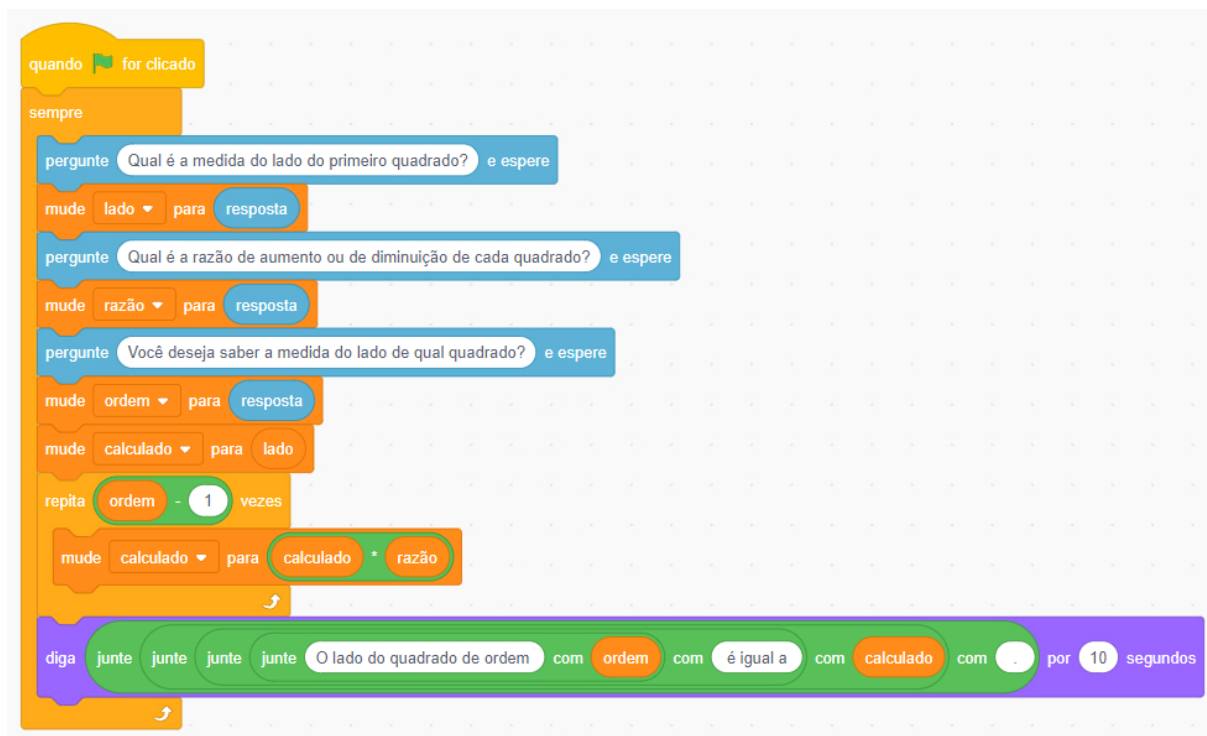
Figura 4.13: Resposta após processamento da resolução.



Fonte: Elaborada pela autora (2024).

Após armazenar os valores de entrada nas variáveis lado, razão e ordem, o algoritmo interno realiza reiteradas multiplicações da medida do lado pela razão em um bloco de código repetir. E, por último, retorna uma resposta concatenando trechos de texto com valores numéricos.

Figura 4.14: Bloco de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

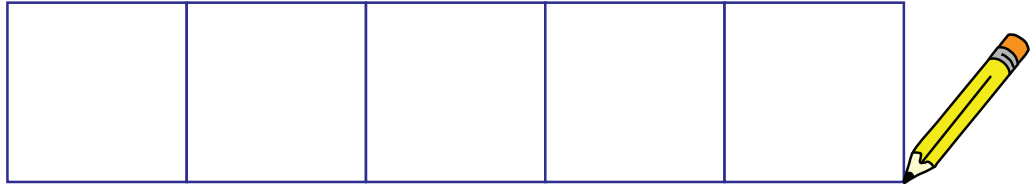
4.2.3 Resolução utilizando de Pensamento Computacional: composição a partir do problema inicial

Na segunda resolução, compomos outro problema a partir do enunciado do ENEM, primeiramente porque a partir dele, fomos inspirados a explorar outras possibilidades, como a de construir a figura com uma razão específica, por exemplo, e o Pensamento Computacional nos deu condições de implementarmos uma resolução; em um segundo momento, porque identificamos nesta questão um processo iterativo de construção geométrica que nos permitiu expandir a forma de analisar, abstrair e resolver o problema, conforme abordaremos na próxima seção, destinada à análise dos processos de Pensamento Computacional utilizados na resolução.

Focamos atenção no seguinte trecho do enunciado: “Inspirado por ele, um artista fez um rascunho de uma obra na qual propunha a ideia de construção de uma sequência de infinitos quadrados”. Assim, nosso objetivo foi obter uma sequência de quadrados adjacentes a partir de medidas digitadas pelo usuário. Imaginamos um lápis que desenhasse n quadrados adjacentes de lados de mesmo comprimento, ou n quadrados adjacentes cujos

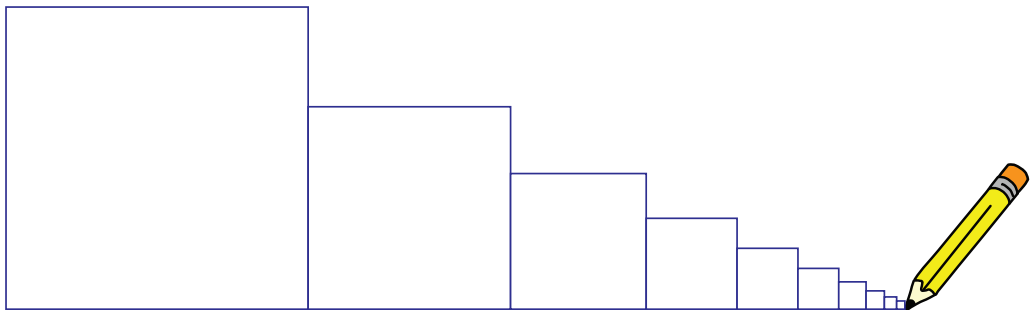
comprimentos dos lados estivessem em uma progressão geométrica crescente, ou decrescente.

Figura 4.15: Cinco quadrados construídos com lado inicial 70 e razão 1.



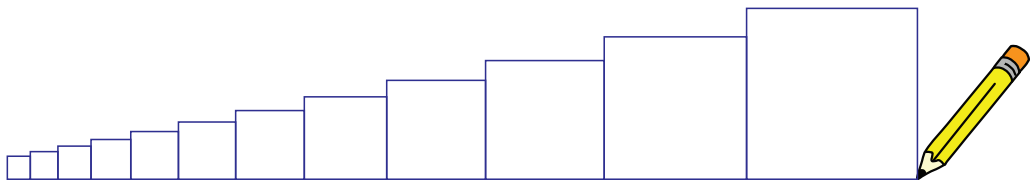
Fonte: Elaborada pela autora (2024).

Figura 4.16: Dez quadrados construídos com lado inicial 120 e razão 0.67.



Fonte: Elaborada pela autora (2024).

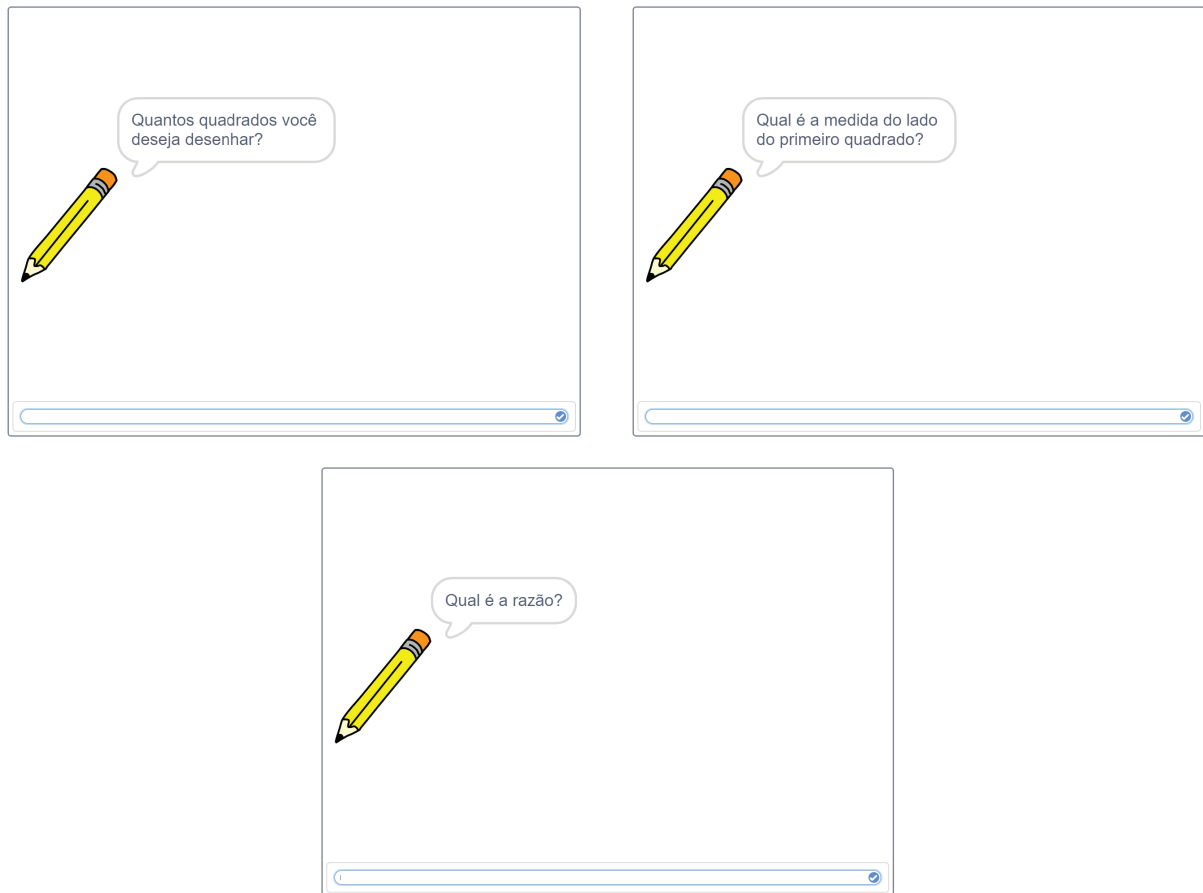
Figura 4.17: Doze quadrados construídos com lado inicial 10 e razão 1.2.



Fonte: Elaborada pela autora (2024).

Inicialmente o programa deveria solicitar quantos quadrados deveriam ser desenhados. Em seguida, solicitar a medida do lado do primeiro quadrado. E, por último, solicitar a razão incremental.

Figura 4.18: Interface da construção no Scratch.

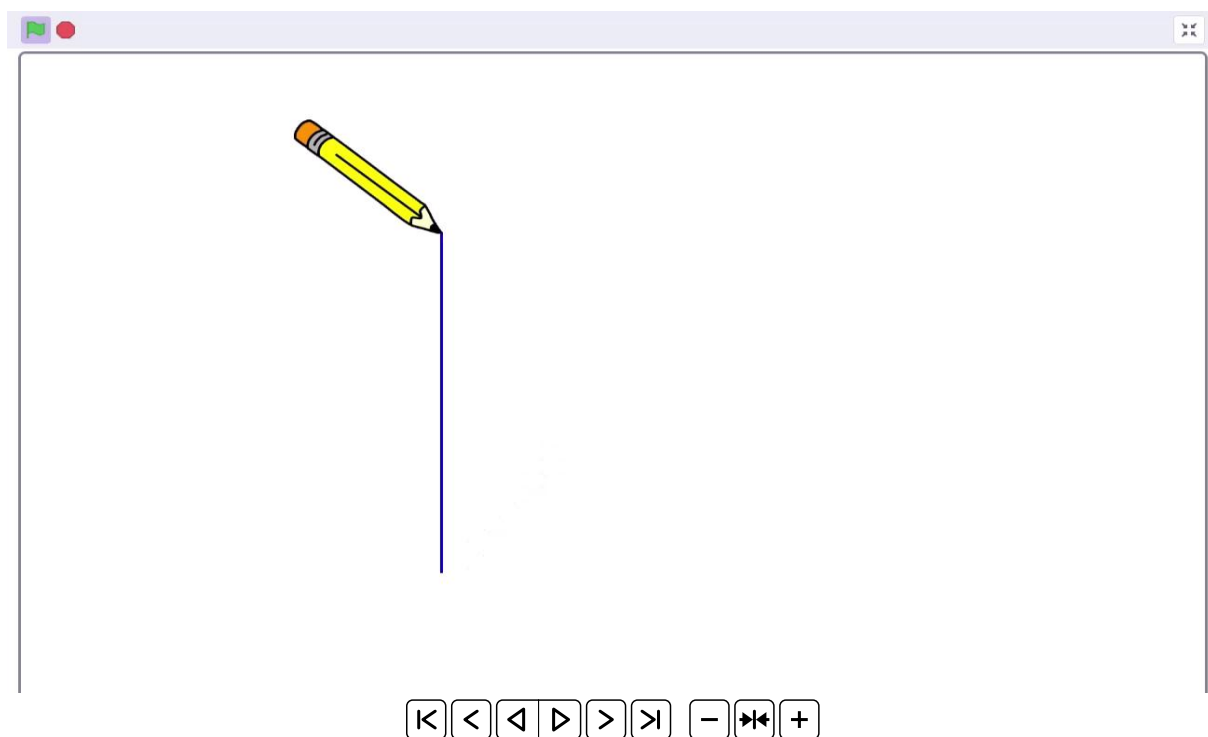


Fonte: Elaborada pela autora (2024).

Essas medidas são armazenadas em três variáveis internas: *Quantidade*, *Lado* e *Razão*.

Para desenhar cada quadrado foi construído um bloco de título *Desenhar* quadrado que, ao ser acionado, desloca para o vértice inferior direito do quadrado a ser construído e constrói o quadrado traçando quatro segmentos em um movimento vertical para cima, horizontal para a esquerda, vertical para baixo e horizontal para a direita.

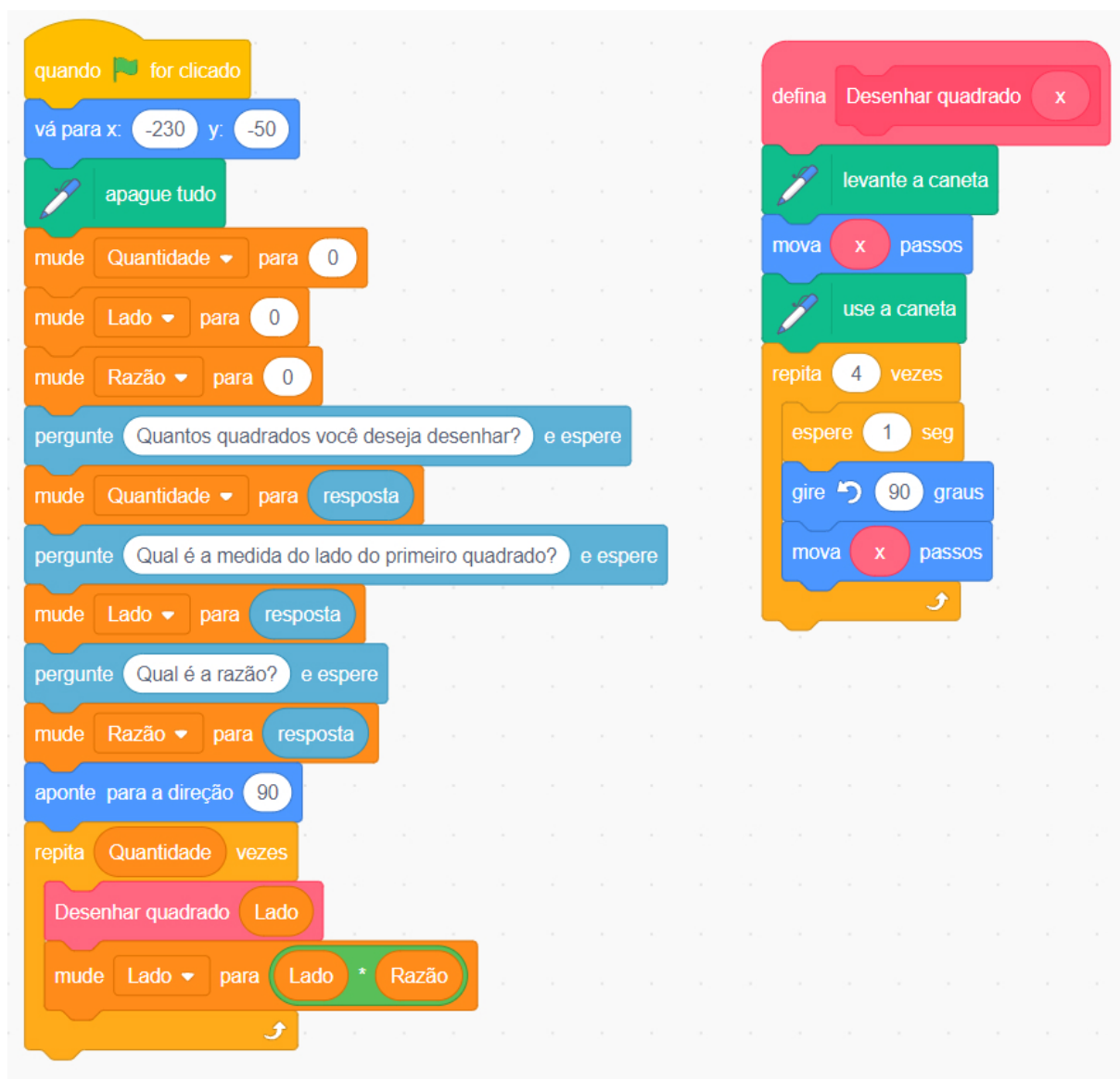
Figura 4.19: Quadrados adjacentes em construção



Fonte: Elaborada pela autora (2024)

Essa abstração permite que a ponta do lápis esteja no vértice inferior direito do quadrado no início e no término, possibilitando assim a construção de quadrados adjacentes sem a necessidade de cálculos de seus distanciamentos. Dessa forma, os deslocamentos não dependem de cálculos explícitos de distância, uma vez que a posição do próximo quadrado é determinada pelo procedimento construtivo adotado. Os blocos de código construídos no Scratch ficaram configurados conforme apresentado na Figura 4.20.

Figura 4.20: Blocos de comando no Scratch.



Fonte: Elaborada pela autora (2024).

4.2.4 Análise dos processos de Pensamento Computacional utilizados na resolução da questão 2

A segunda questão a ser analisada refere-se ao artista gráfico holandês Maurits Cornelius Escher que criou belíssimas obras nas quais as imagens se repetiam, com diferentes tamanhos, induzindo ao raciocínio de repetição infinita das imagens. No enunciado original do ENEM deseja-se saber qual o lado do centésimo quadrado.

Formulação do problema

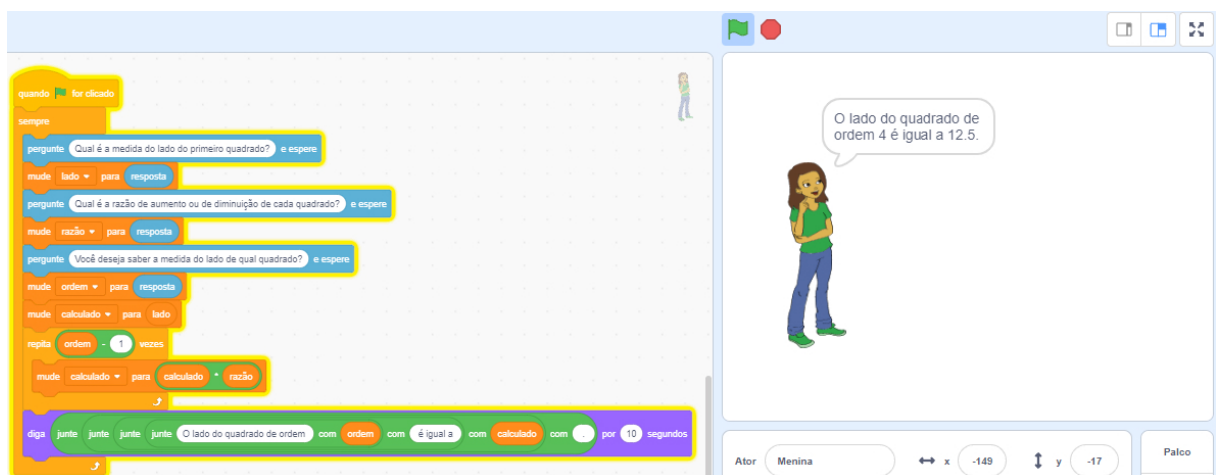
Em nossa resolução no Scratch, formulamos o problema para calcular a medida do lado do quadrado nas mesmas condições do enunciado do ENEM. No entanto, como já

descrevemos anteriormente, incluímos a opção de o usuário especificar quantos quadrados ele deseja construir. Desta maneira, nossa resolução consiste em um modelo computacional que permite ao usuário escolher a quantidade de quadrados quanto a medida do lado.

Em termos de necessidade, objeto e pergunta, nosso objetivo é construir quadrados com medidas que se adaptem a qualquer entrada fornecida pelo usuário. Ou seja, a pergunta que buscamos responder não é sobre uma quantidade fixa de quadrados ou tamanhos predefinidos, mas sim a criação de um modelo computacional flexível, que permita ao usuário determinar tanto o número de quadrados quanto as medidas de seus lados.

Após formularmos o problema, construímos o conjunto de blocos apresentado na Figura 4.21, o qual foi totalmente repensado em outro encontro via Meet. Nesse bloco, não havíamos programado os blocos para construir quadrados, o objetivo era saber se realmente encontraríamos os resultados esperados.

Figura 4.21: Blocos de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

Decomposição

Assim como na questão 1, o processo de decomposição se deu nas seguintes etapas: entrada das informações, armazenamento das informações, processamento ou realização dos cálculos necessários e composição de um texto para o usuário responder.

Como se trata da construção de vários quadrados, o usuário pode escolher quantos quadrados o programa vai desenhar, a medida do lado e qual será a razão. Sendo assim, precisamos definir uma forma de obter as informações necessárias.

Escolhemos um cenário de fundo branco e o objeto lápis para desenhar os quadrados com as medidas desejadas. Para armazenar esses dados e utilizá-los posteriormente, fizemos o uso de três variáveis: a variável *Quantidade* guarda a escolha do usuário em relação ao número de quadrados. As variáveis *Razão* e *Lado* são responsáveis pelo processo

de aumentar ou diminuir o tamanho dos quadrados.

Para o processamento/cálculo da quantidade de quadrados, utilizamos um loop que realiza cálculos iterados, os quais serão detalhados no processo de produção de algoritmo.

Finalmente, a última parte da decomposição obtém os dados calculados e apresenta a resposta ao usuário em uma mensagem textual.

Reconhecimento de padrões e abstração

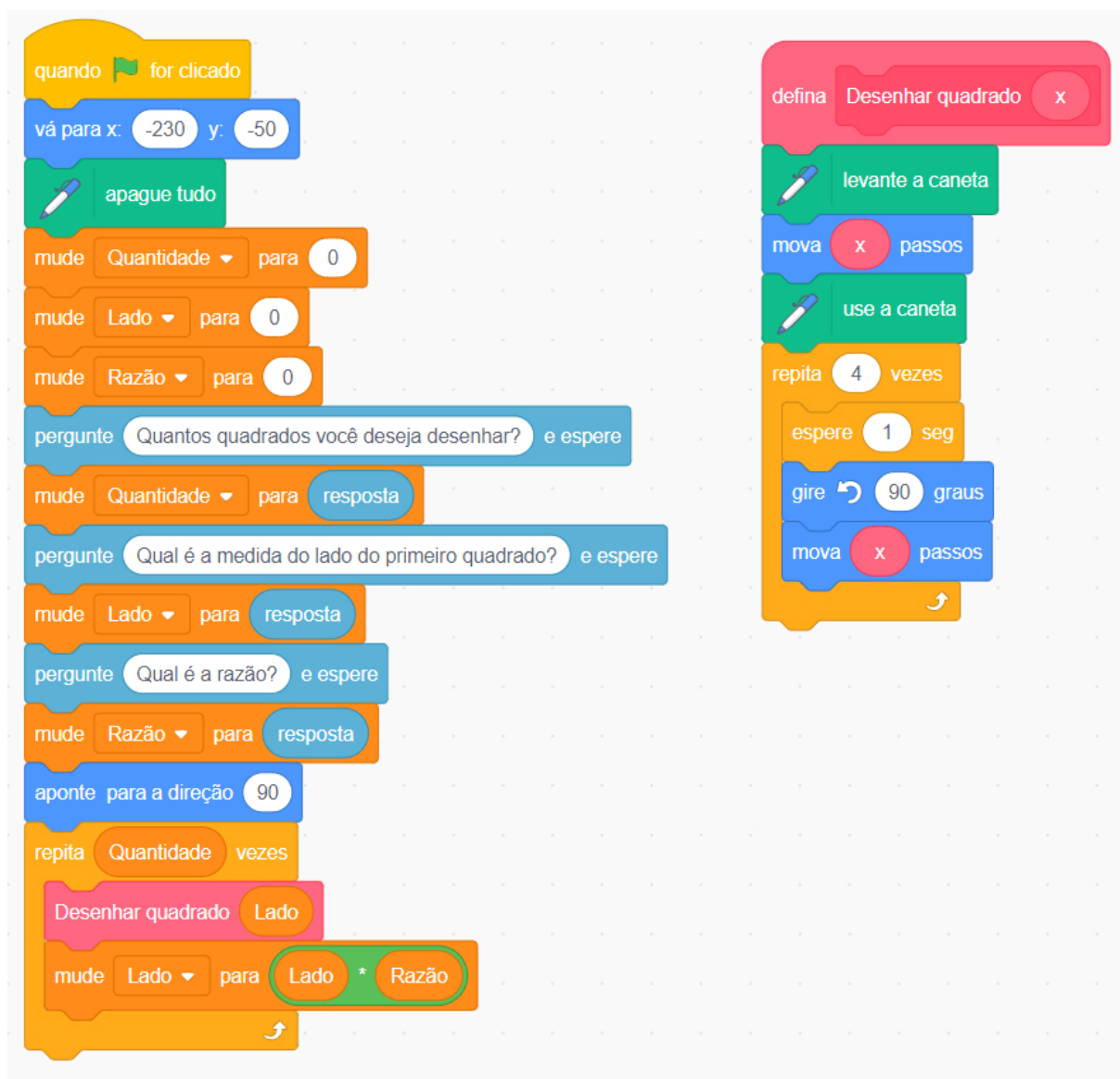
Conforme abordado nas seções 4.2.2 e 4.2.3, as resoluções que mobilizaram o conhecimento matemático basearam-se em dois padrões distintos. No primeiro, o problema é resolvido a partir de uma medida inicial do lado do quadrado e de uma razão definida conforme a escolha do usuário. No segundo, o padrão de construção de cada quadrado segue uma progressão geométrica (PG), na qual o programa constrói e representa graficamente os quadrados de acordo com os parâmetros solicitados pelo usuário.

A abstração da sequência está no padrão de redução do tamanho dos quadrados. Se considerarmos que o lado do primeiro quadrado é L , o lado do próximo quadrado será $L/2$, e assim sucessivamente. Quando deixamos de tratar cada quadrado de forma isolada e passamos a discutir a necessidade de “diminuir sempre pela metade”, esse movimento revela a construção progressiva da abstração, visto que a atenção se desloca dos valores específicos para a identificação de uma regra geral que orienta a continuidade da resolução.

Produção de algoritmo

Após identificar os padrões no processo descrito no parágrafo anterior, bem como da decomposição, foi possível criar o seguinte algoritmo no Scratch.

Figura 4.22: Blocos de comando no Scratch.



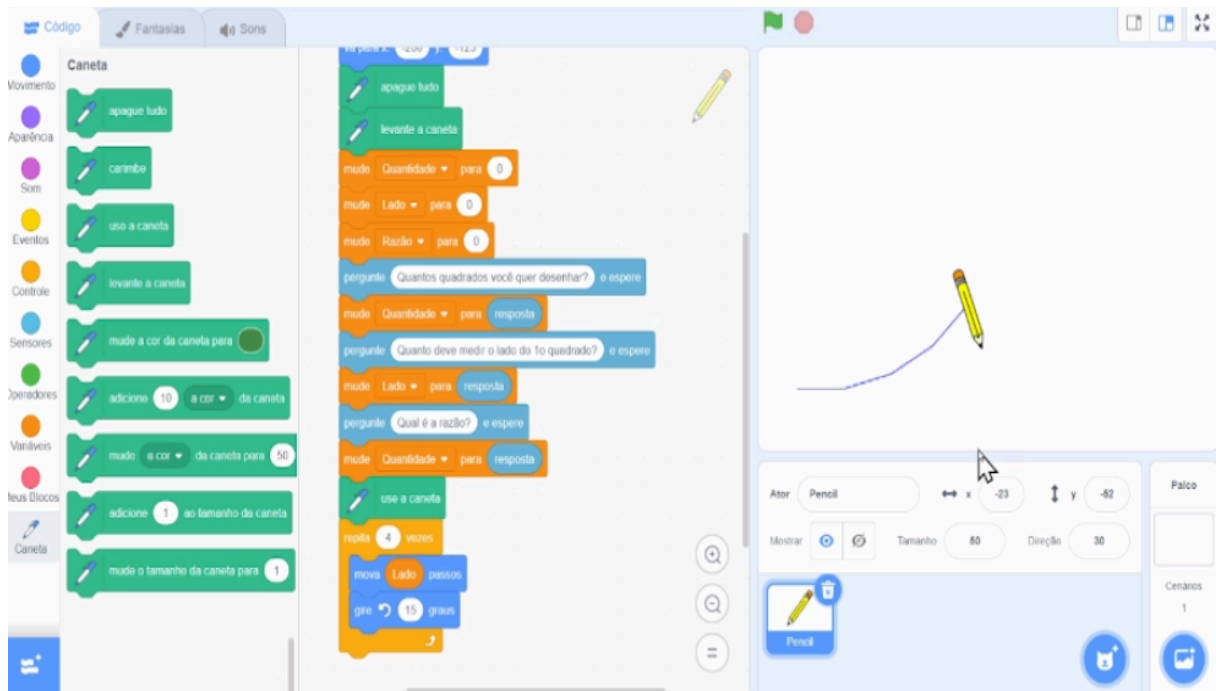
Fonte: Elaborada pela autora (2024).

Este conjunto de blocos oferece ao usuário a possibilidade de desenhar quantos quadrados ele desejar. Um exemplo é se ele quiser desenhar 10 quadrados com medida de lado 120 e uma razão 0,7 que é possível executar no Scratch.

Depuração

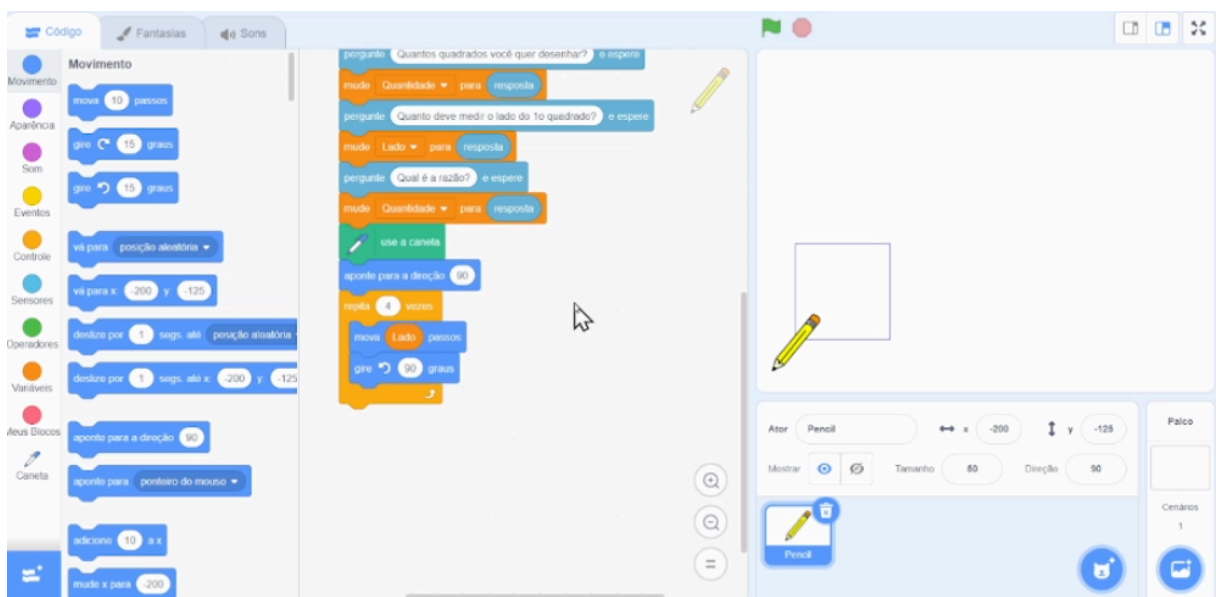
No início da construção no Scratch alguns comandos não funcionaram como esperávamos, o objeto lápis, por exemplo, permaneceu no mesmo ponto do palco ou desenhou um quadrado muito pequeno. Diante desses resultados, os blocos de programação foram revistos e ajustados, possibilitando o funcionamento correto do algoritmo. Nas Figuras 4.21 e 4.22, são apresentados os registros dos erros identificados durante o processo de resolução das questões.

Figura 4.23: Blocos de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

Figura 4.24: Blocos de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

Outro detalhe interessante é que dependendo do tamanho dos quadrados, o lápis pode desenhar fora da tela. É importante ajustar a posição inicial de x e y ou permitir que o usuário escolha essa posição.

4.3 Questão 3

A questão 3 foi retirada da prova do ENEM de 2022 e corresponde a questão de número 177 do caderno amarelo. Segue seu enunciado:

Figura 4.25: Questão 177 do Caderno Amarelo do ENEM de 2022.

Um atleta iniciou seu treinamento visando as competições de fim de ano. Seu treinamento consiste em cinco tipos diferentes de treinos: treino T_1 , treino T_2 , treino T_3 , treino T_4 e treino T_5 . A sequência dos treinamentos deve seguir esta ordem:

Dia	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º	12º	13º
Treino	T_1	R	R	T_2	R	R	T_3	R	T_4	R	R	T_5	R

A letra R significa repouso. Após completar a sequência de treinamentos, o atleta começa novamente a sequência a partir do treino T_1 e segue a ordem descrita. Após 24 semanas completas de treinamento, se dará o início das competições. A sequência de treinamentos que o atleta realizará na 24ª semana de treinos é

- Ⓐ $T_3, R, T_4, R, R, T_5, R.$
- Ⓑ $R, T_3, R, T_4, R, R, T_5.$
- Ⓒ $R, T_4, R, R, T_5, R, T_1.$
- Ⓓ $R, R, T_5, R, T_1, R, R.$
- Ⓔ $R, T_5, R, T_1, R, R, T_2.$

Fonte: ENEM - 2022.

4.3.1 Resolução utilizando conhecimento matemático

A sequência de treinos ao longo de um ciclo de 13 dias é a seguinte: $T_1, R, R, T_2, R, R, T_3, R, T_4, R, R, T_5, R$, onde R representa um dia de repouso e T_1, T_2, T_3, T_4, T_5 representam os diferentes tipos de treinos.

Para determinar a sequência de treinos na 24ª semana, precisamos calcular em qual parte do ciclo de 13 dias a 23ª semana termina. Primeiro, calculamos o número total de dias em 23 semanas: $23 \times 7 = 161$.

Em seguida, calculamos o resto de 161 ao ser dividido por 13, ou seja, 5.

Assim, os últimos 5 dias da 23ª semana correspondem aos cinco primeiros itens na sequência de treinamentos:

$$T_1, R, R, T_2, R, R, T_3, R, R, T_5, R$$

Portanto, na 24ª semana, a sequência de treinos será: $R, T_3, R, T_4, R, R, T_5$.

4.3.2 Resolução utilizando de Pensamento Computacional

A resolução no Scratch como aplicação de Pensamento Computacional é uma generalização do cálculo matemático anterior que permite encontrar uma sequência semanal a partir de qualquer configuração de treinos e repousos, adicionados em uma tabela. Em outras palavras, o modelo permite reconfigurar a sequência de treinos e repousos modificando a quantidade para mais ou para menos que 13 e sob o mesmo algoritmo encontra a sequência de treinos/repousos de uma semana solicitada.

Figura 4.26: Interface da resolução no Scratch.



Fonte: Elaborada pela autora (2024).

Por exemplo, na imagem abaixo aparece uma sequência de nove dias T_1 , R , T_2 , R , T_3 , R , T_4 , R , T_5 e o treinador indica como resposta a sequência de treinamentos na semana 52.

Figura 4.27: Sequência de treinamentos na semana 52.



Fonte: Elaborada pela autora (2024).

Para obter a solução computacional da questão e determinar a sequência de treinos realizada pelo atleta na 24ª semana, identificamos a existência de um padrão de treinos que se repete a cada 13 dias. A partir dessa constatação, foi possível mapear a sequência de treinos e períodos de repouso ao longo de um ciclo completo. Em seguida, calculamos o número total de dias correspondentes a 23 semanas completas, para localizar a posição dos dias dentro do ciclo ao final desse período.

O algoritmo construído determina a posição no ciclo de 13 dias e após 161 dias lista a sequência de treinos a partir dessa posição.

Os códigos desenvolvidos no Scratch para a solução do problema fornecem uma organização ordenada de itens na sequência para a semana específica. O cálculo envolve determinar a posição inicial na sequência, repetidamente aumentando essa posição, e ajustando o tamanho dessa sequência, possibilitando utilizar a resolução a outros casos. Dessa forma, a construção do algoritmo não se limita à situação particular apresentada no enunciado, mas constitui um modelo computacional flexível, capaz de acomodar diferentes entradas e configurações, explicitando processos do Pensamento Computacional.

4.3.3 Análise dos processos de Pensamento Computacional utilizados na resolução da questão 3

A terceira questão a ser analisada trata de um atleta que deseja organizar seu treino visando as competições de fim de ano, sendo que possui cinco tipos diferentes de treinos, além de espaços reservados para repouso. No enunciado original do ENEM, pretende-se determinar qual é a sequência de treinamentos que o atleta realizará na 24^a semana, visto que a sequência apresentada ao leitor segue uma ordem definida e se reinicia ao final de um ciclo de 13.

Formulação do problema

A resolução da questão no Scratch consiste na construção de um modelo que representa a situação proposta no ENEM para determinar a sequência de treinos na 24^a semana. O modelo também permite ao usuário definir a sequência de treinos a ser utilizada, ampliando as possibilidades de análise. Dessa forma, a proposta envolve a implementação de um modelo computacional flexível, que possibilita a reconfiguração da sequência e da quantidade de treinos, bem como o cálculo e a apresentação da sequência correspondente a qualquer semana, para além das 24 consideradas no problema original.

Em termos de necessidade, objeto e pergunta de pesquisa, o objetivo foi apresentar uma proposta que se adeque às entradas fornecidas pelo usuário, em um ciclo de registros passíveis de reorganização e alteração (maior, menor ou igual aos treze dias apresentados inicialmente), compondo, assim, um modelo computacional condicionado às entradas do usuário, tendo como saída a sequência de treinos solicitada conforme a semana informada.

Decomposição

A decomposição adotada para a resolução da questão foi organizada em diferentes blocos, cada um responsável por tratar subproblemas específicos, a saber: entrada de dados, armazenamento das informações, processamento dos cálculos necessários e composição da resposta apresentada ao usuário.

Inicialmente, foi criada a lista *Sequência*, destinada ao armazenamento dos treinos e repousos. Em seguida, inserimos um bloco informando ao usuário a possibilidade de alterar a sequência de treinos, bem como um bloco de interação responsável por solicitar a semana de treino desejada. A informação fornecida é armazenada na variável *Semana*. Posteriormente, outro bloco realiza o cálculo da sequência com base no valor informado, cujo resultado é armazenado na variável *dias*. A essa variável, adicionamos 1, que será a próxima informação resgatada da lista.

Observamos que quando os valores da variável *dias* era menor que o tamanho da lista, o problema estava resolvido, porém, em caso contrário, a sequência ficaria incompleta, pois não havia um *looping* de retorno ao início da lista. Por exemplo, se *dias* fosse igual a 9, por meio da execução dos comandos, a posição inicial da sequência seria o 10º elemento, mas a lista apresentaria apenas quatro resultados (das posições 10, 11, 12 e 13). Para resolver este problema, criamos um bloco de controle para que se a variável *dias* for maior que o tamanho da lista, então, execute-se o cálculo para encontrar o resto de *dias* pelo tamanho da lista.

Por fim, por meio de uma estrutura de repetição, utilizada para recuperar os sete elementos da lista, associada ao uso de operadores, a sequência obtida é apresentada ao usuário.

Reconhecimento de padrões e abstração

Reconhecer padrões é especialmente útil na resolução de situações problema que envolve a repetição de uma sequência. Conforme demonstrado na resolução utilizando conhecimento matemático, identificamos dois padrões diferentes no problema. O primeiro é a sequência de treinos fornecidos, repetida a cada 13 dias - ciclo, conforme a tabela.

Figura 4.28: Quadro com sequência.

Dia	1º	2º	3º	4º	5º	6º	7º	8º	9º	10º	11º	12º	13º
Treinos	T ₁	R	R	T ₂	R	R	T ₃	R	T ₄	R	R	T ₅	R

Fonte: Elaborada pela autora (2024).

Observamos que a sequência de treinos se repete a cada 13 dias, nos dias 1, 4, 7, 9, e 11, o atleta realiza treinos diferentes (T_1, T_2, T_3, T_4, T_5); nos demais dias ele repousa (R).

Para determinar em que ponto da sequência o atleta se encontra após um certo número de dias (n), podemos usar a operação de módulo (resto da divisão de n por 13).

Essa compreensão durante a resolução do problema possibilitou a construção de um algoritmo com um padrão modular, capaz de operar com quaisquer entradas fornecidas pelo usuário. Assim, concentramos nossa atenção em reproduzir no Scratch, por meio de blocos específicos, as expressões que exploramos recorrendo ao conhecimento matemático,

permitindo que o usuário insira apenas o número da semana, enquanto todos os cálculos internos são realizados.

Ademais, no momento da composição da resposta ao usuário, novamente nos ocupamos em relacionar as variáveis, tendo em vista o objetivo (resolver o problema) e o resultado a alcançar (dar resposta a problema final).

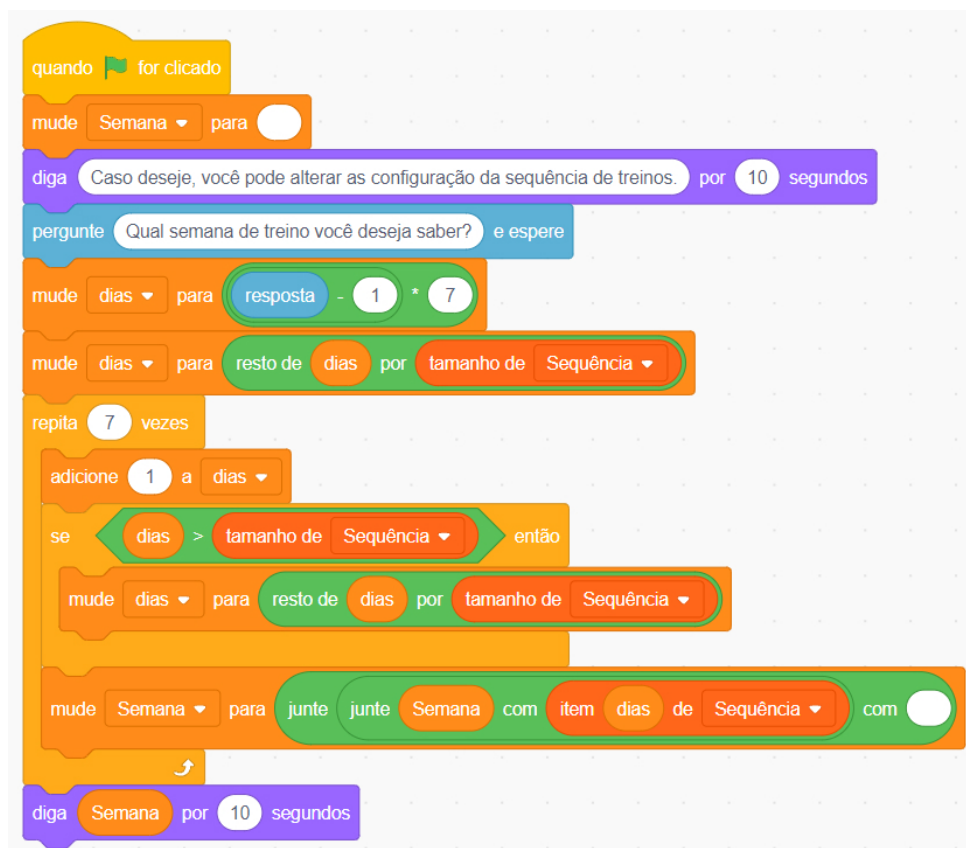
A abstração se reflete na forma como o código lida com elementos como Semana, dias e Sequência, sem exigir que o usuário entenda ou interaja com cada etapa do processo.

Produção de algoritmo

Ao resolver matematicamente a questão, pensamos em como construir um arquivo no Scratch capaz de obter a sequência de treinos em uma semana de ordem escolhida pelo usuário e também, como esse arquivo retornaria a semana, dada uma sequência prévia de treinos.

Como já tínhamos identificado os padrões envolvidos na questão, iniciamos a estruturação de um conjunto de blocos no Scratch com a finalidade de atender as duas necessidades por nós estabelecidas, conforme ilustrado na Figura 4.29.

Figura 4.29: Blocos de comandos no Scratch.

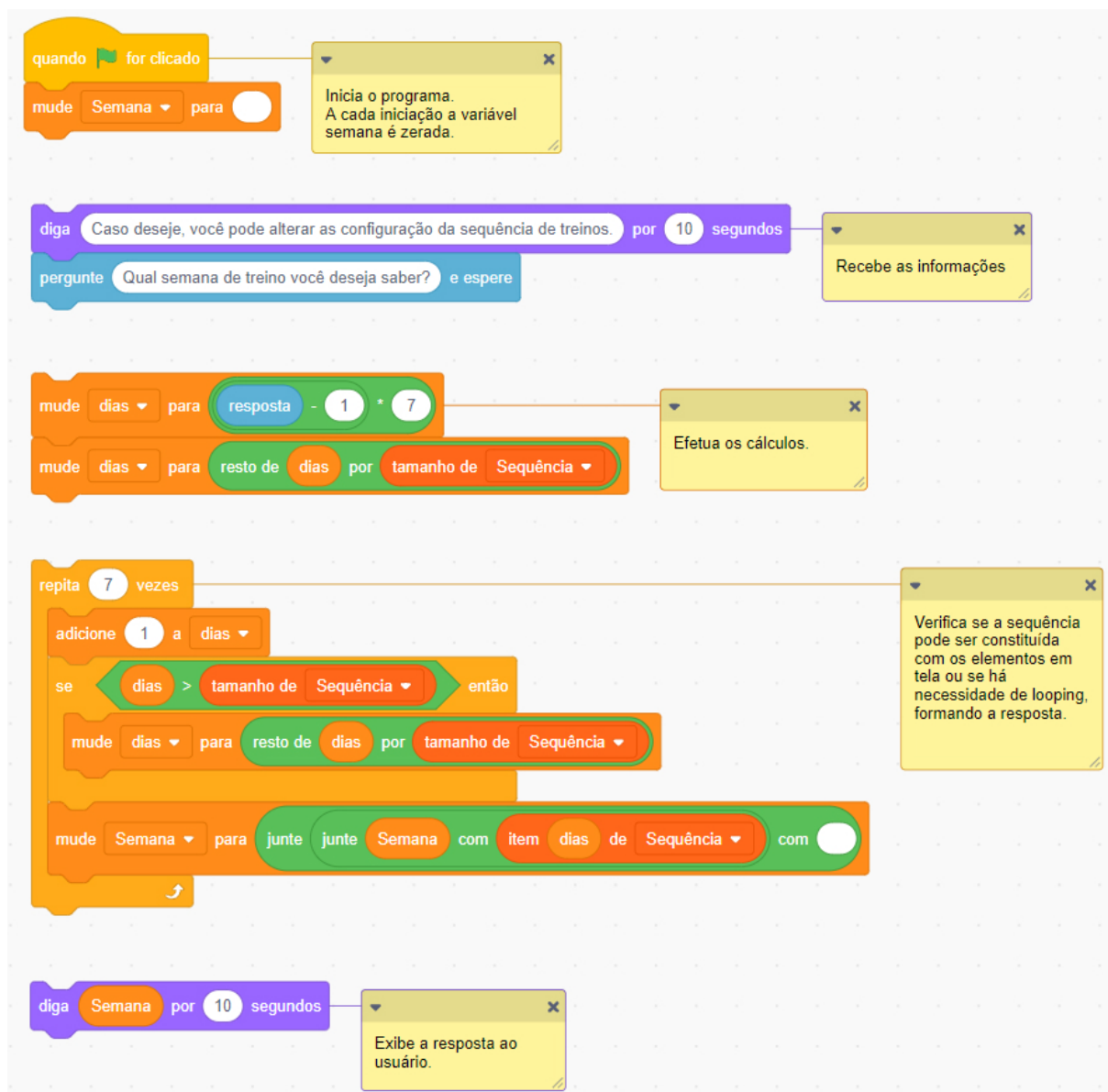


Fonte: Elaborada pela autora (2024).

Esse conjunto de blocos permite informar ao usuário que outra sequência de treinos

pode ser organizada, conforme foi apresentado anteriormente na Figura 4.29, e também possibilita encontrar a sequência de treinos que se deseja saber. Na Figura 4.30, subdividimos o bloco a fim de esclarecer seu funcionamento:

Figura 4.30: Blocos de comandos no Scratch.



Fonte: Elaborada pela autora (2024).

Dessa forma, a organização dos blocos no Scratch apresenta os padrões matemáticos previamente identificados na questão, garantindo que o programa funcione de maneira precisa a cada nova interação do usuário. A reinicialização da variável Semana a cada execução assegura a confiabilidade dos resultados obtidos, evitando o emprego de execuções anteriores.

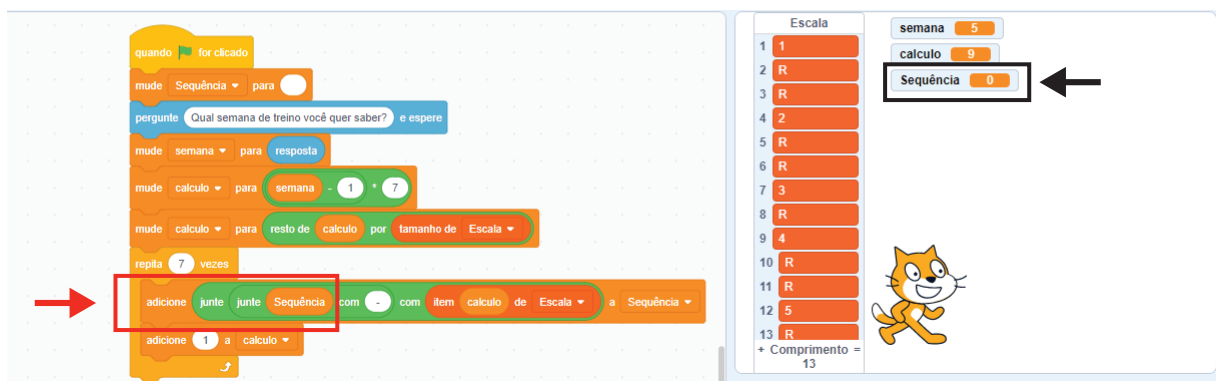
Assim, o arquivo construído não apenas resolve a situação proposta no ENEM, como também se configura como um modelo flexível e reutilizável, permitindo a exploração

de diferentes sequências de treinos e semanas de interesse, ampliando as possibilidades de análise e compreensão do padrão envolvido.

Depuração

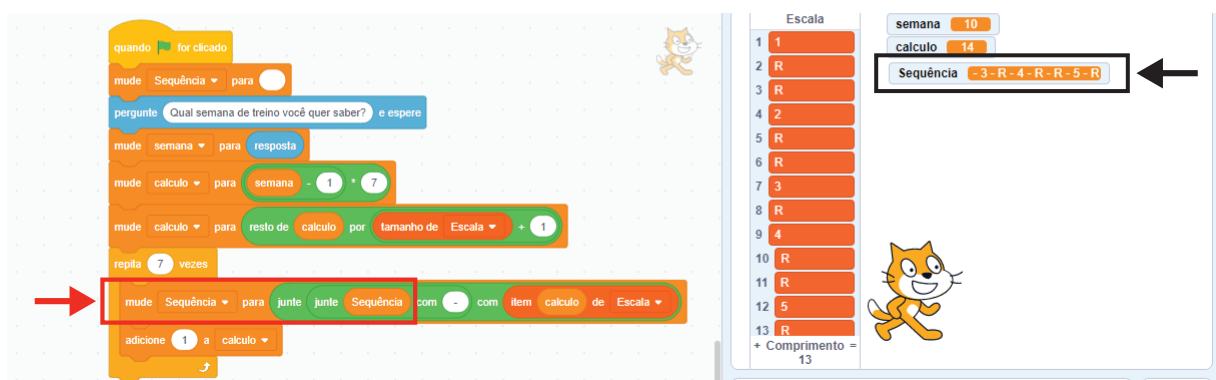
A depuração envolve testar repetidamente o código para garantir que cada parte funcione como planejado. Ao passo que testamos os códigos clicando na bandeira verde, identificamos algumas imprecisões, então, modificamos a abordagem para corrigir essas falhas. Cabe ressaltar que o vídeo mencionado em nosso terceiro capítulo foi fundamental para identificar este processo. A seguir, trazemos um momento que capturamos do vídeo em que desejávamos obter a resposta de forma sequencial, porém, invertemos os comandos, ao invés de utilizar “Mude” usamos “Adicione”, para compor a resposta, portanto, o programa não funcionou e precisamos revisar o algoritmo.

Figura 4.31: Processo de depuração e troca de comandos.



Fonte: Elaborada pela autora (2024).

Figura 4.32: Processo de depuração e troca de comandos.



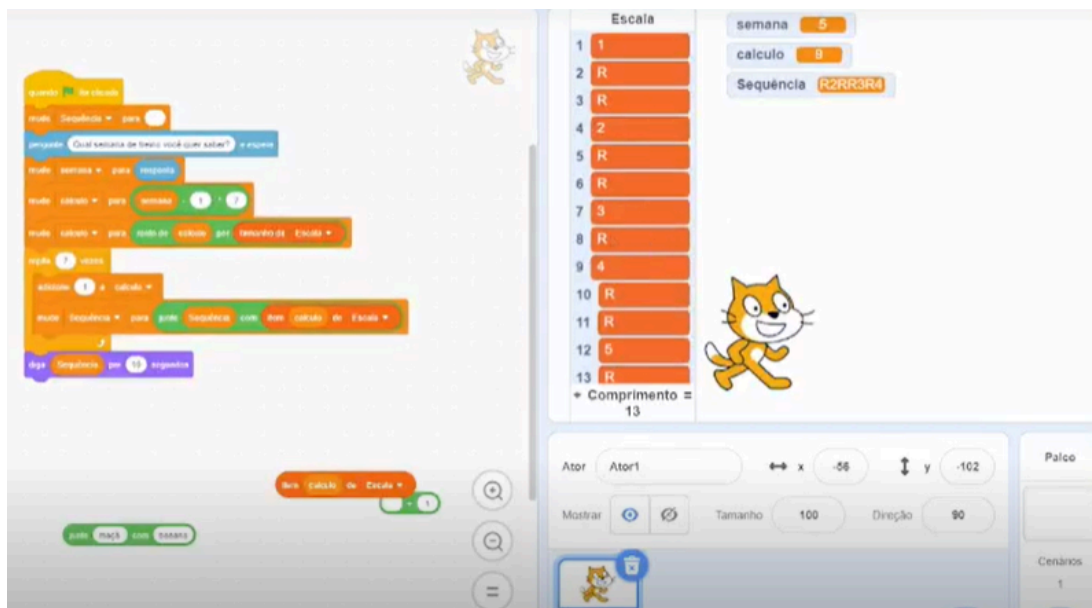
Fonte: Elaborada pela autora (2024).

Nas imagens anteriores, destacamos em vermelho os blocos trocados e, em preto, indicamos a falha e sua correção, ou seja, da primeira vez, não obtivemos o retorno da sequência desejada, mas quando substituímos o comando, a sequência foi mostrada

ordenadamente.

Enquanto estávamos testando a construção, outro detalhe nos inquietou: a forma visual como a sequência era mostrada, embora estivesse ordenada, estava esteticamente inadequada para o propósito. Então, passamos a reestruturar o código até adequar ao nosso planejamento, porém, esse processo foi desafiador, visto que precisamos manipular em um mesmo bloco controles, operadores, variáveis e a lista. Na Figura 4.33 é possível observar alguns blocos soltos na área de blocos de comandos, além do ator dizendo a sequência sem a formatação desejada.

Figura 4.33: Processo de depuração e apresentação estética.



Fonte: Elaborada pela autora (2024).

Na análise, observamos que, durante o processo de construção, o conjunto de blocos responsável por concatenar os elementos e retornar a resposta foi acidentalmente descartado, comprometendo o funcionamento do algoritmo. Essa falha foi identificada posteriormente, ao longo do processo de depuração, quando constatamos que o algoritmo não operava conforme o esperado. A partir da revisão e reorganização dos blocos, foi possível realocá-los adequadamente e concluir a depuração, resultando em um algoritmo funcional.

Testamos outras configurações, inclusive com outros caracteres, porém, havíamos idealizado um projeto que contemplasse a imagem do treinador, a separação da sequência por espaços e a possibilidade de reorganização dos treinos. Dessa forma, o resultado apresentado na Figura 4.29 foi fruto de um trabalho aprofundado, fundamentado nos processos do Pensamento Computacional.

4.4 Questão 4

A quarta questão analisada foi retirada da prova do ENEM de 2022, correspondendo à questão de número 146 do caderno amarelo.

A seguir, trazemos o enunciado original da questão, que servirá de base tanto para a resolução matemática quanto para a construção do modelo computacional desenvolvido no Scratch.

Figura 4.34: Questão 146 do Caderno Amarelo do ENEM de 2022.

Uma pessoa precisa contratar um operário para fazer um serviço em sua casa. Para isso, ela postou um anúncio em uma rede social.

Cinco pessoas responderam informando preços por hora trabalhada, gasto diário com transporte e tempo necessário para conclusão do serviço, conforme valores apresentados no quadro.

Se a pessoa pretende gastar o mínimo possível com essa contratação, irá contratar o operário

- A I.
- B II.
- C III.
- D IV.
- E V.

Operário	Preço por hora (real)	Preço do transporte (real)	Tempo até conclusão (hora)
I	120	0,00	8
II	180	0,00	6
III	170	20,00	6
IV	110	10,00	9
V	110	0,00	10

Fonte: ENEM - 2022.

4.4.1 Resolução utilizando conhecimento matemático

Para determinar os gastos com cada operário, consideramos o cálculo do custo total a partir da soma do valor referente à mão de obra e o custo do transporte, conforme ilustrado na questão. Realizamos os seguintes cálculos matemáticos:

$\text{Custo total} = (\text{preço por hora} \times n^{\circ} \text{ de horas}) + \text{preço do transporte}.$

Adotamos como pressuposto que um operário pode trabalhar mais do que oito horas diárias, visto que o problema não oferece nenhuma informação sobre o limite de horas por dia. Esse pressuposto permitiu estabelecer a seguinte relação:

Figura 4.35: Quadro.

Operário	Preço da hora x quant. horas	Valor do transporte	Custo total
Operário I	120 . 8	0	960
Operário II	180 . 6	0	1080
Operário III	170 . 6	20	1040
Operário IV	110 . 9	10	1000
Operário V	110 . 10	0	1100

Fonte: Elaborada pela autora (2024).

A partir desses resultados, verificamos que o Operário I apresenta o menor custo total. Do ponto de vista matemático, a resolução mobiliza conceitos de aritmética, como multiplicação e adição, bem como relações entre grandezas, especialmente a relação entre preço por hora e tempo de trabalho. Além disso, envolve a interpretação de dados apresentados na tabela do enunciado e o cálculo de custo total, aspectos fundamentais para a tomada de decisão no contexto do problema.

4.4.2 Resolução utilizando de Pensamento Computacional

Neste caso específico, decidimos resolver o problema segundo o seu enunciado, visto que, em nossas análises, percebemos que uma ampliação necessitaria da transformação de unidades de medida, por exemplo, se em outro formato a questão trouxesse dias trabalhados, reorganizaríamos o tempo para calcular as horas e também o valor do transporte, havendo maiores desdobramentos da situação. Dessa forma, optamos em implementar um projeto capaz de otimizar as operações matemáticas, permitindo ao usuário a investigação de outros aspectos relacionados à questão.

Analisando a resolução no Scratch, obtemos a generalização do cálculo matemático que nos permitiu gerar uma lista com o valor final para encontrarmos o menor custo possível. A dinâmica de funcionamento entre as listas será detalhada na próxima seção.

Figura 4.36: Interface da construção no Scratch.

Preço/Hora		Transporte		Tempo		Valor Final	
1	120	1	0	1	8	1	960
2	180	2	0	2	6	2	1080
3	170	3	20	3	6	3	1040
4	110	4	0	4	10	4	1100
5	110	5	10	5	9	5	1000
+ Comprimento = 5		+ Comprimento = 5		+ Comprimento = 5		+ Comprimento = 5	

Menor Valor
960

Fonte: Elaborada pela autora (2024).

4.4.3 Análise dos processos de Pensamento Computacional utilizados na resolução da questão

A quarta questão trata da necessidade de uma pessoa contratar um operário para realizar um trabalho, sendo que deseja gastar o mínimo possível. No valor a ser pago estão inclusas as despesas com o transporte e o preço da hora trabalhada. Do enunciado original, temos cinco diferentes opções a serem analisadas para tal contratação.

Essa questão envolve uma situação-problema que exige a comparação entre diferentes possibilidades, considerando múltiplas variáveis, visando tomar uma decisão que minimize um determinado custo. Tais características tornam a questão particularmente adequada para a mobilização de processos do Pensamento Computacional, especialmente no que se refere à organização de dados, à generalização de cálculos e à construção de procedimentos para a resolução do problema.

Formulação do problema

Formulamos o problema a fim de encontrar o menor custo possível na contratação de mão-de-obra nas mesmas condições do enunciado do ENEM, porém, visando construir um arquivo no Scratch que permita ao usuário indicar o preço por hora, preço do transporte e tempo de trabalho, para que, a partir destas informações, seja calculado o preço total para a realização do serviço.

Consideramos criar listas dinâmicas, sendo que três delas armazenam os dados fornecidos pelo usuário e a quarta lista recebe o resultado do cálculo matemático que apresentamos na seção 4.4.1. Decidimos exibir as quatro listas para oportunizar maior debate em torno dos dados fornecidos e seus resultados. Por exemplo, pensamos na possibilidade de os valores fornecidos por cada operário serem diferentes, mas resultarem

em um mesmo custo final. Tal situação geraria um novo debate em torno de cada variável que compõe o problema. Para tanto, ilustramos nossa abordagem na Figura 4.37 :

Figura 4.37: Variáveis diferentes e valor final igual.



Fonte: Elaborada pela autora (2024).

Situações como a descrita anteriormente não compõem o enunciado inicial do problema, porém, consideramos primordial discorrer sobre tais pressupostos presentes durante nossas discussões e que fortaleceram nossa compreensão sobre os processos constituintes do Pensamento Computacional.

Decomposição

Inicialmente, observamos a aplicação do processo de decomposição ao criarmos as listas a fim de organizar e armazenar as informações fornecidas pelo usuário, que posteriormente seriam resgatadas para a execução dos cálculos.

Para esta construção não utilizamos ator, optamos em interagir com o usuário por meio do acionamento da bandeira verde e de teclas específicas, responsáveis pelo funcionamento de partes específicas do programa. Assim sendo, decompomos o problema em: entrada das informações, armazenamento das informações, realização dos cálculos necessários e devolução da resposta.

Figura 4.38: Obtenção de informações para armazenamento nas listas.

Preço/Hora	Transporte	Tempo	Valor Final
(vazio)	(vazio)	(vazio)	(vazio)
+ Comprimento = 0	+ Comprimento = 0	+ Comprimento = 0	+ Comprimento = 0

Menor Valor

Fonte: Elaborada pela autora (2024).

Na Figura 4.38 apresentamos a forma de interação com o usuário para o preenchimento das listas. A decisão por armazenar os valores nessas listas deve-se ao fato de podermos operar e recuperar tais informações para efetuar os cálculos, diferente do que foi feito utilizando o conhecimento matemático; além disso, essa disposição mantém os dados disponíveis em nosso campo visual para outras análises.

Reconhecimento de padrões e abstração

Esse tipo de enunciado mostra que podemos identificar uma regularidade na estrutura do problema, percebendo que diferentes cenários compartilham a mesma relação entre as variáveis. Ao reconhecer esse padrão, a atenção se volta para o procedimento recorrente, e não para os valores específicos, o que permite construir um algoritmo que pode ser aplicado a diferentes situações.

O primeiro padrão observado refere-se ao cálculo do custo total, que se repete para todos os operários, variando apenas os valores numéricos. Esse custo é expresso pela relação $(\text{valor por hora} \cdot \text{tempo}) + \text{transporte}$, uma expressão algébrica abstraída durante a resolução com o uso de conhecimento matemático. Ao ser reescrita por meio de códigos de programação, essa expressão tornou-se um modelo aplicado ao programa, o que permitiu concentrar esforços, tempo e atenção na organização das listas e dos blocos de comando, em vez da repetição de instruções para a execução do algoritmo.

Após o cálculo dos custos totais, identificamos um padrão de comparação e ordenação de valores numéricos, cujo objetivo é determinar o menor resultado e, assim, responder ao problema proposto. Esse raciocínio caracteriza problemas de minimização, nos quais a melhor escolha não é necessariamente aquela associada ao menor preço por hora, mas sim à que resulta no menor custo total.

A abstração está na seleção das informações essenciais à resolução, desconsiderando

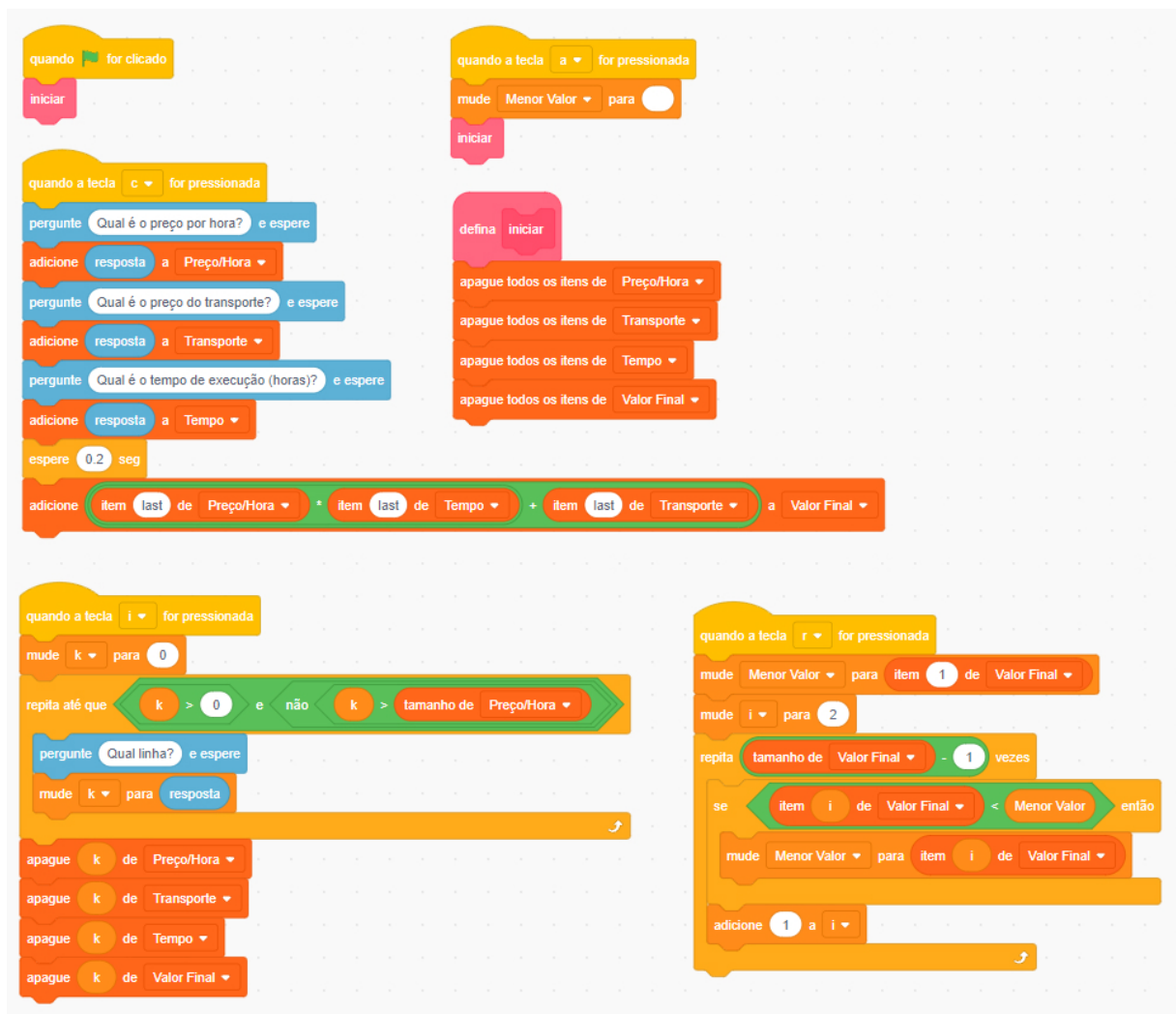
variáveis que não interferem diretamente na tomada de decisão, o que contribui para a simplificação e organização do modelo construído.

Produção de algoritmo

A resolução do problema mobiliza diferentes processos do Pensamento Computacional, destacando-se a produção do algoritmo, uma vez que o procedimento construído não se limita aos valores específicos apresentados no enunciado, mas pode ser aplicado a quaisquer dados que atendam às condições do problema. Essa característica confere à solução um caráter reutilizável e revela a passagem de uma resolução pontual para um modelo computacional mais amplo.

Na Figura 4.39 , apresentaremos os blocos que compõem o algoritmo da construção:

Figura 4.39: Algoritmo.



Fonte: Elaborada pela autora (2024).

Em relação à funcionalidade, o programa é iniciado ao clicar na bandeira verde ou ao pressionar a tecla a. Em seguida, são apresentadas quatro listas e a variável *Menor*

Valor, conforme ilustrado na Figura 4.38. Para a inserção das informações, o usuário deve pressionar a tecla *c*, que aciona as mensagens: *Qual é o preço por hora?*, *Qual é o preço do transporte?* e *Qual é o tempo de execução (em horas)?*. Os dados informados pelo usuário são inseridos, respectivamente, nas listas *Preço/Hora*, *Transporte* e *Tempo*.

Para o cálculo do custo total, são utilizados operadores que relacionam os elementos das listas, organizados na parte final do bloco de programação, logo abaixo das perguntas. O resultado dessas operações é exibido na lista *Valor Final*.

Caso o usuário deseje excluir informações, foi criado um bloco acionado quando a tecla *d* (*delete*) é pressionada, que exibe a pergunta *Qual linha?*. Após o recebimento da resposta, a linha selecionada é removida de todas as listas.

Por fim, foi concebido um bloco responsável por verificar os elementos da lista *Valor Final*, realizar a comparação entre eles e retornar o menor valor, respondendo, assim, à questão proposta.

Depuração

A depuração ocorreu ao longo de todo o processo de resolução do problema, manifestando-se tanto na interação com o objeto construído no Scratch quanto, de maneira mais explícita, nos vídeos que registraram o processo de elaboração do algoritmo.

Durante a implementação no Scratch, foram realizados testes com os dados da tabela para garantir a exatidão dos cálculos e da comparação. A depuração envolveu a verificação dos blocos de multiplicação, soma e controle condicional para a escolha do menor valor. Essa prática fortaleceu a compreensão da lógica aplicada e permitiu identificar falhas operacionais de forma iterativa.

5 CONSIDERAÇÕES FINAIS

Esta pesquisa teve como objetivo investigar como os processos constituintes do Pensamento Computacional (PC), formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração foram mobilizados na resolução de questões de Matemática do ENEM. Para isso, foi desenvolvido e aplicado o método de análise apresentado no Capítulo 3, que se fundamenta na articulação entre dois eixos complementares: o objeto computacional construído no Scratch e o registro em vídeo do processo de resolução, compreendido como fonte privilegiada para identificar ações, decisões, reformulações e momentos de teste e depuração ao longo da atividade.

A análise das questões apresentadas no Capítulo 4, realizada a partir das categorias analíticas derivadas dos seis processos do Pensamento Computacional descritas no capítulo teórico, evidenciou que esses processos se manifestam de forma recorrente e significativa na resolução de problemas matemáticos do ENEM quando mediados pelo Scratch.

Um primeiro achado relevante diz respeito à generalização. A construção de modelos computacionais permitiu transformar situações específicas descritas nos enunciados em estruturas generalizáveis, capazes de resolver não apenas o caso apresentado, mas também situações análogas a partir de novas entradas fornecidas pelo usuário. Esse movimento reforça o potencial do Pensamento Computacional para ampliar o alcance conceitual das soluções matemáticas.

Outro aspecto refere-se à interdependência dos processos. Embora, para fins analíticos, os processos do Pensamento Computacional tenham sido descritos separadamente, a investigação mostrou que eles atuam de maneira articulada e simultânea. A decomposição direciona o reconhecimento de padrões, que, por sua vez, sustenta a abstração e a produção de algoritmos, em um fluxo não linear e recursivo. Tal característica reforça a natureza dinâmica do Pensamento Computacional, diante de uma sequência rígida de etapas.

Destaca-se ainda o papel da depuração como processo de aprendizagem, tal como operacionalizada no método de análise ao considerar tanto as modificações realizadas no código quanto as ações registradas em vídeo. Os registros em vídeo evidenciaram que o erro, o teste e o ajuste constante são elementos constitutivos da construção do conhecimento. Mesmo quando não aparecem no código final frequentemente apresentado de forma “limpa” e funcional, esses momentos foram decisivos para a compreensão do problema e para o refinamento do modelo computacional, corroborando a ideia de que aprender envolve iterar, revisar e reconstruir.

Os resultados confirmam a estreita relação entre o Pensamento Computacional e o conhecimento em Matemática, conforme discutido no Capítulo 2, refere-se à compreensão do Pensamento Computacional como um tratamento de entes abstratos em interface com

a Matemática. Ao modelar problemas do ENEM no Scratch, observou-se que conceitos matemáticos como proporcionalidade, regularidade, operações aritméticas e relações funcionais foram reinterpretados à luz de estruturas algorítmicas.

Nesse contexto, o Pensamento Computacional oferece formas de abstração que, por vezes, não são explicitadas na matemática tradicional, especialmente quando se trata de descrever procedimentos passo a passo e dependentes de condições. O Scratch atuou, assim, como um micromundo construcionista, no qual foi possível “pensar com” a máquina e, simultaneamente, “pensar sobre” o próprio pensar, tornando visíveis processos cognitivos que geralmente permanecem implícitos nas resoluções convencionais.

Ao dialogar criticamente com as orientações da Base Nacional Comum Curricular (BNCC), com seu complemento voltado à Computação e com as discussões promovidas pela Sociedade Brasileira de Computação (SBC) acerca da definição de Pensamento Computacional, este trabalho busca a operacionalização por meio da resolução de questões do ENEM, buscando explicitar seus processos constituintes para além de uma abordagem restrita à elaboração de algoritmos e fluxogramas e evidenciar suas implicações para a Educação Matemática na Educação Básica.

Os achados sugerem que o uso de ambientes como o Scratch vai além de um emprego meramente instrumental ou ilustrativo. Quando integrado à resolução de problemas matemáticos, esse uso favorece o protagonismo do estudante, a tomada de decisões e o desenvolvimento da autonomia intelectual. Estratégias exploradas ao longo da pesquisa, como o uso de listas dinâmicas, variáveis, operadores e blocos de eventos, mostram-se passíveis de adaptação ao contexto escolar, fortalecendo a compreensão conceitual da matemática ao mesmo tempo em que desenvolvem habilidades associadas ao Pensamento Computacional.

Entre as limitações do estudo, destaca-se o desafio de categorizar processos que se manifestam de forma sobreposta e não linear, questão já antecipada na discussão metodológica do Capítulo 3, ao se reconhecer que a separação dos processos do Pensamento Computacional atende a uma necessidade analítica, e não a uma ordem natural de ocorrência. A necessidade de separar analiticamente ações que, na prática, ocorrem de modo integrado exigiu constantes retornos aos dados e ajustes no método de análise.

Como desdobramentos possíveis, sugere-se a aplicação desse método com estudantes da Educação Básica, a fim de investigar se a mobilização dos processos do Pensamento Computacional ocorre de maneira semelhante à observada entre pesquisadores. Além disso, a exploração de outras ferramentas computacionais, para além do Scratch, pode ampliar a compreensão sobre diferentes formas de operacionalizar o Pensamento Computacional na resolução de questões do ENEM.

Assumindo que o Pensamento Computacional constitui uma capacidade criativa, crítica e estratégica, esta pesquisa reafirma sua relevância para a formação integral do cidadão na sociedade contemporânea. Ao evidenciar como os processos do Pensamento

Computacional se articulam à Matemática na resolução de problemas do ENEM, o estudo contribui para fortalecer o diálogo entre teoria e prática, indicando caminhos possíveis para uma educação que valorize o pensar, o fazer e o refletir de forma integrada, em consonância com os referenciais teóricos que fundamentaram esta investigação.

Ao concluir este trabalho, reafirmamos também a motivação que originou. Esta pesquisa não emergiu apenas de uma inquietação teórica, mas de uma trajetória acadêmica e profissional comprometida com a compreensão de como conceitos matemáticos podem ganhar novos significados quando articulados a práticas computacionais. Ao longo do percurso descrito nos Capítulos 2, 3 e 4, fui desafiada a revisitar minhas próprias concepções sobre ensinar e aprender Matemática, reconhecendo no Pensamento Computacional não apenas um conjunto de processos, mas uma possibilidade concreta de ressignificação pedagógica.

Assumo, portanto, que a operacionalização dos processos: formulação do problema, decomposição, reconhecimento de padrões, abstração, produção de algoritmos e depuração, não se constituiu somente como objeto de análise, mas como experiência formativa enquanto pesquisadora. O movimento de dissecar as ações registradas em vídeo, confrontar o objeto construído no Scratch com os referenciais teóricos e explicitar as inter-relações entre os processos do Pensamento Computacional também produziu deslocamentos em minha própria prática investigativa. Essa dimensão formativa reforça a convicção de que integrar o Pensamento Computacional à Educação Matemática não é uma tendência passageira, mas um caminho epistemológico e pedagógico que amplia horizontes para professores e estudantes.

Assim, encerro este trabalho sustentando que o Pensamento Computacional, compreendido como capacidade criativa, crítica e estratégica conforme defendem Brackmann e Wing constitui-se como elemento estruturante para uma formação matemática que dialogue com as demandas contemporâneas. Mais do que responder à questão de pesquisa, esta dissertação consolida um posicionamento acadêmico, o de que pensar algoritmicamente, modelar, testar, depurar e generalizar são práticas intelectuais que fortalecem a autonomia e a autoria no aprender Matemática.

REFERÊNCIAS

ARAÚJO, A. P. F.; SOUZA, P. R.; SILVA, J. A. M. Uso do Scratch no processo de aprendizagem em sala de aula: relato de experiências de alunos do mestrado profissional de ensino tecnológico. **Revista Brasileira de Ensino Profissional e Tecnológico**, Amazonas, 2014.

BRACKMANN, C. P. **Desenvolvimento do Pensamento Computacional através de atividades desplugadas na educação básica**. 2017. Tese (Doutorado em Informática na Educação) – Universidade Federal do Rio Grande do Sul, Porto Alegre. Disponível em: <https://lume.ufrgs.br/handle/10183/172208>. Acesso em: 7 jan. 2026.

BRASIL. Ministério da Educação (MEC). **Base Nacional Comum Curricular (BNCC)**. Brasília: [s. n.], 2018. <https://basenacionalcomum.mec.gov.br/>. Acesso em: 7 jan. 2026.

BRENNAN, K.; RESNICK, M. New frameworks for studying and assessing the development of computational thinking. *In*: PROCEEDINGS of the 6th Annual Meeting Program. Washington, DC, USA: [s. n.], 2012. p. 1–25.

DANTAS, S. C. Pensando e resolvendo problemas com o GeoGebra. **Revista do Instituto GeoGebra Internacional de São Paulo**, [s. l.], v. 12, n. 2, p. 133–164, 2023. Disponível em: <https://revistas.pucsp.br/index.php/IGISP/article/view/63719>. Acesso em: 7 jan. 2026.

ESPADEIRO, Rui Gonçalo. O Pensamento Computacional no currículo de Matemática. **Educação e Matemática: Revista da Associação de Professores de Matemática**, n. 162, p. 5–10, 2021. Revista temática sobre Pensamento Computacional.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2002.

ISTE; CSTA. **Operational Definition of Computational Thinking**. [S. l.: s. n.], 2011. <https://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf>. Acesso em: 7 jan. 2026.

PAPERT, Seymour. **Mindstorms: Children, Computers, and Powerful Ideas**. New York: Basic Books, 1980.

PRADO, S. P. do. **Perspectivas sobre Pensamento Computacional construídas a partir do diálogo com educadores matemáticos**. 2024. Dissertação (Mestrado em Educação Matemática) – Universidade Estadual do Paraná, União da Vitória. Disponível em: <https://prpgem.unespar.edu.br/dissertacoes/defesas-2024>. Acesso em: 7 jan. 2026.

RESNICK, Mitchel. **Jardim de Infância para a Vida Toda: Por Uma Aprendizagem Criativa, Mão na Massa e Relevante para Todos**. Porto Alegre: Penso, 2020. ISBN 9786581334123.

RIBEIRO, L.; FOSS, L.; CAVALHEIRO, S. A. C. Entendendo o Pensamento Computacional. *In*: RAABE, A.; ZORZO, A. F.; BLIKSTEIN, P. (ed.). **Computação na Educação Básica: fundamentos e experiências**. Porto Alegre: Penso, 2020. p. 16–30.

SBC, Sociedade Brasileira de Computação. **Nota Técnica da Sociedade Brasileira de Computação sobre a BNCC-EF e a BNCC-EM**. [S. l.], 2018. Disponível em: <https://bit.ly/3uJmMAA>. Acesso em: 7 jan. 2026.

SBC, Sociedade Brasileira de Computação. **Referenciais de Formação em Computação: Educação Básica**. [S. l.], 2017. Disponível em: <https://bit.ly/3qSqzur>. Acesso em: 7 jan. 2026.

WING, Jeannette M. Computational thinking. **Communications of the ACM**, [s. l.], v. 49, n. 3, p. 33–35, 2006. Disponível em: <https://www.cs.columbia.edu/~wing/ct-portuguese.pdf>. Acesso em: 7 jan. 2026.

WING, Jeannette M. **Computational Thinking: definição refinada (2014)**. [S. l.: s. n.], 2014. Blog “40th anniversary” em site da CMU/SIGCAS.

WING, Jeannette M. **Computational thinking: What and Why?** Disponível em: <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>. Acesso em: 7 jan. 2026. 2011.