FACULDADE DE TECNOLOGIA DE SÃO PAULO
CURSO SUPERIOR EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROPOSTA DE APRENDIZADO BÁSICO DE LÓGICA DE PROGRAMAÇÃO COM SALA DE AULA INVERTIDA PARA INGRESSANTES NOS CURSOS SUPERIORES DA ÁREA DE TECNOLOGIA DA INFORMAÇÃO

Adilson de Jesus Candido Oliveira

Orientadora: Prof.ª Me. Simone Cristina Gonçalves Vianna

SÃO PAULO
2024

Adilson de Jesus Candido Oliveira

PROPOSTA DE APRENDIZADO BÁSICO DE LÓGICA DE PROGRAMAÇÃO COM SALA DE AULA INVERTIDA PARA INGRESSANTES NOS CURSOS SUPERIORES DA ÁREA DE TECNOLOGIA DA INFORMAÇÃO

Trabalho de Conclusão de Curso – TCC – apresentado como exigência parcial para obtenção do diploma de Tecnólogo em Análise e Desenvolvimento de Sistemas pela FATEC–SP.

Orientadora: Prof.ª Me. Simone Cristina Gonçalves Vianna

SÃO PAULO

2024

Oliveira, Adilson de Jesus Candido

PROPOSTA DE APRENDIZADO BÁSICO DE LÓGICA DE PROGRAMAÇÃO COM SALA DE AULA INVERTIDA PARA INGRESSANTES NOS CURSOS SUPERIORES DA ÁREA DE TECNOLOGIA DA INFORMAÇÃO / Adilson de Jesus Candido Oliveira. São Paulo, 2024.
62 p.

Monografia (Trabalho de Conclusão de Curso de Graduação) – Faculdade de Tecnologia de São Paulo – SP, Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas.

Área de concentração: Algoritmos e Lógica de Programação
Orientadora: Prof.ª Me. Simone Cristina Gonçalves Vianna

1. Lógica de Programação. 2. Sala de Aula Invertida. 3. Aprendizado Ativo.

DEDICATÓRIA

Dedico esta monografia ao Prof. Dr. Juarez Antonio Delibo, *in memoriam,* que foi meu professor de língua inglesa na FATEC–SP que me motivou e aconselhou a trilhar uma carreira no âmbito da docência.

AGRADECIMENTOS

Inicialmente, devo meus agradecimentos à Prof.ª Me. Simone Cristina Gonçalves Vianna, que proporcionou um acompanhamento de excelência durante o desenvolvimento da presente monografia, e sem o qual, o trabalho não teria sido realizado com êxito.

Aos professores da Faculdade de Tecnologia de São Paulo (FATEC-SP) que se dispuseram a participar da entrevista e que contribuíram com a presente monografia.

Aos alunos do curso de Análise e Desenvolvimento de Sistemas da FATEC-SP que responderam ao questionário e auxiliaram, desta maneira, na coleta de dados para a pesquisa realizada.

À organização Oxford Cursos Ltda. onde eu exerço o cargo de *professor de cursos livres* atualmente.

“Programming is the art of telling another human being what one wants the computer to do”

Donald Knuth

RESUMO

A lógica matemática é fundamental para o desenvolvimento do pensamento computacional, constituindo a base para o aprendizado de algoritmos e da lógica de programação. Muitos alunos, no entanto, apresentam dificuldades no aprendizado em lógica, sobretudo, provindo do ensino básico, que somada às metodologias tradicionais do ensino superior, afetam o desempenho acadêmico dos estudantes ingressos nos cursos superiores da área de Tecnologia da Informação, podendo resultar em baixo rendimento e, até mesmo, na evasão da graduação. Neste trabalho, aborda-se a metodologia da Sala de Aula Invertida, voltada para o aprendizado da lógica de programação. Assim, o objetivo é conceber um material de estudo prévio, para consumo em momento pré-aula (antes), para que o aluno se familiarize com as estruturas básicas da lógica de programação e desenvolva melhor compreensão quando em aula (durante). Para isso, são abordados os benefícios da Sala de Aula Invertida; em seguida, é apresentada a linguagem de blocos e as estruturas básicas da lógica de programação. Finalmente, para que o estudante averigue a sua compreensão em relação aos conteúdos previamente estudados, são propostos exercícios com as suas respectivas soluções sugeridas.

Palavras-chave: Lógica de Programação. Sala de Aula Invertida. Aprendizado Ativo.

ABSTRACT

Mathematical logic is fundamental to the development of computational thinking, constituting the basis for learning algorithms and programming logic. Many students, however, show difficulties in learning logic, especially from basic education, which, added to traditional higher education methodologies, affect the academic performance of students enrolled in higher education courses in the area of Information Technology, which may result in low performance and even dropout from graduation. In this paper, the Flipped Classroom methodology aimed at learning programming logic is addressed. Thus, the objective is to design prior study material to consume in pre-class (before) moment, so that the student becomes familiar with the basic structures of programming logic and develops better understanding in class (during). To this end, the benefits of the Flipped Classroom are discussed; then, it's presented the block language and the basic structures of programming logic. Finally, so that the student can check their understanding of previously studied content, exercises are proposed with their respective suggested solutions.

Keywords: Programming Logic. Flipped Classroom. Active Learning.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

INTRODUÇÃO

A lógica matemática é um componente essencial nos cursos superiores da área de Tecnologia da Informação. Isto se deve pelas competências de argumentação e de raciocínio lógico que constituem a base do pensamento computacional, o que as fazem amplamente utilizadas nas tradicionais disciplinas de Algoritmos e Lógica de Programação, assim como em disciplinas mais avançadas que, do mesmo modo, desenvolvam estas competências. É possível observar, todavia, que a lógica matemática, em geral, é pouco trabalhada no ensino básico, e por consequência, os alunos do ensino superior se veem com dificuldades quando ingressam em um curso superior que tenha estas competências como pré-requisito.

Tendo em vista essas dificuldades inicialmente apresentadas, procura-se abordar o uso da metodologia de Sala de Aula Invertida no estudo da lógica de programação para alunos ingressantes dos cursos superiores da área de Tecnologia da Informação. Neste sentido, levantam-se três questionamentos: (i) qual o papel da metodologia ativa de aprendizado na lógica de programação? (ii) a metodologia da Sala de Aula Invertida pode contribuir para o desenvolvimento da argumentação e do raciocínio lógico? (iii) a linguagem de blocos auxilia o aprendizado de algoritmos e de lógica de programação?

A partir dos questionamentos feitos, pretende-se, através de explicação e exemplificação de recursos básicos da lógica de programação, conceber um roteiro de estudos para auxiliar o aluno em momento pré-aula, o qual possa contribuir com o treino e desenvolver visão crítica e reflexiva acerca dos algoritmos e recursos que são comumente empregados na lógica de programação, ao mesmo tempo em que se procura estimular a característica autodidata e autônoma deste estudante.

Para tanto, uma abordagem ativa de aprendizado, sustentada pela análise qualitativa (entrevista com docentes) e quantitativa (questionário com estudantes) de dados e informações, respectivamente, é proposta, evidenciando a forma como a metodologia do aprendizado pode influenciar o protagonismo estudantil e estimular o raciocínio não linear que é típico da lógica de programação.

Utilizando-se, portanto, da técnica de pesquisa dedutiva, em que se parte do método geral da Sala de Aula Invertida para o aspecto particular do estudo dos algoritmos e da lógica de programação, esse trabalho é consolidado por intermédio de revisão bibliográfica de âmbito didático-pedagógica e também outras, relacionadas com Algoritmos e Lógica de Programação, aliado de entrevistas com docentes da área, bem como de um questionário

aplicado junto aos alunos do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da Faculdade de Tecnologia de São Paulo (FATEC-SP).

No capítulo um, analisa-se a deficiência de lógica matemática no ensino básico e das metodologias tradicionalmente empregadas e a sua devida influência no nível superior, apresentando a metodologia da Sala de Aula Invertida e suas vantagens, como proposta de contribuição ao aprendizado e diminuição da evasão de alunos ingressantes.

O capítulo dois, por sua vez, apresenta a linguagem de programação em blocos, elucidando as suas características, assim como as suas vantagens no processo de aprendizado de lógica. Ademais é explicado que os recursos básicos da lógica de programação são imprescindíveis na construção dos algoritmos.

Já no capítulo três, são propostos exercícios de elaboração de algoritmos referentes às estruturas básicas apresentadas no capítulo dois deste trabalho, junto de suas respectivas soluções sugeridas.

Finalmente, indicam-se as considerações finais, analisando o auxílio que essa monografia pretende fornecer para o ensino e aprendizado da lógica de programação para alunos ingressantes nos cursos superiores da área de Tecnologia da Informação.

CAPÍTULO 1 – OS DESAFIOS DO APRENDIZADO LÓGICO MATEMÁTICO

1.1 A Lógica Matemática no Ensino Básico

Conforme afirma Nascimento (2016, p.14-15), “a lógica matemática se faz presente desde uma operação simples de aritmética como a adição entre dois números quaisquer, até nas mais refinadas e complexas demonstrações de teoremas”. Deste modo, observa-se que a lógica matemática permeia as mais diversas vertentes do conhecimento, sendo, portanto, de fundamental importância para a compreensão de disciplinas da área de ciências exatas e suas tecnologias, bem como os demais componentes correlatos.

Tendo em vista que a lógica matemática tem uma aplicação demasiadamente abrangente dentro do contexto acadêmico, verifica-se que o seu ensino e aprendizado é de suma importância na educação básica, capacitando o estudante a lidar com diversas teorias e mecanismos abstratos que se subseguem conforme ele avança em seus estudos e alcança o ensino superior. Neste sentido, ressaltam-se as entrevistas realizadas, por meio deste trabalho, com quatro docentes da área de programação da FATEC-SP *(APÊNDICE–A)*, em que se foi possível constatar que a matemática é fundamental na área de computação, haja vista que a ciência da computação é um dos ramos da matemática aplicada.

Verifica-se também que a lógica matemática deve ser trabalhada no ensino básico através de uma noção abrangente e sistêmica, uma vez que esta ciência tem coligações com outras áreas do conhecimento, que igualmente dizem respeito ao raciocínio lógico, assim como Santos; Santos; Kleber (2007, p. 9) indicam, que não se trata de um eixo tecnológico isolado, mas sim de escopo multidisciplinar.

Orsano; Roberto (2018, p. 2) reforçam que a lógica matemática atinge o seu objetivo no ensino básico quando o estudante se torna apto a estabelecer conexões entre assuntos de natureza abstrata, seja numérica ou não, que o leve para a resolução de questões das mais diversas vertentes e de complexidade ímpar. E continuam dizendo que “[...] a maneira como a lógica é integrada à matemática escolar pode determinar o fracasso ou o êxito do desenvolvimento argumentativo dessa ciência.” (Orsano, Roberto, 2018, p. 3). Neste sentido, entende-se que não se trata, tão somente, de dominar a aplicação de estruturas pré-concebidas através de fórmulas, mas de compreender a argumentação presente no pensamento matemático que é norteado pelo raciocínio lógico.

Nascimento (2016, p.14), em adição, sugere que o componente curricular de matemática do ensino básico apresenta uma notável deficiência em relação à abordagem da lógica matemática, sendo que este tópico, em muitos casos, acaba sendo pouco trabalhado em

relação a outros temas da matemática, a exemplo de função, equação, geometria, trigonometria etc.

1.2 O Papel das Metodologias de Ensino

A metodologia de ensino é "conjunto de procedimentos didáticos, representados por seus métodos e técnicas de ensino" (Nérice, 1987, p.284), abrangendo, assim, as mais diversas formas e estratégias de se buscar e propor um estudo acerca do conhecimento que deve ser adquirido pelos alunos, empregando, neste processo, atividades que visem a compreensão e aplicação daquilo que se é trabalhado dentro e fora do ambiente escolar.

Logo, é possível observar que as metodologias são projetadas para se adequarem às habilidades e competências dos estudantes e capacitá-los a refletir, compreender e utilizar as informações da melhor forma possível, e ainda permitir com que assuntos trabalhados sejam adaptados para as suas práticas, inclusive, para além do contexto escolar.

Sobre o caráter interdisciplinar da metodologia de ensino, Brighenti; Biavatti; Souza (2015, p.290) comentam que ela pode ser adaptada para diversos contextos, tanto no que se refere ao ensino e aprendizagem individual (de apenas um estudante em específico), quanto para grupos constituídos de um pequeno ou grande número de estudantes.

Deste modo, é possível afirmar que cabe aos professores, instrumentalizar a metodologia de ensino, de modo que possam lidar com os mais distintos obstáculos de aprendizado, considerando que enquanto há alunos que apresentam facilidade com a abordagem e aplicação de determinados conteúdos, há outros que sentem uma notável dificuldade em dominá-los (Santos, Santos, Kleber, 2007, p. 12). Para lidar com tal situação, deve-se empregar diferentes técnicas de ensino, de maneira que alunos que, eventualmente, tenham maior dificuldade, também possam construir aprendizado acerca do conteúdo, promovendo, assim, o desenvolvimento acadêmico de todos.

Atualmente, a metodologia de ensino lida com variadas Tecnologias Digitais da Informação e Comunicação (DTICs), as quais podem ser utilizadas em prol do ensino-aprendizagem, assim como Brighenti; Biavatti; Souza (2015, p.283) sugerem, a tecnologia impacta largamente na forma de aprendizado, concedendo aspectos como a imersão e profundidade com que conteúdos podem ser elaborados e desenvolvidos, mas também reflete uma forma diferente de se preparar para lidar com as dificuldades dos alunos, resultando em um contínuo esforço por conceber metodologias mais eficazes.

A partir das diferentes formas de estruturar a metodologia de ensino, ressalta-se que não há uma que seja reconhecida como mais ou menos eficiente na promoção de um

aprendizado significativo, mas sim, aquela que melhor se relacione com o contexto educacional em que se está trabalhando, e que se adeque à ementa da disciplina abordada, o que corrobora com o mencionado durante as entrevistas realizadas *(APÊNDICE A)*, uma vez que, conforme considerado pelos entrevistados, a quantidade de horas-aula disponíveis em uma disciplina pode ampliar ou limitar uma abordagem de trabalho mais dinâmica do conteúdo programado.

1.2.1 Metodología da Sala de Aula Invertida

Assim como Lovato; Michelotti; Silva; Loreto (2018, p.157) indicam, as metodologias ativas são aquelas que se diferenciam das formas tradicionais de ensino por atribuir ao estudante o caráter de protagonista de seu próprio desenvolvimento, não se limitando estritamente à condução do docente para a apresentação dos conteúdos a serem estudados. Neste meio, a metodologia ativa confere ao professor um caráter de intermediador do aprendizado, consolidando-se como a ligação entre o estudante e o conteúdo.

"As metodologias ativas são pontos de partida para avançar para processos mais elaborados de reflexão, de integração cognitiva, de generalização, de reelaboração de novas práticas." (Morán, 2015, p.18), portanto, cabe às técnicas diversificadas de ensino a atuação de modo eficaz na maneira como o processo de ensino-aprendizado se concretiza, buscando atingir maior domínio do conteúdo estudado, melhorando, consequentemente, a capacidade do estudante de compreender e aplicar o conteúdo aprendido.

Dentre as diferentes técnicas abrangidas pelas metodologias ativas, enfatiza-se a Sala de Aula Invertida, que pressupõe um aprendizado em ordem inversa ao que é comumente é empregado no sistema de ensino tradicional. Como salientado por Lovato; Michelotti; Silva; Loreto (2018, p.165), na Sala de Aula Invertida, o conteúdo a ser estudado é repassado aos estudantes em momento prévio às aulas, muitas vezes de modo digital, o que garante que o contato com o conteúdo seja feito num formato autodidata, em que o aluno tem autonomia para estudá-lo no seu tempo e ritmo, corroborando para que durante a aula, o assunto possa ser tratado com maior participação, aprofundamento e protagonismo estudantil.

Para que a metodologia da Sala de Aula Invertida seja concretizada, é necessário, entretanto, maior comprometimento por parte do docente que, por sua vez, deve elaborar conteúdos e materiais para serem expostos em momento pré-aula, conforme foi indicado por Schneiders (2018, p.7), o que faz com que a atenção destinada à preparação das aulas seja intensificada.

E Schneiders (2018, p.8) segue indicando que os conteúdos para estudo em momento pré-aula refletem uma qualidade melhor do tratamento da disciplina durante a aula, fazendo com que a participação do aluno seja ampliada, o que é confirmado pelos docentes entrevistados *(APÊNDICE A)*, uma vez que aspectos como a motivação e senso de responsabilidade são acentuados, haja vista que é atribuído aos estudantes o papel de difusores de ideias, opiniões e de construtores do próprio processo de aprendizagem do qual participam ativamente.

1.2.2 Vantagens do Aprendizado em Momento Pré-Aula

O aprendizado em momento pré-aula fornece diversos benefícios, dentre eles o de preparar o estudante para exercer uma função mais ativa no processo de aprendizado, uma vez que ele já traz consigo parte do conhecimento daquilo que lhe será exposto durante a aula.

Assim como afirmam Bergmann; Sams (2016 apud Magalhães et al, 2023, p. 19), quando o processo da sala de aula inversa ocorre, há maior proximidade dos alunos com a matéria exposta, permitindo, inclusive, uma adaptação mais apropriada das limitações individuais dos alunos, tanto no que se refere à facilidade de aprendizado, quanto no momento mais propício para o estudo de cada um.

Bergmann; Sams (2016 apud Magalhães et al, 2023, p. 19) ainda reforçam que a metodologia da Sala de Aula Invertida permite uma aproximação do tratamento dos alunos com os docentes, tendo em vista que os estudantes assumem um papel de maior protagonismo no aprendizado da matéria.

O processo de ensino-aprendizado se dá de modo abrangente na metodologia de Sala de Aula Invertida, visto que “o ensinar e aprender acontece (sic) numa interligação simbiótica, profunda, constante entre o que chamamos mundo físico e mundo digital”. (Morán, 2015, p.16), desta maneira, compreende-se que o estudo abrange os recursos tecnológicos que são disponíveis para a educação no momento atual, permitindo um envolvimento cada vez maior do aluno no processo de aprendizagem.

Rodrigues; Correia (2023, p.3) complementam sobre os benefícios da Sala de Aula Invertida (SAI):

> É esperado que a SAI tenha muitos efeitos benéficos para os discentes, como desenvolvimento de autonomia e iniciativa para aprender; possibilidade de estudar em seu próprio ritmo; desenvolvimento de habilidades de resolução de problemas.

Compreende-se, assim, que o estímulo ao desenvolvimento da autonomia da abordagem dos assuntos estudados e da autossuficiência de pensamento, aliado à

conveniência de propor ao estudante a aprendizagem no momento em que ele se encontrar mais capaz para fazê-lo e mediante a sua própria facilidade, induzem para que a metodologia da Sala de Aula Invertida proponha ganho notório no processo de ensino-aprendizagem.

Em resumo, entende-se que a Sala de Aula Invertida busca não somente com que os conteúdos sejam ministrados de modo a cumprir com aquilo que é proposto no projeto pedagógico dos mais variados cursos, mas também que o aprendizado seja consolidado de maneira mais efetiva e internalizada, assim como o que foi defendido pelos professores entrevistados, atuantes no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da FATEC-SP *(APÊNDICE A)*, visto que afirmam que exercícios para resolução no momento pré-aula permitem com que os alunos tentem fazê-los por si próprios, refaçam quantas vezes desejarem, e postem quando chegarem em uma solução final.

1.3 A Evasão dos Cursos de Tecnologia da Informação

Mesmo que as entrevistas para este trabalho tenham sido desenvolvidas apenas no contexto da educação superior tecnológica pública, apresenta-se, inicialmente, apuração realizada pelo SEMESP, o qual traz resultados da educação superior, tanto pública quanto privada. Na sequência, são abordadas bibliografias que ressaltam o impacto que um ensino básico deficitário acarreta no ensino superior e, finalmente, são abordados os resultados obtidos a partir de questionários aplicados junto a alunos da FATEC-SP.

Conforme mensurado pelo SEMESP (2024, p.32-33) através do Mapa do Ensino Superior no Brasil, na rede pública de ensino superior, a graduação de Sistemas de Informação é o 6° curso com mais alunos matriculados na modalidade presencial e Ciência de Dados é o 5° curso com mais alunos matriculados na modalidade a distância até 2022.

Segundo o SEMESP (2024, p.39), o percentual de desistência acumulada de alunos matriculados em cursos de graduação (tanto na área de TI quanto de outras) em instituições públicas está na margem de 38,3% para a modalidade presencial e 46,6% para a modalidade a distância durante o período de 2018 a 2022.

Ainda de acordo com o SEMESP (2024, p.31-33), no que se refere às instituições privadas de ensino superior até o período de 2022, a graduação em Sistemas de Informação é o 16° curso com mais alunos matriculados na modalidade presencial e este mesmo curso é o 4°com mais alunos matriculados na modalidade a distância.

Em adição, o percentual de desistência acumulada destes alunos (tanto da área de TI quanto de outras) em instituições privadas está na base de 57,4% para a modalidade presencial

e 64,9% para a modalidade a distância de 2018 a 2022 como foi ilustrado por SEMESP (2024, p. 39).

Assim, nota-se que os cursos de Sistemas de Informação e Ciência de Dados, tanto no âmbito do ensino superior público quanto no privado, como graduações no eixo da Tecnologia da Informação, podem ser alvos de grande quantidade das evasões ocorridas, e ao investigar quais fatores contribuem para isso, encontram-se motivos distintos, como abordado por Archimedes (2020, p. 44-45), sendo eles: “tipos de evasão, cancelamento de matrícula, trancamento de matrícula, transferência interna de curso, transferência externa de curso, transferência de instituição de ensino superior, abandono, reprovação.”

Embora existam diversos fatores que corroborem para as evasões, observa-se que o mau desempenho em disciplinas é um fator determinante para a continuidade no curso, e em geral, quando os estudantes sentem dificuldades em grande medida, muitos se sentem inclinados a abandonar o curso de tecnologia que estão cursando.

A dificuldade existente no aprendizado de disciplinas como a lógica de programação é, em diversos casos, apontada como sendo decorrência de um base de lógica matemática ineficiente, assim como Blissari (2018, p.22) ressalta, o ensino básico que não foi concebido de maneira adequada, afeta a aptidão que o estudante tem quando prossegue em sua carreira acadêmica no nível superior.

Através de questionário realizado com alunos do curso de Análise e Desenvolvimento de Sistemas da FATEC-SP *(APÊNDICE C)* no período de 21/09/2023 até 08/10/2023, foi possível obter 149 respostas *(APÊNDICE D)*, ilustrando a perspectiva dos referidos alunos quanto à problemática no ensino-aprendizagem de lógica de programação no contexto atual de seus cursos de graduação.

Inicialmente, é possível verificar que uma quantidade expressiva de alunos (48,3%) responderam que sentem dificuldades muito frequentes ou frequentemente nas disciplinas que exigem a elaboração de algoritmos. Além disso, verifica-se que essa dificuldade está usualmente relacionada à natureza não linear que o desenvolvimento do pensamento computacional exige dos estudantes aliada a eventuais abordagens que são empregadas no contexto do ensino de tais disciplinas.

Ainda com relação ao questionário aplicado aos alunos, ressalta-se que a maioria destes, perfazendo 70,4%, enxergam que o raciocínio lógico-matemático é muito importante ou importante para as disciplinas que envolvem Algoritmos e Lógica de Programação, fato este que reforça o quão imprescindível o raciocínio lógico é para o estudo da programação e das demais vertentes que norteiam o pensamento algorítmico.

No que se refere às dificuldades em acompanhar as aulas de algoritmos (entende-se aulas que envolvem algoritmos como Algoritmos e Lógica de Programação, Estrutura de Dados e Análise de Algoritmos), uma quantidade notável de alunos (36,2%) declara já ter trancado alguma disciplina que envolvia algoritmos por não conseguir acompanhar o ritmo das aulas, sendo que esta frequência mencionada pode oscilar, desde raramente até muito frequentemente *(APÊNDICE D)*.

Contudo, boa parte dos alunos consultados (62,4%) informou que já se sentiu propenso a trancar uma disciplina de algoritmos após não ir bem em provas ou trabalhos iniciais. Neste sentido, apesar de que a quantidade de alunos que efetivamente trancam tal disciplina seja menor do que aquela que cogita o seu trancamento, pode-se notar preocupação considerável neste quesito por parte dos alunos.

Em adição, 68,4% dos alunos consultados acreditam que têm um aproveitamento maior na aula quando se preparam antes para um determinado conteúdo, o que reflete a provável maior facilidade de aprendizado quando há um contato prévio com o conteúdo que está sendo estudado.

Surpreendentemente, a metodologia da Sala de Aula Invertida é reconhecida entre os alunos consultados, alcançando um percentual de 75,8% nas declarações dos que já ouviram falar sobre tal abordagem. Embora tal reconhecimento se dê em diferentes graus de profundidade; enquanto alguns alunos já ouviram falar da metodologia com muita frequência e outros raramente, é possível averiguar a presença de tal assunto no meio acadêmico.

Finalmente, 52,4% dos alunos demonstraram acreditar que a Sala de Aula Invertida pode representar em ajuda considerável para cursar Algoritmos e Lógica de Programação, o que reflete que as metodologias ativas começam a ganhar espaço no universo dos estudantes, passando a ser conhecidas e aceitas por eles.

CAPÍTULO 2 – ALGORITMOS EM BLOCOS

2.1 A Linguagem em Blocos

A linguagem em blocos segundo Sousa; Farias; Carvalho (2020, p.1514) "[...] utiliza ações de arrastar e soltar blocos gráficos ou físicos, que correspondem aos componentes de um programa escrito em linguagem textual". Compreende-se, assim, que a maneira de ilustrar os comandos constituintes do código fonte é representada por segmentos de código que são arranjados de modo a estabelecer uma sequência de operações.

Esta ordenação visual permite melhor compreensão de como a lógica de programação é constituída e quais as suas características fundamentais. A construção do algoritmo através de blocos permite a elaboração do raciocínio de modo lúdico e didático, fazendo com que o(s) que o esteja(m) elaborando conte(m) com um auxílio de estruturas pré-determinadas, como indicado pelos professores entrevistados *(APÊNDICE A)*, os quais colocam, em linhas gerais, que a construção de algoritmos básicos envolvem, muitas vezes, operações matemáticas, e a utilização da linguagem em blocos envolvendo tais operações pode ajudar o estudante a desenvolver o raciocínio lógico-matemático.

Corroborando com essa proposta, como Junior (2017, n.p.) indica, a abordagem da lógica de programação através de blocos permite que suas operações sejam sugestivas e intuitivas, auxiliando no processo de reconhecimento, compreensão e de solução de erros, uma vez que problemas de compilação são mais facilmente encontrados e solucionados através da dinâmica dos blocos. Esta abordagem possibilita com que o pensamento algorítmico não linear seja aprimorado de modo contínuo por parte dos estudantes.

Muitas das vantagens no processo de aprendizado da lógica de programação são conferidas às linguagens em bloco visto que uma das funções primordiais destas linguagens é evidentemente induzir quem está construindo o algoritmo a pensar de maneira lógico-computacional, assim como acontece com as linguagens de programação textuais. Como colocam Sousa; Farias; Carvalho (2020, p.1514), tanto as linguagens em bloco quanto as textuais pressupõem o domínio do pensamento algorítmico em sua elaboração.

Do ponto de vista metodológico, segundo Rodrigues (2019, p.16), o método de ensino da lógica de programação é melhor conduzida através do emprego de técnicas dinâmicas de ensino e aprendizagem que permitam ao estudante se aproximar da solução proposta do algoritmo, ao mesmo tempo que desenvolve a sua visão crítica no segmento lógico matemático, e a programação em blocos busca associar tais objetivos.

2.2 Visualização de Algoritmos

No processo de aprendizado da lógica de programação, a visualização dos algoritmos em blocos configuram uma das maiores vantagens frente aos algoritmos textuais, conforme é denotado por Rodrigues (2019, p.16), visto que o aspecto visual permite aos estudantes a compreensão e distinção das funcionalidades de cada parte do código fonte com maior facilidade, ampliando a sua capacidade de percepção de sequências lógicas e conduzindo-os à diversas formas de elaboração de algoritmos para a solução de problemas de programação, eventualmente encontrados nas aulas.

A fim de que a visualização efetiva dos algoritmos seja alcançada, as ferramentas de programação em blocos, de modo geral, visam expor um painel conciso e objetivo contendo os recursos que podem ser empregados no código fonte, não se tratando de digitar tanto quanto na linguagem textual, mas sim de prover conexão entre blocos de códigos que exercem determinadas funcionalidades de modo dinâmico e adaptativo, assim como Souza; Falcão; Mello (2021, p.11) sugerem.

Deste modo, conforme indicado pelas respostas do questionário aplicado aos alunos do curso superior em Análise e Desenvolvimento de Sistemas da FATEC-SP *(APÊNDICE D)*, acredita-se que o estudo prévio de um conteúdo pode influir em uma melhor facilidade de assimilação e, consequentemente, de aprendizado. No contexto deste trabalho, o estudo prévio proposto faz uso de algoritmos em blocos, por sua vez, mais visuais do que algoritmos em linha de comando, fazendo com que as estruturas clássicas de programação sejam observadas a partir de suas características singulares e da maneira como se encadeiam.

Atribui-se uma notável importância para o aspecto visual no âmbito do aprendizado, pois as ilustrações representam uma linguagem não verbal que indicam com detalhes ímpares aquilo que se expressa, em geral, com mais dificuldade na vertente textual. Particularmente, no que se refere à lógica de programação, a visualização representa uma forma tangível de se esboçar o raciocínio abstrato que se está concebendo.

Desta maneira, como Souza; Falcão; Mello (2021, p.11) indicam, um dos fatores primordiais das linguagens em blocos é a usabilidade, pois a capacidade de aprendizado e adaptação ao uso de tais plataformas constituem fator base para a compreensão e construção da estrutura fundamental que integra o raciocínio típico de Algoritmos e Lógica de Programação.

2.3 Algoritmos em Blocos para Análise Pré-Aula

Através da pesquisa realizada acerca da metodologia da Sala de Aula Invertida, foi possível verificar que os algoritmos em blocos analisados pelos alunos, em momento anterior à aula, permitem que eles tenham contato prévio com os conceitos básicos que regem a programação, capacitando-os, assim, para melhor entendimento de como os recursos da programação são alocados com vistas à construção de um algoritmo capaz de solucionar um problema em questão.

Segundo Junior (2017, n.p.), a programação em blocos proporciona com que o estudante, independentemente da idade, tenha a capacidade de pré-visualizar como os elementos fundamentais da programação são elencados, e como eles interagem entre si dentro do código fonte em questão.

Para desenvolver algoritmos em blocos, é possível fazer uso da plataforma Scratch[1], que conforme Rodrigues (2019, p.17), é uma linguagem que permite realizar a construção de seu código fonte através de blocos e também fornece uma melhor visualização para a detecção de caminhos assumidos pelo código, e mesmo, de problemas relacionados à lógica de programação e que, eventualmente, ocorram. Até mesmo as variáveis que fazem parte do programa podem ser acompanhadas, passo a passo, permitindo assim verificar a mudança de seus valores a partir de cada iteração.

Reitera-se que o Scratch é uma plataforma digital gratuita alcançada através de navegadores e pode ser acessada livremente mediante a criação de um cadastro em sua página, além de propiciar a publicação de diversos projetos elaborados por estudantes do mundo todo.

Ainda de acordo com Rodrigues (2019, p.17), o Scratch viabiliza a reunião de elementos estáticos, saídas que resultam de operações simples ou complexas, saídas que reproduzem animações, dando possibilidade para receber diversos tipos de interação por parte de quem elabora o algoritmo.

O Scratch concede, ainda, a visualização do comportamento do algoritmo a todo o instante, uma vez que a ordem com que os elementos interpretados são destacados na interface de codificação, o que garante a respectiva visualização da resposta imediata do algoritmo mediante a ajustes realizados pelo estudante, fator este que é decisivo no processo de aprendizado.

Souza; Falcão; Mello (2021, p.11) indicam que a usabilidade de plataformas dinâmicas de programação em blocos se dá essencialmente por meio das interações que são facilitadas

[1] SCRATCH, 2023. Disponível em: < https://scratch.mit.edu/ > Acesso em: 01 set. 2023

pela colocação dos blocos de comandos, pois as diversas funcionalidades possíveis são exibidas ao estudante, para que ele possa selecionar aquelas que dizem respeito ao algoritmo que ele está projetando no momento.

“Esse desenvolvimento está aliado à lógica de programação de uma maneira mais intuitiva e agradável, pois esta linguagem proporciona o encapsulamento de detalhes sobre a sintaxe de linguagens de programação textuais”. (Rodrigues, 2019, p.17), deste modo, a distinção dos comandos que são possíveis na codificação e a compreensão das características de cada recurso que deve ser elencado na linguagem Scratch para a construção do algoritmo é amplamente facilitada, e permite também que a estrutura geral do programa seja compreendida pelo estudante, de modo que a sua preocupação não passe a se limitar tanto à sintaxe da linguagem em si, mas sim ao desenvolvimento raciocínio que conduza ao algoritmo solucionador de um dado problema.

Vale ressaltar que a interface de construção de blocos do Scratch é altamente intuitiva, em que comandos são classificados em subconjuntos, de acordo com a finalidade dos mesmos, elencando os recursos de interação com o usuário na categoria “eventos”, os laços na classe “controle”, os testes lógicos e matemáticos na área “operadores” e assim por diante, o que promove uma facilidade em sua identificação, compreensão e utilização por parte do estudante.

Em adição, ressalta-se que o ambiente de interpretação do código é realizado em tempo real através de uma janela de simulação que exibe o comportamento dos componentes, podendo ser interpretado o código completo ou, até mesmo, segmentos do código, o que facilita o processo de experimentação e a correção de eventuais erros durante a lógica do desenvolvimento.

A seguir são abordados os operadores relacionais, lógicos, condicionais, laços, listas e funções que compõem os recursos fundamentais no encadeamento do fluxo lógico de um programa, fazendo com que, a partir de cada iteração, seja possível aproximar-se de um dado objetivo.

2.3.1 Operadores relacionais

Os operadores relacionais cumprem o papel de efetuar uma comparação entre dois elementos que são do mesmo tipo. Estes elementos conforme indicado por Forbellone; Eberspächer (2005, p.21) podem ser “constantes, variáveis ou expressões aritméticas".

A relação dos operadores relacionais é averiguada no quadro ilustrado a seguir, indicando a notação que é comumente empregada nas diversas linguagens de programação atuais.

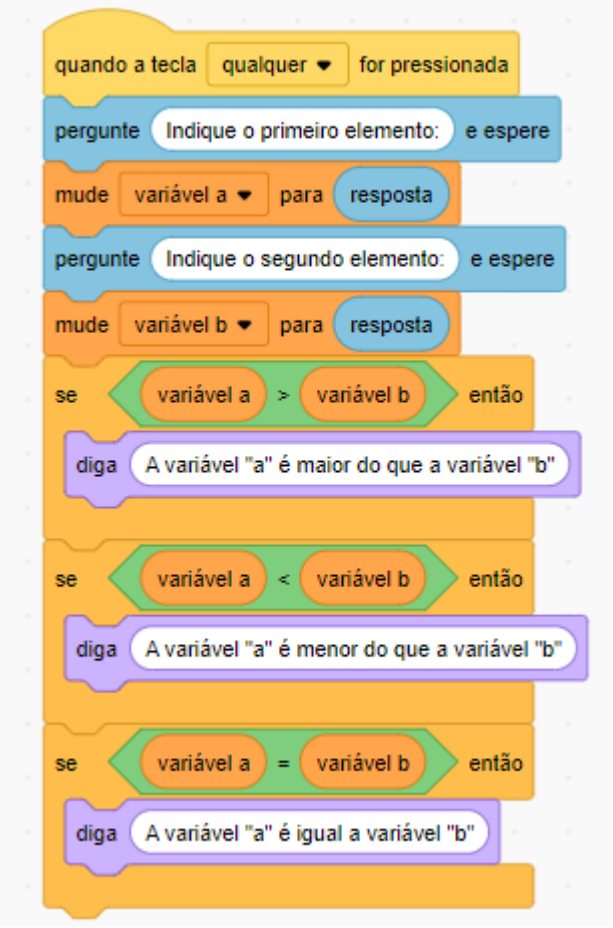
Quadro 1 – Operadores relacionais

>	>=	<	<=	==	!=
maior	maior ou igual	menor	menor ou igual	igual	diferente

Fonte: Adaptado de SCHILDT (1996, p.44)

Na sequência, exemplifica-se um algoritmo em Scratch que recebe duas variáveis, e faz uma série de testes para verificar a relação entre as variáveis.

Figura 1 – Algoritmo para averiguação de relação entre variáveis



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

2.3.2 Operadores lógicos

Os operadores lógicos, assim como os operadores relacionais, são voltados para comparações entre valores, porém os operadores lógicos não se limitam à comparação de apenas dois valores, podendo ser efetuadas comparações de quantidade arbitrária de elementos.

De acordo com Puga; Rissetti (2009, p.41) "a avaliação das expressões nas quais são utilizados os operadores lógicos está resumida na tabela-verdade".

Tabela 1 – Tabela Verdade

Premissa a	Premissa b	a OU b	a E b	NÃO a	NÃO b
V	V	V	V	F	F
V	F	V	F	F	V
F	V	V	F	V	F
F	F	F	F	V	V

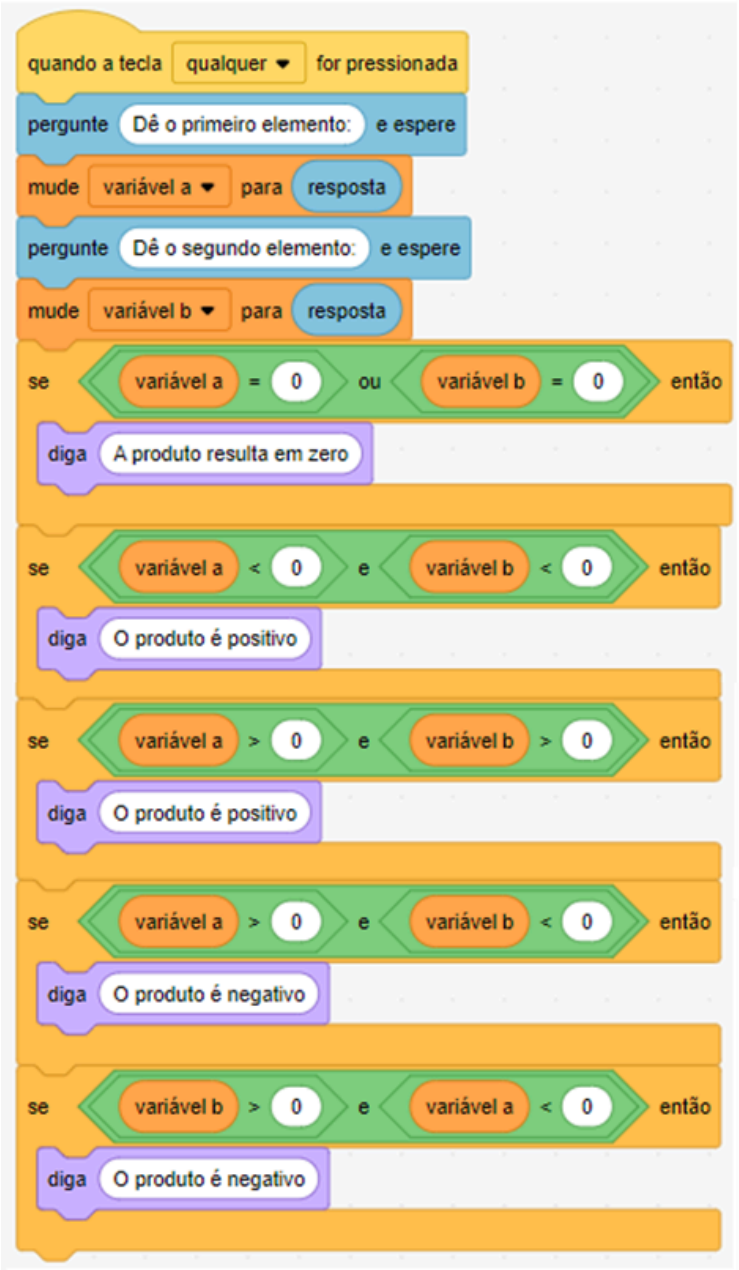
Fonte: Adaptado de PUGA; RISSETTI (2009, p.42)

Desta maneira, os operadores lógicos são: OU que representa a disjunção, dando resultado verdadeiro quando pelo menos um elemento dentre os comparados for verdadeiro; E que, por sua vez, indica a conjunção, retornando verdadeiro quando todos os elementos comparados são verdadeiros, e NÃO que ilustra a negação que nega um elemento qualquer. (Mathias, 2017, p.82).

Deve-se ressaltar que a notação empregada na tabela-verdade pode variar ligeiramente de acordo com a autoria, enquanto alguns autores utilizam a indicação de premissas pelas letras “a” e “b”, outros indicam por “p” e “q”, sendo que tal diferença também pode ser encontrada na simbologia adotada nos operadores encontrados no contexto das diversas sintaxes das linguagens de programação.

Na sequência, exemplifica-se um algoritmo em Scratch que utiliza operadores lógicos a fim de analisar o produto da multiplicação entre dois números racionais que foram indicados pelos usuários, e então sugere que o produto da operação seja: zero, positivo ou negativo.

Figura 2 – Algoritmo para análise lógica



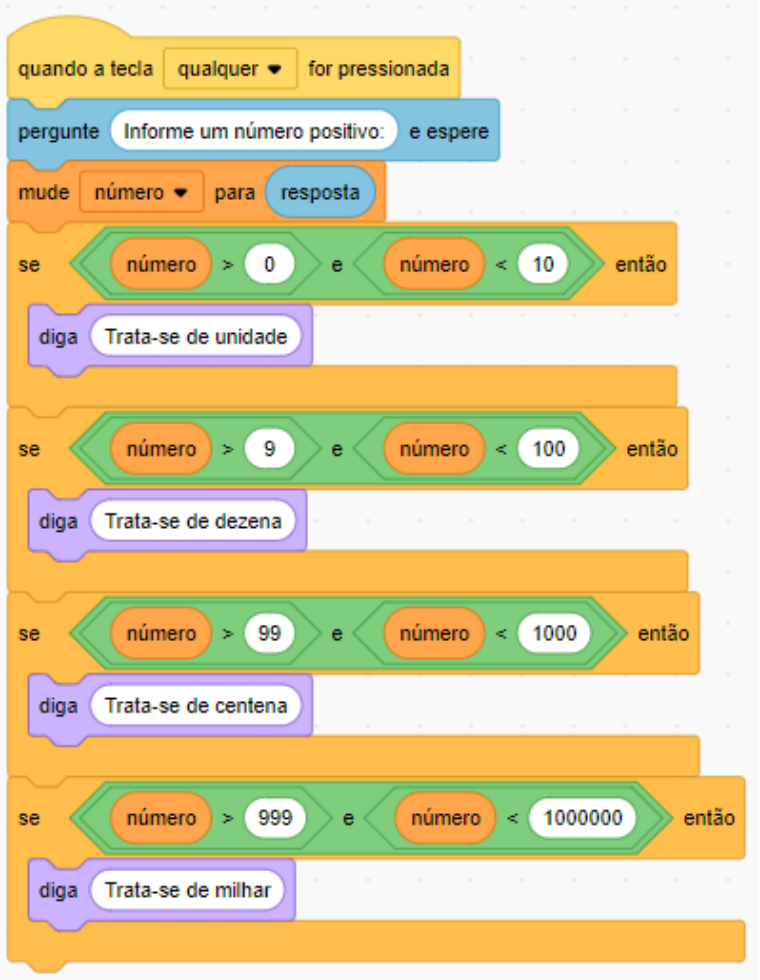
Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

2.3.3 Condicionais simples

Conforme indicado por Forbellone; Eberspächer (2005, p.33), o condicional simples se faz útil no instante em que é necessário realizar a avaliação de uma dada expressão, assim, a ação é realizada se o resultado verdadeiro, mas a mesma ação não é realizada se o resultado for falso. Lutz, Ascher (2007, p.163) reforçam que cabe ao condicional simples escolher dentre as expressões do código fonte, aquela que vai ser efetivamente reproduzida pela linguagem em questão.

Na sequência, apresenta-se um exemplo de algoritmo em Scratch que classifica um número através de testes encadeados pelo condicional simples.

Figura 3 – Algoritmo comparação com condicional simples



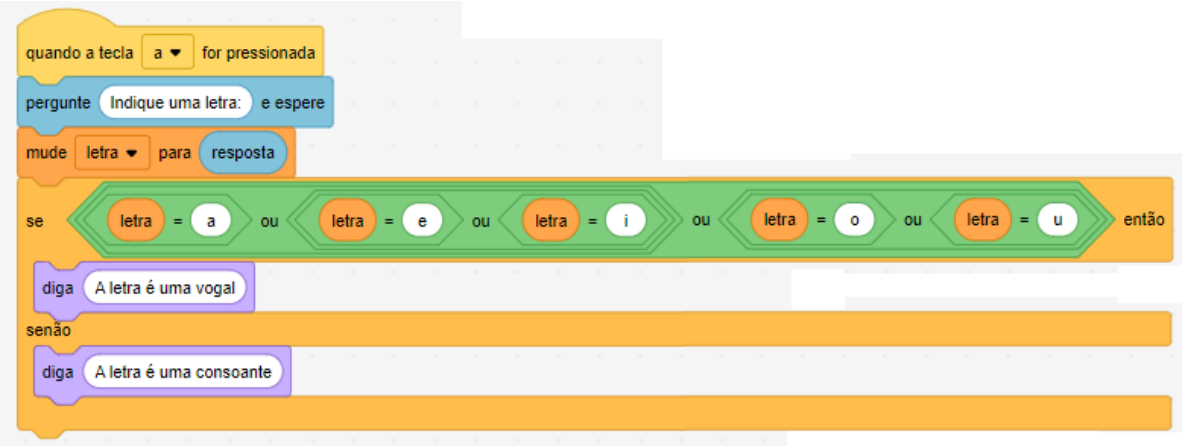
Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

2.3.4 Condicionais compostos

Assim como os condicionais simples, os condicionais compostos também permitem a escolha de qual expressão vai ser executada dentre um agrupamento de diversas expressões, mas tal como reforça Schildt (1996, p.65), os condicionais compostos permitem a execução mandatória de uma dada condição apesar das demais condições testadas serem classificadas como sendo falsas.

Puga; Rissetti (2009, p.58) reforçam que os condicionais compostos abordam conjuntos de expressões, e quando o resultado é verdadeiro, um conjunto de elementos é realizado, mas quando o resultado avaliado é falso, outro conjunto de elementos é realizado. Tal estrutura é particularmente útil quando não se pode prever as possibilidades de encadeamento de elementos que são alinhados, tais como os operadores e condicionais que são ilustrados na sequência.

Figura 4 – Algoritmo para comparação com condicional composto



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

No exemplo, ilustrado conforme figura 4, um algoritmo em Scratch que distingue se uma letra indicada é vogal ou consoante. Para tanto, emprega-se a estrutura de condicionais compostos que primeiramente verifica se a letra informada se enquadra na categoria de vogal e, caso contrário, assume-se que a mesma é uma consoante.

Reitera-se que também é possível efetuar a comparação da letra digitada com as consoantes, porém devido à grande quantidade de consoantes no alfabeto, o algoritmo se torna menos eficiente e mais complexo de ilustrar no trabalho.

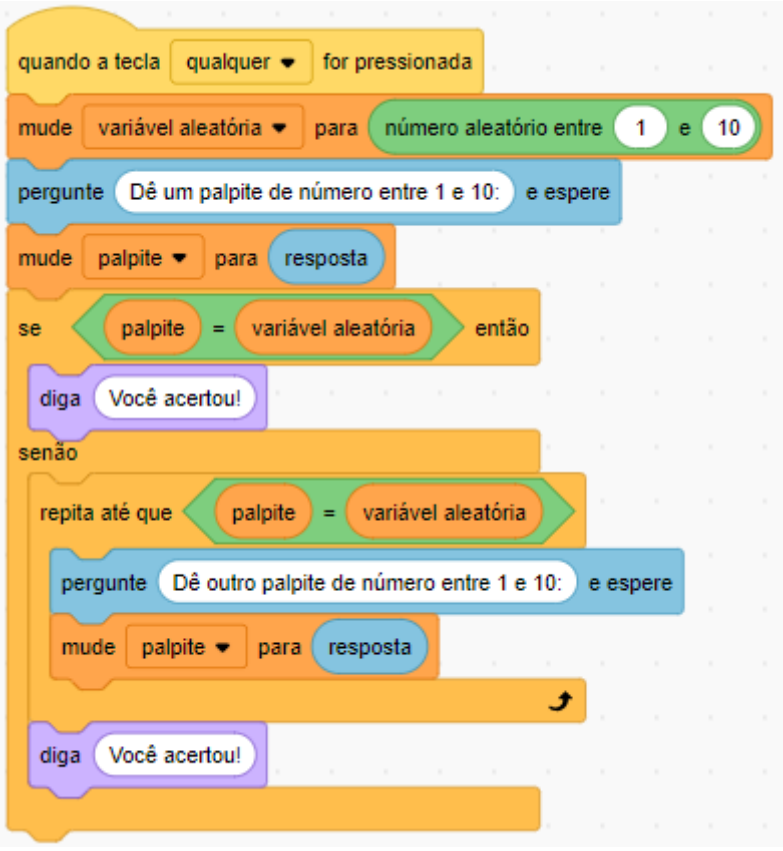
2.3.5 Laços de repetição

Os laços de repetição são estruturas utilizadas para reproduzir uma determinada etapa do código fonte por certa quantidade de vezes. Este recurso é comumente empregado a fim de efetuar a resolução de algoritmos iterativos (aqueles em que ocorre um incremento em suas variáveis). Os referidos laços de repetição são tradicionalmente de dois tipos distintos, o laço de repetição com teste no início e o laço de repetição com contador.

A repetição com teste no início, como indicado por Forbellone; Eberspächer (2005, p.48), cumpre o propósito de avaliar antes de cada iteração do laço se uma dada expressão é verdadeira. Caso o resultado seja verdadeiro, os códigos do seu corpo são reproduzidos, mas caso falso, os códigos do seu corpo não são reproduzidos.

Exemplo de algoritmo em Scratch que realiza uma repetição enquanto o número inserido for diferente do número aleatório determinado:

Figura 5 – Algoritmo para repetição mediante condição de guarda

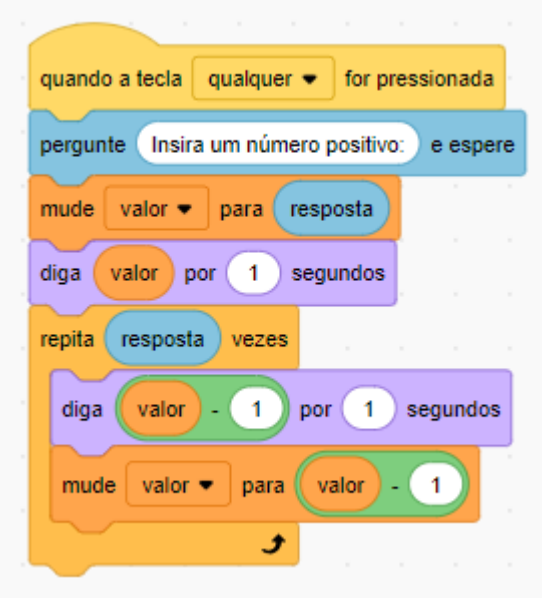


Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

Além da estrutura de repetição com condição de guarda, também há a repetição com uma variável que cumpra a função de efetuar uma contagem de acordo com parâmetros determinados. A repetição com contador atribui à variável uma quantidade específica que, por sua vez, faz o papel de controle da contagem, sendo que o laço é repetido de acordo com o número de vezes que essa variável está definida. Esta estrutura também é amplamente utilizada para percorrer elementos sequenciais como listas e conjuntos, assim como foi salientado por Lutz; Ascher (2007, p.177).

Exemplo de algoritmo em Scratch que faz uma contagem regressiva utilizando laço de repetição com contador:

Figura 6 – Algoritmo para repetição mediante contagem



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

É possível observar que, no caso indicado na Figura 6, a variável que determina a quantidade de vezes que este laço é repetido está sendo indicada através do respectivo valor de entrada, assim, consequentemente, quanto maior for o valor de entrada, mais vezes o laço será repetido e quanto menor for o valor de entrada, menos vezes o laço será repetido.

2.3.6 Listas

Segundo Puga; Rissetti (2009, p.84), as listas são estruturas que reúnem diversos elementos em apenas uma variável, e permitem realizar o acesso dos elementos através do índice que eles possuem dentro da lista, isto é, a posição ordenada em que determinado elemento está devidamente alocado na relação.

As listas podem assumir tamanhos arbitrários de acordo com o contexto onde estão sendo empregadas, todavia, reitera-se que a utilização de listas longas influi em maior esforço computacional.

Luz; Ascher (2007, p.117) reforçam ainda que são efetuados tratamentos de coleção de dados nas listas, permitindo atribuições como inserção, remoção, reatribuição e exclusão de elementos constituintes da sequência. Além disso, destaca-se que a manipulação de listas contendo diversos elementos diminui o esforço computacional em relação à movimentação de grande quantidade de variáveis isoladas.

Exemplo de algoritmo em Scratch que cria uma lista de compras de tamanho arbitrário e preenche as suas posições com itens que são inseridos sequencialmente.

Figura 7 – Algoritmo para criação e atribuição de elementos em uma lista de compras

Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

2.3.7 Funções

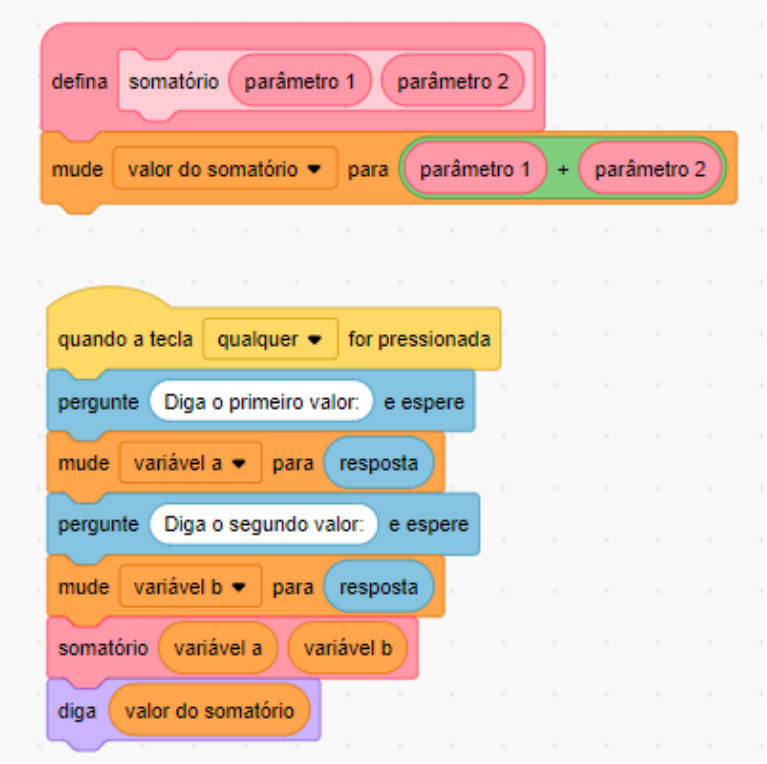
As funções constituem uma forma de melhorar o desempenho de códigos evitando retrabalho, o que ocorre, uma vez que elas permitem segmentar parte do algoritmo que é repetida em determinadas ocasiões, conforme a função é chamada. Assim como é defendido por Schildt (1996, p.138), o código fonte é percorrido e quando há uma chamada de função, o processamento desvia para essa função, executa os seus comandos, e, em seguida, torna a percorrer o código fonte.

De acordo com Forbellone; Eberspächer (2005, p.128), uma das grandes atribuições das funções é a de reduzir uma tarefa em subtarefas, e automatizar estas tarefas menores por meio de funções, para que seja possível alcançar a resolução da tarefa maior.

E as funções também permitem otimizar o espaço de armazenamento computacional volátil, pois o código principal fica menor e, consequentemente, o processamento do algoritmo se dá de modo mais rápido e eficiente, uma vez que as subtarefas ficam disponíveis e só sobem para a memória de chamadas quando solicitadas.

Exemplo de algoritmo em Scratch que contém um programa principal chamando a função somatório para efetuar a soma de dois valores e retorná-los para si:

Figura 8 – Algoritmo para a chamada de uma função separada



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

CAPÍTULO 3 – EXERCÍCIOS PROPOSTOS

3.1 Exercícios de Algoritmos em Blocos

Os exercícios de algoritmos propostos visam oferecer ao estudante visão crítica e reflexiva de como abordar os enunciados, induzindo na construção do raciocínio lógico, conforme fundamentado nos mecanismos apresentados no capítulo anterior, todavia, vale ressaltar, entretanto, que a resolução está pautada na visão e experiência individual. Noutros termos, os enunciados demandam o desenvolvimento de um raciocínio lógico que possa resolver o problema proposto, reitera-se, contudo, que as soluções podem ser múltiplas, não havendo uma única resposta correta.

Os exercícios desenvolvidos em momento pré-aula corroboram, assim como informam as entrevistas com os docentes *(APÊNDICE A),* para o contato e envolvimento prévio do estudante com o assunto abordado, estimulando-os ao treino.

Recomenda-se que a abordagem dos algoritmos seja realizada utilizando as operações apresentadas anteriormente, haja vista que há funções prontas em diversas linguagens de programação, a exemplo do próprio Scratch, que automatizam processos sem ter a necessidade de que o estudante desenvolva uma lógica para conceber a resolução do problema proposto. Sugere-se, além disso, que sejam feitos testes para averiguar a exatidão dos algoritmos construídos e, eventualmente, a correção de erros de lógica, visando o reconhecimento e correção dos códigos.

3.1.1 Exercício de operador relacional

Elabore um algoritmo relacional que receba a idade como entrada e verifique se ela está entre 0 a 19 anos e indique a saída como jovem, se entre 20 e 59 indique a saída como adulto e, se com 60 anos ou mais, indique a idade como idoso.

Exemplo de entrada: **30**
Exemplo de saída: **A sua faixa etária é adulto**

3.1.2 Exercício de operador lógico

Desenvolva um algoritmo lógico que receba dois números e analise a divisão entre eles, indicando se a divisão não existe, se ela resulta em zero, se resulta em negativo ou positivo.

Exemplo de entrada: **-15, 5**
Exemplo de saída: **A divisão resulta em negativo**

3.1.3 Exercício de condicional simples

Monte um algoritmo condicional simples que mensure se a quantidade de calorias consumida é ideal, menor do que a ideal ou maior do que a ideal. Como valor ideal, considere entre 2000 e 2400, incluindo esses valores.

Exemplo de entrada: **1800**
Exemplo de saída: **Você consome menos do que a quantidade ideal.**

3.1.4 Exercício de condicional composto

Construa um algoritmo condicional composto que receba dois números, e verifique se o primeiro número é múltiplo do segundo número.

Exemplo de entrada: **25, 7**
Exemplo de saída: **Os números não são múltiplos**

3.1.5 Exercício de laço de repetição

Apresente um algoritmo contendo um laço de repetição com condição de guarda que receba um número natural e faça a contagem de 0 até este respectivo número. Em seguida, faça adequações neste algoritmo para conter um laço de repetição com contador.

Exemplo de entrada: **5**
Exemplo de saída: **0, 1, 2, 3, 4, 5**

3.1.6 Exercício de lista

Indique um algoritmo que receba um número entre 1 e 10 e calcule até a sua tabuada de 10, indexando os seus elementos em uma lista.

Exemplo de entrada: **4**
Exemplo de saída: **[4, 8, 12, 16, 20, 24, 28, 32, 36, 40]**

3.1.7 Exercício de função

Forneça um algoritmo que receba um número positivo, e calcule a sua inversa, ou seja, 1/número positivo.

Exemplo de entrada: **4**
Exemplo de saída: **0,25**

3.2 Solução dos Exercícios de Algoritmos em Blocos

As soluções apresentadas constituem apenas sugestões de algoritmos resolutivos, havendo numerosas possibilidades para a construção de códigos que atendam ao desenvolvimento proposto pelos enunciados e que estão igualmente corretos para as respectivas respostas das problemáticas.

Neste sentido, evidencia-se que as resoluções sugeridas servem para nortear o estudante quanto à possibilidade de construção de algoritmos, auxiliando-o a averiguar possíveis alternativas lógicas que são semelhantes, ou não, àquelas esperadas para cada enunciado e que, naturalmente, também seriam válidas.

Se o leitor julgar pertinente, antes de efetuar a elaboração de um algoritmo, ele pode elaborar um esboço da lógica que está sendo mentalmente construída para a resolução dos exercícios através de fluxogramas, isto pode culminar em uma melhor visualização do encadeamento das ideias visando à obtenção da solução.

Salienta-se que os exercícios propostos possuem um grau de complexidade progressivo, deste modo, sugere-se que a resolução das questões seja feita na ordem em que está elencada neste capítulo.

Recomenda-se que as soluções sejam lidas pelo estudante, somente após o processo de tentativa de resolução dos enunciados, pois o intuito é que ele alcance as devidas soluções através da sua própria autonomia e pensamento crítico, fazendo com que as soluções sirvam como uma ferramenta de apoio, ou mesmo, de verificação da compreensão por parte deste estudante.

3.2.1 Solução do exercício de operador relacional

Figura 9 – Algoritmo para a solução do exercício de operador relacional

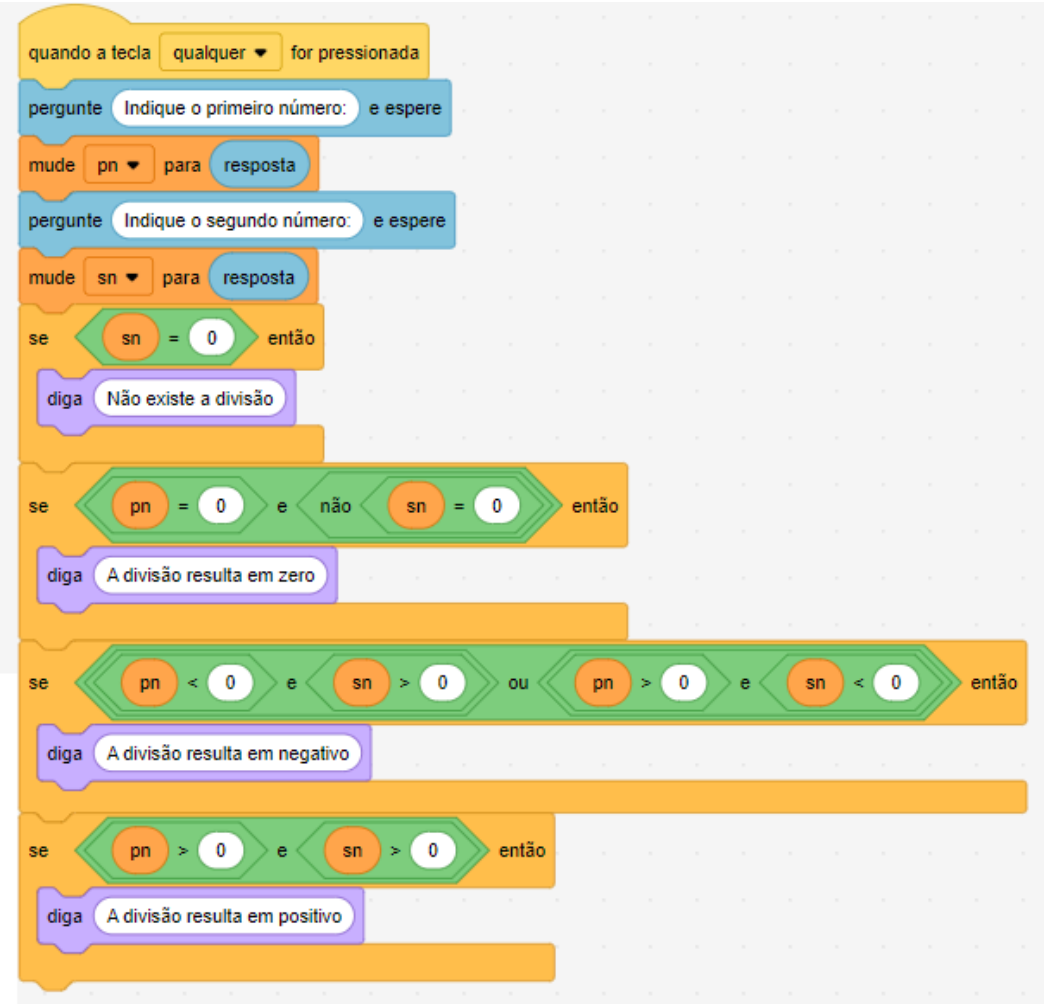
Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

No contexto desta questão, reitera-se que também é possível realizar comparações utilizando o operador relacional de mais de uma maneira. Assim, além de comparar através da operação "resposta < 20", também seria possível averiguar "resposta < = 19" e o mesmo se aplicaria com as demais comparações que estão sendo prontamente efetuadas neste algoritmo.

Além disso, é oportuno enfatizar que, em geral, quando se efetuam mais comparações ou etapas em um algoritmo, compreende-se que ele tende a ser menos eficiente. Neste exercício, quando se utiliza a comparação dupla "< =" em detrimento à comparação simples "<", entende-se que ela pode refletir em um algoritmo de eficiência não otimizada.

3.2.2 Solução do exercício de operador lógico

Figura 10 – Algoritmo para a solução do exercício de operador lógico

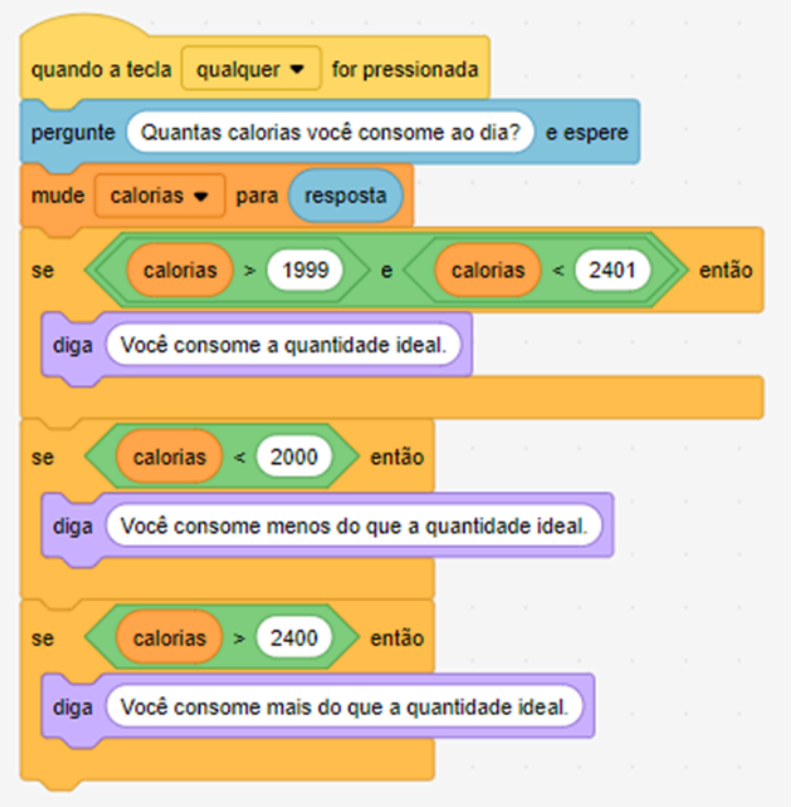


Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

Observe que a solução sugerida utiliza em seu algoritmo as variáveis “pn” para representar o primeiro número, e “sn” para indicar o segundo número, feito, por sua vez, para evitar extensão horizontal excessiva do código fonte. Entretanto, recomenda-se, sempre que possível, a utilização de variáveis com nomes autoexplicativos de acordo com as suas funcionalidades, a fim de que a leitura do algoritmo fique mais clara tanto para o desenvolvedor do código, quanto para um eventual analista.

3.2.3 Solução do exercício de condicional simples

Figura 11 – Algoritmo para a solução do exercício de condicional simples



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

Note que é possível simplificar os testes lógicos “igual a 2000” ou “maior do que 2000”, por apenas efetuar o teste “maior que 1999”. Isto se deve ao fato de que essa última forma de realizar o teste abrange tanto a possibilidade da variável assumir o valor 2000, quanto a possibilidade da mesma assumir um valor maior do que 2000. Tal técnica prejudica a leitura do algoritmo, mas beneficia a eficiência do mesmo, uma vez que uma quantidade menor de comandos será interpretada.

Evidencia-se que a ordem com que os elementos são comparados neste algoritmo não é essencial para que a saída seja correta, uma vez que fazer o teste: se a variável assume valor maior, menor ou dentro do intervalo, ou em outra ordem, pressupõe o mesmo resultado concebido.

3.2.4 Solução do exercício de condicional composto

Figura 12 – Algoritmo para a solução do exercício de condicional composto

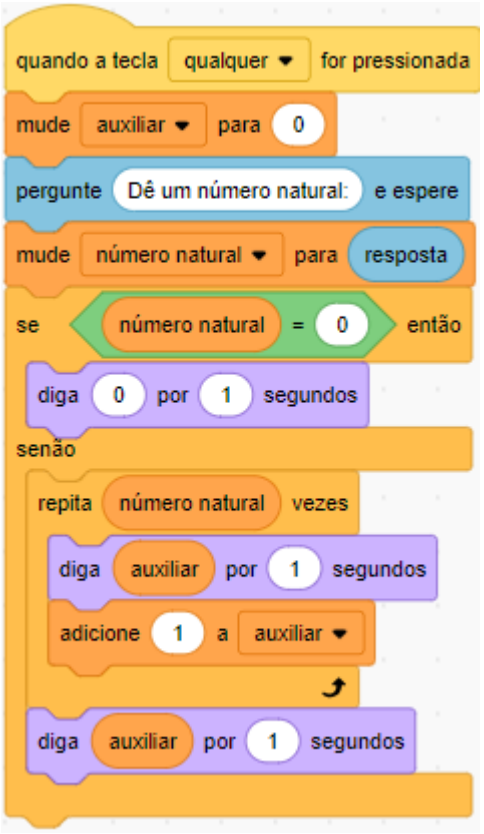
Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

É possível verificar que quando o condicional composto é utilizado no algoritmo, ele representa todos os demais casos possíveis que não são abrangidos pela condição previamente testada e que encerraria a execução do programa.

Deste modo, ao averiguar que o número não é igual ao múltiplo informado, e o resto da divisão do número pelo seu múltiplo não resulta em zero, mas sim em um outro número que é impossível de prever no momento da montagem do algoritmo, deve-se estabelecer uma ação a ser realizada quando o resultado estiver efetivamente fora do escopo das saídas que são esperadas pelo programa.

3.2.5 Solução do exercício de laço de repetição

Figura 13 – Algoritmo para a solução do exercício de laço de repetição com condição de guarda



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

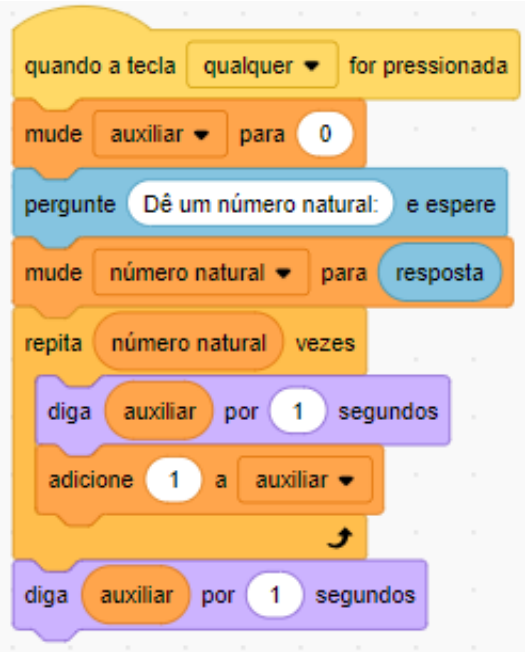
Deve-se analisar que a condição de guarda é importante para determinar o critério de parada do respectivo laço. Um erro comum de lógica que pode ser realizado durante a construção de algoritmos, no entanto, é atribuir à condição de guarda um dado valor que é impossível de ser alcançado, o que implica na condição de guarda não ser considerada falsa em momento algum, impossibilitando uma parada.

A condição de guarda que verifica se o número natural é zero indica o caso trivial do algoritmo, em que não seria necessário entrar no laço e a resposta imediata poderia ser retornada pelo código.

Desta modo, a referida condição de guarda ilustrada na Figura 13 pressupõe que se o número natural for igual a zero, deve-se imediatamente dizer o número zero, porém se este não for o caso, o laço começa a valer e a repetição até o número buscado passa então a ser realizada.

A Figura 14 disposta a seguir indica que este mesmo algoritmo pode ser ajustado para empregar o laço com contador de repetição, e neste momento, a condição de guarda não está mais presente, não sendo mais um fator influenciador da parada da repetição do laço, mas sim o fato de atingir ou não a quantidade da variável que faz o controle da respectiva contagem no algoritmo.

Figura 14 – Algoritmo para a solução do exercício de laço de repetição com contador



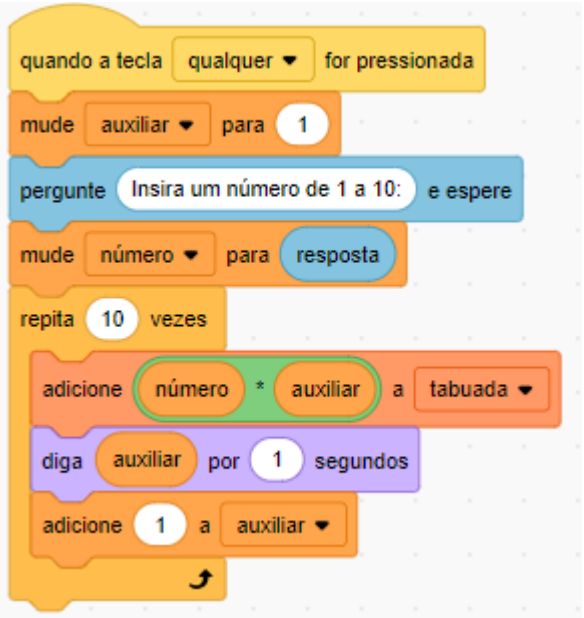
Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

Embora ambas as soluções sejam adequadas para a proposta deste exercício, compreende-se que quando há a condição de guarda o algoritmo se torna mais eficiente, isto se deve pelo fato da condição de guarda ser capaz de identificar o caso mais simples possível, e já de imediato retornar uma resposta sem precisar processar as etapas constituintes do laço de contagem.

Por outro lado, o algoritmo com o laço de repetição com contador permite a chegada do número independentemente de se tratar ou não de um caso trivial, mas devido à não capacidade de distingui-lo, o laço sempre será executado, o que poderia refletir em uma menor eficiência do algoritmo.

3.2.6 Solução do exercício de lista

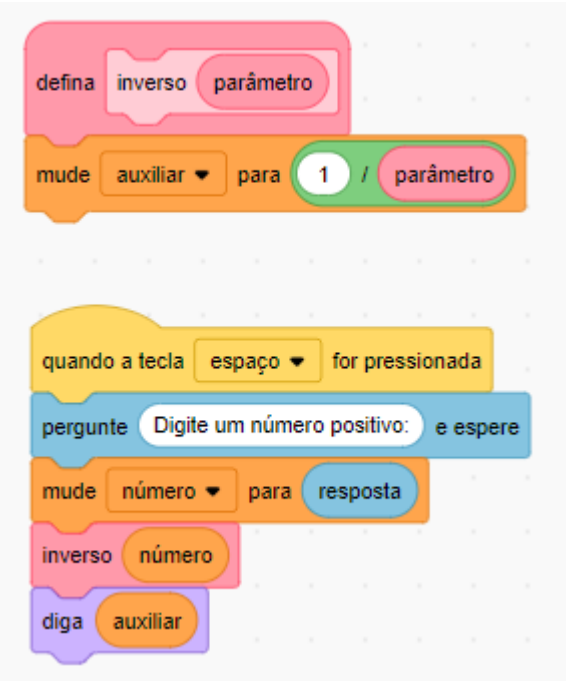
Figura 15 – Algoritmo para a solução do exercício de lista



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

3.2.7 Solução do exercício de função

Figura 16 – Algoritmo para a solução do exercício de função



Fonte: Elaborador pelo autor utilizando a ferramenta Scratch

CONSIDERAÇÕES FINAIS

Os estudantes ingressantes em curso de Tecnologia de Informação geralmente têm contato com a disciplina de Algoritmos e Lógica de Programação quando ainda estão no processo de familiarização com o pensamento computacional. Aliado a isso, uma base insuficiente de matemática provinda da educação básica, pode fazer com que estes estudantes enfrentam dificuldades no desenvolvimento do raciocínio lógico tanto no contexto da FATEC-SP quanto de outras instituições de ensino superior.

A opção por abordar o desafio de que a lógica de programação, via de regra, representa para os alunos ingressantes dos cursos superiores de Tecnologia da Informação foi decorrente da dificuldade percebida pelo autor durante as suas próprias aulas de Algoritmos e Lógica de Programação, enquanto estudante do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da FATEC-SP.

Essa mesma dificuldade foi endossada por meio das entrevistas realizadas com os docentes da FATEC-SP, que afirmaram ter identificado dificuldade de raciocínio lógico-matemático por parte de seus alunos durante as aulas, resultando, em casos extremos, até mesmo no trancamento da disciplina. No que se refere aos estudantes, o questionário aplicado mostrou que boa parte deles reconhece a dificuldade em lidar com Algoritmos e Lógica de Programação no contexto de suas graduações.

A partir deste contexto, o presente trabalho buscou lidar com a temática do estudo de Algoritmos e Lógica de Programação, utilizando abordagens provindas da metodologia de Sala de Aula Invertida, em que se procurou sistematizar os recursos comumente utilizados na lógica de programação de modo visual e interativo, instruindo o aluno ingressante numa jornada autodidata e autônoma em momento pré-aula, contribuindo, assim, para o seu protagonismo e desenvolvimento contínuos.

Com base nas definições conceituais e exemplificação dos elementos empregados na construção de algoritmos, foram propostos exercícios a fim de estimular o raciocínio lógico do aluno no que se refere ao reconhecimento e aplicação dos recursos básicos de Algoritmos e Lógica de Programação. Em seguida, a solução destes exercícios foi demonstrada para que esse mesmo aluno possa, efetivamente, comparar a solução desenvolvida por ele com a solução proposta pelo autor do trabalho.

Desta maneira, esta monografia buscou contribuir para o aprendizado de lógica de programação por parte de alunos ingressantes dos cursos superiores na área de Tecnologia da Informação, elucidando que embora possam surgir dificuldades no estudo do pensamento

computacional, com apoio de metodologias de ensino e aprendizagem eficientes, pode-se superar desafios e corroborar para o sucesso do referido aluno. Assim como reforçam os dados do questionário aplicado e das entrevistas realizadas, a metodologia da Sala de Aula Invertida pode apresentar um ganho para o processo de aprendizado de Algoritmos e Lógica de Programação.

A presente monografia apresentou uma forma de estudo e aprendizado básico de Algoritmos e Lógica de Programação com a metodologia de Sala de Aula Invertida, recomendando-a para futuros trabalhos que abordem, dentre outros, o ensino e o aprendizado de Estrutura de Dados, da Análise de Algoritmos e, até mesmo, da Programação Orientada a Objetos.

REFERÊNCIAS BIBLIOGRÁFICAS

ARCHIMEDES, F. N. **Estudo sobre a evasão nos cursos de graduação na área de tecnologia da informação.** São Paulo, 2020. 89 f. Dissertação (Ensino de Ciências e Matemática) – Universidade Cruzeiro do Sul. Disponível em: < https://repositorio.up.edu.br/jspui/bitstream/123456789/2301/1/ARCHIMEDES%20FERRARI%20NETO.pdf > Acesso em: 31 ago. 2023.

BLISSARI, I. R. **Análise dos possíveis fatores da evasão no curso de tecnologia da informação e comunicação.** Araranguá, 2018. 66 f. Monografia (Tecnologia da Informação e Comunicação) – Universidade Federal de Santa Catarina. Disponível em: < https://repositorio.ufsc.br/bitstream/handle/123456789/191929/TCC%20-%20ISABELA%20ROSSO%20BLISSARI%20.pdf?sequence=1 > Acesso em: 31 ago. 2023.

BRIGHENTI, J; BIAVATTI, V. T; SOUZA, T. R. de. Metodologias de ensino-aprendizagem: uma abordagem sob a percepção dos alunos. **Revista Gestão Universitária na América Latina-GUAL,** Florianópolis, v. 8, n. 3, p. 281-304, 2015. Disponível em: < https://www.redalyc.org/pdf/3193/319342694014.pdf > Acesso em: 31 ago. 2023.

FORBELLONE, A. L. V; EBERSPÄCHER, H. F. **Lógica de programação: a construção de algoritmos e estruturas de dados.** 3. ed. São Paulo: Pearson Prentice Hall, 2005. 217 p.

JUNIOR, E. C. Linguagem de blocos: uma proposta construcionista no processo de ensino e aprendizagem de algoritmo. **ETIC-Encontro de Iniciação Científica,** São Paulo, v. 13, n. 13, 2017. Disponível em: < http://intertemas.toledoprudente.edu.br/index.php/ETIC/article/download/6124/5826 > Acesso em: 31 ago. 2023.

LOVATO, F. L; MICHELOTTI, A.; SILVA, C. B. da; LORETO, E. L. da S. Metodologias ativas de aprendizagem: uma breve revisão. **Acta Scientiae,** Canoas, v. 20, n. 2, 2018. Disponível em: < http://www.periodicos.ulbra.br/index.php/acta/article/viewFile/3690/2967 > Acesso em: 31 ago. 2023.

LUTZ, M; ASCHER, D. **Aprendendo python.** Trad. João Tortello. 2. ed. Porto Alegre: Bookman, 2007. 568 p.

MAGALHÃES, M. S. et al. Sala de aula invertida: o que é e quais os benefícios para a educação atual?. **Revista Ilustração,** Cruz Alta, v. 4, n. 2, p. 15-22, 2023. Disponível em: < https://journal.editorailustracao.com.br/index.php/ilustracao/article/download/149/94 > Acesso em: 31 ago. 2023.

MATHIAS, I. M. **Algoritmos e programação.** Ponta Grossa: UEPG/NUTEAD, 2017. 175 p.

MORÁN, J. Mudando a educação com metodologias ativas. **Coleção mídias contemporâneas. Convergências midiáticas, educação e cidadania: aproximações jovens**, Ponta Grossa, v. 2, n. 1, p. 15-33, 2015. Disponível em: < https://edisciplinas.usp.br/pluginfile.php/4941832/mod_resource/content/1/Artigo-Moran.pdf > Acesso em: 31 ago. 2023.

NASCIMENTO, J. A. do. **Explorando a lógica matemática no ensino básico.** Natal, 2016. 182 f. Dissertação (Matemática) – Centro de Ciências Exatas e da Terra, Universidade Federal do Rio Grande do Norte. Disponível em: < https://repositorio.ufrn.br/bitstream/123456789/21925/1/JeffersonAlexandreDoNascimento_DISSERT.pdf > Acesso em: 31 ago. 2023.

NÉRICE, I. G. **Didática geral dinâmica.** 10. ed. São Paulo: Atlas, 1987.

ORSANO, R; ROBERTO, D. **Uma abordagem do ensino de lógica matemática no ensino Médio.** São Luís, 2018. 81 f. Dissertação (Matemática) – Universidade Federal do Maranhão. Disponível em: < https://tedebc.ufma.br/jspui/bitstream/tede/2553/2/DiegoOrsano.pdf > Acesso em: 31 ago. 2023.

PUGA, S; RISSETTI, G. **Lógica de programação e estruturas de dados, com aplicações em java.** 2. ed. São Paulo: Pearson Prentice Hall, 2009. 281 p.

RODRIGUES, N. C; CORREIA, D. A sala de aula invertida no ensino de ciências e matemática: uma revisão sistemática. **Revista de Ensino de Ciências e Matemática,** São Paulo, v. 14, n. 3, p. 1-22, 2023. Disponível em: < https://revistapos.cruzeirodosul.edu.br/index.php/rencima/article/download/3858/2068 > Acesso em: 31 ago. 2023.

RODRIGUES, R. K. **Estudo do uso da linguagem de blocos scratch no ensino do pensamento computacional.** Curitiba, 2019. 61 f. Monografia (Ciência da Computação) –Universidade Tecnológica Federal do Paraná. Disponível em: < https://riut.utfpr.edu.br/jspui/bitstream/1/6014/1/linguagemblocospensamentocomputacional.pdf > Acesso em: 31 ago. 2023.

SANTOS, J. A; SANTOS, L. S. B. dos; KLEBER, V. F. **Dificuldades na aprendizagem de matemática.** São Paulo, 2007. 41 f. Monografia (Matemática) – Centro Universitário Adventista de São Paulo. Disponível em: < http://www.educadores.diaadia.pr.gov.br/arquivos/File/2010/artigos_teses/MATEMATICA/Monografia_Santos.pdf > Acesso em: 31 ago. 2023.

SCHILDT, H. **C completo e total.** Trad. Roberto Carlos Mayer. 3. ed. São Paulo: Makron Books, 1996. 848 p.

SCHNEIDERS, L. A. O método da sala de aula invertida (flipped classroom). **Lajeado: ed. da UNIVATES.** Lajeados, 2018. Disponível em: < https://grupos.moodle.ufsc.br/pluginfile.php/855093/mod_data/content/4259/Sala%20de%20aula%20invertida%202.pdf > Acesso em: 31 ago. 2023.

SEMESP – Secretaria de Modalidades Especializadas de Educação. 14º Mapa do ensino superior no Brasil. São Paulo, 2024. Disponível em: < https://www.semesp.org.br/mapa/edicao-14/download/ > Acesso em: 12 maio 2024.

SOUSA, L. de L; FARIAS, E. J; CARVALHO, W. V. de. Programação em blocos aplicada ao ensino do pensamento computacional: Um mapeamento sistemático. In: **Anais do XXXI Simpósio Brasileiro de Informática na Educação.** SBC, Manaus, 2020. p. 1513-1522. Disponível em: < https://sol.sbc.org.br/index.php/sbie/article/download/12907/12761 > Acesso em: 31 ago. 2023.

SOUZA, F. A. de; FALCÃO, T. P; MELLO, R. F; Avaliação heurística de ferramentas de programação em blocos. In: **Anais Estendidos do I Simpósio Brasileiro de Educação em Computação.** SBC, Manaus, 2021. p. 11-11. Disponível em: < https://sol.sbc.org.br/index.php/educomp_estendido/article/download/14850/14695 > Acesso em: 31 ago. 2023.

APÊNDICE A – TRANSCRIÇÃO DE ENTREVISTAS COM DOCENTES

A seguir são transcritas as entrevistas[2] realizadas com os docentes que lecionam na área de algoritmos, lógica de programação e linguagem de programação do Departamento de Tecnologia da Informação da Faculdade de Tecnologia de São Paulo.

Transcrição da entrevista com a Profa. Esp. Elisabete da S. Santos:

1) Os alunos do curso de ADS, apresentam, de modo geral, deficiência em raciocínio lógico provinda do ensino básico? Por quê? **Essa proporção é meio a meio. Há alunos que têm uma boa base de raciocínio lógico, mas há alunos que apresentam um pouco de dificuldade, então sempre damos exemplos de coisas da vida real para que eles consigam acompanhar o raciocínio nas aulas.**

2) Você acredita que um ensino básico com maior rigor em matemática, deixaria o aluno melhor preparado para aprender Algoritmos e Lógica de Programação? Por quê? **Com certeza. A matemática desenvolve o raciocínio lógico do aluno, e isso é fundamental na linguagem de programação. E não somente o conhecimento em matemática, mas também o domínio de outras disciplinas como a língua portuguesa, no caso da construção de *Webpages*, e de inglês, que é um facilitador para o aprendizado de programação.**

3) Em sua disciplina, ao longo dos semestres, você reconhece evasão após as avaliações iniciais? Por quê? **Há pouquíssima evasão na minha disciplina, porque a programação para a web é um assunto que os alunos se interessam muito, então eles acabam superando quaisquer dificuldades.**

4) Você observa evasão da sua disciplina por parte dos alunos em decorrência de não estarem preparados para aprender determinados assuntos? Por quê? **Verifico uma dificuldade dos alunos quando estamos abordando programação em si como Javascript, Php. Em que embora haja estruturas lógicas que são comuns às outras linguagens, há comandos particulares que os alunos aprendem na prática. Assim, eles têm uma base teórica, e quando veem funcionando é que conseguem fazer a junção da teoria com a prática.**

5) Você conhece a metodologia da Sala de Aula Invertida? **Sim, eu conheço a metodologia da Sala de Aula Invertida. Nesta metodologia, o aluno é protagonista na aula. Na minha disciplina,**

[2] As transcrições foram revisadas pelos docentes entrevistados que autorizaram a sua reprodução na presente monografia conforme o APÊNDICE B - AUTORIZAÇÕES DE REPRODUÇÃO DE ENTREVISTAS.

busco sempre a participação dos alunos, assim a aula fica mais interessante. Especialmente porque a parte de web muda bastante, e assim eu posso aprender com os alunos também.

6) A metodologia de sala de aula invertida poderia refletir em uma facilidade de aprendizado por parte dos alunos? Por quê? **Certamente. É muito importante estimular a participação dos alunos através da Sala de Aula Invertida, porque assim eles também têm a responsabilidade pelo desenvolvimento de um aprendizado mais dinâmico. E principalmente, quando o aluno faz um projeto com carinho e esmero e apresenta para os outros colegas, ele acaba repassando os conhecimentos e as técnicas utilizadas para os demais, e isto torna a aula mais proveitosa.**

Transcrição da entrevista com a Profa. Dra. Maria C. de .M. Fabiani:

1) Os alunos ingressantes do curso de ADS, apresentam, de modo geral, deficiência em raciocínio lógico provinda do ensino básico? Por quê? **Na minha disciplina de programação em microinformática não sinto esse problema. Tratando-se de uma disciplina complementar a programação, eu não exijo muito o raciocínio lógico. Normalmente, a ideia da lógica a ser desenvolvida é fornecida e os alunos seguem essa ideia.**

2) Você acredita que um ensino básico com maior rigor em matemática, deixaria o aluno melhor preparado para aprender Algoritmos e Lógica de Programação? Por quê? **Para a nossa área de Tecnologia da Informação, qualquer conhecimento a mais de matemática seria bom, porque isto desenvolve o raciocínio de maneira geral.**

3) Em sua disciplina, ao longo dos semestres, você reconhece evasão após as avaliações iniciais? Por quê? **Sim, embora essa evasão se dê em menor proporção nas turmas da manhã e da noite, há uma evasão maior nas turmas da tarde. Porém, não se pode afirmar que a evasão seja decorrente do mau desempenho nas avaliações.**

4) Você observa evasão da sua disciplina por parte dos alunos em decorrência de não estarem preparados para aprender determinados assuntos? Por quê? **Eu não observo evasão devido a eles estarem ou não preparados para aprender determinados assuntos. E também não encontro problemas de conhecimento básico de matemática por parte dos alunos de ADS, mas observo este problema básico de matemática com alunos de outros cursos.**

5) Você conhece a metodologia da Sala de Aula Invertida? **Eu já ouvi palestras que abordavam a metodologia da Sala de Aula Invertida, portanto eu a conheço de modo geral, mas não com muitos detalhes. Sei que esta metodologia propõe um problema e prepara material para os alunos estudarem e tentarem resolver.**

6) A metodologia de sala de aula invertida poderia refletir em uma facilidade de aprendizado por parte dos alunos? Por quê? **Dependendo do conteúdo, sim. Porém há conteúdos em que esta abordagem pode não ser muito apropriada, pois o aluno poderia se frustrar em não conseguir ler o material e resolver o problema proposto. Usualmente, propomos o exercício baseado no que foi ensinado na aula.**

Transcrição da entrevista com o Prof. Me. Sergio L. Banin:

1) Os alunos ingressantes do curso de ADS, apresentam, de modo geral, deficiência em raciocínio lógico provinda do ensino básico? Por quê? **É misturado, há alunos que fizeram curso técnico e já sabem programar bem, mas há alunos que têm dificuldade e até precisariam de um apoio de monitoria.**

2) Você acredita que um ensino básico com maior rigor em matemática, deixaria o aluno melhor preparado para aprender Algoritmos e Lógica de Programação? Por quê? **Acredito que sim, porque o raciocínio matemático ajuda bastante no desenvolvimento de algoritmos.**

3) Em sua disciplina, ao longo dos semestres, você reconhece evasão após as avaliações iniciais? Por quê? **Sim, reconheço que há uma certa evasão após as primeiras avaliações.**

4) Você observa evasão da sua disciplina por parte dos alunos em decorrência de não estarem preparados para aprender determinados assuntos? Por quê? **Pode-se observar uma evasão, mas não é possível dizer se é decorrência de estarem ou não preparados, porque a evasão também ocorre devido a motivos externos, como problemas familiares, financeiros.**

5) Você conhece a metodologia da Sala de Aula Invertida? **Sim, conheço a metodologia da Sala de Aula Invertida e uso nas aulas.**

6) A metodologia de sala de aula invertida poderia refletir em uma facilidade de aprendizado por parte dos alunos? Por quê? **Certamente. Eu já uso a metodologia da Sala de Aula Invertida há sete anos e percebo que ajuda muito no aprendizado dos alunos. Por exemplo, eu desenvolvi um site em que os alunos acessam, leem a tarefa, desenvolvem e entregam até o momento anterior à aula valendo nota. Isto incentiva os alunos a estudarem o assunto antes da aula. E, posteriormente, quando iniciamos a aula, abordamos essa tarefa, assim, os alunos que fizeram a entrega comentam dizendo as estratégias que usaram para resolver e eventuais dificuldades que tiveram.**

Transcrição da entrevista com o Prof. Dr. Silvio do L. Pereira:

1) Os alunos do curso de ADS, apresentam, de modo geral, deficiência em raciocínio lógico provinda do ensino básico? Por quê? **Sim, geralmente, os alunos apresentam deficiência em raciocínio lógico. Muitos alunos têm dificuldade até mesmo com relação a conceitos simples, como o uso de conectivos lógicos "e" e "ou". Não há como saber se esses conteúdos não foram devidamente abordados no ensino básico ou se foram abordados, mas os alunos não conseguiram absorvê-los.**

2) Você acredita que um ensino básico com maior rigor em matemática, deixaria o aluno melhor preparado para aprender Algoritmos e Lógica de Programação? Por quê? **Com certeza, um ensino básico com maior rigor matemático prepararia melhor o aluno para aprender Algoritmos e Lógica de Programação. Isso porque a maior parte dos algoritmos, especialmente aqueles que são dados para programadores iniciantes, envolve cálculos. Então, se o aluno não tem um bom embasamento em lógica e matemática, ele terá maior dificuldade em um curso da área de computação pois, afinal de contas, a computação é um ramo aplicado da matemática.**

3) Em suas disciplinas, ao longo dos semestres, você reconhece evasão após as avaliações iniciais? Por quê? **Sim, certamente, há evasão após as avaliações iniciais das disciplinas que eu leciono. Observo que os alunos conseguem entender os conceitos e o raciocínio, quando eu explico em sala de aula como desenvolver os algoritmos para resolver os problemas propostos. No entanto, tenho a impressão que, justamente por entenderem com relativa facilidade as explicações dadas em aula, os alunos acabam subestimando a dificuldade dos conteúdos e não se preparam adequadamente para realizar as avaliações. A maioria dos alunos não faz os exercícios propostos (para serem feitos fora da aula), acreditando que serão capazes de resolvê-los durante a prova, e acabam não conseguindo tirar boas notas. Assim, acredito que a evasão não seja devida à dificuldade dos conteúdos abordados nas aulas, mas sim à falta de dedicação dos alunos ao estudo extraclasse.**

4) Você observa evasão da sua disciplina por parte dos alunos em decorrência de não estarem preparados para aprender determinados assuntos? Por quê? **Observo essa evasão também. Normalmente, eu explico de modo detalhado o conteúdo que é abordado, e eu percebo que a dificuldade do aluno está, muitas vezes, relacionada a outras disciplinas que ele passou sem ter base boa, especialmente na matéria de Estrutura de Dados, em que parte dos alunos desiste porque não sabe programar em Linguagem C.**

5) Você conhece a metodologia da Sala de Aula Invertida? **Sim, eu conheço essa metodologia.**

6) A metodologia de sala de aula invertida poderia refletir em uma facilidade de aprendizado por parte dos alunos? Por quê? **A metodologia da Sala de Aula Invertida tem os seus méritos, ela funciona bem para disciplinas que envolvem leitura, discussão e debate. Entretanto, essa metodologia pode não ser a mais eficaz nas disciplinas de algoritmos que ensino que não tratam de opinião, discussão e leitura, e cujos conteúdos não são tão simples de serem lidos e entendidos por parte de leigos. Para a metodologia da sala de aula invertida funcionar nas minhas disciplinas, precisaríamos de mais aulas (i.e., uma carga horária semestral maior) para que se pudesse introduzir um conteúdo mais básico, pedir para o aluno ler algo mais avançado, e abordar esse conteúdo novamente na aula seguinte. Com a carga horária atual, se eu adotasse tal metodologia, não conseguiria abordar todos os conteúdos previstos nas ementas das disciplinas.**

APÊNDICE B – AUTORIZAÇÕES DE REPRODUÇÃO DE ENTREVISTAS

FACULDADE DE TECNOLOGIA DE SÃO PAULO

TERMO DE AUTORIZAÇÃO DE REPRODUÇÃO DE ENTREVISTA

Eu, **Prof.ª Esp. Elisabete da S. Santos**, autorizo a reprodução da transcrição da entrevista que concedi ao aluno **Adilson de J. C. Oliveira** matriculado no 6° semestre do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas no ano de 2024 para ser utilizada no trabalho de conclusão de curso com o tema **"Proposta de Aprendizado Básico de Lógica de Programação com Sala de Aula Invertida para Ingressantes nos Cursos Superiores da Área de Tecnologia da Informação"**, sob a orientação da **Prof.ª Me. Simone C. G. Vianna**.

Eu declaro que estou de acordo com o descrito acima e assino esta autorização.

São Paulo, 15 de março de 2024.

Prof.ª Esp. Elisabete da S. Santos
Faculdade de Tecnologia de São Paulo

FACULDADE DE TECNOLOGIA DE SÃO PAULO

TERMO DE AUTORIZAÇÃO DE REPRODUÇÃO DE ENTREVISTA

Eu, **Prof.ª Dra. Maria C. de M. Fabiani**, autorizo a reprodução da transcrição da entrevista que concedi ao aluno **Adilson de J. C. Oliveira** matriculado no 6° semestre do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas no ano de 2024 para ser utilizada no trabalho de conclusão de curso com o tema **"Proposta de Aprendizado Básico de Lógica de Programação com Sala de Aula Invertida para Ingressantes nos Cursos Superiores da Área de Tecnologia da Informação"**, sob a orientação da **Prof.ª Me. Simone C. G. Vianna**.

Eu declaro que estou de acordo com o descrito acima e assino esta autorização.

São Paulo, 12 de março de 2024.

MC Fabiani

Prof.ª Dra. Maria C. de M. Fabiani
Faculdade de Tecnologia de São Paulo

FACULDADE DE TECNOLOGIA DE SÃO PAULO

TERMO DE AUTORIZAÇÃO DE REPRODUÇÃO DE ENTREVISTA

Eu, **Prof. Me. Sergio L. Banin**, autorizo a reprodução da transcrição da entrevista que concedi ao aluno **Adilson de J. C. Oliveira** matriculado no 6° semestre do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas no ano de 2024 para ser utilizada no trabalho de conclusão de curso com o tema **"Proposta de Aprendizado Básico de Lógica de Programação com Sala de Aula Invertida para Ingressantes nos Cursos Superiores da Área de Tecnologia da Informação"**, sob a orientação da **Prof.ª Me. Simone C. G. Vianna**.

Eu declaro que estou de acordo com o descrito acima e assino esta autorização.

São Paulo, 21 de março de 2024.

Prof. Me. Sergio L. Banin

Faculdade de Tecnologia de São Paulo

FACULDADE DE TECNOLOGIA DE SÃO PAULO

TERMO DE AUTORIZAÇÃO DE REPRODUÇÃO DE ENTREVISTA

Eu, **Prof. Dr. Silvio do L. Pereira**, autorizo a reprodução da transcrição da entrevista que concedi ao aluno **Adilson de J. C. Oliveira** matriculado no 6° semestre do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas no ano de 2024 para ser utilizada no trabalho de conclusão de curso com o tema **"Proposta de Aprendizado Básico de Lógica de Programação com Sala de Aula Invertida para Ingressantes nos Cursos Superiores da Área de Tecnologia da Informação"**, sob a orientação da **Prof.ª Me. Simone C. G. Vianna**.

Eu declaro que estou de acordo com o descrito acima e assino esta autorização.

São Paulo, 12 de março de 2024.

Prof. Dr. Silvio do L. Pereira
Faculdade de Tecnologia de São Paulo

APÊNDICE C – QUESTIONÁRIO COM ALUNOS

O questionário a seguir foi distribuído aos alunos da graduação em Análise e Desenvolvimento de Sistemas da Faculdade de Tecnologia de São Paulo de 21/09/2023 a 8/10/2023.

1) Com que frequência você sente dificuldade nas disciplinas que exigem a elaboração de algoritmos?
( ) Muito frequentemente
( ) Frequentemente
( ) Ocasionalmente
( ) Raramente
( ) Nunca

2) Na sua visão, qual a importância do raciocínio lógico-matemático para disciplinas que envolvem Algoritmos e Lógica de Programação?
( ) Muito importante
( ) Importante
( ) Moderado
( ) Às vezes é importante
( ) Não é importante

3) Com que frequência você trancou uma disciplina que envolvia algoritmos por não conseguir acompanhar o ritmo das aulas?
( ) Muito frequentemente
( ) Frequentemente
( ) Ocasionalmente
( ) Raramente
( ) Nunca

4) É verdade que você já esteve propenso a trancar uma disciplina que envolva algoritmos por não ir bem nas avaliações inicias?
( ) Quase sempre verdade
( ) Geralmente verdade

( ) Às vezes verdadeiro
( ) Geralmente falso
( ) Quase sempre falso

5) Quando você se prepara antes para um determinado conteúdo, você tem um aproveitamento maior na aula?
( ) Concordo totalmente
( ) Concordo
( ) Não estou decidido
( ) Discordo
( ) Discordo totalmente

6) Com que frequência você ouviu falar sobre a metodologia da Sala de Aula Invertida?
( ) Muito frequentemente
( ) Frequentemente
( ) Ocasionalmente
( ) Raramente
( ) Nunca

7) A metodologia da Sala de Aula Invertida poderia ajudar você a cursar Algoritmos e Lógica de Programação?
( ) Concordo totalmente
( ) Concordo
( ) Não estou decidido
( ) Discordo
( ) Discordo totalmente

8) Você gostaria de adicionar comentários extras? (Opcional)

_____
_____
_____
_____

APÊNDICE D – RESPOSTAS AO QUESTIONÁRIO COM ALUNOS

Data	1) Com que frequência você sente dificuldade nas disciplinas que exigem a elaboração de algoritmos?	2) Na sua visão, qual a importância do raciocínio lógico-matemático para disciplinas que envolvem algoritmos e lógica de	3) Com que frequência você trancou uma disciplina que envolvia algoritmos por não conseguir acompanhar o ritmo	4) É verdade que você já esteve propenso a trancar uma disciplina que envolva algoritmos por não ir bem nas avaliações iniciais?	5) Quando você se prepara antes para um determinado conteúdo, você tem um aproveitamento maior na aula?	6) Com que frequência você ouviu falar sobre a metodologia da Sala de Aula Invertida?	7) A metodologia da Sala de Aula Invertida poderia ajudar você a cursar algoritmos e lógica de programação?
21/9	Muito frequentemente	Muito importante	Nunca	Às vezes verdadeiro	Concordo totalmente	Frequentemente	Concordo totalmente
21/9	Muito frequentemente	Muito importante	Muito frequentemente	Às vezes verdadeiro	Concordo totalmente	Nunca	Não estou decidido
21/9	Frequentemente	Às vezes é importante	Nunca	Geralmente falso	Concordo	Nunca	Concordo totalmente
21/9	Ocasionalmente	Muito importante	Nunca	Geralmente falso	Concordo totalmente	Frequentemente	Discordo
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo	Muito frequentemente	Não estou decidido
21/9	Raramente	Moderado	Nunca	Quase sempre falso	Concordo	Muito frequentemente	Concordo
21/9	Ocasionalmente	Muito importante	Nunca	Geralmente falso	Concordo	Muito frequentemente	Não estou decidido
21/9	Muito frequentemente	Muito importante	Nunca	Às vezes verdadeiro	Não estou decidido	Nunca	Não estou decidido
21/9	Frequentemente	Importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Muito frequentemente	Muito importante	Ocasionalmente	Às vezes verdadeiro	Concordo	Ocasionalmente	Não estou decidido
21/9	Ocasionalmente	Importante	Raramente	Geralmente verdade	Concordo totalmente	Ocasionalmente	Não estou decidido
21/9	Raramente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Muito frequentemente	Concordo totalmente
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Ocasionalmente	Importante	Nunca	Quase sempre falso	Concordo totalmente	Raramente	Concordo
21/9	Raramente	Moderado	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Raramente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Frequentemente	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Não estou decidido	Nunca	Não estou decidido
21/9	Frequentemente	Importante	Raramente	Geralmente verdade	Concordo	Nunca	Não estou decidido
21/9	Frequentemente	Moderado	Nunca	Quase sempre falso	Concordo	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Ocasionalmente	Geralmente verdade	Concordo	Nunca	Não estou decidido
21/9	Frequentemente	Importante	Muito frequentemente	Geralmente verdade	Não estou decidido	Raramente	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Geralmente falso	Concordo totalmente	Raramente	Não estou decidido
21/9	Frequentemente	Importante	Frequentemente	Geralmente verdade	Concordo totalmente	Ocasionalmente	Concordo
21/9	Raramente	Muito importante	Nunca	Geralmente falso	Concordo totalmente	Raramente	Concordo
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Raramente	Concordo totalmente
21/9	Frequentemente	Importante	Nunca	Geralmente verdade	Concordo	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Não estou decidido	Frequentemente	Não estou decidido
21/9	Ocasionalmente	Muito importante	Raramente	Às vezes verdadeiro	Concordo totalmente	Ocasionalmente	Não estou decidido
21/9	Frequentemente	Moderado	Nunca	Quase sempre falso	Concordo	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Ocasionalmente	Concordo totalmente
21/9	Ocasionalmente	Muito importante	Nunca	Geralmente falso	Concordo	Raramente	Não estou decidido
21/9	Raramente	Importante	Nunca	Quase sempre falso	Concordo	Ocasionalmente	Não estou decidido
21/9	Raramente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Ocasionalmente	Concordo totalmente
21/9	Frequentemente	Muito importante	Nunca	Geralmente falso	Concordo	Frequentemente	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Raramente	Muito importante	Nunca	Quase sempre falso	Discordo	Nunca	Não estou decidido
21/9	Muito frequentemente	Muito importante	Nunca	Geralmente falso	Discordo	Raramente	Concordo totalmente
21/9	Muito frequentemente	Importante	Nunca	Às vezes verdadeiro	Não estou decidido	Nunca	Discordo
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
21/9	Muito frequentemente	Muito importante	Muito frequentemente	Geralmente verdade	Concordo totalmente	Nunca	Concordo totalmente
21/9	Frequentemente	Importante	Nunca	Às vezes verdadeiro	Concordo	Ocasionalmente	Não estou decidido
21/9	Raramente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Frequentemente	Concordo
21/9	Frequentemente	Moderado	Nunca	Quase sempre falso	Concordo	Raramente	Concordo
21/9	Frequentemente	Importante	Nunca	Quase sempre falso	Concordo	Raramente	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Quase sempre falso	Não estou decidido	Nunca	Não estou decidido
21/9	Ocasionalmente	Muito importante	Nunca	Geralmente falso	Não estou decidido	Ocasionalmente	Discordo totalmente
22/9	Muito frequentemente	Importante	Muito frequentemente	Geralmente verdade	Concordo totalmente	Raramente	Não estou decidido
22/9	Nunca	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Muito frequentemente	Concordo totalmente
22/9	Frequentemente	Importante	Raramente	Geralmente falso	Não estou decidido	Nunca	Não estou decidido
22/9	Muito frequentemente	Importante	Frequentemente	Às vezes verdadeiro	Concordo	Nunca	Não estou decidido
27/9	Muito frequentemente	Muito importante	Nunca	Quase sempre verdade	Concordo	Ocasionalmente	Não estou decidido
1/10	Frequentemente	Importante	Raramente	Às vezes verdadeiro	Concordo	Ocasionalmente	Concordo
1/10	Frequentemente	Importante	Frequentemente	Geralmente verdade	Concordo	Frequentemente	Concordo
1/10	Raramente	Às vezes é importante	Ocasionalmente	Geralmente falso	Discordo totalmente	Ocasionalmente	Concordo totalmente
1/10	Muito frequentemente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Muito frequentemente	Concordo totalmente
1/10	Frequentemente	Moderado	Frequentemente	Geralmente verdade	Concordo totalmente	Ocasionalmente	Concordo
1/10	Muito frequentemente	Importante	Muito frequentemente	Quase sempre verdade	Concordo	Muito frequentemente	Concordo totalmente
1/10	Frequentemente	Importante	Muito frequentemente	Quase sempre verdade	Concordo totalmente	Muito frequentemente	Concordo totalmente
1/10	Raramente	Às vezes é importante	Frequentemente	Geralmente verdade	Concordo totalmente	Muito frequentemente	Concordo
1/10	Raramente	Às vezes é importante	Nunca	Geralmente falso	Concordo	Ocasionalmente	Não estou decidido
1/10	Muito frequentemente	Muito importante	Raramente	Geralmente verdade	Concordo totalmente	Frequentemente	Concordo totalmente
1/10	Muito frequentemente	Muito importante	Muito frequentemente	Geralmente verdade	Concordo totalmente	Frequentemente	Concordo totalmente
1/10	Frequentemente	Muito importante	Raramente	Geralmente verdade	Concordo	Frequentemente	Concordo totalmente
1/10	Ocasionalmente	Moderado	Frequentemente	Às vezes verdadeiro	Discordo	Raramente	Não estou decidido
1/10	Ocasionalmente	Importante	Frequentemente	Geralmente verdade	Concordo totalmente	Raramente	Concordo
1/10	Ocasionalmente	Moderado	Frequentemente	Geralmente verdade	Concordo	Frequentemente	Concordo totalmente
1/10	Frequentemente	Muito importante	Muito frequentemente	Geralmente verdade	Concordo	Frequentemente	Concordo
1/10	Muito frequentemente	Moderado	Ocasionalmente	Quase sempre verdade	Concordo	Ocasionalmente	Concordo
1/10	Frequentemente	Muito importante	Nunca	Às vezes verdadeiro	Concordo	Ocasionalmente	Concordo
1/10	Frequentemente	Importante	Raramente	Às vezes verdadeiro	Concordo	Ocasionalmente	Concordo
1/10	Ocasionalmente	Moderado	Nunca	Geralmente verdade	Concordo totalmente	Raramente	Não estou decidido

1/10	Ocasionalmente	Importante	Nunca	Às vezes verdadeiro	Concordo	Raramente	Concordo
1/10	Muito frequentemente	Muito importante	Nunca	Geralmente verdade	Concordo	Raramente	Concordo
1/10	Frequentemente	Moderado	Frequentemente	Geralmente verdade	Não estou decidido	Frequentemente	Concordo
1/10	Ocasionalmente	Importante	Nunca	Às vezes verdadeiro	Concordo	Muito frequentemente	Concordo totalmente
1/10	Muito frequentemente	Importante	Nunca	Quase sempre falso	Concordo	Ocasionalmente	Concordo
1/10	Raramente	Importante	Raramente	Geralmente verdade	Concordo totalmente	Ocasionalmente	Concordo
3/10	Muito frequentemente	Importante	Nunca	Às vezes verdadeiro	Concordo	Raramente	Concordo totalmente
3/10	Muito frequentemente	Importante	Frequentemente	Às vezes verdadeiro	Concordo	Raramente	Concordo
3/10	Frequentemente	Importante	Raramente	Geralmente verdade	Concordo	Raramente	Concordo
3/10	Frequentemente	Moderado	Frequentemente	Geralmente verdade	Concordo totalmente	Muito frequentemente	Concordo totalmente
3/10	Frequentemente	Importante	Nunca	Geralmente verdade	Concordo	Ocasionalmente	Concordo
3/10	Frequentemente	Muito importante	Frequentemente	Geralmente verdade	Concordo	Frequentemente	Concordo
3/10	Frequentemente	Muito importante	Frequentemente	Às vezes verdadeiro	Não estou decidido	Frequentemente	Concordo
3/10	Frequentemente	Importante	Muito frequentemente	Às vezes verdadeiro	Concordo	Ocasionalmente	Concordo
3/10	Muito frequentemente	Importante	Frequentemente	Geralmente verdade	Não estou decidido	Ocasionalmente	Não estou decidido
3/10	Muito frequentemente	Importante	Raramente	Geralmente verdade	Concordo	Raramente	Concordo totalmente
3/10	Frequentemente	Moderado	Nunca	Quase sempre verdade	Concordo	Ocasionalmente	Concordo totalmente
3/10	Muito frequentemente	Importante	Nunca	Geralmente verdade	Concordo	Muito frequentemente	Não estou decidido
3/10	Muito frequentemente	Importante	Ocasionalmente	Geralmente verdade	Não estou decidido	Muito frequentemente	Concordo
3/10	Ocasionalmente	Moderado	Frequentemente	Às vezes verdadeiro	Não estou decidido	Nunca	Não estou decidido
3/10	Frequentemente	Muito importante	Nunca	Geralmente verdade	Não estou decidido	Ocasionalmente	Não estou decidido
3/10	Frequentemente	Às vezes é importante	Nunca	Geralmente verdade	Discordo	Nunca	Não estou decidido
3/10	Nunca	Moderado	Nunca	Geralmente falso	Não estou decidido	Nunca	Não estou decidido
3/10	Nunca	Às vezes é importante	Nunca	Quase sempre falso	Não estou decidido	Nunca	Não estou decidido
3/10	Nunca	Às vezes é importante	Raramente	Geralmente falso	Discordo	Nunca	Discordo
3/10	Nunca	Importante	Nunca	Geralmente verdade	Não estou decidido	Nunca	Não estou decidido
3/10	Raramente	Moderado	Raramente	Às vezes verdadeiro	Não estou decidido	Muito frequentemente	Concordo totalmente
3/10	Nunca	Muito importante	Nunca	Geralmente verdade	Discordo	Ocasionalmente	Discordo totalmente
3/10	Muito frequentemente	Muito importante	Frequentemente	Quase sempre verdade	Concordo totalmente	Frequentemente	Concordo totalmente
3/10	Nunca	Moderado	Nunca	Geralmente verdade	Não estou decidido	Frequentemente	Concordo
3/10	Ocasionalmente	Importante	Nunca	Geralmente falso	Concordo	Frequentemente	Concordo
8/10	Ocasionalmente	Moderado	Nunca	Às vezes verdadeiro	Concordo	Ocasionalmente	Concordo
8/10	Raramente	Às vezes é importante	Raramente	Geralmente verdade	Não estou decidido	Frequentemente	Concordo
8/10	Ocasionalmente	Importante	Nunca	Quase sempre falso	Concordo totalmente	Ocasionalmente	Concordo totalmente
8/10	Frequentemente	Importante	Ocasionalmente	Às vezes verdadeiro	Concordo totalmente	Frequentemente	Concordo
8/10	Ocasionalmente	Às vezes é importante	Nunca	Geralmente verdade	Não estou decidido	Frequentemente	Concordo
8/10	Ocasionalmente	Importante	Frequentemente	Às vezes verdadeiro	Concordo	Frequentemente	Concordo totalmente
8/10	Nunca	Às vezes é importante	Raramente	Geralmente verdade	Concordo totalmente	Muito frequentemente	Concordo totalmente
8/10	Raramente	Importante	Frequentemente	Quase sempre verdade	Concordo totalmente	Ocasionalmente	Concordo
8/10	Nunca	Não é importante	Nunca	Às vezes verdadeiro	Discordo	Raramente	Não estou decidido
8/10	Frequentemente	Moderado	Nunca	Às vezes verdadeiro	Discordo	Raramente	Não estou decidido
8/10	Muito frequentemente	Não é importante	Nunca	Geralmente falso	Não estou decidido	Raramente	Não estou decidido
8/10	Raramente	Moderado	Nunca	Às vezes verdadeiro	Não estou decidido	Frequentemente	Concordo
8/10	Raramente	Às vezes é importante	Raramente	Às vezes verdadeiro	Não estou decidido	Ocasionalmente	Não estou decidido
8/10	Ocasionalmente	Às vezes é importante	Frequentemente	Geralmente verdade	Não estou decidido	Frequentemente	Concordo
8/10	Nunca	Às vezes é importante	Nunca	Geralmente verdade	Não estou decidido	Frequentemente	Não estou decidido
8/10	Ocasionalmente	Moderado	Nunca	Geralmente falso	Concordo	Nunca	Não estou decidido
8/10	Frequentemente	Importante	Nunca	Geralmente verdade	Concordo	Frequentemente	Concordo
8/10	Frequentemente	Moderado	Nunca	Às vezes verdadeiro	Não estou decidido	Frequentemente	Concordo
8/10	Ocasionalmente	Moderado	Nunca	Geralmente verdade	Concordo	Ocasionalmente	Discordo
8/10	Raramente	Importante	Nunca	Às vezes verdadeiro	Concordo	Frequentemente	Concordo
8/10	Ocasionalmente	Importante	Nunca	Às vezes verdadeiro	Não estou decidido	Ocasionalmente	Discordo totalmente
8/10	Ocasionalmente	Às vezes é importante	Nunca	Geralmente verdade	Não estou decidido	Raramente	Não estou decidido
8/10	Ocasionalmente	Não é importante	Nunca	Às vezes verdadeiro	Não estou decidido	Raramente	Concordo
8/10	Ocasionalmente	Às vezes é importante	Nunca	Geralmente verdade	Não estou decidido	Raramente	Concordo
8/10	Nunca	Muito importante	Nunca	Às vezes verdadeiro	Não estou decidido	Ocasionalmente	Não estou decidido
8/10	Ocasionalmente	Às vezes é importante	Nunca	Às vezes verdadeiro	Não estou decidido	Ocasionalmente	Não estou decidido
8/10	Frequentemente	Muito importante	Nunca	Às vezes verdadeiro	Concordo	Raramente	Não estou decidido
8/10	Ocasionalmente	Importante	Nunca	Às vezes verdadeiro	Concordo	Raramente	Concordo
8/10	Ocasionalmente	Às vezes é importante	Nunca	Às vezes verdadeiro	Concordo	Muito frequentemente	Concordo totalmente
8/10	Raramente	Às vezes é importante	Nunca	Geralmente verdade	Não estou decidido	Ocasionalmente	Concordo
8/10	Ocasionalmente	Importante	Nunca	Às vezes verdadeiro	Não estou decidido	Frequentemente	Concordo
8/10	Ocasionalmente	Importante	Frequentemente	Às vezes verdadeiro	Concordo totalmente	Frequentemente	Concordo
8/10	Muito frequentemente	Muito importante	Raramente	Geralmente falso	Concordo totalmente	Nunca	Não estou decidido
8/10	Muito frequentemente	Muito importante	Frequentemente	Geralmente falso	Concordo totalmente	Muito frequentemente	Não estou decidido
8/10	Muito frequentemente	Muito importante	Nunca	Geralmente falso	Não estou decidido	Nunca	Não estou decidido
8/10	Muito frequentemente	Muito importante	Nunca	Quase sempre falso	Concordo totalmente	Nunca	Não estou decidido
8/10	Frequentemente	Muito importante	Nunca	Geralmente verdade	Concordo totalmente	Nunca	Não estou decidido
8/10	Frequentemente	Muito importante	Nunca	Às vezes verdadeiro	Concordo	Frequentemente	Concordo
8/10	Frequentemente	Importante	Nunca	Às vezes verdadeiro	Concordo	Frequentemente	Concordo
8/10	Frequentemente	Importante	Nunca	Quase sempre falso	Não estou decidido	Nunca	Não estou decidido
8/10	Muito frequentemente	Muito importante	Nunca	Geralmente verdade	Concordo	Ocasionalmente	Concordo
8/10	Muito frequentemente	Muito importante	Nunca	Geralmente falso	Concordo	Frequentemente	Concordo
8/10	Nunca	Muito importante	Nunca	Quase sempre falso	Não estou decidido	Nunca	Não estou decidido
8/10	Muito frequentemente	Muito importante	Raramente	Quase sempre falso	Concordo	Raramente	Concordo