Coleção Livro Aberto de Matemática

Ensino Médio

Pensamento Computacional

Livro do Professor

Leonardo Barichello

Pensamento Computacional
2ª edição, 9 de agosto de 2023
Leonardo Barichello
ISBN: 978-85-244-0523-5
Publicado no Brasil / Published in Brazil

Coleção Livro Aberto de Matemática

Revisão: Dênis Vargas, Letícia Rangel, Diego Lieban, Larissa Monzon, Kátia Rocha

Capa: Enzo Esberard
Ilustrações: Agnes Antonello
Diagramação e Gráficos: Tarso Boudet Caldas

Realização: Instituto de Matemática Pura e Aplicada (IMPA)
Estrada Dona Castorina, 110
Jardim Botânico
22460-320, Rio de Janeiro, RJ
www.impa.br
editora@impa.br

B2521 Barichello, L.,
Pensamento Computacional/Leonardo Barichello — 2ª ed. — Rio de Janeiro, IMPA, 2023.
47 p.: il. color (Coleção Livro Aberto de Matemática)

E-book
ISBN 978-85-244-0523-5 (Versão professor)

1. Matemática. I. Título II. Série

CDD: 510
Carolina Celano Lima/CRB-7: 2438

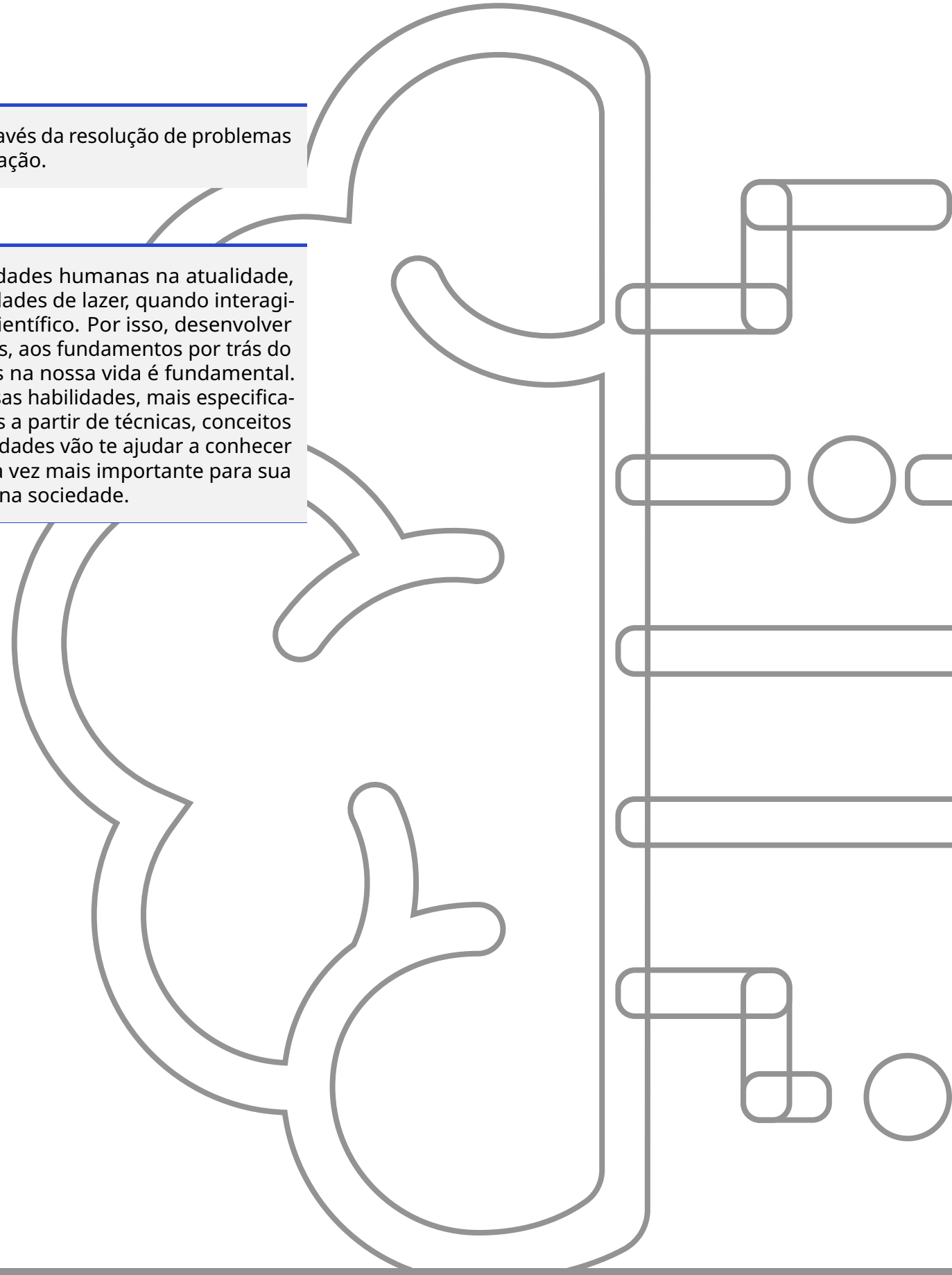
Sumário

O quê?

Neste material trataremos de Pensamento Computacional através da resolução de problemas matemáticos por meio do uso de de linguagens de programação.

Por quê?

Os computadores estão presentes em quase todas as atividades humanas na atualidade, seja nos mais diversos ramos profissionais, nas nossas atividades de lazer, quando interagimos com outras pessoas e na produção de conhecimento científico. Por isso, desenvolver habilidades relacionadas ao uso de softwares e computadores, aos fundamentos por trás do seu funcionamento e ao impacto do uso dessas tecnologias na nossa vida é fundamental. Pensamento computacional é o nome dado a uma parte dessas habilidades, mais especificamente, às habilidades relacionadas à resolução de problemas a partir de técnicas, conceitos e ferramentas típicos da Ciência da Computação. Essas habilidades vão te ajudar a conhecer noções básicas de linguagens de programação, o que é cada vez mais importante para sua vida profissional, trajetória acadêmica e participação efetiva na sociedade.



Introdução ao Professor

A Base Nacional Comum Curricular (BNCC) traz o desenvolvimento do pensamento computacional como um dos objetivos relacionados à área de Matemática desde os Anos Finais do Ensino Fundamental até o Ensino Médio. Essa expressão não aparece nas competências nem nas habilidades, mas é mencionada diversas vezes ao longo das discussões propostas no documento, sempre nas seções sobre Matemática.

Em geral, essas menções sugerem que o desenvolvimento do pensamento computacional deve ser visto como uma consequência do ensino de Matemática:

> Outro aspecto a ser considerado é que a aprendizagem de Álgebra, como também aquelas relacionadas a Números, Geometria e Probabilidade e Estatística, podem contribuir para o desenvolvimento do pensamento computacional dos alunos (BNCC Brasil, 2018, p. 271)

Embora essa recomendação ocorra em diversos trechos do documento, uma busca por "pensamento computacional" nas habilidades apresenta nenhum exemplo objetivo, sendo possível apenas identificar termos relacionados a essa temática, como fluxogramas e algoritmos, em algumas habilidades de Matemática (e apenas de Matemática) do Ensino Fundamental e do Ensino Médio. No caso do Ensino Médio, as duas habilidades que fazem menções a esses termos são:

Habilidades da BNCC

EM13MAT315 Investigar e registrar, por meio de um fluxograma, quando possível, um algoritmo que resolve um problema.

EM13MAT405 Utilizar conceitos iniciais de uma linguagem de programação na implementação de algoritmos escritos em linguagem corrente e/ou matemática.

Mas então, o que seria o pensamento computacional?

Esse termo é resultado de um crescente movimento de incorporação da computação a currículos escolares não apenas como ferramentas pontuais para ensino e aprendizagem de alguns conceitos, mas como um disciplina ou um conjunto de habilidades mais amplo que esteja relacionado ao mundo digital (Raabe et al., 2020).

Para nos ajudar a entender de forma mais concreta, vamos considerar o currículo de referência em Teconologia e Computação proposto pelo Centro de Inovação para a Educação Brasileira (CIEB), disponível em www.curriculo.cieb.net.br/curriculo. Essa proposta, divide o currículo em três grandes eixos: Cultura Digital, Tecnologia Digital e Pensamento Computacional. Para este terceiro eixo, é dada a seguinte definição:

> Pensamento Computacional refere-se à capacidade de resolver problemas a partir de conhecimentos e práticas da computação, englobando sistematizar, representar e analisar.

Além disso, a proposta do CIEB divide o Pensamento Computacional em 4 conceitos: Abstração, Algoritmos, Decomposição e Reconhecimento de Padrões.

Embora o currículo de referência do CIEB seja muito claro e completo, do ponto de vista de um professor de Matemática, a definição e os quatro conceitos soam como habilidades que já são ou poderiam ser desenvolvidas em aulas usuais de matemática, sem que houvesse necessidade de trazer este tal de pensamento computacional para o cenário.

A concepção que vamos desenvolver busca justamente salientar o que o pensamento computacional pode trazer de novidade e de específico para um professor de matemática no Ensino Médio.

Pensamento computacional no Livro Aberto

Vários são os autores que buscam definir o que é pensamento computacional (Shute et al., 2017) e é comum entre eles o reconhecimento de que há uma grande intersecção entre as habilidades e conceitos cobertos por este termo e as habilidades e conceitos cobertos pelo que poderíamos chamar de "pensamento matemático". Entretanto, um elemento que se destaca como exclusivo ao universo do pensamento computacional é o foco em processos de resolução de problemas, os quais, após formalização, são chamados de algoritmos.

Enquanto em Matemática o foco da atividade usualmente recai em dois objetos, a resolução de um problema ou a demonstração de um teorema, em computação o processo de resolução de um problema é o grande objeto de interesse. Nesse sentido, habilidades relacionadas a compreender, sistematizar e analisar esse processo ganham importância.

Isso será feito neste material em duas etapas (Reis et al., 2021). Primeiro, propondo problemas matemáticos e estendendo a discussão no sentido de mudar o foco progressivamente para o processo de resolução através de diferentes representações (textuais, visuais ou em uma linguagem de programação).

Tomemos como exemplo o seguinte cenário inspirado em uma questão da Olimpíada Brasileira de Informática:

> O cometa Halley é um dos cometas de menor período do Sistema Solar, completando uma volta em torno do Sol a cada 76 anos; na última ocasião em que ele ficou visível do planeta Terra, em 1986, várias agências espaciais enviaram sondas para coletar amostras de sua cauda e assim confirmar teorias sobre suas composições químicas.

A partir desse contexto, um professor de matemática poderia propor questões como: Quando será a próxima passagem do cometa Halley? Uma pessoa que nascerá no ano 2500 terá a chance de avistar o cometa Halley pela primeira vez em que ano? Quantas vezes o cometa passará ao longo de todo o terceiro milênio?

De fato, essas questões serão colocadas a cada atividade, mas complementadas por questões como essa:

> Descreva como obter, a partir de um dado ano qualquer, qual é o próximo ano em que o cometa Halley será visível novamente do planeta Terra.

Como mostrado na próxima figura, esse tipo de questão pode ser respondida de forma textual (A), com auxílio de notação matemática (C) e fluxogramas (B) ou como um algoritmo escrito em uma linguagem de programação (D), como mostrado na imagem a seguir.

Ao resolver e discutir questões como essas, o objeto de interesse passa a ser o processo de resolução da questão ou, simplesmente, o algoritmo, como sugerido pela BNCC:

> Associado ao pensamento computacional, cumpre salientar a importância dos algoritmos e de seus fluxogramas, que podem ser objetos de estudo nas aulas de Matemática. (BNCC Brasil, 2018, p. 271)

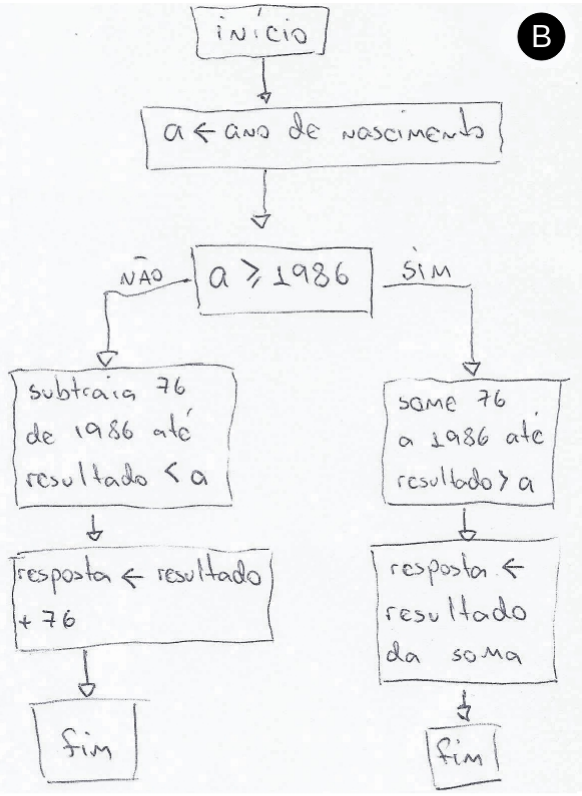
A

ano de nascimento = a

Se $a \geq 1986$, então somamos 76 a 1986 até que o resultado seja maior que a. Essa é a resposta.

Se $a < 1986$, subtraímos 76 de 1986 até que o resultado seja menor que a. A resposta é o valor obtido uma subtração antes.

B



C

a ← ano de nascimento

$n \leftarrow \frac{(a - 1986)}{76}$

arredonde n para cima (inteiro)

resposta ← 1986 + 76 . n

D

```
programa
{
    funcao inicio()
    {
        inteiro a, n, resposta
        leia(a)
        n = (a-1986) / 76
        se (a>1986) {
            n=n+1
        }
        resposta = 1986 + n*76
        escreva(resposta)
    }
}
```

A isso chamamos de resolver um problema sob o ponto de vista computacional. Ao fazer isso, esperamos que os estudantes vivenciem um processo que se inicia com descrições textuais e notação matemática, passa por fluxogramas e se completa com uma linguagem de programação. Entendemos que esse processo cobre as duas habilidades relacionadas ao pensamento computacional colocadas na BNCC do Ensino Médio e ele é desenvolvido ao longo deste material.

A segunda etapa do desenvolvimento do pensamento computacional tem início quando os estudantes saibam utilizar uma linguagem de programação para criar algoritmos que possam ser executadas pelo computador. A partir desse ponto, eles podem usar essa ferramenta para explorar novos problemas matemáticos ou para lançar novos olhares para problemas conhecidos, estendendo-os.

Essa segunda etapa aprofunda as habilidades da BNCC relacionadas ao pensamento computacional, mas demanda algum domínio sobre programação de computadores e, de certa forma, dependendo de conteúdos matemáticos que serão desenvolvidos ao longo do Ensino Médio. Por isso, as atividades relacionadas a ela são apresentadas em um outro material da coleção Livro Aberto, podendo ser utilizadas como oficinas avulsas, como atividades a serem realizadas em momentos oportunos do currículo ou como material de base para uma disciplina eletiva ou de itinerários formativos relacionados à matemática ou à tecnologia.

A organização do material

Este material está dividido em 5 seções.

As 4 primeiras propõem que os estudantes resolvam problemas matemáticos sob o ponto de vista computacional, como exemplificado anteriormente com o problema do cometa Halley. Para isso, além de introduzir esse novo modo de pensar ao estudante (seção 1), também exploramos alguns conceitos básicos de linguagens de programação como variáveis (seção 2), condicional (seção 3) e repetição (seção 4).

Ao longo dessas 4 seções, o estudante será progressivamente apresentado ao Portugol e esperamos que tenha um domínio significativo dessa linguagem ao final do material.

Por fim, na seção 5 são propostos problemas que cobrem os conceitos básicos vistos anteriormente e podem servir como fixação e prática, não sendo obrigatórios para o cumprimento das habilidades da BNCC.

Como usar

Estimamos que a discussão do material completo ocupe cerca de 24 aulas de 50 minutos cada. Além disso, é importante salientar que os pré-requisitos matemáticos para a resolução das atividades propostas neste material se referem a objetos do conhecimento já tratados ao longo do Ensino Fundamental.

Dessa forma, sugerimos que este material seja utilizado de duas maneiras:

a) Como um capítulo independente integrado ao currículo comum de Matemática. Nesse caso, sugerimos também que ele seja usado no início do 1º ano do Ensino Médio, permitindo assim que professores e estudantes possam utilizar linguagens de programação, quando adequado, em outros momentos do currículo;

b) Como material base para disciplinas eletivas ou que sejam parte de um itinerário formativo relacionado à matemática ou à tecnologia. Também sugerimos que essa disciplina seja ofertada o mais cedo possível no Ensino Médio, uma vez que abre portas para outros estudos que envolvam linguagens de programação.

Essa decisão deve ser tomada pelo professor de acordo com o perfil de seus estudantes e as possibilidades de planejamento curricular da escola.

A linguagem escolhida: Portugol

Ao longo deste material será proposta uma transição que começa no registro livre das soluções. A partir delas, e de uma busca por maior clareza nas ações e no encadeamento delas, surgirá a necessidade de utilizar recursos mais específicos para registro das soluções, como fluxogramas ou elementos textuais mais estruturados. Ao final, gostaríamos que os estudantes tivessem a chance de conhecer uma linguagem de programação.

Se o professor tem familiaridade com alguma linguagem de programação, sugerimos fortemente que a utilize nas aulas, pois problemas que propomos não são complexos a ponto de exigirem recursos específicos que não estejam presentes em linguagens de programação comerciais. Porém, se você não conhece nenhuma, nossa sugestão é o Portugol.

Os motivos que determinaram essa escolha foram:

a) Os comandos em Portugol foram criados em português;

b) Portugol foi criado com objetivos didáticos, ou seja, algumas de suas características não foram escolhidas por motivações computacionais, mas sim para facilitar a aprendizagem por programadores iniciantes (Noschang et al., 2014);

c) Existem muitos materiais que permitem o autoestudo da linguagem na internet. Recomendamos especificamente a playlist sobre o Portugol do canal HM Programming no Youtube (www.youtube.com/watch?list=PLJ4lbwalqv3Eaiay2pCeU_QU6vb-Hz989) e os materiais disponíveis no site do LITE (http://lite.acad.univali.br/portugol). Além disso, você também pode se familiarizar com a linguagem através das respostas para os exercícios contidos neste material.

d) Portugol pode ser usado de três formas diferentes: em um ambiente de programação online acessível a partir de um navegador (http://portugol-webstudio.cubos.io/ide), ou seja, sem demandar a instalação de softwares específicos; em um ambiente de programação offline instalado no computador sem fazer uso da internet; e em um aplicativo instalado no celular (como o Portugol Mobile, disponível em lojas de aplicativos).

Ao longo do material, será utilizada apenas a interface de texto, ou seja, a interação entre o usuário e o algoritmo será apenas por texto em ambiente similar ao estereótipo de computação: tela preta e letrinhas brancas. Apesar de parecer antiquada, essa decisão foi tomada para evitar que questões relacionadas à interface fossem trazidas para as discussões, como seria inevitável em ambientes de programação como o Scratch, App Inventor ou ambientes web.

Atualmente, o Portugol Studio, um dos softwares que permite o uso da linguagem Portugol, é mantido pelo Laboratório de Inovação Tecnológica na Educação (LITE) da Univali (Esteves et al., 2019).

O uso do computador

Embora o uso de computadores não seja uma condição para a realização de atividades relacionadas ao pensamento computacional, é claro que esse instrumento é parte importante desse processo. O uso de computadores não apenas foi a fonte de inspiração para as habilidades que o termo pensamento computacional busca promover, mas também favorece, mesmo que implicitamente, o desenvolvimento dessas habilidades por meio das suas características.

Porém, sabemos que o uso de computadores nem sempre é viável nas escolas brasileiras.

Por isso, na concepção deste material, buscamos um balanço na escolha das atividades de modo que um professor que tenha recursos computacionais limitados, ainda pudesse desenvolver as habilidades da BNCC que cobrimos.

Primeiro, acreditamos que muitas das atividades que são propostas no início do material podem ser resolvidas e discutidas sem o uso de computadores e de maneira significativa. Além disso, diferentes dinâmicas para o uso dos computadores podem ser adotadas.

Idealmente, gostaríamos que os estudantes tivessem acesso a um computador ou celular ao final de cada atividade para que pudessem implementar os algoritmos que venham a criar e tenham uma experiência mais completa. Nesse cenário, sugerimos o trabalho em pequenos grupos, cada um com acesso a um computador ou celular.

Outra possibilidade é que a turma como um todo tenha acesso a um computador. Essa dinâmica é proposta em algumas atividades como uma maneira de promover discussões com todos os estudantes participando juntos em torno de um mesmo algoritmo. O uso de um projetor é ideal para essa dinâmica, mas um computador ou notebook também pode funcionar.

Avaliação

Como aspectos ligados ao pensamento computacional não são cobrados em avaliações oficiais ou exames de seleção como a maioria dos tópicos do currículo de matemática, a avaliação das habilidades cobertas por este material pode ser feita utilizando outras práticas diferentes das convencionais.

Essa flexibilidade cria condições para experimentar novas formas de avaliar. Nossa recomendação é de que a avaliação seja feita através de um número pequeno de problemas e com um intervalo de tempo maior para os estudantes resolverem, diferentemente do cenário típico de uma prova composta por várias questões cobrindo, cada uma, diferentes detalhes do conteúdo e cuja resolução tipicamente deve ser feita em um espaço de tempo limitado.

A nossa sugestão vem da percepção de que o objetivo deste material não é desenvolver o domínio do estudantes sobre um conjunto de conceitos, mas sim uma familiaridade geral com o pensamento computacional e com conceitos e práticas básicas de linguagens de programação. Nesse sentido, consideramos importante que, em uma avaliação, o estudante tenha a possibilidade de pesquisar, testar, errar e corrigir.

Temos duas sugestões que podem ajudar no processo de avaliação:

a) As questões propostas na Seção 5 podem ser usadas para avaliação, uma vez que não trazem novos conteúdos e habilidades, apenas exploram o que já foi estudado nas seções anteriores;

b) Nas notas para o professor dessas questões há sugestões de variações que também podem ser usadas na avaliação. De maneira geral, buscamos indicar variações das questões propostas ao longo de todo o material para que o professor utilize para estender as atividades ou para fins de avaliação.

A matemática deste capítulo

Como já deve ter ficado claro, os objetivos deste capítulo não estão relacionados a objetos matemáticos com que estamos acostumados a lidar em um currículo convencional. Porém, é natural que alguns objetos surjam ao longo das atividades uma vez que estas são pautadas por problemas matemáticos. Nesse sentido, cabe salientar que foi feito um esforço explícito durante a elaboração deste material para que não houvessem pré-requisitos de conteúdo além do que se aborda no Ensino Fundamental.

Além disso, a dificuldade matemática dos problemas escolhidos também foi considerada com cuidado, de modo que a dificuldade aumenta progressivamente ao longo do material. Esperamos que dessa forma, dificuldades de natureza matemática não obstruam o foco central: o desenvolvimento do pensamento computacional.

Glossário explicativo

Algoritmo: uma sequência de passos que devem ser seguidos para a realização de uma determinada tarefa. Exemplos podem ir de uma receita de bolo até o algoritmo que permite ao seu navegador de internet enviar sua senha para o banco sem que um outro usuário possa lê-la, passando pelo algoritmo da fatoração de um número natural. Todos esses exemplos são sequências de passos, mais ou menos rígidos e mais ou menos formais, que devem ser seguidos para que se obtenha um resultado final, seja ele um bolo, uma autenticação segura ou a fatoração de um número natural.

Linguagem de programação: é uma linguagem usada para descrever algoritmos, escrita por um ser humano, que pode ser compreendida por um computador. Nesse caso, o termo “compreender” está sendo usado no sentido bem restrito de “seguir os passos”. Normalmente, linguagens

de programação são constituída por um conjunto pequeno de comandos disponíveis e sintaxe muito rígida, para que não reste qualquer ambiguidade no momento da execução dos comandos. Isso pode comprometer a agilidade do seu uso, pois é necessário conhecer vários detalhes específicos que, eventualmente, variam de uma linguagem para outra. Exemplos de linguagens comuns atualmente são: Python, Javascript, C e Portugol.

Fluxograma: é uma forma essencialmente visual de representar algoritmos. O seu uso é recomendado na BNCC do Ensino Fundamental como uma maneira de representar soluções de problemas algorítmicos. Existem versões padronizadas de fluxogramas, com regras sobre como determinadas ações e componentes devem ser representadas. Porém, o uso mais comum deste recurso é mais informal preocupando-se primordialmente com a representação geral do fluxo do algoritmo. Por conta disso, apesar de seguirmos um padrão de cores e formatos nos fluxogramas apresentados, não nos preocuparemos em descrever em detalhe e discutir esse padrão, recomendando que o professor deixe que os estudantes não se preocupem com esses aspectos.

Condicional: Um dos conceitos básicos de qualquer linguagem de programação. É uma instrução que direciona os próximos passos a serem executados pelo computador para um ou outro conjunto de comandos de acordo com uma condição. Normalmente, se expressa no formato "se condição, então faça isso, senão faça aquilo". Por exemplo, se pensarmos em um algoritmo que identifique se um número natural é múltiplo de 5 teríamos basicamente o seguinte condicional: se o algarismo das unidades for igual a 0 ou 5, então "sim", caso contrário "não". Esquematicamente, podemos escrevê-la assim:

```
se (algarismo das unidades=0 ou algarismo das unidades=5) então
escreva(”sim”)
senão
escreva(”não”)
```

Vale salientar a diferença entre o uso do termo condicional na computação e na matemática. Enquanto que na computação ele se refere a um recurso que permite direcionar o fluxo de um processo para uma ou outra direção, de acordo com uma condição (como mostrado no exemplo), em matemática ele costuma se referir a teoremas ou propriedades, como em "se um número inteiro tiver algarismo das unidades igual a 0, então ele é múltiplo de 5". Embora diferentes, esses dois usos podem ser compatíveis, mas em matemática a ênfase recai na leitura lógica da expressão enquanto que na computação a ênfase está no controle do fluxo de um algoritmo.

Variável: Apesar de familiar, no universo de programação de computadores o termo variável também tem um uso um pouco diferente do que fazemos em matemática. Em matemática, variável representa um elemento genérico de um conjunto dado (dito universo da variável), sendo normalmente designada por uma letra minúscula como `x` , `y` ou `t` . Já no contexto de programação de computadores, uma variável é um espaço na memória do computador que armazena um pedaço de informação de um determinado tipo. Alguns dos tipos mais comuns são números inteiros, valores booleanos (verdadeiro ou falso), números com parte decimal e caracteres. Porém, o conteúdo de uma variável é sempre estabelecido: a qualquer momento na execução de um código é possível invocar o conteúdo (ou valor) de uma variável e usá-lo para executar alguma ação. Além disso, esse valor pode ser mudado ao longo da execução. No exemplo abaixo, na primeira linha, duas variáveis, com nomes `num` e `t` são criadas e especifica-se de qual tipo elas são (inteiro). Na segunda linha, armazena-se o valor 5 na variável `num` (o comando diz para o computador armazenar no trecho de memória reservado à variável `num` o valor 5). Na terceira linha, armazena-se o dobro do valor armazenado em `num` na variável `t` . Por conta disso, o valor a ser escrito pelo comando da quarta linha será 10.

```
inteiro num, t
num <- 5
```

```
3 t <- 2*num
4 escreva(t)
5 t <- t+1
6 escreva(t)
```

Como, ao chegar na linha 5 (antes de executá-la), o valor armazenado em `t` era 10, o resultado de `t+1` é 11, portanto, o valor que será escrito pelo comando da linha seguinte é 11.

Um detalhe importante que é necessário salientar é que, sempre que estivermos descrevendo um algoritmo em um fluxograma ou textualmente, usaremos o simbolo ← ou `<-` para descrever uma atribuição de um valor a uma variável. Porém, muitas linguagens de programação (como a que escolhemos para este material) utilizam o símbolo `=`. Essa escolha resulta em comandos que são, no mínimo, ambíguos com a notação matemática. Por exemplo, o comando "`t <- t+1`" seria escrito como "`t=t+1`". Em um contexto matemático, este último comando representa uma equação sem solução, o que nada tem a ver com o significado em contexto computacional.

Repetição: Outro recurso fundamental de qualquer linguagem de programação são as estruturas de repetição. Embora possa haver variações no formato, em geral, elas seguem a seguinte ideia: "repita certas ações enquanto determinada condição seja verdadeira". Por exemplo, se quisermos somar todos os números até 100 e mostrar o resultado, podemos esquematizar esse processo da seguinte maneira:

```
1 soma=0
2 numero=1
3 enquanto (numero<=100) faca
4 soma = soma+numero
5 numero = numero+1
6 escreva("soma")
```

Note que antes da repetição ajustamos as variáveis para que tenham valores iniciais coerentes com o objetivo do código. A cada repetição, duas ações são realizadas:

a) a variável número é somada à variável soma e o resultado é guardado na variável soma;

b) somamos 1 à variável número.

Essas ações serão repetidas enquanto o valor da variável número seja menor ou igual a 100. Depois, o conteúdo da variável soma será escrito na tela.

Habilidades, partes, seções e objetivos específicos

A relação entre as diferentes seções deste material com as habilidades da BNCC e Objetivos Específicos que foram definidos para contemplarmos cada uma das habilidades está apresentada no Quadro 1. Logo em seguida, trazemos a descrição de cada um dos objetivos específicos.

Objetivo Específico 1: Analisar problemas matemáticos visando sua resolução do ponto de vista computacional.

Objetivo Específico 2: Compreender como sistematizar soluções algorítmicas através de recursos como a linguagem matemática, fluxogramas ou linguagens de programação. Esse objetivo específico será desdobrado, para fins de clareza, em casos particulares que farão referência a qual forma de representação será valorizada a cada atividade.

		EM13MAT315		EM13MAT405		
		OE 1	OE 2	OE 3A	OE 3B	OE 3C
Parte 1	Seção 1	X	X			
	Seção 2		X	X		
	Seção 3		X		X	
	Seção 4		X			X
	Seção 5			X	X	X

Quadro 1: Habilidades e Objetivos Específicos de cada seção

Objetivo Específico 3: Compreender os conceitos básicos de uma linguagem de programação. Este objetivo específico foi desdobrado, para fins de clareza, em três: 3A (focado no conceito de variável), 3B (focado no conceito de condicional) e 3C (focado no conceito de estruturas de repetição).

Em termos das competências específicas de Matemática para o Ensino Médio propostas da BNCC, este material contribui com o desenvolvimento de três delas:

- a **Competência Específica 3** (Utilizar estratégias, conceitos, definições e procedimentos matemáticos para interpretar, construir modelos e resolver problemas em diversos contextos, analisando a plausibilidade dos resultados e a adequação das soluções propostas, de modo a construir argumentação consistente), à medida que a resolução de problemas é a tônica geral do material e que esse novo olhar, sob o ponto de vista computacional, pode ampliar o repertório do estudante para lidar com problemas de diversas naturezas;

- a **Competência Específica 4** (Compreender e utilizar, com flexibilidade e precisão, diferentes registros de representação matemáticos (algébrico, geométrico, estatístico, computacional etc.), na busca de solução e comunicação de resultados de problemas), afinal, novas formas de representação, como os fluxogramas e algoritmos escritos em linguagens de programação, serão amplamente empregados ao longo do material;

- a **Competência Específica 5** (Investigar e estabelecer conjecturas a respeito de diferentes conceitos e propriedades matemáticas, empregando estratégias e recursos, como observação de padrões, experimentações e diferentes tecnologias, identificando a necessidade, ou não, de uma demonstração cada vez mais formal na validação das referidas conjecturas), uma vez que o uso de programação de computadores pode abrir novas portas para a experimentação com objetos matemáticos.

Materiais adicionais

Considerando que o tema pensamento computacional é novo nos currículos brasileiros e que, portanto, há pouco material de fato criado para uso em sala de aula e para autoestudo por parte de professores interessados, criamos o site www.mais.mat.br/pensamentocomputacional para disponibilizar materiais que complementam o conteúdo deste texto, como:

- As soluções das questões envolvendo algoritmos na forma textual para que o professor possa copiar, colar, testar e estudar com mais facilidade;

- Soluções em outras linguagens de programação;

- Propostas de novas atividades estritamente relacionadas às atividades presentes nessa material para que o professor possa usar para autoestudo, exercícios adicionais em sala de aula ou para avaliação;
- Exemplos de respostas dadas por estudantes.

A intenção do site é oferecer suporte adicional para o professor, algo especialmente importante considerando que o tema pensamento computacional pouco aparece na formação inicial de professores de matemática.

Para o professor: Um novo jeito de resolver problemas

As atividades desta seção introduzem aos estudantes o que chamamos na introdução para o professor de "resolução de um problema do ponto de vista computacional" e criam um contexto no qual um primeiro aspecto importante para o pensamento computacional pode ser discutido: a clareza das instruções.

Os dois problemas propostos na duas primeiras atividades serão revisitados em seguida. Primeiro, como contexto para a discussão de **fluxogramas**. Não pretendemos usar sempre fluxogramas ao longo do capítulo, mas a discussão da estrutura e uso deste tipo de representação pode salientar aspectos importantes sobre a clareza das instruções descritas pelos estudantes em outras representações. A BNCC propõe o uso de fluxogramas no Ensino Fundamental (como nas habilidades **EF06MA04**, **EF07MA07**, **EF08MA10**, **EF09MA15**), porém, mesmo que os estudantes não tenham estudado o assunto, esperamos que haja alguma familiaridade informal e isso deve bastar para criar discussões proveitosas. Por último, os dois problemas servirão de contexto para introdução da linguagem de programação Portugol. Isso é feito no **Para Saber Mais** que encerra a seção.

Como explicado na introdução, o uso de laboratório de informática para que os estudantes se familiarizem com uma linguagem de programação é muito desejável, mas não é indispensável. Você, professor, deve fazer essa escolha de acordo com as possibilidades e características da sua turma e escola.

Caso decida usar o laboratório, sugerimos que nessa primeira ida, o professor proponha uma atividade menos aberta em que os estudantes tenham apenas que reproduzir os algoritmos dados e possam experimentar resolver vários casos executando-os.

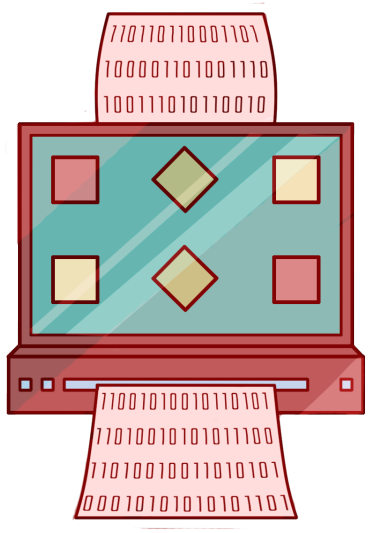
Vale a pena salientar que linguagens de programação precisam ser seguidas à risca e isso pode gerar pequenos erros, especialmente em uma primeira vez. Preste atenção no uso de chaves e na digitação dos comandos. Além disso, alguns símbolos utilizados em linguagens de programação são diferentes dos que utilizamos em notação matemática convencional. Abaixo estão indicados alguns que serão usados já nas primeiras atividades:

- `*` multiplicação
- `!=` diferente
- `==` igualdade
- `<=` menor ou igual a
- `>=` maior ou igual a
- `%` resto, ou seja, `a%b` calcula o valor do resto da divisão inteira de `a` por `b`

Por fim, um último detalhe importante referente a Portugol é o símbolo usado para separação decimal, que é o ponto (como nos países de língua inglesa) e não a vírgula (como no Brasil).

1 Um novo jeito de resolver problemas

Os computadores estão em todos os lugares, seja na forma de celulares, notebooks, computadores de mesa ou dispositivos com funções mais específicas, como máquinas de cobrança. Das atividades mais corriqueiras às atividades mais inovadoras, é possível identificar a influência de um computador, e na matemática não é diferente.



Nas últimas décadas, a comunidade de matemáticos teve que aprender a lidar com o uso intensivo de computadores em uma área do conhecimento que, por muito tempo, aceitava de forma quase única argumentos lógicos na forma textual. Um exemplo ocorreu com o conhecido **problema das quatro cores** que, em resumo, lançava a questão sobre quantas cores são necessárias para pintar um mapa sem que países vizinhos tenham a mesma cor.

Em 1852, uma resposta para esse problema foi sugerida por um matemático: 4 cores seriam suficientes, mas ele não demonstrou que essa resposta de fato estava correta. Mais de 100 anos se passaram até que um grupo de matemáticos conseguiu demonstrar que a tal resposta estava de fato correta, mas a demonstração era feita com o auxílio de computadores e tomava mais de mil horas de processamento nos computadores mais rápidos existentes naquele momento.

O que incomodou os matemáticos não foi apenas o tempo quase sobrehumano para checar essa solução, mas principalmente o questionamento sobre a validade ou não de uma demonstração que dependia de um computador para ser realizada.

Você Sabia?

Você pode saber mais sobre o problema das quatro cores assistindo ao vídeo disponível em youtu.be/TsAuQkbkW50.

O que essa história ilustra é que a disponibilidade de computadores faz mais do que agilizar certas tarefas repetitivas, mas cria a possibilidade de resolvermos problemas de maneira diferente do que poderíamos sem eles. A capacidade de utilizar um computador amplifica as nossas possibilidades como resolvedores de problemas.

Neste material, vamos desenvolver o seu pensamento computacional partindo de problemas matemáticos como os que você já está acostumado a resolver e buscar chegar até a criação de algoritmos que resolvam esses problemas e que possam ser executados por um computador!

Objetivos Específicos

Boas instruções

- Analisar problemas matemáticos visando a sua resolução do ponto de vista computacional.
- Compreender como sistematizar soluções algorítmicas usando linguagem textual e matemática.

Sugestões e discussões

Boas instruções

Organização da turma a última questão requer que os estudantes discutam as respostas criadas por colegas. Essa discussão é importante para que eles tenham a oportunidade de perceber as limitações das suas respostas bem como para que possam se inspirar na estrutura das respostas dos colegas. Sugerimos que você organize a turma em quartetos para que os alunos resolvam em duplas as questões **a)** , **b)** , **c)** e **d)** e depois façam a troca com a outra dupla do seu quarteto para discutir a questão **e)** .

Dificuldades previstas para essa atividade, os estudantes devem imaginar o destinatário do bilhete como sendo uma pessoa qualquer, não um outro estudante do mesmo nível ou que conheça o problema. Essa abstração pode ser difícil para alguns deles, por isso a importância de interação entre eles.

Enriquecimento da discussão se for viável para o seu contexto, seria rico observar alguém que realmente não conhece a atividade tentando fazer o cálculo com base em um bilhete.

Conexões Você pode discutir o conceito de função contínua e de função crescente comparando as funções "preço final" sugeridas pelas duas ideias descritas na atividade.

Duração 2 aulas. É importante não apressar essa atividade, pois o objetivo principal é mostrar aos estudantes como não é simples ser claro em instruções que serão seguidas por outras pessoas.

Nota 1

Atividade 1

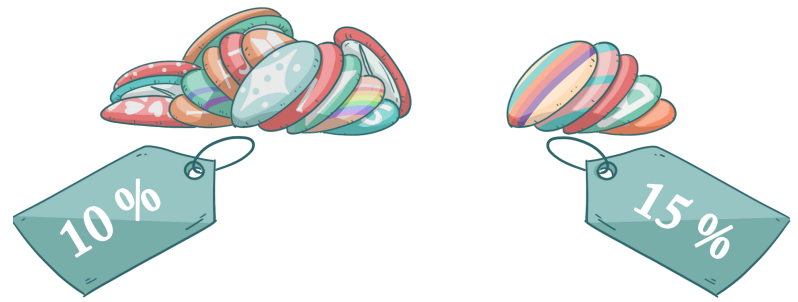
Boas instruções

O proprietário de uma empresa que fabrica e vende *bottons* gostaria de criar uma promoção para incentivar a compra de grandes quantidades, uma vez que isso agiliza a produção. Na sua loja, todos os *bottons* são vendidos pelo mesmo preço: R$ 2,00.

A primeira ideia foi oferecer um desconto de 10 % no valor total da compra caso o comprador adquirisse mais do que 100 unidades.

a) Com essa promoção, qual seria o valor de uma compra de 95 *bottons*? E de uma compra de 105 *bottons*?

Como você deve ter notado, se essa proposta de desconto for utilizada compras com poucas unidades acima de 100 podem ficar mais baratas do que compras que não qualificaram para o desconto.



Uma outra ideia do proprietário foi a seguinte: cada unidade além da centésima recebe 15 % de desconto no seu valor. Por exemplo, se alguém comprar 105 *bottons*, o comprador paga o preço normal para 100 deles e depois recebe 15 % de desconto no valor dos outros 5.

b) Com essa promoção, qual seria o valor de uma compra de 95 *bottons*? E de 105? E de 200 *bottons*?

c) Você acha que essa ideia evita o problema de compras com pouco mais de 100 *bottons* ficarem mais baratas do que compras com quantidades menores? Justifique a sua resposta.

O proprietário da empresa decide adotar essa segunda ideia, mas notou que ela cria um problema: o cálculo do valor final da compra não pode mais ser feito de maneira imediata na calculadora.

d) Descreva com as suas palavras como um funcionário deve proceder para calcular o valor de uma compra a partir da informação de quantos *bottons* foram comprados usando uma calculadora simples. Dê a sua resposta na forma de um bilhete que será lido por uma pessoa que você não irá encontrar pessoalmente.

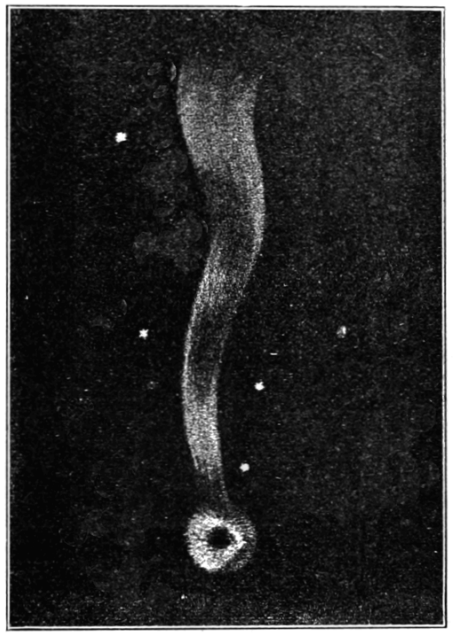
e) Troque o seu bilhete com um colega e discuta com ele se vocês seriam capazes de compreender como o preço de uma compra é calculado se vocês tivessem apenas lido o bilhete escrito por cada um.

Atividade 2

Uma resolução boa de repetir

O cometa Halley é um dos cometas de menor período do Sistema Solar, completando uma volta em torno do Sol a cada 76 anos. Na última ocasião em que ele ficou visível do planeta Terra, em 1986, várias agências espaciais enviaram sondas para coletar amostras de sua cauda e assim confirmar teorias sobre suas composições químicas.

Figura 1: Registro da passagem do cometa Halley em 1682



Fonte: Doolittle (1910)

a) Uma pessoa que nasceu em 2020 vai ter a oportunidade de avistar o cometa Halley pela primeira vez em que ano? E uma que venha a nascer no ano 2200? E no ano 3000?

b) Qual foi o ano em que o cometa Halley pode ter sido avistado pela primeira vez para uma pessoa que nasceu em 1900? E uma que nasceu em 1500?

c) Em uma folha de papel à parte, descreva como obter a resposta para questões como as anteriores para um amigo fictício que tenha terminado o Ensino Médio, mas não tenha resolvido essas questões. Tenha em mente que o objetivo não é explicar como resolver o problema, mas descrever um procedimento que permita ao seu colega obter respostas. Use texto, expressões matemáticas, esquemas visuais ou outros recursos que possam ajudar a deixar as suas instruções o mais claras possível.

d) Troque as instruções com um colega, e seguindo os passos que ele descreveu, obtenha o ano do primeiro possível avistamento do cometa Halley para alguém que tenha nascido nos anos: 2500, 1000 e 2290.

e) Antes de devolver a resolução para o seu colega, avalie-a considerando os seguintes aspectos: Ele seguiu a mesma estratégia que você? Na sua opinião, qual dos dois conjuntos de instruções é mais fácil de seguir e porquê?

Objetivos Específicos

Uma resolução boa de repetir

- Analisar problemas matemáticos visando a sua resolução do ponto de vista computacional.
- Compreender como sistematizar soluções algorítmicas usando linguagem textual e matemática.

Sugestões e discussões

Uma resolução boa de repetir

Organização da turma apesar do **d)** sugerir a troca de resoluções entre estudantes, as questões podem ser resolvidas individualmente.

Dificuldades previstas o problema não é difícil quando as questões **a)** e **b)** são resolvidas. A dificuldade pode surgir quando os estudantes tentarem descrever suas soluções, na questão **c)** , não pelo conteúdo matemático em si, mas por se tratar de algo que eles não estão acostumados a fazer.

Enriquecimento da discussão é comum estudantes não contemplarem dois casos especiais nas suas instruções: o primeiro ocorre quando o ano de nascimento é um ano de passagem do cometa, como 2062; o segundo é o próprio ano de 1986. A descrição não cobre esses casos intencionalmente, para que isso possa ser discutido com os estudantes de acordo com a coerência com o contexto e com a solução proposta.

Conexões este problema está relacionado com progressão aritmética e pode ser discutido nesses termos se o professor desejar.

Duração 2 aulas. Sugerimos que as questões **a)** e **b)** sejam resolvidas e discutidas com a turma toda antes de serem propostas as seguintes.

Fonte Olimpíada Brasileira de Informática.

Nota 2

Objetivos Específicos

Usando fluxogramas

- Compreender como sistematizar soluções algorítmicas usando fluxogramas.

Sugestões e discussões

Usando fluxogramas

Organização da turma esta atividade pode ser resolvida individualmente.

Dificuldades previstas de acordo com a BNCC, os estudantes já devem ter interagido com fluxogramas no Ensino Fundamental e como os fluxogramas que propomos são simples, não devem apresentar desafios. O formato e a cor das partes dos fluxogramas segue um padrão internacional, mas isso não é importante para os usos que faremos neste material.

Duração 2 aulas, considerando a discussão do **Organizando** seguinte.

Solução

Usando fluxogramas

a) R$ 120,00 e R$ 285,00.

b) 2366.

Organizando — Um novo jeito de resolver problemas

As atividades anteriores mostraram que não é simples criar um conjunto de instruções que resolve um determinado problema de modo que uma pessoa seja capaz de compreendê-las e segui-las sem precisar fazer mais perguntas ou pedir esclarecimentos. Por isso é importante que fique claro tanto o significado de cada instrução, quanto a ordem em que elas devem ser executadas.

No caso de criarmos instruções para um computador seguir, essa exigência fica ainda maior, pois o computador não consegue usar bom senso ou conhecimentos adicionais sobre a situação para interpretar as instruções ou tomar decisões que não estejam claras, caso isso seja necessário. Para um computador, todas as instruções dadas devem ser conhecidas por ele e o fluxo, ou seja, a ordem em que elas devem ser executadas, deve ser absolutamente claro na descrição.

Um conjunto finito de instruções bem definidas e com fluxo claro é chamado de algoritmo.

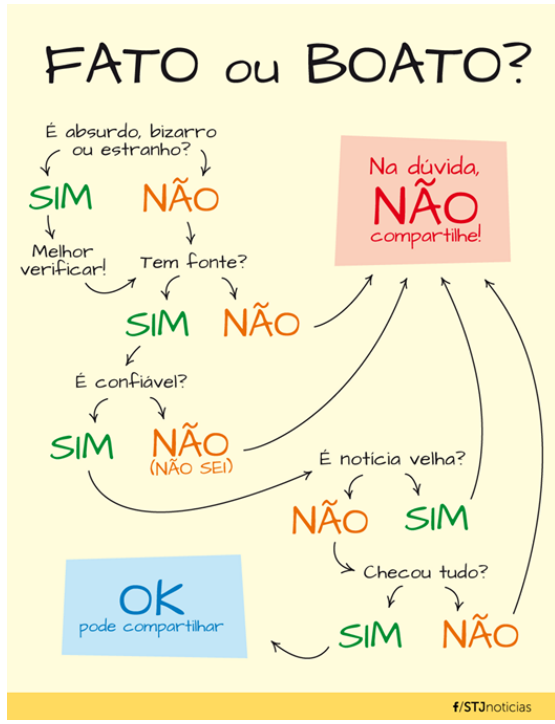
Explorando — A representação via fluxogramas

Ao longo deste material, vamos conhecer várias ferramentas que nos permitam representar um conjunto de instruções. Em última instância, estamos procurando desenvolver habilidades e conhecer ferramentas que nos permitam criar algoritmos, e algoritmos podem ser usados para criar programas e aplicativos que realizem certas tarefas para nós!

Uma dessas formas de representação é o **fluxograma**. Trata-se de uma forma de representar visualmente a maneira como as instruções (ou comandos) se relacionam e qual a ordem em que devem ser realizadas (ou executados).

Você já deve ter visto esquemas visuais como o mostrado a seguir. Este esquematiza o processo de decisão sobre a pertinência de compartilhar uma notícia.

Figura 2: Campanha do Superior Tribunal de Justiça (STJ)



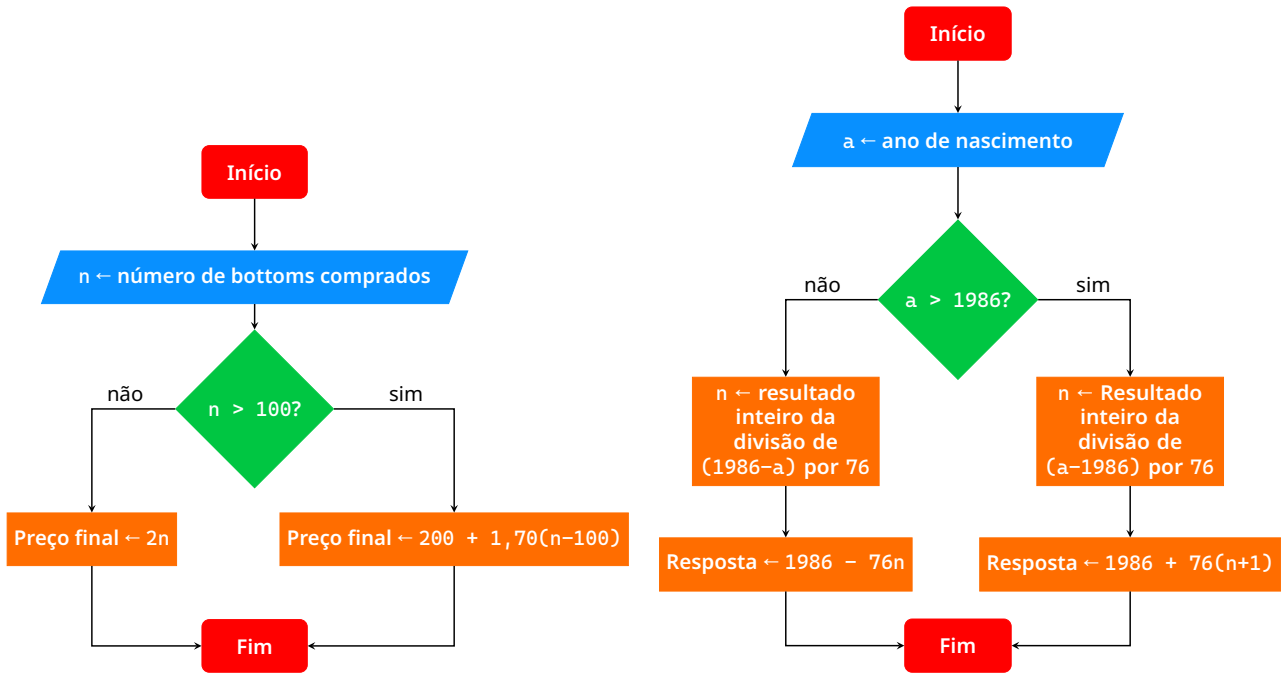
Fonte: Página do Facebook do STJ

Em algumas atividades deste material, faremos um uso mais específico e rigoroso desse recurso, mas a intenção é a mesma da imagem anterior: representar visualmente um processo com múltiplas etapas e ações.

Atividade 3

Usando fluxogramas

A imagem a seguir mostra dois fluxogramas que resolvem as atividades anteriores. Leia-os com atenção e, se necessário, volte as Atividades 1 e 2 para relembrar.



a) Utilize o fluxograma da esquerda para determinar qual seria o valor de uma compra de 60 e outra de 150 bottons

b) Utilize o fluxograma da direita para determinar em que ano o cometa Halley poderá ser avistado pela primeira vez por alguém que tenha nascido no ano de 2300.

Organizando — A representação via fluxogramas

Os fluxogramas são uma maneira de representar algoritmos que fazem uso de um arranjo visual para deixar claro o fluxo, ou seja, a sequência em que comandos devem ser realizados. Embora seja conveniente em muitas situações, eles não são recomendados para algoritmos muito complexos ou longos.

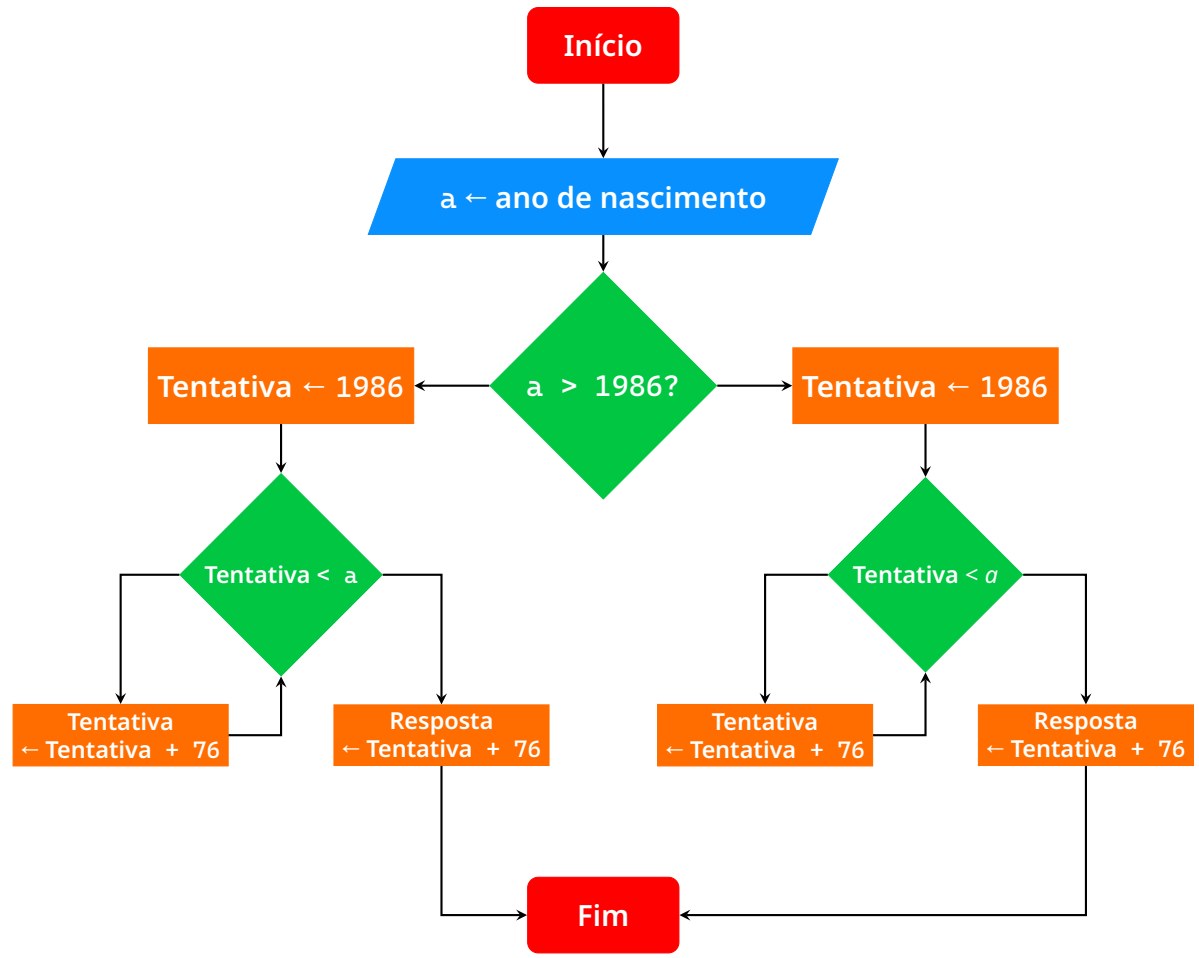
Mesmo assim, eles ainda podem ser uma forma rápida para organizar o seu raciocínio quando estiver pensando em um algoritmo. Por isso, consideramos que vale a pena mostrar mais um exemplo que ilustra uma situação não mostrada anteriormente: um processo repetitivo.

Considere a seguinte solução (descrita na forma textual) para a Atividade 2:

Se o ano de nascimento for maior que 1986, então some 76 a 1986 até que o resultado seja maior ou igual ao ano de nascimento. Quando isso ocorrer, o resultado da última soma é a resposta. Se o

ano de nascimento for menor do que 1986, então subtraia 76 a 1986 até que o resultado seja menor que ano de nascimento. Quando isso ocorrer, some 76 ao resultado da última soma e essa será a resposta.

Como fluxograma, ela pode ser representada como mostrado a seguir.



Dois detalhes merecem atenção neste fluxograma.

O primeiro deles se refere aos termos **`tentativa`**, **`a`** e **`resposta`**. Concretamente, podemos pensar neles como nomes que damos a certos valores que serão usados ao longo na nossa resolução. Em computação, esses termos são chamados de **variáveis** e representam espaços na memória do computador que foram reservados para o seu algoritmo armazenar alguma informação. Vamos considerar o caso do variável **`tentativa`**. Ela é usada para armazenar em qual ano estamos à medida que somamos ou subtraímos 76 até encontrar a resposta. O ato de armazenar uma informação em uma variável nesse fluxograma é representada por ←, como em tentativa ← tentativa + 76. Esse comando diz ao computador para calcular o valor do lado direito de ← (somando 76 ao valor atual da variável **`tentativa`**) e então guardar na variável indicada à esquerda do ← (**`tentativa`**) o resultado. Com isso, o resultado do cálculo é escrito "em cima" do valor que estava armazenado até antes da execução desse comando na variável em questão.

O segundo detalhe é a ocorrência de ciclos: esses ciclos representam processos iterativos, ou seja, que envolvem repetição sucessiva, são repetitivos. Focando no ciclo que está mais à esquerda, ele diz o seguinte: cheque se **`tentativa`** é menor do que **`a`**, se não for, subtraia 76 e volte para a checagem. Isso faz com que o algoritmo repita essa subtração até que o resultado encontrado seja menor do que **`a`**. No ciclo do lado direito, ocorre o mesmo mas com adições (veja que o lado direito do fluxograma é acionado quando **`a`** é maior do que 1986).

Processos repetitivos são muito comuns em algoritmos e voltaremos a eles em breve.

Para que um computador seja capaz de seguir instruções que realizem alguma tarefa, é necessário que o algoritmo seja escrito em uma linguagem de programação. Uma linguagem de programação nada mais é do que uma maneira muito estruturada de descrever um algoritmo. A necessidade de clareza e estrutura para que um computador consiga interpretar um algoritmo faz com que certos elementos aparentemente estranhos sejam usados para organizar o fluxo das instruções.

Existem centenas de linguagens de programação em uso na atualidade, cada uma com certas vantagens e desvantagens e com diferentes níveis de popularidade. Atualmente, Python é uma linguagem muito utilizada em vários contextos e conta com muitos materiais para autoestudo na internet. Javascript é um outro exemplo, mas seu uso é mais comum em páginas de internet. A linguagem C é muito usada, mas normalmente para programas mais técnicos que precisam de alto rendimento.

A imagem a seguir mostra dois algoritmos escritos em uma linguagem de programação chamada **Portugol**, que tem seus comandos em português. Apesar de não ser usada comercialmente, essa é a linguagem que sugerimos para você neste capítulo.

As instruções descritas abaixo em Portugol são as mesmas descritas anteriormente com fluxogramas na Atividade 3. Leia com calma e tente compreender como as representações se relacionam.

```
programa {
  funcao inicio() {
    inteiro n
    real preco
    leia(n)
    se (n>100) {
      preco = 200 + (n-100)*1.70
    }
    senao {
      preco = 2*n
    }
    escreva(preco)
  }
}
```

```
programa {
  funcao inicio() {
    inteiro a, n, resposta
    leia(a)
    n = (a-1986) / 76
    se (a>1986) {
      n=n+1
    }
    resposta = 1986 + n*76
    escreva(resposta)
  }
}
```

Embora não seja necessário entender todos os detalhes que fazem partes desse algoritmo escrito em Portugol, como o uso dos símbolos { e }, um detalhe é importante: o uso do símbolo =.

Sugestões e discussões

Instruções para um computador

Esta seção traz pela primeira vez um algoritmo escrito em Portugol. Embora não seja proposta como uma atividade, pode ser interessante para o professor dedicar ao menos 1 aula para a discussão dessa seção com os estudantes caso deseje, mais adiante, usar essa linguagem de programação para resolver as atividades propostas.

Não é imprescindível que os estudantes entendam a função de todas as linhas dos algoritmos mostrados, pois essa familiarização pode ser construída ao longo das próximas atividades, que também trarão algoritmos escritos em Portugol como informação complementar.

Se houver disponibilidade, pode ser interessante rescrever esses algoritmos em um computador ou celular, fazer pequenas alterações e executá-las para conferir os seus efeitos no funcionamento do algoritmo. Um bom ponto de partida é alterar os valores numéricos ou acrescentar novos comandos `escreva`.

Em Portugol, assim como na maioria das linguagens de progamação, o símbolo = não é usado para expressar uma igualdade entre dois termos (como em uma equação), mas sim para representar a ação de atribuição: a variável à esquerda do sinal deve receber o valor à direita. Esse é o significado do símbolo ← nos fluxogramas que discutimos na atividade anterior.

Por causa disso, é comum vermos expressões como "`t=t+76`" em algoritmos escritos em linguagens de programação.

O sinal de igual — **Observação**

Em um contexto matemático, o sinal de igual nessa expressão significaria uma equivalência entre os valores do seu lado direito e esquerdo. Nesse caso específico, a expressão é uma equação sem solução (pois não existe valor de `t` que satisfaça a igualdade proposta).

Em linguagens de programação, essa expressão significa "calcule o valor da expressão ao lado direito do símbolo = e armazene na variável indicada no lado esquerdo".

Por isso, tenha muito cuidado com o contexto ao usar o símbolo =, pois ele pode ser interpretado de maneira incorreta se o contexto não estiver claro.

Ao longo das próximas atividades vamos incentivar o uso de Portugol para escrever algoritmos e, pouco a pouco, vamos esclarecer o significado de todos os elementos que apareceram nas imagens anteriores.

O Portugol Studio é um software que interpreta algoritmos escritos em Portugol e pode ser instalado no seu computador (http://lite.acad.univali.br/portugol) ou no seu celular. Você também pode acessá-lo diretamente do navegador em http://portugol-webstudio.cubos.io. Caso você não esteja familiarizado com esse tipo de recurso, sugerimos o vídeo http://youtu.be/6OIADpFImtc como ponto de partida.

Para o professor: Variáveis e operações

Nesta seção, a resolução das atividades propostas dependem de maneira bastante explícita de cálculos feitos com valores dados. A intenção aqui é colocar em foco o conceito de variável no sentido computacional, que é diferente do sentido matemático (como discutido no **Organizando** anterior) e como efetuar cálculos aritméticos com elas.

A intenção do **Organizando** que encerra esta seção é salientar uma característica de linguagens de programação: a limitação dos comandos disponíveis. Na verdade, essa é uma limitação que não vale apenas para linguagens de programação, mas para qualquer software. Conhecer as limitações e como elas podem ser contornadas nos torna usuários mais hábeis. No caso específico desta seção, a limitação refere-se ao fato de Portugol não possuir um comando “arredonde para cima”, portanto, isso precisa ser implementado a partir de outros comandos disponíveis na linguagem (combinando o cálculo do resto de uma divisão inteira com um condicional).

O que importa aqui é a consciência da limitação e não o domínio sobre como implementar o arredondamento para cima em uma linguagem de programação. Além disso, essa discussão traz de volta o conceito de condicional (o comando `se` do algoritmo mostrado) que será o foco da próxima seção.

2 Variáveis e operações

Dois recursos são muito comuns na criação de algoritmos, não importando se descrevemos na forma textual, com fluxograma ou em uma linguagem de programação: condicionais e repetições. Além desses dois recursos, o conceito de variável é especialmente importante quando pensamos em linguagens de programação.

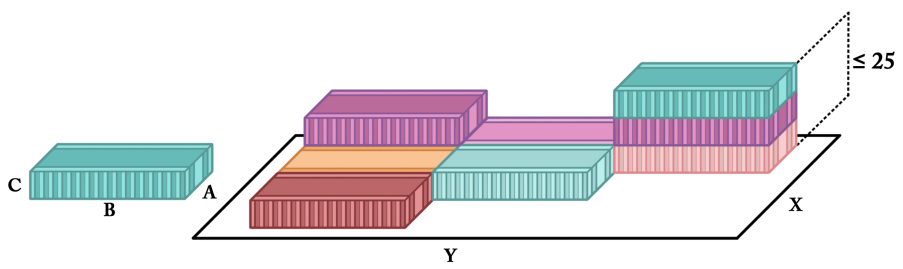
Já conhecemos esses três elementos nas atividades anteriores, mas nas atividades a seguir vamos ter a chance de explorá-los com mais profundidade.

Explorando — Variáveis e operações

Atividade 4

Contêineres

O transporte em grandes navios de carga se dá através de contêineres, que são caixas metálicas grandes dentro das quais as empresas acomodam seus produtos. A maneira como os contêineres são colocados nos navios costuma ser determinada pelo sistema de carregamento disponível nos portos. Imagine um porto que, por restrição nos equipamentos disponíveis, carregue seus contêineres sempre alinhados da mesma forma, como mostrado na figura abaixo.



A administradora desse porto está com problemas para determinar quantos contêineres de dimensões A, B e C (em metros) podem ser colocados em um navio que tenha área de carregamento nas dimensões X e Y (em metros). Observe que a dimensão A dos contêineres deve ser carregada paralelamente à dimensão X no navio, o mesmo ocorre para as dimensões B e Y.

Além dessas restrições, há uma limitação de altura que diz que a pilha de contêineres não pode ultrapassar 25 metros de altura.

a) Quantos contêineres de dimensão (em metros) $A = 3$, $B = 4$, $C = 1$ cabem em um navio com área de carregamento com dimensões $X = 30$ e $Y = 60$?

b) Quantos contêineres de dimensão (em metros) $A = 4$, $B = 8$, $C = 2$ cabem em um navio com área de carregamento com dimensões $X = 30$ e $Y = 60$?

c) Descreva, com suas palavras, como determinar quantos contêineres de dimensões A, B e C cabem em um navio com espaço de carregamento igual a X e Y.

Objetivos Específicos

Contêineres

- Compreender como sistematizar soluções algorítmicas usando linguagem matemática.
- Compreender os conceitos básicos de uma linguagem de programação: variáveis.

Sugestões e discussões

Contêineres

Organização da turma sugerimos a resolução desta e da próxima atividade conjuntamente e, especialmente por causa da segunda atividade, recomendamos a organização da turma em duplas.

Duração 3 aulas para a resolução desta e da próxima atividade e discussão de ambas ao final.

Comentários para o Portugol a implementação da solução a este problema em Portugol é muito próxima ao que os estudantes devem descrever aqui, pois ao fazermos a divisão entre dois números inteiros e armazenarmos em uma variável inteira, o Portugol armazena o quociente da divisão inteira (com resto), causando o efeito de arredondar para baixo.

Conexões você pode discutir o conceito de volume de paralelepípedos e relembrar o significado da divisão em termos de medida.

Solução

Contêineres

a) 3750

b) 588

c) Calculamos a quantidade de conteineres que cabem em cada uma das dimensões: $n_1 = \frac{X}{A}$, $n_2 = \frac{Y}{B}$ e $n_3 = \frac{25}{C}$, sempre considerando a parte inteira do quociente. O total de conteineres que podem ser levados será: total $= n_1 \times n_2 \times n_3$.

Objetivos Específicos

Um professor em cada van

- Compreender como sistematizar soluções algorítmicas usando linguagem matemática.
- Compreender os conceitos básicos de uma linguagem de programação: variáveis.

Sugestões e discussões

Um professor em cada van

Organização da turma em duplas.

Duração 3 aulas para a resolução desta atividade e da anterior e discussão de ambas ao final.

Dificuldades previstas a interpretação do enunciado da questão pode gerar algumas dúvidas, por isso dê tempo para que os alunos compreendam matematicamente o problema antes de partir para os itens de natureza computacional.

Comentários para o Portugol a implementação desta solução é um pouco mais difícil do que a descrição verbal por causa do procedimento de "arredondar para cima", que não é oferecido de forma direta em Portugol, nem na maioria das linguagens de programação. Na resolução da atividade essa dificuldade não deve emergir, mas ela será aprofundada na seção **Organizando** logo a seguir.

Solução

Um professor em cada van

a) 16 e 5

b) Calcule n dividido por 13 e arredonde o resultado para cima se houver parte decimal

c) Sim, Não

d) Calcule n dividido por 13 e arredonde o resultado para cima se houver parte decimal. Esse resultado, vamos chamar de p, é igual ao número de vans necessárias e igual ao número mínimo de professores. O número máximo de professores, chamemos de max, é dado por $\max = 14p - n$

Um professor em cada van

Uma escola costuma organizar passeios com frequência e para levar os alunos conta com vários motoristas de vans. Essas vans possuem 14 lugares, além do assento do motorista. Por questão de segurança, a direção da escola exige que sempre haja um professor em cada van, não importando o número de alunos. Felizmente, a escola tem contato com muitos motoristas de vans.

a) Quantos professores serão necessários para acompanhar os estudantes em um passeio em que 200 estudantes desejem participar? E em um passeio em que 65 estudantes desejem participar?

b) Descreva com suas palavras, e de forma clara, como obter a quantidade mínima de professores necessários para acompanhar um passeio em que n estudantes estejam interessados em participar.

Em algumas ocasiões, porém, há mais professores dispostos a acompanhar os estudantes do que o necessário. Em geral, a direção gosta dessa situação pois cada professor fica responsável por menos estudantes. Porém, já aconteceu de não haverem lugares disponíves nas vans para os professores adicionais, e a escola não pretende contratar vans adicionais.

c) Se houverem 200 estudantes interessados e 20 professores disponíveis, haverá lugar para todos os professores nas vans sem que seja necessário contratar vans adicionais? E se forem 75 estudantes e 10 professores?

d) Descreva com suas palavras e de forma clara como descobrir o número máximo de professores que podem acompanhar n estudantes em um passeio sem que seja necessário contratar mais vans.

Para resolver as duas atividades anteriores você deve ter utilizado apenas algumas operações: soma, subtração, multiplicação, divisão e arredondamentos. As quatro primeiras estão diretamente disponíveis em qualquer linguagem de programação e são suficientes para resolver a Atividade 4. Vejamos como seria o algoritmo para resolvê-la em Portugol:

```
1  programa {
2    funcao inicio() {
3      inteiro a, b, c, x, y, capacidade
4      leia(a)
5      leia(b)
6      leia(c)
7      leia(x)
8      leia(y)
9      capacidade = (x/a)*(y/b)*(25/c)
10     escreva(capacidade)
11   }
12 }
```

Veja que na linha 9 é realizada a divisão de cada uma das dimensões do navio pelas dimensões de cada contêiner para determinar quantos contêineres cabem em cada dimensão e, então, as quantidades são multiplicadas para determinar a capacidade do navio.

Vale a pena salientar um aspecto sobre o cálculo do linha 9 acima: como estamos lidando com variáveis inteiras, o Portugol entende que a divisão a ser realizada também é uma divisão inteira, ou seja, sem parte decimal no quociente. Assim, o resultado de `x/a` é a quantidade (inteira) de contêineres que cabem na dimensão `x` do navio, como queríamos.

Já na Atividade 5, você deve ter precisado de uma ação além das quatro operações: arredondamento. O recurso de arredondar para cima é necessário para descobrir o número mínimo de professores necessários para acompanhar a turma.

Por exemplo, se 30 alunos quiserem participar de uma viagem, calculamos $30/13 \approx 2,3$. Logo, precisamos de mais de 2 professores, ou seja, 3. Matematicamente, dizemos que obtivemos o menor número inteiro maior ou igual ao resultado da divisão. Informalmente, dizemos que estamos "arredondando para cima". Porém, o Portugol (e outras linguagens de programação) não entende a instrução "arredonde para cima" ou "obtenha o menor número inteiro maior ou igual". Então, como poderíamos descrever para o computador esse processo?

Para fazer isso, podemos utilizar duas operações que a maioria das linguagens de programação oferece: quociente da divisão (representada por `/` e resto da divisão entre dois números inteiros (representada por `%`).

- A operação `/` retorna um número real igual ao quociente da divisão se os valores envolvidos forem números reais, por exemplo, `7.2/4.8` terá como resultado 1.5. Caso os valores envolvidos sejam números inteiros, ela retorna a parte inteira do quociente, por exemplo, `9/4` terá como resultado 2.

- Já a operação `%` só pode ser realizada entre dois números inteiros e retorna o resto da divisão, por exemplo, `9%4` terá como resultado 1.

No caso do problema posto na atividade, devemos adicionar uma van se a divisão do número de interessados por 13 tiver resto maior do que zero. Em Portugol, uma forma de resolver o problema é essa:

Sugestões e discussões

Variáveis e operações

O professor pode aproveitar seções como essa para promover discussões com os estudantes que os familiarizem com a linguagem de programação progressivamente. Assim, eles estarão familiarizados com a sintaxe geral do Portugol antes mesmo de chegarem às atividades que pedem a criação de um algoritmo nessa linguagem.

O quadro **Para Refletir** a seguir tem justamente a intenção incentivar essa familiarização dos estudantes com o Portugol.

Note que os dois comandos escreva utilizados trazem não apenas os valores finais que interessam, mas também um pouco de texto (escritos entre aspas) explicando o que significa cada valor.

Esse recurso pode ser usado para explicar as saídas de um algoritmo, ou seja, os valores que o algoritmo escreve após processar os valores dados (chamados de entrada), como foi feito nesse algoritmo.

Já no algoritmo que resolve a Atividade 4, faria mais sentido explicar as entradas, uma vez que são cinco valores. Isso poderia ser feito com um comando `escreva` antes de cada `leia` com algum conteúdo do tipo *Digite a primeira dimensão do contêiner*.

```portugol
programa {
  funcao inicio() {
    inteiro n, interessados,maximoprofs
    leia(interessados)
    n = interessados/13
    se (interessados%13 > 0) {
      n=n+1
    }
    maximoprofs = 14*n-interessados
    escreva("São necessários no mínimo ", n)
    escreva(" e podem ir no máximo ", maximoprofs)
  }
}
```

Veja que esse algoritmo primeiro considera que o número mínimo de professores (**n**) é igual ao resultado inteiro da divisão do número de interessados por 13. Depois, se o resto da divisão por 13 for maior do que zero, o algoritmo soma 1 ao número mínimo de professores.

Para refletir

Agora que você sabe como realizar algumas operações em Portugol, você já consegue criar algoritmos que realizem cálculos matemáticos simples a partir de um conjunto de valores dados. Você pode, por exemplo, criar um algoritmo que leia três notas e calcule a média entre elas. Que tal tentar?

Para o professor: Condicionais

Esta seção foca no uso de estruturas condicionais. Apesar de já terem aparecido nas seções anteriores, a atividade a seguir propõe uma situação em que é necessário utilizar condicionais de forma mais sofisticada do que nas anteriores. A sofisticação não vem da dificuldade matemática da questão, mas pela atenção necessária para que as instruções fiquem claras.

A intenção do **Organizando** desta seção é refletir um pouco sobre algoritmos com fluxos mais complexos devido ao uso de condições encadeadas. Isso é muito comum em algoritmos e por isso é importante compreender como criá-los e como interpretá-los.

Embora a seção seja composta por uma única atividade, o **Organizando** da seção traz muitos elementos que devem ser discutidos com os estudantes, permitindo que eles relacionem o conceito de condicional com fluxogramas e com Portugol.

3 Condicionais

Condicionais são um componente central em qualquer linguagem de programação e, na verdade, em todos os recursos computacionais que utilizamos. Elas servem para que o computador realize ações diferentes de acordo com o caso com que estiver lidando. Por exemplo, na Atividade 1 haviam dois casos que exigiam um conjunto de comandos diferentes para que se calculasse o preço final da compra: o caso com até 100 bottons e o caso com mais de 100 bottons.

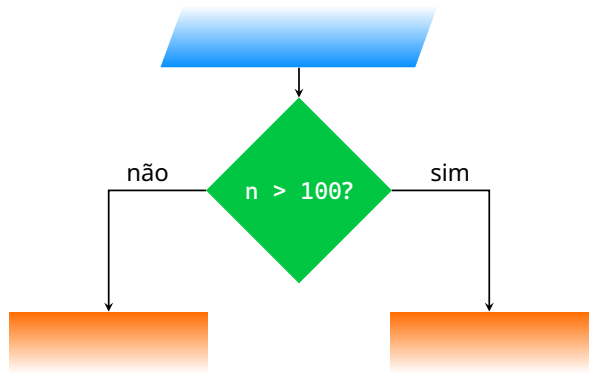
No caso das planilhas eletrônicas, por exemplo, elas podem ser usadas para que uma célula mostre um conteúdo que depende do conteúdo de uma outra célula. Um exemplo simples é mostrado a seguir. Imagine que um professor tenha uma planilha em que registra as notas dos seus estudantes em duas provas. Além de calcular a média das duas notas, ele deseja que a planilha mostre se cada estudante está, ou não, de recuperação de acordo com o seguinte critério: se a média for menor do que 6 então o estudante está de recuperação, senão está aprovado.

	A	B	C	D
1	**Prova 1**	**Prova 2**	**Média**	**Situação**
2	6,5	4,5	=(A2+B2)/2	=SE(C2<6;"Recuperação";"Aprovado")

	A	B	C	D
1	**Prova 1**	**Prova 2**	**Média**	**Situação**
2	6,5	4,5	5,5	Recuperação

Note que o comando descrito na frase “se a média for menor do que 6, então o estudante está de recuperação, senão está aprovado” é composto por três partes: uma condição (C2<6), o que deve ser feito se a condição for verdadeira (escrever a palavra “Recuperação” na célula) e o que deve ser feito se a condição for falsa (escrever “Aprovado” na célula). Isso está condensado no conteúdo da célula D2 mostrada anteriormente.

Nas atividades anteriores, usamos o mesmo raciocínio para resolver o problema do cometa Halley (se o ano de nascimento fosse maior do que 1986 a resolução seguia por um caminho, se fosse menor seguia por outro) e o problema sobre o número de professores (se houvesse resto na divisão, era necessário contratar mais uma van). Quando representamos um algoritmo na forma de um fluxograma, condicionais são representadas como bifurcações no fluxo das instruções, de modo que um ou outro lado deve ser seguido conforme a condição estipulada.



A próxima atividade vai exigir um uso mais sofisticado de condicionais do que as atividades que resolvemos até agora.

Objetivos Específicos

Cubos conectados

- Compreender como sistematizar soluções algorítmicas usando fluxogramas ou linguagens de programação.
- Compreender os conceitos básicos de uma linguagem de programação: condicional.

Sugestões e discussões

Cubos conectados

Organização da turma sugerimos que os estudantes resolvam essa atividade em grupo, pois ela exige não apenas a compreensão da situação proposta como também atenção aos detalhes na elaboração da solução do **e)** e esmero na apresentação dessa solução.

Duração 2 aulas.

Dificuldades é comum os estudantes não representarem ou descrevem a relação entre as diferentes condições de maneira adequada. Os condicionais que resolvem este problema podem estar um dentro do outro ou um depois do outro, e o fluxo entre os comandos são diferentes nesses casos, como será discutido na seção **Organizando** a seguir.

Conexões o início desta atividade cria a oportunidade para uma revisão sobre volume de paralelepípedos antes que os estudantes se engajem com as questões da atividade.

Fonte esta questão foi inspirada em uma questão da Olimpíada Brasileira de Informática.

Solução

Cubos conectados

a) 500 litros, 400 litros, 0 litros

b) 500 litros, 500 litros, 250 litros

c) 600 litros, 600 litros, 600 litros

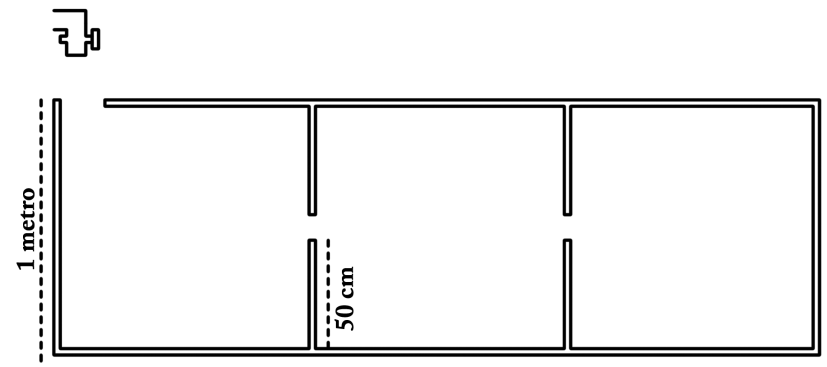
d) Existem muitas maneiras de resolver esse problema. Duas delas são discutidas na seção **Organizando** a seguir. Sugerimos que as soluções dadas pelos estudantes sejam comparadas levando em conta as discussões propostas a seguir.

Atividade 6

Cubos conectados

Uma empresa que produz detergente transporta o produto bruto em cubos metálicos com lados iguais a 1 metro. Para facilitar o enchimento destes cubos, eles possuem uma abertura pequena, como mostrado na figura abaixo, que permite conectar três cubos durante o enchimento. Assim, à medida que o líquido é despejado, ao atingir a altura da abertura, o líquido começa a escoar para o segundo cubo. Se houver líquido suficiente, o terceiro cubo também poderá receber uma parte.

Por exemplo, se forem injetados 600 litros de detergente, os cubos ficarão como mostrado a seguir, com 500 litros no primeiro, 100 litros no segundo e 0 no terceiro.



a) Quantos litros haverá em cada cubo se forem injetados 900 litros de detergente?

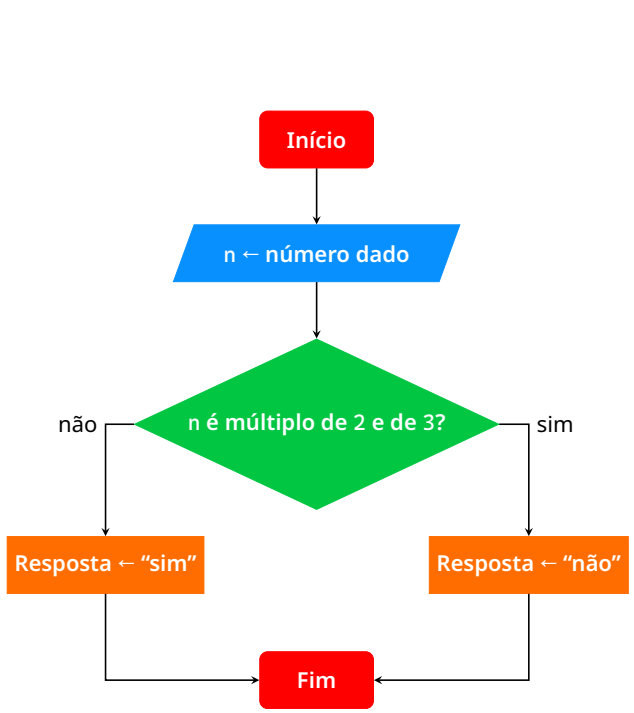
b) Quantos litros haverá em cada cubo se forem injetados 1250 litros de detergente?

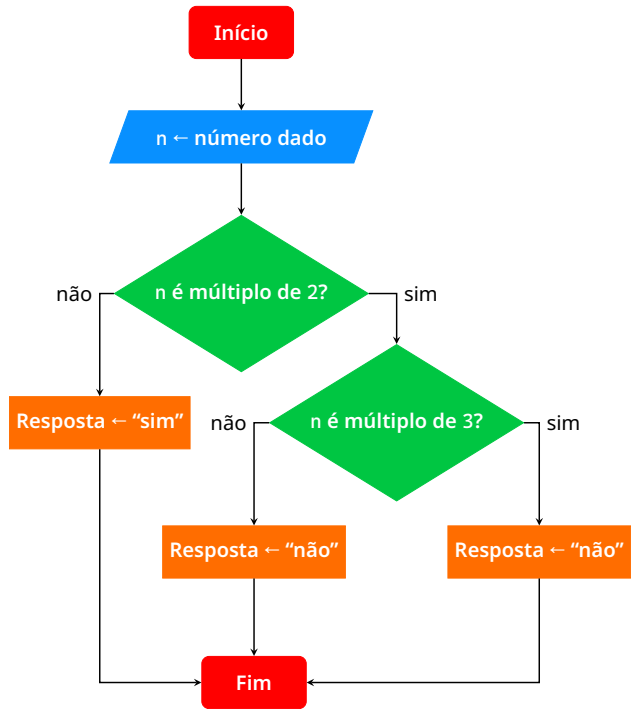
c) Quantos litros haverá em cada cubo se forem injetados 1800 litros de detergente?

d) Quantos litros haverá em cada cubo se forem injetados 2400 litros de detergente?

e) A empresa quer que você crie um algoritmo que permita saber qual é o volume de detergente em cada um dos três cubos quando uma quantidade N (em litros) é despejada no conjunto. Apresente a sua solução de maneira esquemática, isto é, sem usar muito texto.

Leia com atenção os dois fluxogramas abaixo. Ambos foram criados para determinar se um número dado é múltiplo de 6 a partir do seguinte resultado: um número é múltiplo de 6 se for múltiplo de 2 e de 3.





Para refletir

Os dois fluxogramas são visualmente diferentes, mas o que você pode dizer sobre os resultados que cada um deles produz para diferentes valores de **n**?

Veja que no fluxograma da esquerda, temos uma única condição e ela testa, de uma só vez, se **n** é múltiplo de 2 e se **n** é múltiplo de 3. No fluxograma da direita temos duas condições e a segunda está "dentro" da primeira, sendo acionada apenas se a primeira for verdadeira.

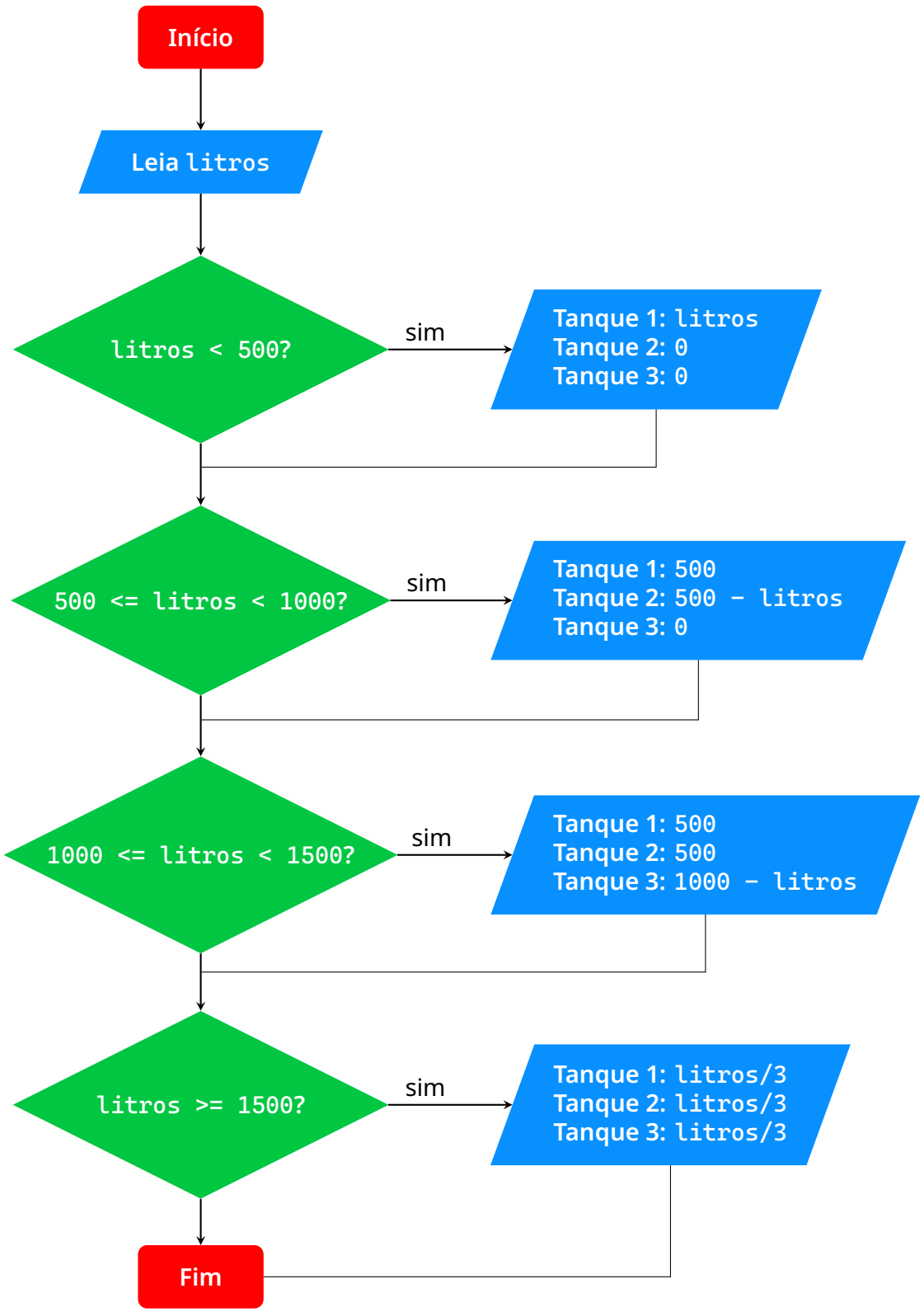
Apesar de estruturalmente diferentes, ambos produzem os mesmos resultados para qualquer valor de **n**. Como ambas estão corretas, a escolha por uma ou outra solução deve ser feita por outros critérios, como a clareza para o leitor, os recursos disponíveis na linguagem de programação que você esteja usando ou até mesmo considerações sobre a eficiência de cada algoritmo em termos de velocidade de processamento computacional (mas essa discussão é muito mais avançada do que os nossos objetivos no momento).

Agora, vejamos duas soluções para a atividade Cubos Conectados representadas como um fluxograma e como um algoritmo em Portugol.

Apesar de não se tratar de uma atividade propriamente dita, pode ser interessante dedicar uma aula toda ao que é discutido nesta seção, tanto no que diz respeito às diferenças entre as duas soluções quanto às diferentes representações (fluxograma e Portugol) de um mesmo algoritmo.

Um problema simples, mas interessante, que pode ser usado tanto para avaliar o trabalho desenvolvido nesta seção quanto para estender a discussão é o algoritmo para determinar se uma pessoa é maior de idade a partir do dia, mês e ano de seu nascimento e da data de hoje.

Note que se a diferença entre os anos for maior do que 18 não é necessário checar o mês. Também não é necessário checar o dia do nascimento se o mês do 18° aniversário já passou. Construir e discutir o algoritmo para este problema, seja na forma de um fluxograma quanto escrito em Portugol, pode ser bastante proveitoso.



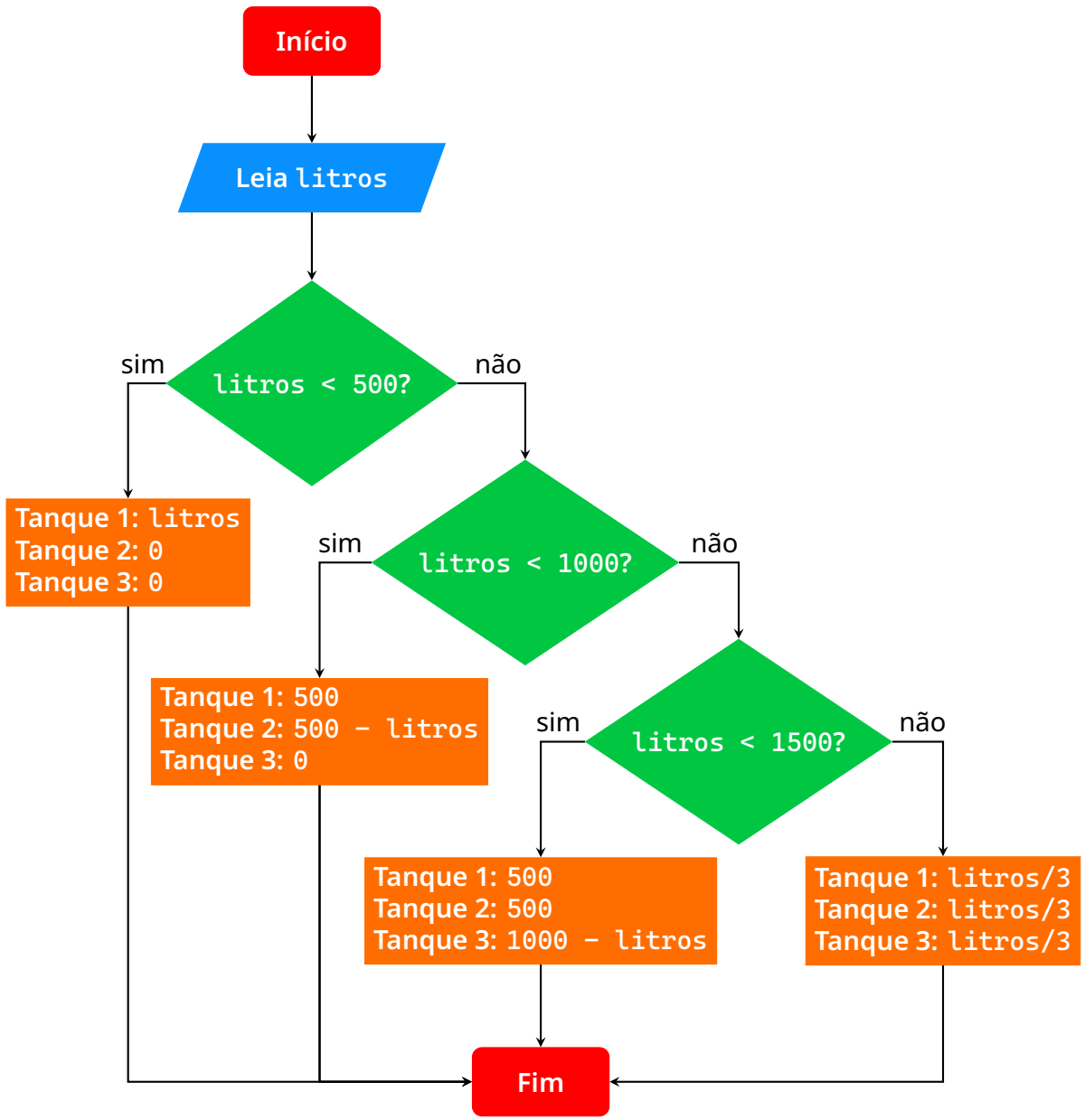
Nesse fluxograma, as quatro condições são sempre analisadas, não importando se alguma delas já foi satisfeita e produziu a resposta corretamente. O algoritmo em Portugol que reproduz esse comportamento esta colocado a seguir:

```
1 programa {
2   funcao inicio() {
3     real litros
4     leia(litros)
```

```
    se (litros<500) {
      escreva(litros+", 0, 0")
    }
    se (litros>=500 e litros<1000) {
      escreva("500, " + (litros-500) + ", 0")
    }
    se (litros>=1000 e litros<1500) {
      escreva("500, 500, " + (litros-1000))
    }
    se (litros>=1500) {
      escreva((litros/3) + ", " + (litros/3) + ", " + (litros/3))
    }
  }
}
```

Os quatro comandos `se` que compõem o código são independentes um do outro e o algoritmo, ao ser executado, checará as quatro condições.

Uma outra opção é mostrada no fluxograma a seguir:



Note que nesse fluxograma, as condições estão encadeadas: se o total de litros colocados for menor do que 500, o fluxo do algoritmo passa pelo bloquinho azul mais à esquerda e depois já vai para o fim, sem sequer considerar as demais condições. Dessa forma, o algoritmo evita checagens desnecessárias.

O algoritmo em Portugol que reproduz esse comportamento esta colocado a seguir:

```
programa {
  funcao inicio() {
    real litros
    leia(litros)
    se (litros<500) {
      escreva(litros+", 0, 0")
    }
    senao {
      se (litros<1000) {
        escreva("500, " + (litros-500) + ", 0")
      }
      senao {
        se (litros<1500) {
          escreva("500, 500, " + (litros-1000))
        }
        senao {
          escreva(litros/3+" "+litros/3+" "+litros/3)
        }
      }
    }
  }
}
```

Veja que agora todo comando `se` é seguido por um `senao`. Isso faz com que o algoritmo economize algumas checagens, pois os comandos que estão dentro de um `senao` só são executados se a condição do `se` ao qual ele está ligado não for satisfeita.

Ainda há outras formas corretas e diferentes de resolver esse problema, mas o importante neste momento é ter clareza de que as soluções acima são equivalentes, apesar de estruturadas de forma diferente.

Para o professor: Repetições

Esta seção é focada em estruturas de repetição, como o `enquanto`, que já usamos em problemas anteriores. O objetivo aqui é salientar o poder desse recurso para realizar tarefas que poderiam ser muito monótonas, e sujeitas a erro, se fossem realizadas à mão.

Além disso, essa é a primeira seção em que as atividades pedem explicitamente para que se escreva os algoritmos em Portugol. Espera-se que as discussões sugeridas nas seções anteriores já tenham promovido alguma familiaridade dos estudantes com essa linguagem de programação.

Sugerimos que se faça a leitura e interpretação das duas representações discutidas no **Explorando** com os estudantes, tendo certeza de que eles compreendem dois aspectos importantes em uma estrutura de repetição: a condição (m<10, nesse caso) e a delimitação dos comandos que são repetidos (delimitados pelas chaves em Portugol e pela sequência que parte da condição e volta para ela no fluxograma).

As duas atividades propostas abordam conteúdos com grande potencial do ponto de vista matemático: números primos e uma sequência numérica (chamada de sequência de Collatz) que envolve uma conjectura ainda em aberto.

A intenção do exemplo mostrado no **Organizando** é promover uma discussão, através de um algoritmo simples, sobre duas práticas comuns em estruturas de repetição: o uso de uma variável como contador e a passagem de valores entre variáveis.

A segunda atividade abre portas para muitos outros exemplos interessantes envolvendo sequências numéricas. Se seus estudantes estiverem com dificuldades, pode ser um bom momento para explorar sequências numéricas diversas e fixar alguns conceitos tratados até aqui.

4 Repetições

Repetir muitas vezes uma mesma ação é algo considerado cansativo para a maioria das pessoas. Porém, para muitas tarefas isso é necessário, e esse é um dos motivos que fazem dos computadores uma ferramenta tão útil: eles podem repetir comandos sem se cansar, sem perder a motivação ou a atenção e sem se distrair a ponto de cometerem erros.

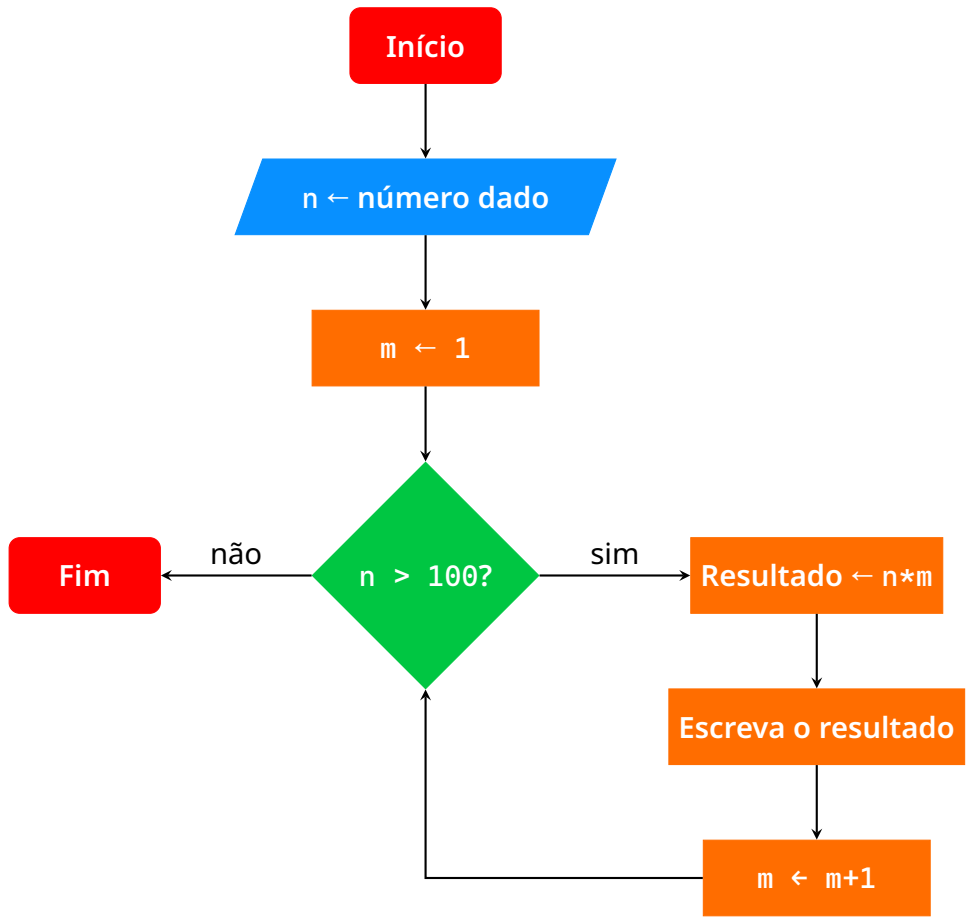
Historicamente, essa foi a motivação de diversos cientistas que, a partir do século 17, começaram a propor máquinas que fossem capazes de realizar cálculos aritméticos e que podem ser consideradas as avós dos computadores modernos.

Você Sabia?

As primeiras máquinas concebidas para realizar cálculos eram puramente mecânicas, ou seja, baseadas em engrenagens e encaixes físicos. Pesquise na internet pelas máquinas desenvolvidas pelos matemáticos Blaise Pascal e Gottfried Leibniz.

Nas próximas atividades, vamos lidar com problemas que envolvem muitas repetições e que ainda são muito relevantes na atualidade. Por isso, vamos aprender como criar algoritmos que realizem essas repetições por nós.

Vamos começar com um algoritmo simples que contém um ciclo de repetição representado na forma de um fluxograma:



Note que as três ações mostradas na parte de baixo à direita do fluxograma são repetidas diversas vezes, enquanto a condição `m<10` for verdadeira.

Objetivos Específicos

É primo ou não?

- Compreender como sistematizar soluções algorítmicas usando linguagem matemática e linguagens de programação.
- Compreender os conceitos básicos de uma linguagem de programação: repetição.

Sugestões e discussões

É primo ou não?

Organização da turma sugerimos que essa atividade seja resolvida em conjunto, pela turma toda, uma vez que a etapa final depende do uso de algoritmo completo e correto.

Duração 2 aulas.

Conexões vários tópicos do Ensino Fundamental podem ser revisados ou discutidos em conexão com essa atividade, como os conceitos de divisor e múltiplo, critérios de divisibilidade e o Crivo de Eratóstenes (que pode ser discutido sob o ponto de vista computacional se os estudantes souberem usar vetores).

Para refletir os itens que podem ser identificados nessa discussão são: só é necessário verificar se um número d é divisor de um número n enquanto $d \leq \sqrt{n}$; a busca pode ser feita apenas até encontrarmos o primeiro divisor diferente de 1 e de n; só é necessário checar se números primos são divisores (mas como não temos uma lista desses números disponível, essa propriedade não pode ser utilizada neste contexto).

Variação a biblioteca Util do Portugol traz funções quer permitem a contagem de tempo dentro de um algoritmo. Com isso, é possível propor o desafio de encontrar o maior primo em 1 minuto, por exemplo, estimulando a busca por melhorias nos algoritmos. O uso de bibliotecas em Portugol é simples e você pode aprender mais sobre elas em https://youtu.be/rs8ihN08bgU.

Nota 3

A cada repetição, a variável `m` tem o seu conteúdo aumentado em uma unidade, o que faz com que, em algum momento, o seu valor seja maior ou igual a 10, violando a condição e fazendo com que o algoritmo saia do ciclo para o seu fim.

Para refletir

O que este algoritmo escreverá se o número dado for 8?

Em Portugol, ciclos são realizados pelo comando `enquanto`: quando o computador chega neste comando, ele entende que deve repetir os comandos dentro das chaves seguintes enquanto a condição colocada (`m<10`) for verdadeira.

```
programa {
funcao inicio()  {
    inteiro n, m, resultado
    leia(n)
    m=1
    enquanto (m<10) {
        resultado = m*n
        escreva(resultado + "\n")
        m=m+1
      }
  }
}
```

Esse é um exemplo simples de uma estrutura de repetição, que poderia ser feita por uma pessoa sem grande esforço. Mas ela transmite muito bem a ideia de repetir alguma ação enquanto uma condição for verdadeira.

Nas atividades a seguir, vamos propor algumas tarefas repetitivas que são muito trabalhosas quando feitas manualmente, por isso é ainda mais relevante utilizar um computador para fazê-las.

Explorando — Repetir e repetir

Atividade 7

É primo ou não?

Você deve se lembrar que um número é chamado primo se tiver apenas dois divisores inteiros positivos: 1 e ele mesmo. Por exemplo, o número 9 não é primo, pois ele também é divisível por 3 enquanto que 13 é primo, pois só é divisível por 1 e por ele mesmo.

Até hoje, não existem métodos realmente rápidos para identificar se um número é primo ou não. Basicamente, todos os métodos existentes baseiam-se em testar se o número dado deixa resto zero quando é dividido pelos números menores do que ele. Imagina a quantidade de repetições necessárias para verificar se o número 4.000.037 é primo!

a) Descreva um algoritmo que verifica se um número é primo ou não.

b) Em conjunto com toda a turma, escreva um dos algoritmos criados no item anterior em Portugol.

c) Utilize esse algoritmo para verificar se o número 4.000.037 é primo.

Você deve ter notado que o algoritmo responde quase que imediatamente se o número dado é primo ou não, mesmo sendo um número grande.

d) Se você conseguisse verificar um divisor a cada 1 segundo, estime aproximadamente quanto tempo você teria levado para verificar se 4.000.037 é primo.

Para refletir

Embora muitas checagens precisem ser feitas, não é necessário dividir um número dado por todos os números naturais menores do que ele para verificar se ele é primo. Várias melhorias podem ser feitas nesse processo para torná-lo mais eficiente. Discuta com a turma essas melhorias.

Atividade 8

A sequência de Collatz

A sequência de Collatz é uma sequência numérica formada por números inteiros que é construída a partir de um valor inicial, que podemos chamar de a_1, por meio da seguinte regra: o próximo termo da sequência é igual à metade do anterior, se o anterior for par, e igual ao triplo do anterior mais um, se o anterior for ímpar. Matematicamente, podemos descrever essa regra de formação dessa maneira:

$$a_n = \begin{cases} a_{n-1}/2, & \text{se } a_{n-1} \text{ é par} \\ 3a_{n-1} + 1, & \text{se } a_{n-1} \text{ é ímpar} \end{cases}$$

Objetivos Específicos

A sequência de Collatz

- Compreender como sistematizar soluções algorítmicas usando linguagem matemática e linguagens de programação.
- Compreender os conceitos básicos de uma linguagem de programação: repetição.

Sugestões e discussões

A sequência de Collatz

Dificuldades este problema propõe uma sequência que é obtida de forma recursiva e os estudantes podem não estar acostumados com esse tipo de sequência numérica. Do ponto de vista computacional, ele é mais simples que a atividade anterior e essa escolha foi intencional, para servir como uma atividade de fixação.

Organização da turma por ter objetivo maior de fixação e pela simplicidade matemática do problema, sugerimos que essa atividade seja resolvida individualmente ou em duplas, após uma discussão inicial com a turma toda para que se compreenda o pedido descrito no problema.

Duração 2 aulas.

Discussão com o algoritmo em mãos, várias explorações podem ser feitas, por exemplo: você consegue encontrar um valor para a_1 que resulte em uma sequência de tamanho exatamente igual a 7? Qual é o número inteiro menor do que 50 que resulta na sequência com o maior número de termos?

Variação você pode sugerir aos estudantes que criem um algoritmo que, ao invés de gerar a sequência a partir de um valor específico de a_1 dado pelo usuário, utilize um segundo comando `enquanto` para gerar as sequências de Collatz para todos os números naturais de um intervalo dado.

Solução

A sequência de Collatz

a) 12, 6, 3, 10, 5, 16, 8, 4, 2, 1

b) 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1

c) Você pode usar o algoritmo abaixo para gerar a sequência a partir de qualquer valor de a_1.

d)
```
programa
{
  funcao inicio()
  {
    inteiro a, b
    leia(a)
    enquanto (a!=1) {
      se (a%2==0) {
        b=a/2
      }
      senao {
        b=3*a+1
      }
      escreva(b + " ")
      a=b
    }
  }
}
```

e) 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1

Por exemplo, se tomarmos, temos a seguinte sequência: 5, 16, 8, 4, 2, 1, 4, 2, 1, 4, 2, 1, ... Note que a sequência entra em um ciclo depois que chega em 1 pela primeira vez. Por isso, dizemos que ela termina ao chegar em 1, ou seja, a sequência de Collatz com $a_1 = 5$ tem 6 termos: 5, 16, 8, 4, 2, 1.

a) Escreva a sequência de Collatz para $a_1 = 12$.

b) Escreva a sequência de Collatz para $a_1 = 17$.

c) Escolha um número entre 5 e 20 e obtenha a sequência de Collatz iniciada por esse número.

d) Escreva em Portugol um algoritmo que obtenha a sequência de Collatz a partir de um valor dado para a_1.

e) Com auxílio do seu algoritmo, determine a sequência de Collatz para $a_1 = 26$.

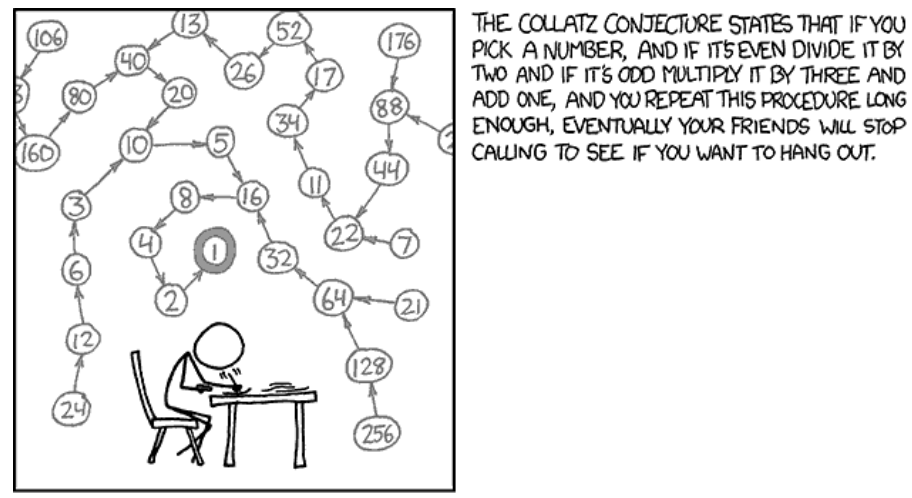


Figura 3: Fonte: https://xkcd.com/710. O texto diz: A conjectura de Collatz afirma que se você escolher um número, sendo par dividi-lo por 2, sendo ímpar multiplicá-lo por 3 e somar 1, e se você repetir isso por tempo suficiente, seus amigos vão parar de te ligar pra saber se você quer encontrá-los.

O comportamento da sequência de Collatz pode ser surpreendente. No caso do item **e)** acima, a sequência tem apenas 11 termos, mas se usarmos $a_1 = 27$ a sequência precisa de 113 termos para terminar! Dica: você pode criar uma nova variável para contar os termos da sequência.

É claro que repetir esse processo tantas vezes pode ser bem entediante, mas esse é o tipo de tarefa que um computador pode fazer sem dificuldade e muito mais rapidamente. Com o algoritmo que você criou em mãos, é mais simples testar casos e eventualmente levantar conjecturas sobre essa sequência: é possível prever valores de que geram sequências bem curtas? Há alguma família de números que, se atingida, encaminha a sequência para o seu final? Será que qualquer que seja o valor de a_1 a sequência sempre termina?

Um detalhe interessante é que até hoje os matemáticos não conseguiram responder satisfatoriamente a essa última pergunta! Com o auxílio de computadores, muitos valores de a_1 já foram testados e a sequência gerada a partir deles sempre chega, cedo ou tarde, em 1. Mas ninguém conseguiu provar que isso é verdadeiro para qualquer número inteiro positivo desde que o problema foi proposto (há mais de 80 anos).

Processos que envolvem muitas repetições são bons exemplos de como os computadores podem ser usados como ferramenta: podemos programá-los para executar essa parte da tarefa e focarmos nas partes que envolvem criatividade e o reconhecimento de padrões. No caso da sequência de Collatz, poder analisar e comparar várias sequências rapidamente nos permite levantar conjecturas, testá-las e ajustá-las de acordo com as observações.

Para compreendermos bem o uso de repetição, vamos analisar o algoritmo abaixo.

Ele escreve os termos de uma sequência cujos dois primeiros termos valem 1 (veja que as variáveis **a1** e **a2** começam recebendo esses valores). A variável **n**, que é lida no início do algoritmo determina a quantidade de termos que serão escritos. Veja que os dois primeiros são escritos antes do comando **enquanto**, por isso a variável **i** recebe o valor 3 (quando o enquanto começar, já estaremos escrevendo o terceiro termo).

```
1   programa {
2   funcao inicio()  {
3       inteiro a1,a2,an,penultimo,ultimo,proximo,n,i
4       a1=1
5       a2=1
6       escreva(a1+" "+ a2+" ")
7       ultimo=a2
8       penultimo=a1
9       leia(n)
10      i=3
11      enquanto (i<=n) {
12          proximo=penultimo+ultimo
13          escreva(proximo+" ")
14          i=i+1
15          penultimo=ultimo
16          ultimo=proximo
17      }
18    }
19  }
```

Para refletir

Quais valores seriam escritos considerando $n = 6$? Você conhece essa sequência numérica?

O aspecto que é importante compreender neste algoritmo é o uso das variáveis **i** e **proximo**.

Note que a variável **i** começa com o valor 3 e ela aparece na condição que determina quantas vezes o comando **enquanto** vai ser repetido. Ao final dos comandos que são executados dentro do **enquanto**, o valor de **i** é incrementado em uma unidade, até que ela fica maior do que **n** e o **enquanto** para de ser repetido. O papel dessa variável é servir como um contador de repetições.

Já as variáveis **ultimo**, **penultimo** e **proximo** têm uma função diferente. Para entendê-las, é necessário analisar os comandos das linhas 7 e 8 e depois das linhas 12, 15 e 16.

Nas linhas 7 e 8, as variáveis **ultimo** e **penultimo** recebem o valor das variáveis **a1** e **a2**. Note que a partir desse ponto, **a1** e **a2** não são mais utilizadas. Elas foram usadas para registrar o início da sequência numérica e depois os termos seguintes serão criados usando as variáveis **ultimo**, **penultimo** e **proximo**. Veja que na linha 7, a variável **ultimo** recebe o valor do último

Trata-se da sequência de Fibonacci, cujos 6 primeiros termos são: 1, 1, 2, 3, 5 e 8.

É possível usar esse algoritmo para explorar algumas propriedades dessa sequência, mas uma variação do algoritmo nos parece especialmente importante do ponto de vista computacional: que alteração deve ser feita no código se quisermos que ele escreva os termos até ultrapassar um valor dado?

Esse efeito pode ser obtido alterando apenas a condição do comando **enquanto** de modo que o valor da variável **ultimo**, ao invés de **i**, seja comparado com o valor dado pelo usuário.

termo conhecido, nesse caso o **a2**, e na linha 8, a variável **penultimo** recebe o valor do penúltimo termo conhecido, nesse caso o **a1**.

Na linha 12, a variável **proximo** recebe o valor de **penultimo+ultimo** e depois o seu conteúdo é escrito, ou seja, **proximo** é o próximo termo da sequência.

Por fim, os comandos das linhas 15 e 16 estão fazendo o trabalho de "avançar na sequência". Nessas linhas, variáveis **penultimo** e **ultimo** deixam de ser o primeiro e segundo termos e passam a ser os próximos: **penultimo** recebe o valor do **ultimo** e **ultimo** recebe o valor de **proximo**, como se estivéssemos "andando" na sequência. Dessa forma, na próxima repetição, **penultimo** e **ultimo** estarão armazenando os dois últimos valores calculados e **proximo** será a soma deles, ou seja, o termo seguinte.

Você pode adaptar esse algoritmo, e o que aprendeu com as atividades anteriores, para obter outras sequências numéricas, como progressões aritméticas, progressões geométricas e os números triangulares. Pesquise na internet sobre essas sequências e tente construir algoritmos em Portugol para cada uma delas.

Para o professor: Praticando tudo que foi estudado

Nesta seção, são propostas quatro atividades cujas resoluções passam por tudo que foi estudado ao longo das seções anteriores, de modo que não é estritamente necessário resolver todas elas para completar os objetivos educacionais deste material.

Elas podem ser discutidas em sala de aula, sem uso de dispositivo eletrônico que permita a execução dos algoritmos pedidos, uma vez que os problemas são matematicamente interessantes. Porém, o uso do laboratório poderá enriquecer o engajamento e potencializar o desenvolvimento do pensamento computacional dos seus estudantes, já que o uso do computador favorece o exercício de algumas habilidades ligadas a esse tipo de pensamento.

As atividades são colocadas de maneira mais direta, deixando espaço para que o professor as desdobre no formato que achar mais adequado, dando mais ou menos ênfase para o lado computacional ou matemático. Por conta disso, não indicamos a duração de cada atividade nem a organização da turma, mas salientamos alguns aspectos que podem ser relevantes para o caso de uso em laboratório.

Além disso, essas atividades também podem ser usadas para avaliação. Duas possibilidades para tal uso são:

- Atividades em dupla ou em pequenos grupos de modo que as duplas devem escolher uma das quatro atividades e resolvê-la em um período de uma aula dupla;
- Dividir a turma em 8 grupos, de modo que cada atividade seja resolvida por 2 grupos diferentes. Os grupos deverão preparar uma apresentação curta e, a cada aula, os dois grupos que resolveram as mesmas atividades apresentarão suas resoluções.

5 Praticando tudo que foi estudado

Nesta seção, vamos praticar tudo que foi estudado até este momento.

Todas as atividades propostas envolvem dois aspectos, o matemático e o computacional. Isso significa que conhecimentos matemáticos serão necessários para compreender e resolver os problemas e você também terá que pensar computacionalmente sobre o processo de resolução para transformar a sua resolução em um algoritmo.

Apesar das atividades não pedirem explicitamente, você pode criar todos os seus algoritmos em Portugol e testá-los usando as respostas que você obteve ao longo da atividade.

Praticando — Tudo o que você aprendeu

Atividade 9

Imposto de renda retido na fonte

O imposto de renda é um dos principais impostos no Brasil. Uma das formas em que esse imposto é cobrado ocorre na folha de pagamento, ou seja, quando um funcionário com carteira de trabalho assinada recebe o seu salário. Essa forma de cobrança é chamada de imposto de renda retido na fonte.

O cálculo do valor a ser descontado do salário (e imediatamente repassado ao governo federal) é feito de forma que ele seja percentualmente maior à medida que o salário aumenta. Em 2020 eram adotados 5 intervalos de salário bruto (isto é, ainda sem o desconto do imposto). Para cada um desses intervalos, o cálculo é feito da seguinte maneira: calcula-se a porcentagem indicada em "alíquota" do salário bruto e, do resultado, subtrai-se a "parcela a deduzir". O resultado obtido é o valor que será descontado do salário no momento do pagamento.

Base de cálculo (R$)	Alíquota (%)	Parcela a deduzir do IR (R$)
Até 1.903,98	—	—
De 1.903,99 até 2.826,65	7,5	142,80
De 2.826,66 até 3.751,05	15,5	354,80
3.751,06 até 4.664,68	22,5	636,13
Acima de 4.664,68	27,5	869,36

Tabela 1: Fonte: Receita Federal

Por exemplo, para um salário de R$ 2.000,00, começamos identificando que ele se encontra na segunda faixa. Logo, devemos calcular 7,5 % de 2000 e depois subtrair 142,80 do resultado. Portanto, $0{,}075 \times 2000 - 142{,}80 = 7{,}20$ é o valor que será descontado do salário para pagar o imposto de renda desse trabalhador.

a) Quanto será descontado de uma pessoa que tenha salário bruto igual a R$ 1.500,00? E igual a R$ 3.000,00? E igual a R$ 6.000,00?

b) Crie um algoritmo em Portugol que calcule o valor do imposto de renda a ser retido na fonte para um dado salário bruto.

Objetivos Específicos

Imposto de renda retido na fonte

Compreender os conceitos básicos de uma linguagem de programação, como repetição, condicional e variáveis.

Sugestões e discussões

Imposto de renda retido na fonte

Conexões a atividade tem conexões com funções definidas por partes e continuidade de funções, além de estar relacionada com tópicos de Matemática Financeira.

Sugestão geral Você pode transformar essa atividade em uma proposta mais ampla, solicitando aos estudantes que pesquisem como o cálculo do imposto retido na fonte é feito e quais são as alíquotas e valores a deduzir para cada intervalo de salário bruto. Há muitas referências na internet que podem ser consultadas.

Comentários para o Portugol Como as variáveis usadas neste código são do tipo real, recomenda-se que ao atribuir um valor a elas, seja indicada ao menos uma casa decimal, mesmo que seja igual a 0, como na linha `ir=0.0` na solução dada no gabarito. Além disso, o Portugol pode considerar como erro as situações em que uma variável só recebe um valor dentro de comandos condicionais, pois caso a condição não seja satisfeita, o algoritmo poderia terminar sem que a variável receba um valor. Isso pode ser resolvido atribuindo um valor qualquer, ou que faça sentido para a situação, à variável no início do código.

Variações Você pode pedir que o algoritmo calcule o valor do salário após o desconto ou a alíquota efetivamente cobrada.

Nota 4

Objetivos Específicos

Xadrez

Compreender os conceitos básicos de uma linguagem de programação, como repetição, condicional e variáveis.

Sugestões e discussões

Xadrez

Discussão Este problema pode ser resolvido aritmeticamente com o auxílio do conceito de paridade (aplicada tanto às coordenadas de um dado quadradinho quanto à soma das coordenadas), mas também admite soluções com repetições (em que o código alterna entre as duas soluções à medida que "percorre" um caminho até o quadradinho de destino). Incentive a discussão dessas duas soluções em termos de simplicidade, clareza e eficiência (qual delas obtém a solução realizando o menor número de ações?).

Variação Uma questão análoga pode ser proposta considerando o movimento de um cavalo do jogo de Xadrez: dada a cor da posição inicial do cavalo e o número de vezes que ele foi movido, qual é a cor da posição em que ele se encontra?

Fonte: Olimpíada Brasileira de Informática.

Nota 5

Objetivos Específicos

MMC

Compreender os conceitos básicos de uma linguagem de programação, como repetição, condicional e variáveis.

Sugestões e discussões

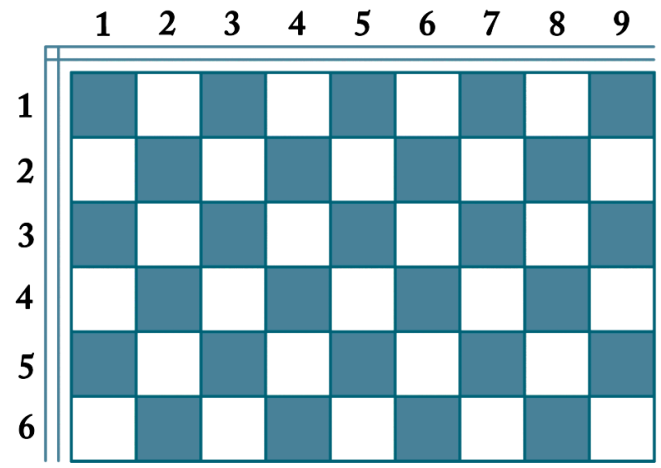
MMC

Comentários gerais Esta atividade reforça o uso de estruturas de repeticão através de um procedimento matemático conhecido, o cálculo do Mínimo Múltiplo Comum entre dois números. O problema pode ser resolvido com um algoritmo bastante curto, o que ilustra bem o poder dessas estruturas.

Atividade 10

Xadrez

No tabuleiro de xadrez, a casa na linha 1, coluna 1 (canto superior esquerdo) é sempre branca e as cores das casas se alternam entre branca e preta. Dessa forma, como o tabuleiro tradicional tem oito linhas e oito colunas, a casa na linha 8, coluna 8 (canto inferior direito) será também branca. Neste problema, entretanto, queremos saber a cor da casa no canto inferior direito de um tabuleiro com dimensões quaisquer: L linhas e C colunas. No exemplo da figura, para $L = 6$ e $C = 9$, a casa no canto inferior direito será preta.



a) Qual seria a cor da casa no canto inferior direito se o tabuleiro tiver 7 colunas e 4 linhas? E se tiver 10 colunas e 8 linhas? E se tiver 23 colunas e 40 linhas?

b) Descreva como descobrir a cor da casa no canto inferior direito de um tabuleiro como esse, sabendo a quantidade de linhas e colunas que o compõe.

c) Escreva um algoritmo em Portugol que implemente o processo que você descreveu na questão anterior.

Atividade 11

MMC

Existem vários métodos para o cálculo do mínimo múltiplo comum (MMC) entre dois números inteiros. Você deve ter aprendido como fazer isso ainda no Ensino Fundamental e, de vez em quando, ainda deve utilizar esse procedimento para resolver algumas questões de matemática.

a) Descreva como você procede para obter o mínimo múltiplo comum entre dois números dados. Se quiser, comece obtendo o mínimo múltiplo comum para os números 12 e 18 e depois tente descrever o método que você utilizou de forma genérica, isto é, para dois números quaisquer a e b.

Embora não seja o método mais eficiente, vamos implementar o método da lista. Ele consiste em listar os múltiplos do primeiro número (começando por ele mesmo) até encontrar um múltiplo que seja divisível pelo segundo número.

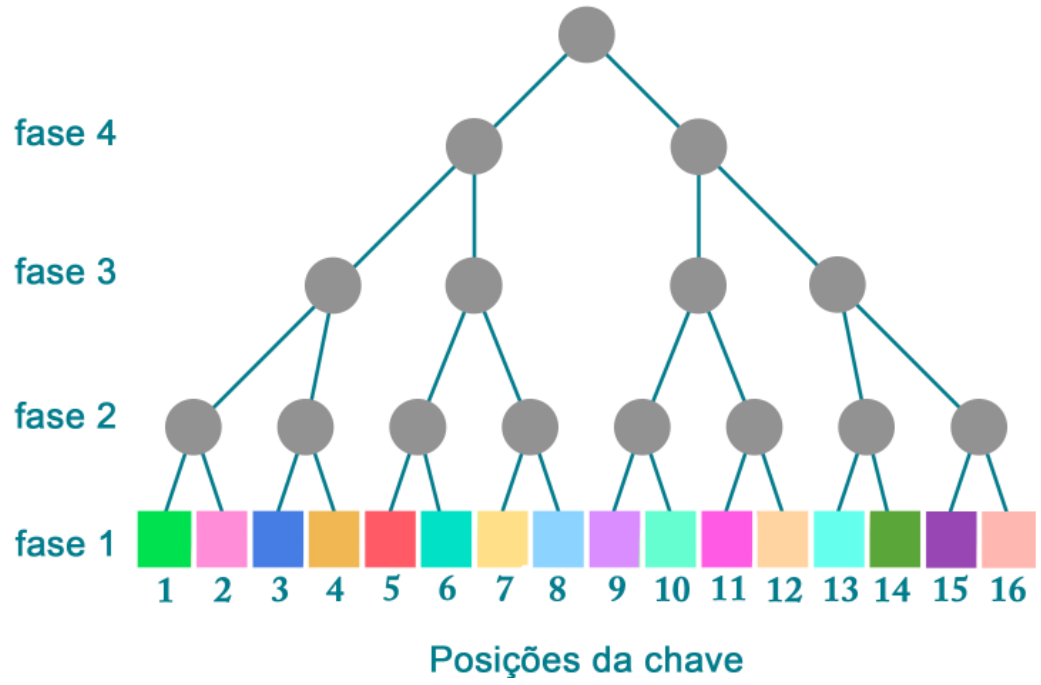
Exemplo: para o caso 12 e 18, vamos listar os múltiplos de 12. Primeiro, o próprio 12, que não é divisível por 18. Depois 24 ($12 \cdot 2$), que não é divisível por 18. Depois 36 $12 * 3$, que é divisível por 18 e, portanto, é o mínimo múltiplo comum entre esse 12 e 18.

b) Escreva um algoritmo em Portugol que obtenha o mínimo múltiplo comum entre dois números naturais dados, usando o método da lista.

Atividade 12

Campeonato

Um torneio esportivo mundial é organizado no tradicional formato de chaves: dois competidores se enfrentam e quem vence passa para a próxima etapa, até que os dois que venceram todas as partidas se enfrentem na final. Nesta edição, o torneio vai contar com 16 competidores e os confrontos ocorrerão como mostrado abaixo.



O grande adversário do Brasil neste torneio é os Estados Unidos e, por isso, todos querem saber quando os competidores desses dois países poderão vir a se enfrentar. Isso será determinado um dia antes do início do torneio, quando será sorteada uma bolinha com um número de 1 a 16 para cada país, indicado a posição em que eles serão alocados nas chaves.

a) Escreva um algoritmo em Portugol que, dadas as duas posições do Brasil e Estados Unidos nas chaves, determina em qual fase do torneio os países poderão se enfrentar.

Para refletir

Você conseguiria modificar o seu algoritmo de modo que ele possa ser usado para outras quantidades de competidores?

Observação: você pode assumir que essas quantidades sempre serão uma potência de 2.

Variação Você pode pedir aos estudantes que usem as ideias desta atividade para construir um algoritmo que calcula o MDC entre dois números usando a relação abaixo ou através de um algoritmo concebido especificamente para esse fim.

$$\text{mdc}(a, b) = \frac{a \cdot b}{\text{mmc}(a, b)}$$

Nota 6

Objetivos Específicos

Campeonato

Compreender os conceitos básicos de uma linguagem de programação, como repetição, condicional e variáveis.

Nota 7

Referências

Brasil. (2018). *Base Nacional Comum Curricular* [Disponível em: http://basenacionalcomum.mec.gov.br]. Ministério da Educação. Brasília, Brasil.

Disessa, A. A. (2018). A Computation Literacy and “The Big Picture” Concerning Computers in Mathematics Education. *Mathematical Thinking and Learning*, *20*(1), 3–31.

Doolittle, C. L. (1910). Halley’s Comet. *The Popular Science Monthly*.

Esteves, A., Noschang, L., Raabe, A., & Filho, A. (2019). Portugol Studio: Em direção a uma comunidade aberta para pesquisa sobre o aprendizado de programação. *Anais do XXVII Workshop sobre Educação em Computação*, 513–522.

Li, Y., Schoenfeld, A. H., diSessa, A. A., Graesser, A. C., Benson, L. C., English, L. D., & Duschl, R. A. (2020). Computational Thinking Is More about Thinking than Computing. *Journal for STEM Education Reserach*, 1–18.

Noschang, L., Pelz, F., de Jesus, E., & Raabe, A. (2014). Portugol Studio: IDE para Iniciantes em Programação. *Anais do XXII Workshop sobre Educação em Computação*, 1–10.

Raabe, A., Zorzo, A. F., & Blikstein, P. (2020). *Computação na Educação Básica: Fundamentos e Experiências*. Penso Editora.

Reis, S. R. d., Barichello, L., & Mathias, C. V. (2021). Novos conteúdos e novas habilidades para a área de Matemática e suas Tecnologias. *Revista Internacional De Pesquisa Em Educação Matemática*, *11*(1), 37–58. https://doi.org/https://doi.org/10.37001/ripem.v11i1.2539

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158.

Notas

Nota 1. (Página 2)

Solução

Boas instruções

a) R$ 190,00 e R$ 189,00

b) R$ 190,00, R$ 208,50 e R$ 370,00

c) Sim, pois a promoção não afeta o valor que já seria pago pelos primeiros bottons ao dar desconto apenas para os bottons adicionais. Diferentemente da primeira opção, aqui o a função "valor a ser pago" é estritamente crescente em todo o seu domínio.

d) Essa questão admite muitas respostas, vide nota lateral para esclarecimento sobre aspectos importantes a serem considerados na discussão e correção.

Nota 2. (Página 3)

Solução

Uma resolução boa de repetir

a) 2.062, 2.214 e 3.050.

b) 1.910 e 1.530

c) As soluções podem variar muito, mas duas abordagens são esperadas: a descrição de um processo repetivo no qual soma-se ou subtrai-se sucessivas parcelas iguais a 76 de 1986 até obter uma resposta; o cálculo de quantos intervalos de 76 anos cabem no intervalo entre 1986 e o ano de nascimento para então calcular o ano de interesse. Ambas estão corretas.

d) O primeiro e segundo casos foram escolhidos de forma a serem um no futuro e um no passado, já o terceiro é o ano exato em que o cometa passa e é comum que esse caso não seja contemplado claramente nas descrições sobre como resolver. Ele foi incluído para que os estudantes tenham a chance de aprimorar suas instruções. Vale salientar que tanto o caso de considerar o próprio ano quanto a passagem seguinte como resposta são válidos nos termos colocados na atividade. Não pretendemos considerar mês e dia nessa atividade e nas próximas atividades em torno deste problema, mas isso seria possível se o professor desejar.

Nota 3. (Página 20)

Solução

É primo ou não?

a) O objetivo das questões a) e b) é obter um algoritmo como o mostrado a seguir.

```
programa
{
  funcao inicio()
  {
    inteiro n, d, resto
    leia(n)
    d=2
    resto=1
```

```
      enquanto ( (d*d<=n) e (resto!=0)) {
        resto = n%d
        d=d+1
      }
      se (resto==0) {
        escreva("não é primo")
      }
      senao {
        escreva("é primo")
      }
    }
  }
```

b) Sim, é primo.

c) Se a turma notou a restrição de procurar divisores até $\sqrt{n}$, serão necessárias aproximadamente 2000 verificações, ou seja, 2000 segundos, que equivale a mais de meia hora. Se a turma tivesse uma lista de números primos, seria necessário apenas cerca de 300 verificações, ou seja, 5 minutos.

Nota 4. (Página 25)

Solução

Imposto de renda retido na fonte

a) R$ 0,00, R$ 95,20 e R$ 780,64

b)

```
programa
{
  funcao inicio()
  {
    real salario, ir
    leia(salario)
    ir = 0.0
    se (salario>1903.98 e salario<=2826.65) {
      ir = salario*0.075-142.80
    }
    se (salario>2826.65 e salario<=3751.05) {
      ir = salario*0.15-354.80
    }
    se (salario>3751.05 e salario<=4664.68) {
      ir = salario*0.225-636.13
    }
    se (salario>4664.68) {
      ir = salario*0.275-869.36
    }
    escreva(ir)
  }
}
```

Nota 5. (Página 26)

Solução

Xadrez

a) Preto, branco, preto.

b) É possível resolver esta questão pensando na paridade da soma das dimensões (mostrado abaixo) ou na paridade das coordenadas isoladamente.

c) Várias soluções são possíveis, sendo essa uma delas:

```
programa
{

  funcao inicio()
  {
    inteiro L, C, distancia
    leia(L)
    leia(C)
    distancia = L-1 + C-1
    se (distancia%2==0) {
      escreva("branca")
    }
    senao {
      escreva("preta")
    }
  }
}
```

Nota 6. (Página 27)

Solução

MMC

a) Vários métodos são ensinados para esse procedimento, mas os dois que parecem mais comuns envolvendo a listagem de múltiplos ou a fatoração simultânea dos dois números. Este último é computacionalmente mais eficiente, mas depende de uma lista de números primos para sua implementação.

b)
```
programa
{
  funcao inicio()
  {
    inteiro a, b, n
    leia(a)
    leia(b)
    n=1
    enquanto ( ((a*n)%b) != 0 ) {
      n=n+1
    }
    escreva(a*n)
  }
}
```

Nota 7. (Página 27)

Sugestões e discussões

Campeonato

Comentários gerais Esta atividade propõe um problema matematicamente mais difícil do que todos os propostos até o momento. Além disso, ele pode ser resolvido algoritmicamente de diversas maneiras.

Sugestões gerais A solução deste problema passa pela ideia de agrupamento dos números de 2 em 2 (se enfrentam na primeira fase), 4 em 4 (se enfrentam na segunda fase), 8 em 8 (se enfrentam na terceira fase) e assim por diante. Esse agrupamento pode ser detectado matematicamente verificando se a divisão das duas posições, menos 1, por 2, 4, 8 e 16 têm o mesmo resultado inteiro. Por exemplo, considere os times nas posições 3 e 8, como o quociente de $(3-1)$ e de $(8-1)$ por 2 é diferente, eles não se enfrentam na fase 1, por 4 também, mas por 8 ambos têm quociente 0, logo, se enfrentam na fase 3. É importante ter certeza de que os estudantes compreenderam a solução do ponto de vista matemático antes que tentem escrever a solução algorítmica. Outro aspecto importante se refere à forma como as posições são numeradas: se fossem de 0 a 15 e não de 1 a 16, a solução do problema seria mais direta.

Variação Você pode aumentar o número de times participantes para 32, 64, etc. Isso pode levar à generalização do problema com 2^n times participando.

Fonte: Olimpíada Brasileira de Informática.

Solução

Campeonato

A solução específica deste problema é mostrada abaixo. Curiosamente, ela é mais longa que a solução geral, porém, o significado das etapas do algoritmo são mais claras.

```
programa
{
  funcao inicio()
  {
    inteiro pos1,pos2
    leia(pos1)
    leia(pos2)
    pos1 = pos1-1
    pos2 = pos2-1
    se (pos1/2 == pos2/2) {
      escreva("fase 1")
    }
    senao se (pos1/4 == pos2/4) {
      escreva("fase 2")
    }
    senao se (pos1/8 == pos2/8) {
      escreva("fase 3")
    }
    senao {
      escreva("fase 4")
    }
  }
}
```

Uma solução geral é mostrada a seguir. A variável **nfases** permite que a solução seja ajustada para torneios com mais ou menos times participantes, desde que siga a mesma estrutura de chaves.

```
programa
{
  funcao inicio()
  {
    inteiro pos1, pos2, nfases, i, potencia
    nfases = 4 //isso pode ser mudado e o algoritmo funciona para
               //torneio com mais fases
    i = 1 //contador
    potencia = 2
    leia(pos1)
    leia(pos2)
    enquanto (i<=nfases) {
      //se a divisão por 2, 4, 8... der o mesmo quociente
      //os times estão no mesmo "agrupamento"
      se ( (pos1-1)/potencia == (pos2-1)/potencia ) {
        escreva("fase " + i)
        pare //se achei a resposta, eu já paro o algoritmo
      }
      potencia = potencia*2 //passando para o proximo agrupamento
      i=i+1
    }
  }
}
```

Coleção Livro Aberto de Matemática

O esforço para produzir livros didáticos de matemática com licença aberta começou em 2016, com a elaboraçao do material de Frações para o Ensino Fundamental I. Desde então, novos elaboradores acreditaram e juntaram-se ao projeto para alcançarmos novos níveis e novos livros.

Hoje, possuímos diversos capítulos escritos e vários livros em produção. Tudo isso a partir de um trabalho colaborativo envolvendo matemáticos, professores universitário e professores da Educação Básica.

Um princípio fundamental desta iniciativa é que sua produção configure uma proposta pedagógica ancorada e acompanhada por pesquisa científica em Ensino de Matemática.

O projeto tem também compromisso com a formação e o desenvolvimento profissional de professores. Em particular, pela composição característica da equipe, destaca-se o entendimento do potencial do projeto para enfrentar um reconhecido desafio: estreitar o diálogo entre a realidade e as demandas próprias da prática docente e a formação acadêmica do professor.

O Livro Aberto de Matemática é um projeto do IMPA, desenvolvido em suas etapas iniciais pela Associação Livro Aberto com financiamento da Fundação Itaú Social e apoio da UNIRIO e da UFRJ.

Realização

impa
Instituto de Matemática Pura e Aplicada

OLIMPÍADA BRASILEIRA DE MATEMÁTICA DAS ESCOLAS PÚBLICAS
OBMEP-IMPA
Somando novos talentos para o Brasil

Patrocínio

Itaú Social

ISBN: 978-85-244-0523-5
9 788524 405235