UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
Departamento de Ciências Exatas
PROFMAT - Mestrado Profissional em Matemática em Rede Nacional

Dissertação de Mestrado

ALGUNS ALGORITMOS EM JAVA PARA MATEMÁTICA BÁSICA

Newton Silva Santos

Orientador: Prof. Dr. Haroldo Gonçalves Benatti

Feira de Santana

Dezembro de 2020

UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA

Departamento de Ciências Exatas

PROFMAT - Mestrado Profissional em Matemática em Rede Nacional

ALGUNS ALGORITMOS EM JAVA PARA MATEMÁTICA BÁSICA

Newton Silva Santos

Dissertação apresentada ao Programa de Mestrado Profissional em Matemática em Rede Nacional - PROFMAT do Departamento de Ciências Exatas, UEFS, como requisito parcial para a obtenção do título de **Mestre**.

Orientador: Prof. Dr. Haroldo Gonçalves Benatti

Feira de Santana

18 de Dezembro de 2020

Ficha Catalográfica – Biblioteca Central Julieta Carteado

S233a Santos, Newton Silva
Alguns algoritimos em Java para matemática básica / NewtonSilva Santos. –, 2020.
90f.: il.

Orientador: Haroldo Gonçalves Benatti
Dissetação (mestrado) – Universidade Estadual de Feira de Santana, Programa de Pós-Graduação Mestrado Profissional em Matemática em Rede Nacional, 2020.

1. Algoritimo – Educação matemática 2. Algoritimo - Computação I. Benatti, Haroldo Gonçalves, orient. II. Universidade Estadual de Feira de Santana. III. Título.

CDU: 51:681.3

Tatiane Souza Santos - Bibliotecária CRB5/1634

UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
DEPARTAMENTO DE CIÊNCIAS EXATAS
MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL

PROFMAT

ATA DA SESSÃO PÚBLICA DE DEFESA DE DISSERTAÇÃO DO DISCENTE NEWTON SILVA SANTOS DO PROGRAMA DE MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL DA UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA

Aos dezoito dias do mês de dezembro de dois mil e vinte às 9h30min, ocorreu a defesa pública não presencial, através da plataforma Google Meet, link: meet.google.com/kxm-tcej-xup, da dissertação apresentada sob o título **"ALGUNS ALGORITMOS EM JAVA PARA MATEMÁTICA BÁSICA"**, do discente **Newton Silva Santos**, do Programa de Mestrado Profissional em Matemática em Rede Nacional - PROFMAT da Universidade Estadual de Feira de Santana, para obtenção do título de MESTRE. A Banca Examinadora foi composta pelos professores: Haroldo Gonçalves Benatti (Orientador, UEFS), Eleazar Madriz (UFRB) e Márcia Braga de Carvalho Ferreira (UEFS). A sessão de defesa constou da apresentação do trabalho pelo discente e das arguições dos examinadores.

Em seguida, a Banca Examinadora reuniu-se em sessão secreta para julgamento final do trabalho e atribuiu o conceito: APROVADO.

Sem mais a tratar, foi lavrada a presente ata, que segue assinada pelos membros da Banca Examinadora e pelo Coordenador Acadêmico Institucional do PROFMAT.

Feira de Santana, 18 de dezembro de 2020.

Haroldo G. Benatti

Prof. Dr. Haroldo Gonçalves Benatti (UEFS)
Orientador

Eleazar

Prof. Dr. Eleazar Madriz (UFRB)

Prof.ª Dra. Márcia Braga de Carvalho Ferreira (UEFS)

Visto do Coordenador:

Agradecimentos

Tenho muito a agradecer, primeiro a essa força cósmica que nos une como seres humanos, alguns nomeiam tal força como Buda, Maomé, Deus, etc. Eu apenas acredito nesta força.

Agradeço a UEFS pois por meio desta instituição obterei o título de mestre e foi mediante a parceria UEFS/PROFMAT que tornou-se possível a minha realização neste mestrado, já que moro em Araci-ba cidade que fica a 200 km de Salvador, onde, na época, era o PROFMAT mais próximo de mim. Com a adesão da UEFS ao programa, ficou mais viável fazer o mestrado.

À CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, que me concedeu uma bolsa para auxílio financeiro que foi muito importante para custeios como deslocamento, livros, etc. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001

Ao prefeito Antônio Carvalho da Silva Neto e a secretaria de educação Manuela Texeira Silva Nery de Almeida, que me ajudaram facilitando meus estudos na UEFS em Feira de Santana.

Aos meus professores que contribuíram para o enriquecimento de meus conhecimentos matemáticos uma vez que minha graduação é em Analise de sistemas , na área de informática, fato que trouxe muita dificuldade para acompanhar o curso. Em especial ao professor Dr. Haroldo Gonçalves Benatti pelo apoio, dedicação, disponibilidade, correções, cobranças, paciência e incentivo na elaboração deste trabalho.

A todos os meus familiares, meus irmãos, amigos, a meu pai, minha mãe, minha avó materna (Lindinha), a meu avô paterno (Fernando) e é claro, a eles, “as três leis de Newton” que são Ana Livia Santos Pimentel, Newton Silva Santos Junior e Anna Edith Andrade Pinho Santos, meus queridos filhos, que são fundamentais para mim.

Resumo

Observamos que os Algoritmos são muito usados na matemática da educação básica, mas alunos e professores não se dão conta de sua importância, pois, nossa sociedade está a cada dia mais informatizada e toda a tecnologia que nos rodeia precisa dos algoritmos para funcionar. O objetivo deste trabalho é demonstrar a importância dos algoritmos para a matemática e dar um contexto para os mesmos na sala de aula. Para objetivar esta intenção pretende-se mostrar o que é Algoritmo e suas estruturas, visualizar alguns algoritmos matemáticos escritos em linguagem Java e demostrar um exemplo de Sequência Didática. Assim os leitores entenderão não só a relevância dos algoritmos mas a relação que ele faz entre a computação e a educação Matemática.

Palavras-chave: Algoritmos, Matemática, Computação.

Abstract

We observed that Algorithms are widely used in the mathematics of basic education, but students and teachers do not realize their importance, because our society is more and more computerized every day and all the technology that surrounds us needs the algorithms to work. The objective of this work is to demonstrate the importance of algorithms for mathematics and to provide a context for them in the classroom. To objectify this intention it is intended to show what Algorithm and its structures are, to visualize some mathematical algorithms written in Java language and to demonstrate an example of Didactic Sequence. This way, readers will understand not only the relevance of the algorithms but the relationship it makes between computing and mathematics education.

Keywords: Algoritmo, Mathematics, computing.

Sumário

Introdução

Este trabalho traz como tema os algoritmos. Uma temática que é utilizada constantemente no cotidiano, seja na escola ou fora dela, algoritmos aparecem em todos os níveis de conhecimentos matemáticos e em outros campos de conhecimento, como na medicina, que por meio de alguns algoritmos os médicos fazem diagnósticos das doenças de seus pacientes. Os algoritmos estão praticamente em tudo que nos rodeia atualmente, entretanto, nem todos percebem este fato.

No capítulo 1, basicamente é explorado o que é algoritmo, trazendo para isso, uma abordagem histórica e como as ciências, como a matemática e a informática, dependem dele, além de demonstrar a importância, características, representações e componentes que um algoritmo pode ter.

No capítulo 2, apresentam-se alguns algoritmos matemáticos da educação básica(focando no ensino fundamental e ensino médio), seu funcionamento, sua estrutura, o código Java e resultados.

No Capítulo 3, é discutida a importância do aprendizado dos algoritmos no ensino Básico, fala porque foi escolhida a linguajem Java e apresenta uma ideia de como introduzir algoritmos na sala de aula.

Capítulo 1

O Começo

A matemática é uma ciência que trabalha muito a percepção e a lógica dos objetos que nos rodeiam tentando entender e explicar o mundo através de cálculos e das relações que existem entre objetos distintos. Na contemporaneidade, a matemática está bastante desenvolvida e tem fórmulas e algoritmos para várias situações do cotidiano. Mas nem sempre foi assim, a matemática foi se desenvolvendo ao decorrer que o tempo passava, foi mudando e se aprimorando à medida que se unia o conhecimento de várias regiões. Com isso, veio a necessidade de aplicá-la a rotina diária e isso se tornou mais viável por conta dos algoritmos.

Na China em 1983, uma escavação levou a uma tumba que tinha uma coleção de 190 tiras de bambu contendo um texto matemático que foi datado do ano 186 e cujo o nome é "Suan shu shu - Writings on Reckonig". Para dar um exemplo na luta "Jiu-jitsu"a última parte da palavra é uma variação do termo chinês "shu"onde um de seus significados é método, procedimento. Jiu-jitsu significa regras procedimentais da flexibilidade, dos movimentos suaves.

A palavra método era usada pelos chineses também na matemática, além de outras áreas e artes marciais. O livro chinês tem várias seções, e muitas destas seções, se referem a resolução de cálculos, sendo um guia destinado aos administradores chineses daquela época para atender necessidades práticas. Outras seções revelam o interesse por conhecer como se estrutura a resolução de problemas[1].

Na cultura ocidental, tem-se registros que o termo algoritmo surgiu devido ao matemático persa Al-Khwarizmi, cujo livro de sua autoria, que em latim se chamava Algorismi, daí vem a origem do nome algoritmo que foi de grande importância para a prática da aritmética. Recentemente a palavra algoritmo tem um significado relevante e fundamental em uma área relativamente nova de conhecimento que é a Informática.

1.1 O que é algoritmo

Algoritmo é uma sequência de instruções finitas, que não podem ser ambíguas, com a finalidade de resolver algum problema. Cada instrução deve ser executada de maneira manual, mecânica ou eletrônica obedecendo um intervalo de tempo e uma quantidade de esforço ou processamento finito.

Segundo Berlinsk [28],

> "O computador digital é uma máquina e, como qualquer objeto, um prisioneiro ao fim de áridas leis da termodinâmica. Quando o tempo acaba, o combustível acaba também. Da mesma forma que o programador de computador martelando em um teclado com a ponta de dois dedos tensos. Como todos nós. Mas um algoritmo é outra coisa. Ocupando o espaço entre a agulhada do desejo e a resultante bolha de satisfação, é um instrumento abstrato de coordenação, que fornece os procedimentos para várias finalidades. Feito de sinais e símbolos, os algoritmos, como os pensamentos, vivem em um mundo além do tempo. Um algoritmo é um procedimento eficaz, um modo de fazer uma determinada coisa em um número finito de passos discretos. No mundo de onde surge um matemático e para o qual o matemático, como nós, deve voltar, um algoritmo, por assim dizer, é um conjunto de regras, uma receita, uma prescrição para a ação, um guia, uma diretiva concatenada e controlada, uma intimidação, um código, um esforço feito para jogar um complexo xale verbal sobre o caos inarticulado da vida."

Uma comparação bastante utilizada para ilustrar um algoritmo é uma receita de bolo, onde se tem os passos pré-determinados para que o bolo fique pronto, então algoritmo nada mais é que passos a serem realizados para completar uma tarefa.

1.2 A importância dos algoritmos

Como já citado acima, algoritmo é uma ferramenta abstrata muito importante que transpassou a matemática e atualmente é mais conhecida na área da computação/informática. Na escola vê-se a todo momento algoritmos para resolução de problemas e, no cotidiano se está rodeado deles em computadores, celulares, tablets, internet, dentre muitos outros. Então, está na hora de começarmos a dar um prestígio maior a essa ferramenta abstrata, para que a sociedade venha a entender a sua importância e a estudar os mesmos, com a finalidade de se ter um conhecimento inicial sobre algoritmo. A cada dia que passa, o mundo se torna mais autônomo, devido a novas tecnologias, e essas novas tecnologias são dependentes dos algoritmos. Existe outro grande benefício em estudar os algoritmos, que é desenvolver a lógica, e assim tornar a sociedade mais apta a compreender e interpretar as milhares de informações na qual se está exposto diariamente.

Pensar desta forma diferenciada é muito importante para muitas áreas do conhecimento, principalmente nas áreas de exatas e tecnológicas, pois um computador com um algoritmo bem elaborado vai executar uma tarefa com mais rapidez e eficiência usando o mesmo volume de dados. Entender como funcionam os algoritmos a as tecnologias do nosso cotidiano é muito importante[3].

1.3 Características de um algoritmo

Para Knuth [4], um algoritmo tem que ter algumas características indispensáveis para que consiga ser efetivo e possa resolver o problema a que se propõe, que são:

Finitude:

Um algoritmo deve sempre terminar depois de um número finito de instruções.

Definição:

Cada instrução do algoritmo deve ser definida com precisão, ou seja, as instruções que serão executadas deverão ser especificadas rigorosamente e não podem ter espaço para interpretações diferentes do que se quer.

Entrada:

São os dados que serão fornecidos ao algoritmo. Essas entradas são tomadas a partir de conjuntos de objetos especificados antes do algoritmo iniciar. É importante ressaltar que existe a possibilidade da entrada ser um dado nulo.

Saída:

Um algoritmo deve ter uma ou mais respostas (saídas), onde o conjunto de dados entregues ao algoritmo será por ele tratado, operado a partir do conjunto de dados da entrada.

1.4 Representações de algoritmos

Existem algumas maneiras de representar um algoritmo. Dentre as existentes as mais comuns e usuais são:

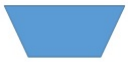
Linguagem Natural

Os algoritmos que usam uma narrativa descritiva, ou seja, na linguagem formal do indivíduo que os fizeram. Assim eram feitos os algoritmos antigamente, sem símbolos utilizando apenas palavras, o que dificultava a precisão do raciocínio descrito.

Fluxograma (ou Diagrama de Fluxo)

Nada mais é que uma representação gráfica de um algoritmo onde cada figura geométrica foi padronizada para representar uma ação ou decisão para resolver o problema a que o algoritmo se propõe. Será utilizado o padrão ANSI (American National Standards Institute)

Tabela 1.1: **Tabela com alguns símbolos.**

Símbolo	Nome	Quando utilizar?
	Inicio ou Fim	Todas as vezes que iniciar ou terminar o Fluxograma de um determinado processo
	Decisão	Todo processo existe um ponto de decisão que dependendo da situação ou decisão tomada poderão sinalizar dois ou mais caminhos. Um exemplo prático poderia ser após a verificação de um produto antes de sua liberação.Nesta situação poderiamos ter um símbolo de decisão com o questionamento: "O produto está conforme?". Caso positivo, o produto poderia ser entregue ao cliente, caso negativo, o produto não pode ser liberado.
	Processo	Serve para indicar as etapas no fluxo continuo do processo. "Embalar o produto", "Atendimento a cliente", este são apenas alguns exemplos de atividades que podem ser inseridos neste símbolo. Este símbolo nomeia quais são as etapas fundamentais de cada processo.
	Operação manual	Indicado para representar tarefas manuais existentes no fluxo de um processo.
	fluxo de linha	Utilizado como conector entre os símbolos de um processo. Serve para indicar a direção em que os processos ocorrem.

Disponível em https://certificacaoiso.com.br/o-que-e-fluxograma-de-processos/

Pseudo linguagem ou pseudo código

Emprega uma linguagem intermediária entre uma linguagem natural qualquer, como o português, e uma linguagem de programação como o Java, uma pseudo linguagem muito utilizada para aprendizado no Brasil é o Portugol. A pseudo linguagem é mais utilizada para fazer algoritmos que tem como objetivo o uso na informática, para que depois seja transcrita nas diversas linguagens de programação. Então, os algoritmos criados em pseudo linguagem devem ser independentes das linguagens de programação, sendo assim um pouco menos restritivas que as linguagens de programação pois cada linguagem tem suas peculiaridades e formalismo.

Por ser o meio termo entre a codificação e a linguagem escrita é de fácil interpretação e de codificação, como a própria ambiguidade de sua definição sugere, é possivel chamar de pseudo linguagem ou pseudo código, é um intermédio entre a linguagem padrão e a linguagem de programação. Não existe nenhuma padronização, mas existem uma série de recomendações e boas práticas para escrever um algoritmo em pseudo linguagem, dentre elas destacam-se:

- Use um comando por linha;
- Imagine que o algoritmo será lido por pessoas que não são da área da informática ou de exatas;
- Use frases breves;
- Seja objetivo;
- Use palavras claras para evitar duplo sentido;
- O algoritmo da pseudo linguagem precisa funcionar;

Código

É o algoritmo escrito em alguma linguagem já existente como Java, Cobol, C e outras, seguindo as peculiaridades e formalismo da linguagem em que o algoritmo é escrito. É bom

lembrar que o algoritmo é compilado e transformado depois em "linguaguem de máquina" do respectivo computador, celular, tablet, e muito mais.

1.5 Composição dos algoritmos

Os algoritmos podem ser extremamente simples mas a grande maioria dos algoritmos são mais complexos e usam alguns componentes e instrumentos que serão especificados abaixo. Vale lembrar que os algoritmos não são estruturas fixas ou engessadas, existem algoritmos que terão todos os componentes, outros algoritmos terão alguns ou apenas um destes componentes e os mais simples nenhum dos instrumentos citados, isso vai depender do que será feito e de como o algoritmo é escrito pois duas pessoas podem escrever algoritmos para a mesma função ou finalidade com estruturas bem diferentes. Veja alguns dos componentes ou instrumentos a seguir.

1.5.1 Operadores e Expressões

As expressões são na verdade instruções constituídas por operadores onde o resultado é dependente dos valores atribuídos aos operadores. Os operadores, como o próprio nome já revela, são estruturas que fazem operações e essas operações podem ser aritméticos, lógicos,relacionais e de atribuição.

Operadores aritméticos são comumente usados em expressões que fazem cálculos, os operadores lógicos basicamente darão resultados como "verdadeiro" e "falso" nas expressões em que aparecem, os operadores relacionais fazem uma comparação entre os termos utilizados em uma relação, e o operador de atribuição, cujo simbologia utilizada será "**:=**", defini um dado inicial ou sobrescrever um dado em uma variável. A tabela 1.2 exibe os principais operadores indicando os tipos e símbolos:

Tabela 1.2: **Principais Operadores**

Operação	Tipo	Símbolo
Adição	Aritmético	+
Subtração	Aritmético	-
Multiplicação	Aritmético	*
Divisão	Aritmético	/
Potência	Aritmético	^
conjunção	lógico	AND (E)
Disjunção	lógico	OR (OU)
Negação	Lógico	NOT(NÃO)
Maior	Relacional	>
Maior ou igual	Relacional	>=
Menor	Relacional	<
Menor ou igual	Relacional	<=
Igual	Relacional	=
Diferente	Relacional	<>
Prioridade	Relacional	()

Assim como nas expressões matemáticas os algoritmos tem uma ordem de prioridade de seus operadores que vai das mais altas prioridades até a mais baixa, sendo que as de maior prioridade são analisadas e processadas primeiro. Veja a ordem de prioridade dos operadores:

1. Parênteses: ( )
2. Potência: ^
3. Multiplicação e divisão: * e /
4. Adição e subtração: + e -

5. NOT (NÃO)

6. AND (E)

7. OR (OU)

Observe o exemplo para visualizar as prioridades:

$$\begin{aligned}(-1+(2^2-5*6))\div(-5+2)+1 &= (-1+(4-30))\div(-3)+1\\ &= (-1+(-26))\div(-3)+1\\ &= (-1-26)\div(-3)+1\\ &= -27\div-3+1\\ &= 9+1\\ &= 10\end{aligned} \tag{1.1}$$

1.5.2 Variáveis e constantes

Os algoritmos são estruturas que basicamente manipulam dados para chegar a um resultado ou solução e existem duas formas de trabalhar com dados em um algoritmo que são as constantes e as variáveis.

Uma variável faz referência a um dado que pode ser modificado à medida que o algoritmo é executado. Em computadores e componentes eletrônicos mais sofisticados a variável é um espaço na memória alocado para armazenar um tipo de dado e para isso as variáveis precisam ser nomeadas para que possa utilizá-la no decorrer do algoritmo. É preciso especificar o tipo de dados que esta variável pode receber e operar pois não é possível, por exemplo, operar uma variável do tipo " texto " com uma do tipo " numérica ".

Uma constante, como o próprio nome sugere, é um valor que não vai mudar em toda a execução do algoritmo, um valor fixo que sempre será o mesmo a qualquer momento do algoritmo. Em uma constante, assim como as variáveis, é preciso dizer a qual tipo de dados

ela pertence. Abaixo, a tabela 1.3 de alguns tipos de dados que variáveis e constantes podem assumir, na linguagem de programação Java, Observe:

Tabela 1.3: **Tipos de variáveis e constantes.**

Tipo	Descrição
Int	Números inteiros de 32 bits. Pode assumir valores entre -2.147.483.648 até 2.147.483.647.
Float	Representa números em notação de ponto flutuante, normalizada em precisão simples de 32 bits. O menor valor positivo é 1.40239846e-46 e o maior é 3.40282347e+38.
Boolean	Pode assumir dois valores. true e false.
String	Armazena caractere em notação de 16 bits. Serve para a armazenagem de dados alfanuméricos. Também pode ser como dado inteiro com valores na faixa 0 e 65535.
Byte	Aloca número de 8 bits. Pode assumir valores entre $-2^7 = -128$ e $2^7 - 1 = 127$.
Short	Armazena números inteiros de 16 bits em notação de complemento de dois. Os valores possíveis cobrem a faixa de $-2^{-15} = -32.768$ a $2^{15} - 1 = 32.767$.
Long	Aloca inteiros de 64 bits. Pode assumir valores entre -2^{63} e $2^{63} - 1$.
Double	Representa números em notação de Ponto flutuante normalizada com a precisão dupla de 64 bits em conformidade com a norma IEEE 754-1985. O menor valor positivo representável é 4.94065645841246544e-324 e o maior é 1.7976931348623157e+308.

Disponível em http://trtrfdfd.blogspot.com/2012/11/tipos-de-dados-Java.html

O bit é a menor unidade de informação que pode ser armazenada ou transmitida, assumi somente os valores 0 ou 1. Olhando na tabela, observe que uma variável do tipo byte, que tem 8 bits, pode assumir $2^8 = 256$ valores, dos quais são 128 negativos, 127 positivos e o zero.

Em um programa as variáveis e constantes ao receberem um tipo de dado são então declaradas, essa ação de especificar o tipo de dados das variáveis e constantes é chamada de declaração. Em alguns casos especifica-se um valor inicial ao declarar a variável.

1.5.3 Condicionais

A estrutura condicional nos possibilita a seleção de uma ação ou um grupo destas a ser executada através de condições, representadas por expressões lógicas.

Comando: SE

O comando SE é usado da seguinte forma, se uma condição é verdadeira uma sequência de comandos é executada, e se for falsa, outra sequência de comandos é executada. O comando SE pode ser simples ou composto como poderemos ver na síntaxe abaixo.

A sintaxe da estrutura condicional simples é:

Se (condição) então

(sequência de comandos)

FimSe

A sintaxe da estrutura condicional composta é:

Se (condição) então

(sequência de comandos A) senão

(sequência de comandos B) senão

(sequência de comandos C) senão

Senão...

Algoritmo 1: Exemplo: O algoritmo compara dois números e identifica o maior deles.

Entrada: n1,n2

Saída: resultado da comparação

```
1  início
2      ler (n1);
3      ler (n2);
4      se (n1 > n2) então
5          Escrever (" O primeiro número é o maior ");
6      senão
7          se (n1 < n2) então
8              Escrever (" O primeiro número é o menor ");
9          senão
10             Escrever (" Os números são iguais ");
11         fim
12     fim
13 fim
```

Comando CASO

O comando CASO é utilizado quando a condição avaliada tem valores diferentes de verdadeiro ou falso. Os comandos são executados de acordo aos resultados da expressão condicional. A sintaxe do comando CASO é:

CASO (Expressão)

(Opção 1): (sequência de comandos 1)

(Opção 2): (sequência de comandos 2)

...

(Opção n): (sequência de comandos N)

Ele funciona assim, caso o resultado da expressão seja igual a opção 1 execute a sequência de comandos 1, caso o resultado da expressão seja igual a opção 2 execute a sequência 2 e assim por diante.

Algoritmo 2: Exemplo: Calcular o dobro ou o triplo de um número fornecido.

Entrada: opção, num

Saída: resultado

```
1  início
2      Escrever (“ Opções: ”);
3      Escrever (“ 1 para calcular o dobro do número ”);
4      Escrever (“ 2 para calcular o triplo do número ”);
5      Escrever (“ Escolha uma opção: ”);
6      Ler (Opção);
7      Escrever (“ Digite o número: ”);
8      Ler (Num);
9      caso (Opção = 1) faça
10         Resultado := Num*2;
11     fim
12     caso (Opção = 2) faça
13         Resultado := Num*3;
14     fim
15     Escrever (Resultado);
16 fim
```

1.5.4 Repetição

Os comandos de repetição dá a possibilidade de repetir uma sequência de comandos até que uma condição o interrompa.

Comando: ENQUANTO

É um comando de repetição que só ira parar quando a condição estipulada for falsa.

Sintaxe:

ENQUANTO (condição) FAÇA

(sequência de comandos)

FimENQUANTO

Algoritmo 3: Exemplo: Algoritmo que mostra os números de 1 a 50.

Entrada: não tem entrada

Saída: números de 1 a 50

```
1 início
2     Contador := 1;
3     enquanto (Contador <= 50) faça
4         Escrever(Contador);
5         Contador := Contador + 1;
6     fim
7 fim
```

Comando: PARA

O comando PARA executa o código um número determinado de vezes.

Sintaxe:

PARA Contador := ValorInicial ATÉ ValorFinal FAÇA

(Sequência de comandos)

FimPARA

Algoritmo 4: Exemplo: Algoritmo que mostra os números de 1 a 50.

Entrada: não tem entrada

Saída: números de 1 a 50

```
1 início
2     Contador := 1;
3     para Contador := 1 até 50 faça
4         Escrever(Contador);
5     fim
6 fim
```

Comando: REPITA

O REPITA é uma estrutura de repetição bem parecida com o ENQUANTO, ele também precisa de uma condição para finalizar. A estrutura do comando REPITA garante que pelo menos uma passagem será feita por ele já que, ao contrário do ENQUANTO, o teste de condições é feito no final da estrutura.

Sintaxe:

REPITA

(Sequência de comandos)

ATÉ (condição)

Algoritmo 5: Exemplo: Algoritmo que mostra os números de 1 a 50.

Entrada: não tem entrada

Saída: números de 1 a 50

```
1 início
2     Contador := 1;
3     repita
4         Escrever(Contador);
5         Contador := Contador + 1;
6     até (Contador >= 50);
7 fim
```

Os tópicos acima apresentados cobrem o que será utilizado neste trabalho e é suficiente para entender o que iremos utilizar no próximo capítulo. Para aprofundar-se nas estruturas, funcionamento, tipos, e outras características de maneira mais imersiva, que não é o objetivo aqui, consultar referência bibliográfica [13] [14] [15].

Capítulo 2

Alguns algoritmos matemáticos

Depois de falar como se estrutura um algoritmo mostrando suas variáveis, representações, operações, e outros, este Capítulo 2 apresenta o funcionamento de alguns algoritmos matemáticos através de pseudolinguagem e/ou fluxograma e os respectivos códigos em linguagem de programação Java. Assim, além de ver o algoritmo em passos, irá vê-lo funcionando na prática.

Caso haja interesse em se aprofundar no código Java aqui apresentado, existem vários livros, cursos e sites na internet. Vale salientar que os algoritmos apresentados neste trabalho usaram apenas operadores, variáveis, estruturas de repetição e condicionais a fim de tornar o entendimento dos algoritmos mais fácil e didático.

Muitos dos algoritmos matemáticos existentes já estão implementados na biblioteca de objetos do Java e não seria interessante utilizá-los para o objetivo deste trabalho, que é o entendimento dos códigos, a utilização desses objetos do Java dificultaria o entendimento do leitor.

2.1 Divisão Euclidiana

A divisão euclidiana que também é conhecida como divisão inteira ou divisão com resto, é um dos temas essenciais dos assuntos de matemática visto nos primeiros anos de aprendizado,

os alunos são introduzidos a tais conceitos através da exemplificação e apresentações de situações - problemas como dividir uma pizza de 10 pedaços em 4 pratos, distribuir uma certa quantidade x de objetos para uma quantidade y de pessoas, dentre muitas outras situações.

É uma operação que os alunos apresentam muita dificuldade, e isso não ocorre apenas no ensino fundamental mas também no ensino médio, pois chegam muitos alunos em salas de aulas com essa dificuldade. Uma operação básica, que mesmo assim, quase sempre, não é completamente dominada pelos alunos [18][19]. Desconfio que a falta da prática da tabuada, como era na minha época, seja um entre vários outros motivos, para tal realidade, que acaba acarretando na falta do domínio da tabela de multiplicação de cor, além as regras básicas de cálculo como adição, subtração, decomposição de números inteiros e diferenciação de números pares e números ímpares, e muito mais, mas essa discussão não cabe aqui no escopo deste trabalho.

A divisão euclidiana é aprendida como uma das quatro operações básicas da matemática na educação básica no Brasil. E o conceito por trás da divisão é na verdade subtrações consecutivas até que se sobra um número menor que o divisor. Veja a figura 2.1.

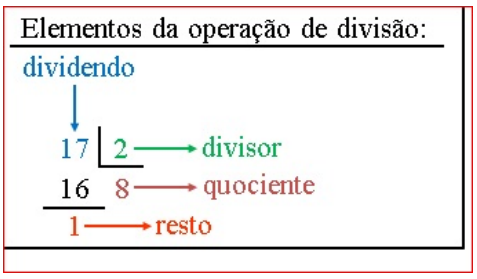


Figura 2.1: Método da chave, fonte:https://brainly.com.br/tarefa/4467950

Dividendo (D): Número que será dividido.

Divisor (d): Número que divide.

Quociente (q): Resultado da divisão.

Resto (r): Algumas vezes, assim que finalizada a divisão, sobra uma quantidade que não pode ser dividida. Esta quantidade recebe o nome de resto.

A partir dos elementos acima, a divisão será definida da seguinte maneira:

$D = d.q + r$, onde D,d e q são naturais e r é um natural com $0 \leq r < d$. Onde para dividir D por d temos que achar um número q que multiplicado por d chegue em D ou o mais próximo possível de D, sendo que o valor que sobrará será o resto r que não pode ser maior que d. Usando esse raciocínio criou-se o método da chave, o método mais usado na Educação básica para ensinar o algoritmo da divisão.

Teorema [16]

Se a, $b \in \mathbb{N}$, com $b > 0$, então existem dois únicos naturais q e r tais que $a = bq + r$, com $0 \leq r < b$.

Demonstração

Existência:

Se $a < b$, então existem $q = 0$ e $r = a$ nas condições exigidas. Assim pode-se assumir que $a \geq b > 0$. Considere que $\mathbb{N}$ não tem o elemento zero e considere também o conjunto $S = \{a - by \mid y \in \mathbb{N}\} \cap (\mathbb{N} \cup \{0\})$. Como $a - b \geq 0$, tem-se que $a - b \in S$, logo o conjunto S é não vazio. Como S é limitado inferiormente, pelo principio da boa ordenação[35], tem-se que S possui um menor elemento, digamos que $r = a - bq \geq 0$. Será mostrado que $r < b$.

Suponha por absurdo que $r \geq b$. Segue-se então que $r = s + b$ com $s \in \mathbb{N} \cup \{0\}$, logo $s = r - b = a - bq - b = a - (q + 1)b \in S$. Note que s é igual a $a - bq - b = r - b$. Assim $s < r$ e $s \in S$. Isto é uma contradição pois r é o menor elemento de S.

Unicidade:

Suponha $a = bq + r = bq' + r'$, sendo $0 \leq r$ e $r' < b$. Sem perda de generalidade, suponha $r \leq r'$. Daí tem-se que $0 \leq b(q - q') = r' - r < b$, o que só é possível se $q = q'$ e consequentemente $r = r'$, Caso r seja maior que r' a prova é análoga. ■

Assim os números q e r são chamados, respectivamente, de quociente e resto da divisão de a por b. A partir da prova pode se gerar um algoritmo, por subtrações sucessivas, $a - b$, $a - 2b$, $a - 3b$, ..., e em cada iteração consulta-se o resultado até que o resto seja menor que o divisor.

2.1.1 Fluxograma

Na Figura 2.2 temos o fluxograma do algoritmo utilizado.

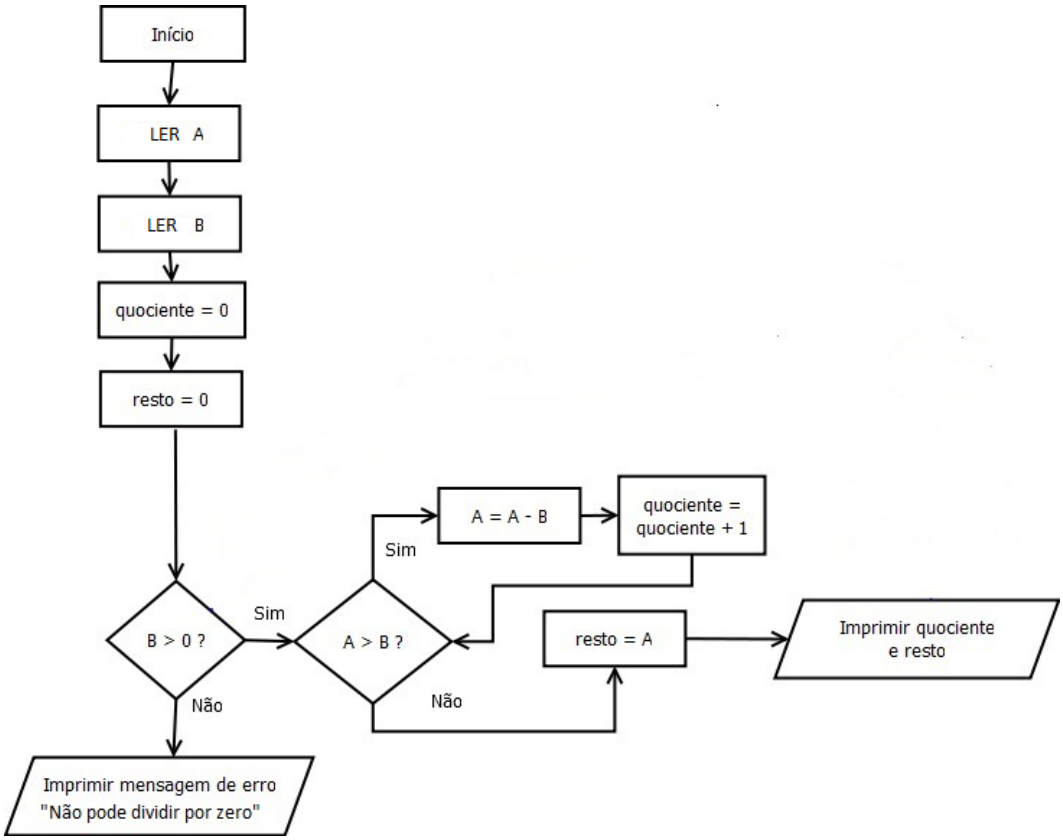


Figura 2.2:

2.1.2 Algoritmo

Algoritmo 6: O Algoritmo abaixo é baseado no fluxograma apresentado na Figura 2.2.

Entrada: A,B,quociente, resto

Saída: quociente, resto

```
1  início
2      Escrever (“ Escreva o Divisor ”);
3      Ler (A)
4      Escrever (“ Escreva o Dividendo ”);
5      Ler (B)
6      quociente := 0 ;
7      resto := 0 ;
8      enquanto (resto >= B) faça
9          resto := resto - B;
10         quociente := quociente + 1;
11     fim
12     Escrever (“O quociente é ”+ quociente + “ e o resto é” + resto );
13 fim
```

2.1.3 Código em Java

```
1  import java.util.Scanner;
2  import javax.swing.JOptionPane;
3  public class Divisao {
4      public static void main(String[] args) {
5        // TODO Auto-generated method stub
6  long a,b,q,r;
7  Scanner ler = new Scanner(System.in);
8  System.out.print("Escreva o dividendo: ");
9  a = ler.nextLong();
```

```
10 System.out.print("Escreva o divisor: ");
11 b = ler.nextLong();
12 q = 0;
13 r = a;
14 System.out.println();
15 System.out.println("passos");
16 System.out.println();
17 while (r >= b) { //garante que o resto é menor que o divisor
18    r = r - b;
19    q = q + 1;
20    System.out.print(a);
21    System.out.print(" = ");
22    System.out.print(b);
23    System.out.print(".");
24    System.out.print(q);
25    System.out.print(" + ");
26    System.out.println(r);
27 }
28 System.out.println();
29 System.out.print("O quociente da divisão de ");
30 System.out.print(a);
31 System.out.print(" por ");
32 System.out.print(b);
33 System.out.print(" é ");
34 System.out.print(q);
35 System.out.print(" e o resto é ");
36 System.out.println(r);
37      }
38
39 }
```

2.1.4 Resultados

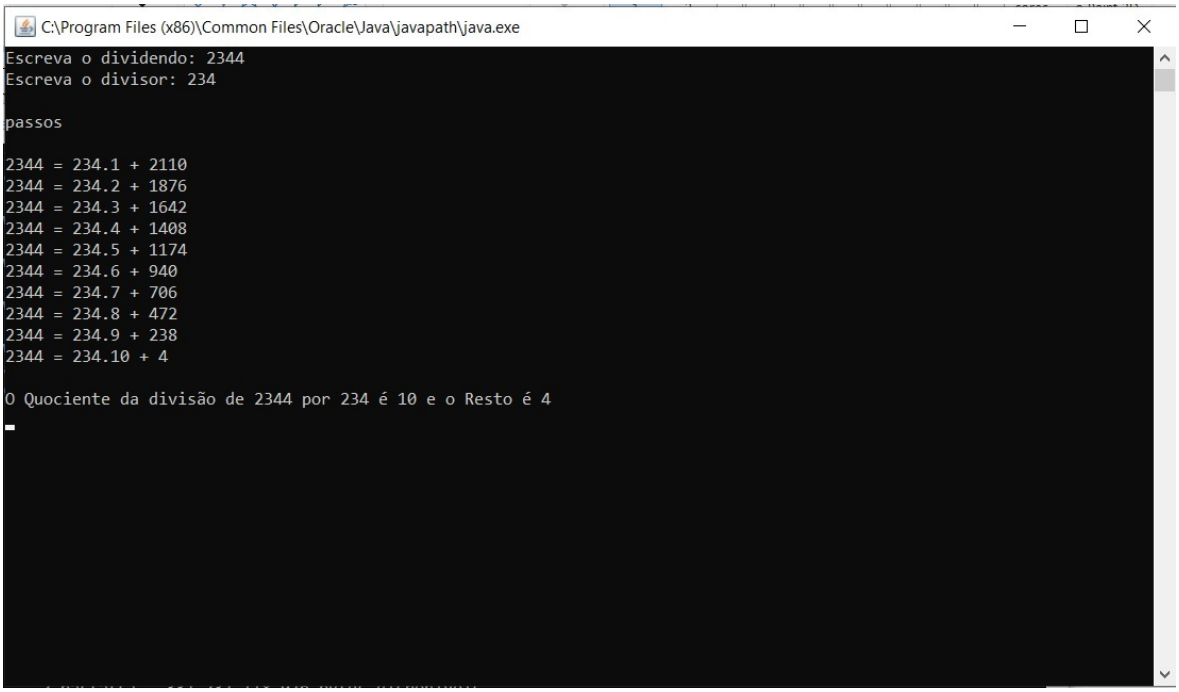


Figura 2.3: Resultado da divisão usando 2344 como dividendo e 234 como divisor.

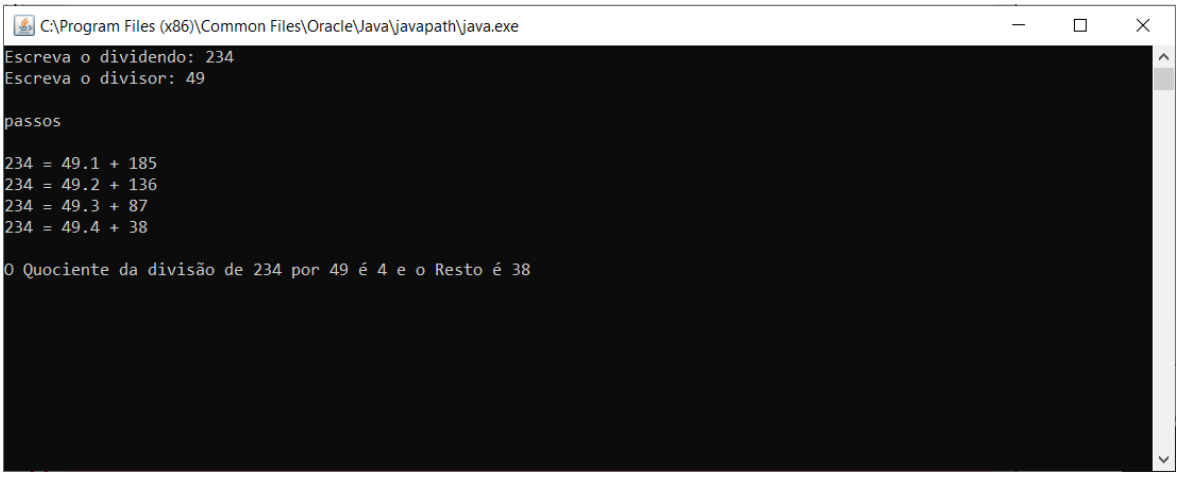


Figura 2.4: Resultado da divisão usando 234 como dividendo e 49 como divisor.

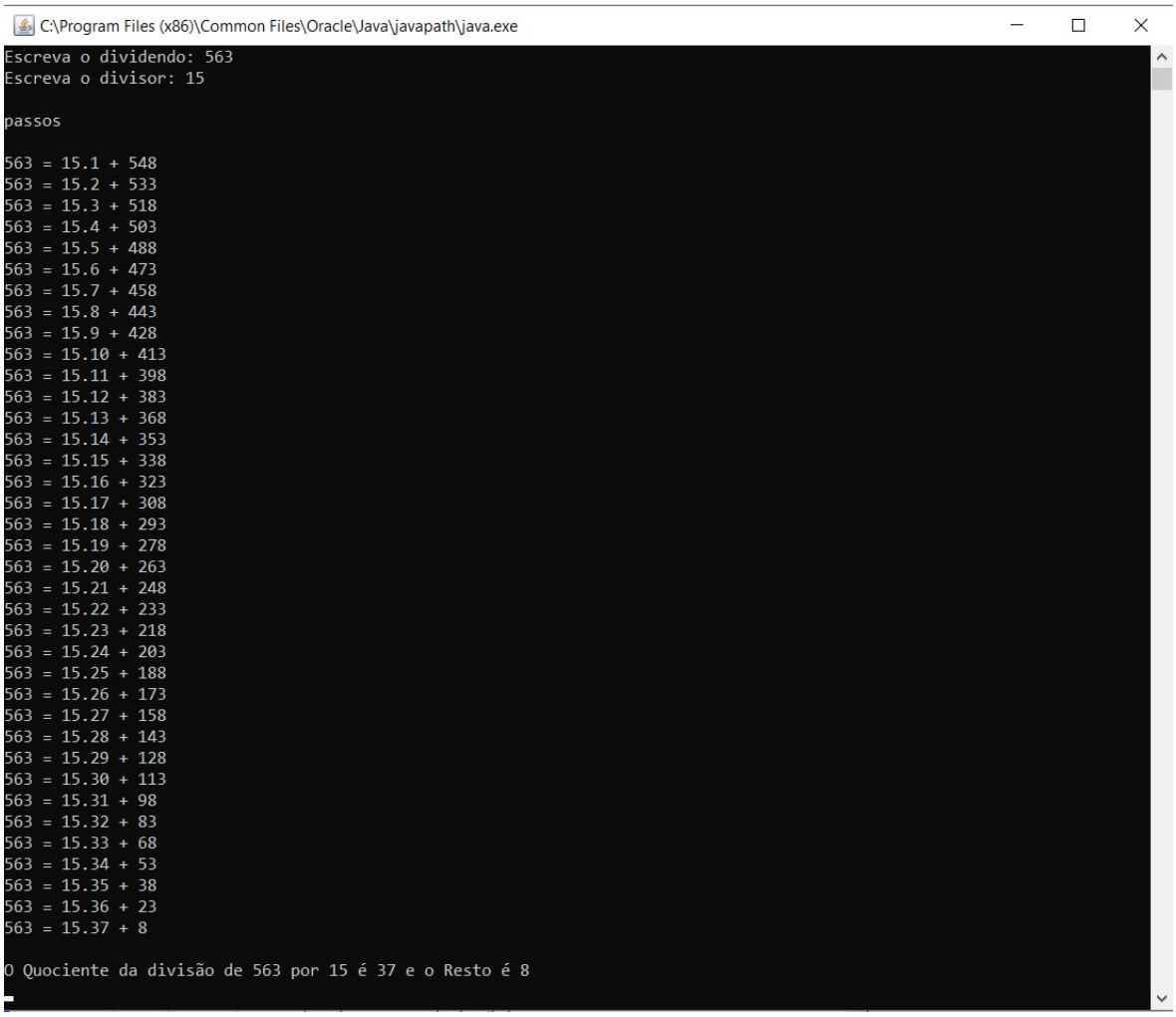


Figura 2.5: Resultado da divisão usando 563 como dividendo e 15 como divisor.

2.2 Algoritmo de Euclides para determinar MDC(Máximo Divisor Comum)

O ensino do conceito do MDC (Máximo Divisor Comum) e o método de encontrá-lo acontece no ensino fundamental, período em que as crianças estão vivenciando algumas experiências do seu dia a dia que utilizam o raciocínio da divisão de dois ou mais elementos em partes iguais, ou seja, justamente o conceito de MDC.

O que acontece, é que, este conhecimento é passado em um contexto apenas Matemático

e os alunos não percebem que o MDC é um ferramenta muito importante para que resolvam situações de sua vida cotidiana.

Os professores devem instigar os alunos a perceberem a importância deste algoritmo por meio de situações problemas contextualizando com acontecimentos de sua vida cotidiana.

A decomposição de um número natural em fatores primos é uma estratégia para se conseguir determinar o MDC de dois ou mais números naturais. Veja o exemplo:

Determine o MDC de 48,36 e 30

Fatoramos os números:

$48 = 2^4.3^1$

$36 = 2^2.3^2$

$30 = 2^1.3^1.5^1$

O MDC será o resultado da multiplicação dos números que aparecem em todas as fatorações sempre com o menor expoente, que no exemplo acima são os números 2^1 e 3^1. Observe que no exemplo aparecem 2^4, 2^2 e 2^1, escolhendo o de menor expoente, o 2^1, o mesmo ocorre para escolher o 3^1, como $2^1 = 2$ e $3^1 = 3$, então o MDC de 48, 36 e 30 será $2.3 = 6$

Mas, quando os números forem muito grandes este método ficará bastante complicado pois a decomposição de números muitos grandes pode ser bastante demorada e de pouca eficiência prática.

No livro sétimo dos Elementos de Euclides [5] há um método, apesar de terem evidências históricas que este método existisse antes do livro, que é o Algoritmo de Euclides para a obtenção do MDC entre dois números naturais.

2.2.1 Como o algoritmo funciona

Obtendo o [mdc] entre dois números naturais X e Y onde $X > Y$.

1) Divida X por Y e obtenha o resto R_1. Se R_1 for zero, o mdc entre X e Y é Y.

2) Se R_1 não for zero, divida Y por R_1 e obtenha o resto R_2. Se R_2 for zero, o mdc entre X e Y é R_1.

3) R_2 não for zero, divida R_1 por R_2 e obtenha o resto R_3. Se R_3 for zero, o mdc entre X e Y é R_2.

...

Se R_n não for zero, divida R_{n-1} por R_n e obtenha o resto R_{n+1}. Se R_{n+1} for zero, o mdc entre X e Y é R_n

Observe abaixo a exemplificação de como funciona o algoritmo.

Calcular mdc (84,76):

$mdc(84, 76)$

$84 = 76.1 + 8$

$mdc(84, 76) = mdc(76, 8)$

$76 = 8.9 + 4$

$mdc(76, 8) = mdc(8, 4)$

$8 = 4.2 + 0$

$mdc(4, 0) = 4$

$mdc(84, 76) = 4$

Para calcular o MDC de mais de dois números naturais ou segmentos, basta escolher 2 desses números, aplica-se o algoritmo, o resultado será aplicado com o próximo número até que se chegue na resposta final que será o MDC de todos os números. Veja a seguir:

Calcular MDC de (30,40,60,75)

MDC de (30,40) = 10

MDC de (10,60) = 10

MDC de (10,75) = 5

então o MDC de (30,40,60,75) = 5

2.2.2 Fluxograma

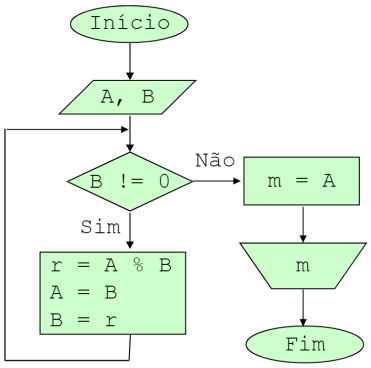


Figura 2.6: Disponível em https://slideplayer.com.br/slide/5591740/

2.2.3 Algoritmo

Algoritmo 7: O Algoritmo MDC

Entrada: A,B,r

Saída: m

```
1  início
2      Escrever ("vamos calcular o MDC de: ");
3      Ler (A)
4      Ler (B)
5      enquanto ( B <> 0) faça
6          r:= A % B;
7          A:= B;
8          B:= r;
9      fim
10     m: = A;
11     Escrever ("O MDC será " + m);
12 fim
```

2.2.4 Código em Java

Observe o código.

```
1  import java.util.Scanner;
2  public class Mdc {
3     public Mdc() {
4        // TODO Auto-generated constructor stub
5     }
6     public static void main(String[] args) {
7        // TODO Auto-generated method stub
8  long a,b,r;
9  Scanner ler = new Scanner(System.in);
10 System.out.println("Digite os dois números para calcular o MDC");
11 a = ler.nextLong();
12 b = ler.nextLong();
13 System.out.print("MDC de (");
14 System.out.print(a);
15 System.out.print(",");
16 System.out.print(b);
17 System.out.print(")");
18 while(b != 0){
19    r = a % b;  // retorna o resto da divisão de a por b
20    a = b;
21    b = r;
22 }
23 System.out.print(" = ");
24 System.out.print(a);
25     }
26
27 }
```

2.2.5 Resultados

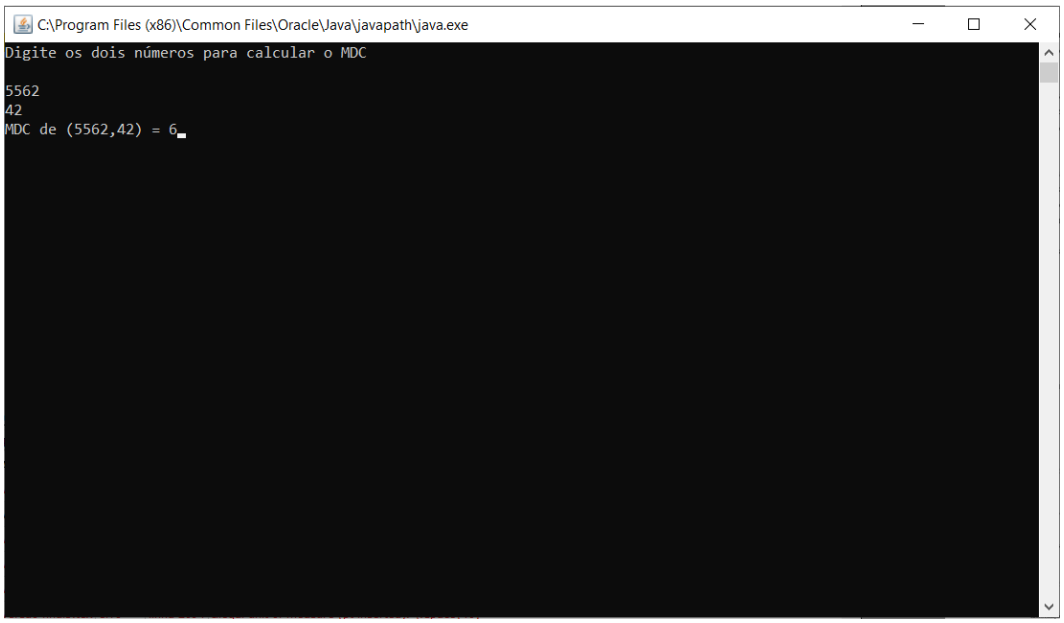


Figura 2.7: MDC de 5562 e 42.

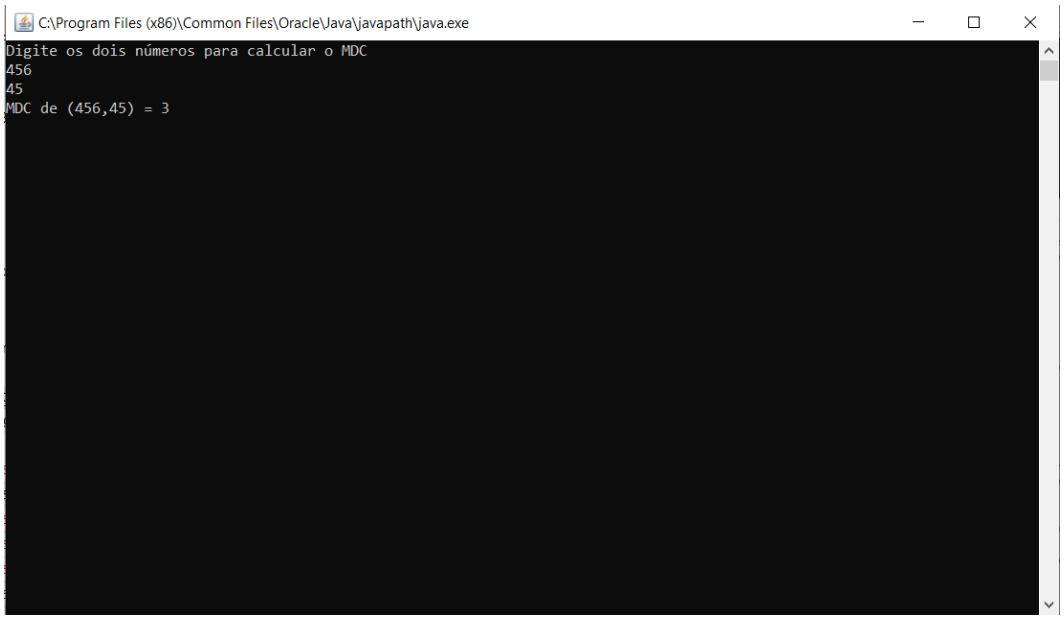


Figura 2.8: MDC de 456 e 45.

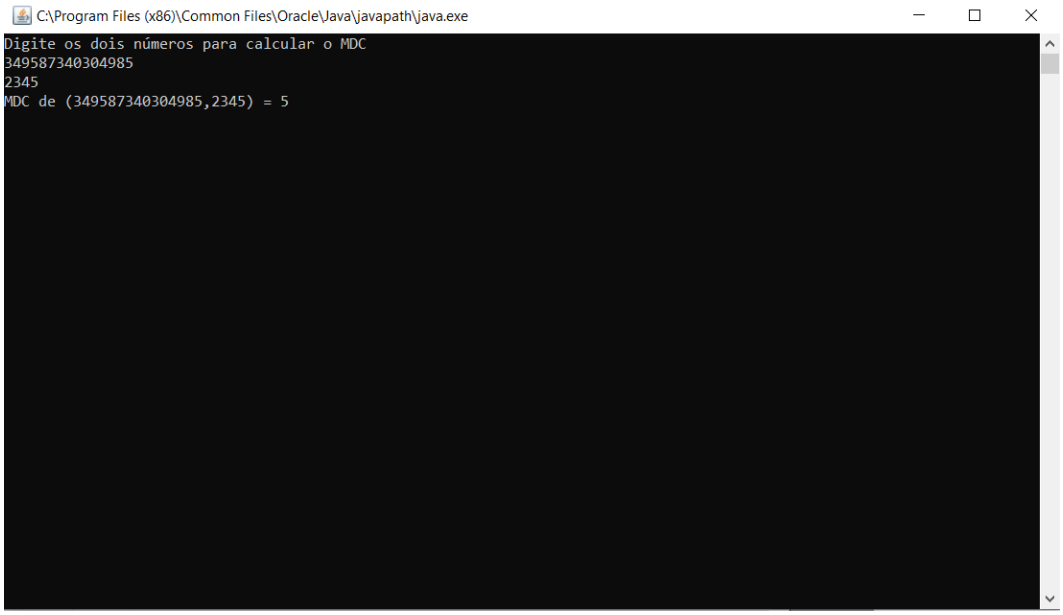


Figura 2.9: MDC de 349587340304985 e 2345.

2.3 Fórmula resolutiva da equação de segundo grau (Teorema de Bhaskara)

Problemas envolvendo a resolução de equações de 2º grau existem há muito tempo, existem textos com as resoluções destas equações muito antes da era cristã. Há relatos que foram achadas placas na mesopotâmia e papiros egípcios que datam de aproximadamente 4 mil anos[11].

> "Babilônios e egípcios utilizavam-se de textos e símbolos como ferramenta auxiliar na resolução. Os gregos conseguiam concluir suas resoluções realizando associações com a geometria, pois eles possuíam uma forma geométrica para solucionar problemas ligados a equações do 2º grau.
>
> Dentre os indianos, os matemáticos Sridhara, Bramagupta e Bhaskara também contribuíram para o desenvolvimento da Matemática, fornecendo importantes informações sobre as equações do 2º grau. Sridhara foi o primeiro a estabelecer uma

fórmula matemática para a resolução das equações biquadradas, pois Bramagupta e Bháskara trabalhavam utilizando textos.

Os árabes foram brilhantemente representados por al-Khowarizmi, que se baseando no trabalho dos gregos, criou metodologias para a resolução de equações do 2^o grau. A representações geométricas utilizadas por al-Khowarizmi são influenciadas por Euclides.

Foi com o francês Viète que o método resolutivo das equações do 2^o grau ganharam como símbolos, as letras. Viète é o responsável pela modernização da álgebra. Seus trabalhos foram desenvolvidos por outro francês, denominado René Descartes."[27]

Uma observação interessante é que a grande maioria dos alunos conseguem identificar instantaneamente uma equação do segundo grau, mas não conseguem resolvê-las. Acredito que a causa seja o fato dele ser um pouco longo, e infelizmente não desperta interesse nos alunos. É um algoritmo bastante usado no cotidiano de vários profissionais. Veja os Exemplos:

- Na engenharia é usada para estudar lançamentos e projéteis em trajetória parabólicas;
- Em física, nos movimentos uniformemente variados, lançamentos, queda livre, entre outros;
- Em administração ou economia, pode ser usada para descobrir o lucro máximo de uma empresa.

Demonstração da fórmula de resolução da equação do segundo grau:

A demonstração desta fórmula consiste basicamente em completar quadrados, veja:

$f(x) = ax^2 + bx + c$, onde a é diferente de zero, logo $ax^2 + bx + c = 0$, passo o termo independente c para a esquerda $ax^2 + bx = -c$, divido tudo por a

$$\mathrm{x}^2 + \frac{bx}{a} = -\frac{c}{a} \tag{2.1}$$

Agora temos que achar o valor que somado a $x^2 + \frac{bx}{a}$ forme um quadrado perfeito da forma $(x+y)^2 = x^2 + 2xy + y^2$, então o termo que deve somar a $x^2 + \frac{bx}{a}$ para que a soma seja um quadrado perfeito é:

$$\begin{aligned} 2\text{xy} &= \frac{bx}{a} \\ 2\text{y} &= \frac{b}{a} \\ \text{y} &= \frac{b}{2a} \end{aligned} \tag{2.2}$$

logo

$$(\text{x} + \frac{b}{2a})^2 = x^2 + \frac{bx}{a} + \frac{b^2}{4a^2} \tag{2.3}$$

acrescento $\frac{b^2}{4a^2}$ nos dois lados da equação 2.1, temos

$$\begin{aligned} \text{x}^2 + \frac{bx}{a} + \frac{b^2}{4a^2} &= -\frac{c}{a} + \frac{b^2}{4a^2} \\ (\text{x} + \frac{b}{2a})^2 &= -\frac{c}{a} + \frac{b^2}{4a^2} \\ (\text{x} + \frac{b}{2a})^2 &= \frac{(-4ac + b^2)}{4a^2} \\ \text{x} + \text{b/2a} &= \pm\sqrt{\frac{-4ac + b^2}{4a^2}} \\ \text{x} + \text{b/2a} &= \pm\frac{\sqrt{b^2 - 4ac}}{2a} \\ \text{x} = \text{-b/2a} &\pm\frac{\sqrt{b^2 - 4ac}}{2a} \\ \text{x} &= \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \blacksquare \end{aligned} \tag{2.4}$$

E, finalmente, obteve-se os dois valores de x, um usando o sinal positivo e outro o sinal negativo, vale observar também que foi encontrado $b^2 - 4ac$ de delta ($\triangle$), e ele é utilizado para saber se as raízes fazem ou não parte do conjunto dos números reais e se são ou não iguais.

2.3.1 Fluxograma

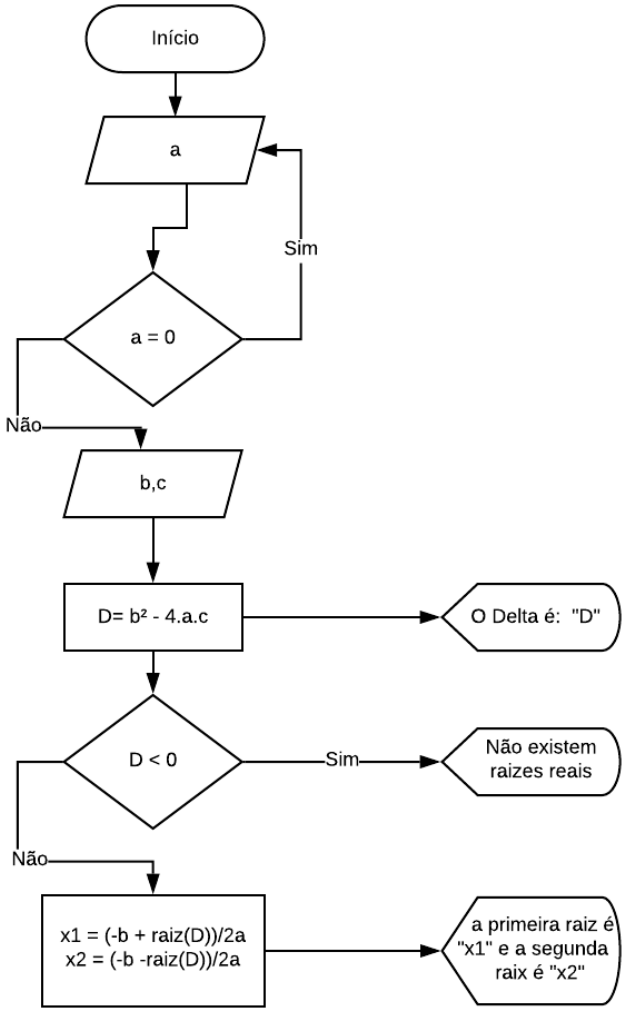


Figura 2.10: Equação do segundo grau

2.3.2 Algoritmo

Algoritmo 8: Equação do Segundo Grau

Entrada: a,b,c,

Saída: D,x1,x2

```
1  início
2      Escrever (Vamos resolver equações de 2º grau);
3      Escrever (Digite os coeficientes 'a' do termo x^2:);
4      ler (a);
5      enquanto (a = 0) faça
6          Escrever ( O termo 'a' não pode ser zero, digite outro valor.);
7          ler(a);
8      fim
9      Escrever (Digite o coeficiente 'b' do termo x:);
10     ler(b);
11     Escrever (Digite o termo independente 'c': );
12     ler(c);
13     D := b^2 - 4 * a * b
14     Escrever ( O delta é: );
15     Escrever (D);
16     se (D < 0) então
17         Escrever (Não existem raízes reais)
18     senão
19         Escrever ( primeira raiz é);
20         x1:= -b + (raiz(D))/2*a;
21         Escrever ( x1);
22         Escrever ( segunda raiz é);
23         x2:= -b - (raiz(D))/2*a;
24         Escrever (x2);
25     fim
26 fim
```

2.3.3 Código em Java

```
1  package equacao_2grau;
2  import java.util.Scanner;
3  public class Equacao2grau {
4      public Equacao2grau() {
5          // TODO Auto-generated constructor stub
6      }
7
8      public static void main(String[] args) {
9          // TODO Auto-generated method stub
10         double a,b,c,d,x1,x2;
11         Scanner ler = new Scanner(System.in);
12         System.out.print("Escreva o termo 'a': ");
13         a = ler.nextFloat();
14         while (a == 0) {
15            System.out.print(" O termo 'a' tem que ser diferente de
                  zero, digite outro valor:");
16            a = ler.nextFloat();
17         }
18         System.out.print("Escreva o termo 'b': ");
19         b = ler.nextFloat();
20         System.out.print("Escreva o termo independente 'c': ");
21         c = ler.nextFloat();
22         d = Math.pow(b,2) - 4*a*c;
23         System.out.print("O delta é: ");
24         System.out.println(d);
25         if (d < 0) {
26           System.out.print("A equação não tem raiz real");
27         } else {
28            x1 = (-b + Math.sqrt(d))/(2*a) ;
```

```
29              System.out.print("A raiz x1 é:");
30              System.out.println(x1);
31              x2 = (-b - Math.sqrt(d))/(2*a) ;
32              System.out.print("A raiz x2 é:");
33              System.out.print(x2);
34
35          }
36      }
37
38  }
```

2.3.4 Resultados

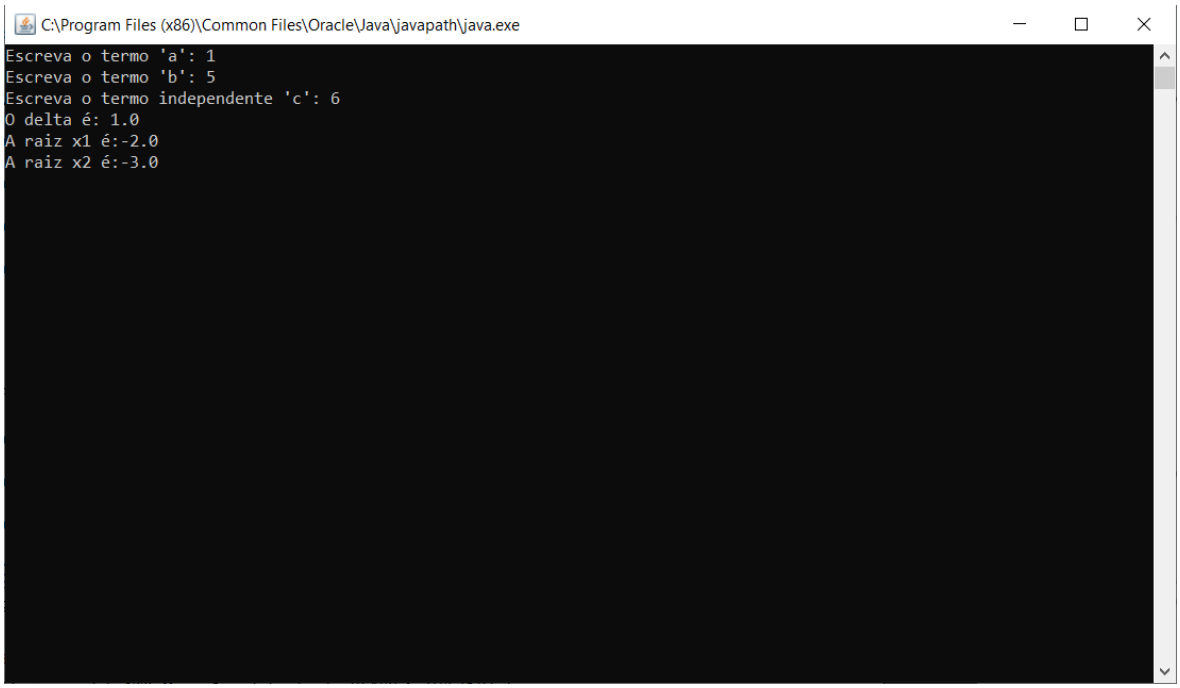


Figura 2.11: Raizes da equação: $x^2 + 5x + 6 = 0$.

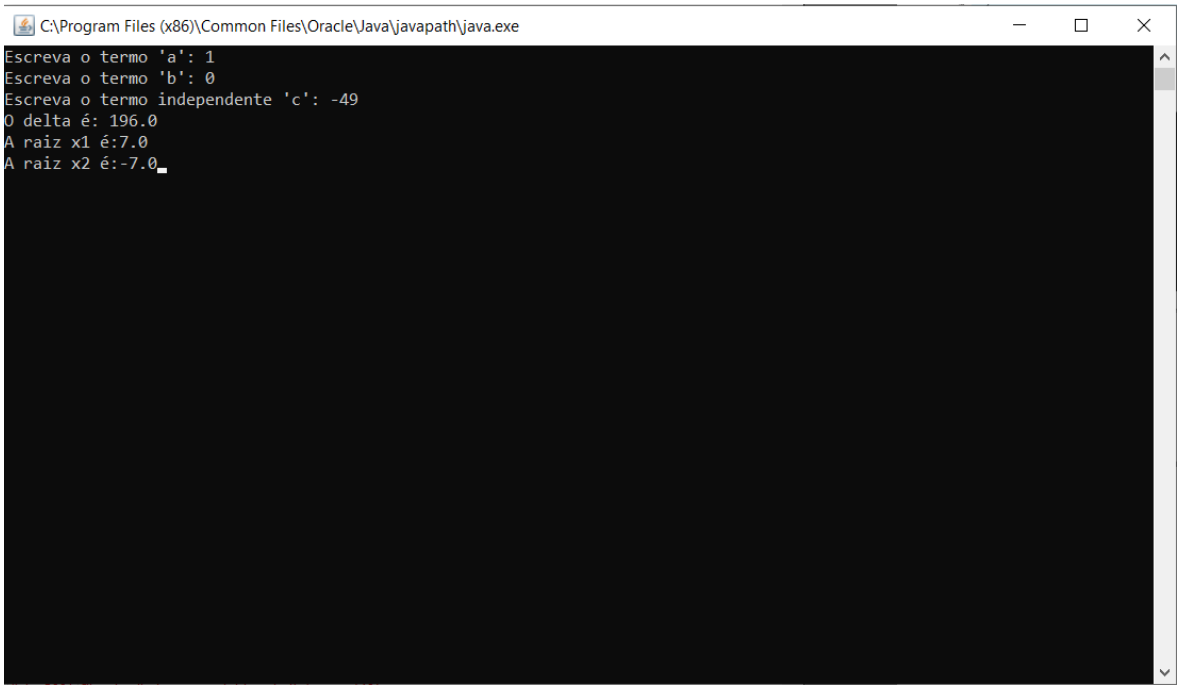


Figura 2.12: Raizes da equação: $x^2 - 49 = 0$.

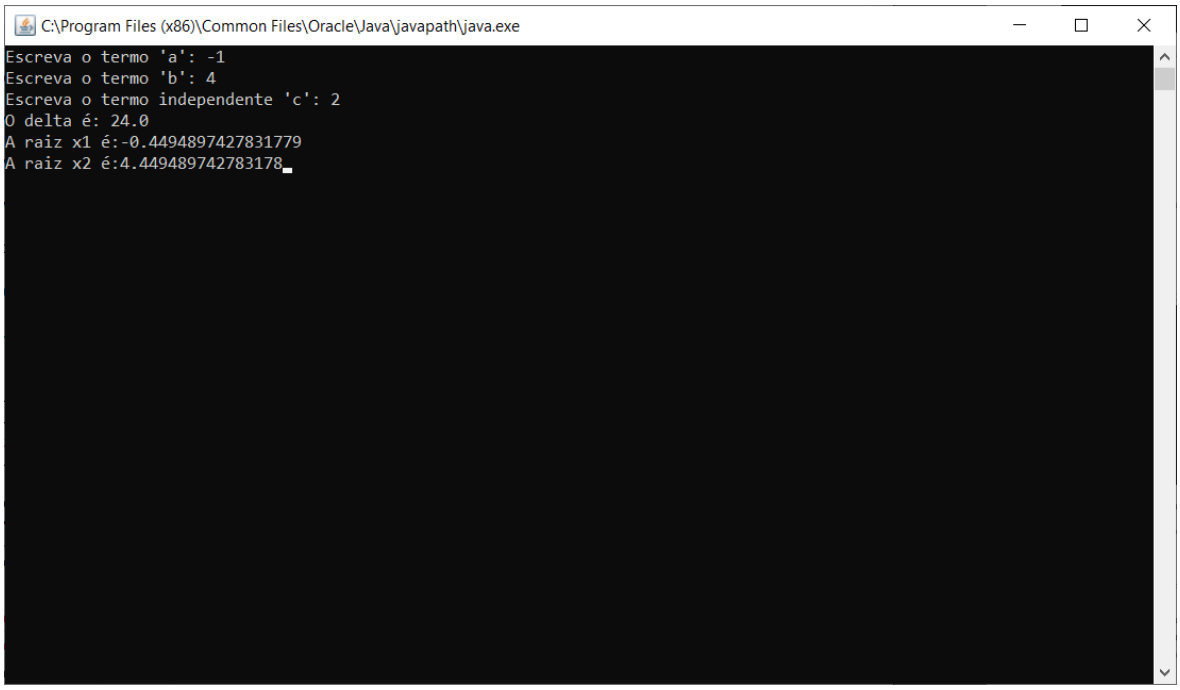


Figura 2.13: Raizes da equação: $-x^2 + 4x + 2 = 0$.

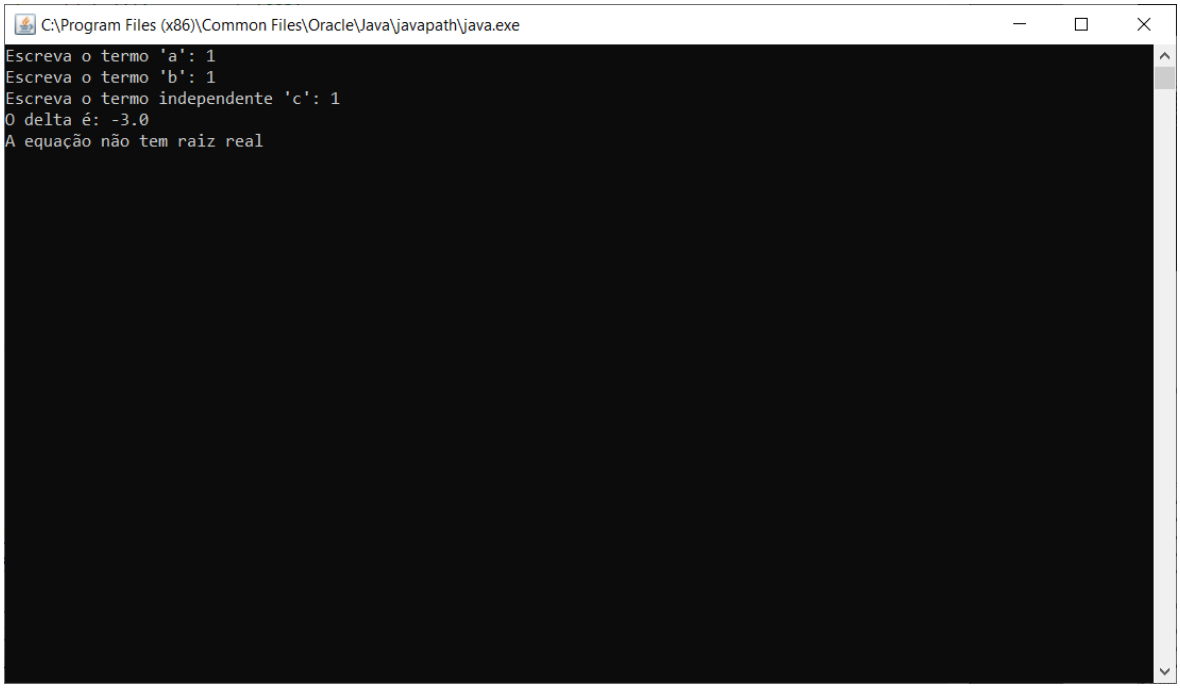


Figura 2.14: Raizes da equação: $x^2 + x + 1 = 0$.

2.4 Briot-Ruffini

A existência de aplicações utilizando equações polinomiais e as técnicas para calculá-las surgiram da necessidade de se ter resultados mais precisos em cálculos. A divisão de polinômio é uma das mais importantes ferramentas de cálculo já desenvolvidas. O algoritmo de Briot-Ruffini é usado principalmente para calcular limites e diminuir o grau da equações além de muitas outras utilidades.

O Algoritmo de Briot-Ruffini consiste em um método de divisão de um polinômio de qualquer grau por um binômio da forma $x - a$. Ele foi criado por Charles Auguste Briot e Paolo Ruffini e pode ser utilizado no ensino médio para resolução de uma enorme quantidade de problemas.

Veja um exemplo obtido da internet [17].

Sejam:

$P(x) = 3x^3 - 2x^2 + x + 5$

$D(x) = x + 1$

Queremos dividir P(x) por D(x).

Primeiro, desenhe dois segmentos de reta, um na horizontal e outro na vertical.

Figura 2.15: método de Briot-Ruffini

Colocar os coeficientes do polinômio P(x) no segmento de reta horizontal e à direita do segmento vertical e repetir o primeiro coeficiente na parte de baixo. No lado esquerdo do segmento vertical, deve-se colocar a raiz do binômio. Para determinar a raiz de um binômio, basta igualá-lo a zero, assim: $x + 1 = 0$, $x = -1$.

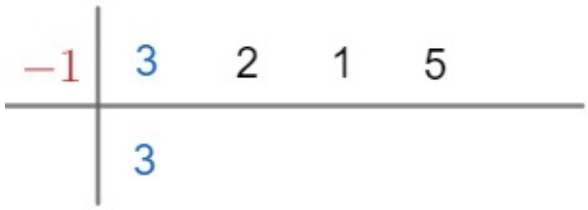


Figura 2.16: método de Briot-Ruffini

Multiplicar a raiz do divisor pelo primeiro coeficiente localizado abaixo da linha horizontal e, em seguida, somar o resultado pelo próximo coeficiente localizado acima da linha horizontal. Repetir o processo até o último coeficiente, no caso da Figura 2.15 o coeficiente será 5. Veja:

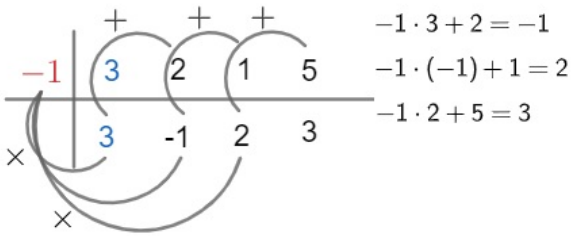


Figura 2.17: método de Briot-Ruffini

Após realizar esses três passos, será analisado o que o algoritmo fornece. Na parte superior da linha horizontal e à direita da linha vertical, temos os coeficientes do polinômio $P(x) = 3x^3 + 2x^2 + x + 5$.

O número -1 é a raiz do divisor e, portanto, o divisor é $D(x) = x+1$. Por fim, o quociente pode ser encontrado com os números localizados abaixo da linha horizontal, sendo o último número o resto da divisão.

Lembre-se de que o grau do dividendo é 3 e o grau do divisor é 1, portanto o grau do quociente é dado por $3 - 1 = 2$. Assim, o quociente é $q(x) = 3x^2 - x + 2$.

Observe novamente que os coeficientes são obtidos com os números abaixo da linha horizontal e que o resto da divisão é $R(x) = 3$.

Demonstração do Algoritmo de Briot-Ruffini

Dado um polinômio $P(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x^1 + a_0$ e o binômio $D(x) = x - d$ sabemos, pelo Teorema de D'Alembert, que a divisão de um polinômio de grau n dividido pelo binômio acima terá como quociente um polinômio um grau menor que P(x) ficando o quociente $Q(x) = q_{n-1}x^{n-1} + q_{n-2}x^{n-2} + ... + q_1 x^1 + q_0$ e sobrará um resto r.

Sabemos que $P(x) = Q(x).D(x) + r$ então

$a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x^1 + a_0 = (x - d).(q_{n-1}x^{n-1} + q_{n-2}x^{n-2} + ... + q_1 x^1 + q_0) + r$

$a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x^1 + a_0 = (q_{n-1}x^n + q_{n-2}x^{n-1} + ... + q0x) - (dq_{n-1}x^{n-1} + dq_{n-2}x^{n-2} + ... + dq_1 x^1 + dq_0) + r$

$a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x^1 + a_0 = q_{n-1}x^n + (q_{n-2} - dq_{n-1})x^{n-1} + ... + (q_0 - dq_1)x + r - dq_0$

como consequência teremos que

$a_n = q_{n-1}$

$a_{n-1} = q_{n-2} - dq_{n-1}$

.

.

.

$a_1 = q_0 - dq_1$

$a_0 = r - dq_0$

ou seja

$q_{n-1} = a_n$

$q_{n-2} = a_{n-1} + dq_{n-1}$

.

.

.

$q_0 = a_1 + dq_1$

$r = a_0 + dq_0$

Observe que as igualdades apresentadas anteriormente são justamente o que faz o algoritmo de Briot-Ruffini:

d	a_n	a_{n-1}	...	a_1	a_0
	q_{n-1}	q_{n-2}	...	q_0	r

Figura 2.18: algoritmo de Briot-Ruffini

como exemplo disso, observe que $q_{n-1}.d + a_{n-1} = q_{n-2}$ o mesmo acontece com os outros termos do polinômio, chega-se então ao fim do Algoritmo.

2.4.1 Algoritmo

Algoritmo 9: AlgoritmoBriotRuffini

Entrada: grau, raiz

Saída: polinomioq[ ]

```
1  início
2      Escrever("Escreva o grau do polinômio"); ler (grau);
3      poli := grau; cont := grau;
4      polinomiox.tamanho:= grau + 1; polinomioq.tamanho:= grau + 1;
5      enquanto (grau <> 0) faça
6          Escreva("Escreva o termo x elevado a" );
7          Escreva(grau);
8          Ler(polinomiox[grau]); grau:= grau -1;
9      fim
10     Escreva("Escreva o termo independente");
11     ler(polinomiox[0]);
12     Escreva("Escreva a raiz do binômio. ex:x + a, raiz = -a"); ler(raiz);
13     polinomioq[poli]:= polinomiox[poli]; poli:= poli -1;
14     enquanto (poli <> -1) faça
15         polinomioq[poli] = polinomioq[poli + 1]*raiz + polinomiox[poli];
16         poli = poli - 1;
17     fim
18     enquanto (cont <> 0) faça
19         Escreva("Termo x elevado a"); Escreva(cont);
20         Escreva(polinonioq[cont]); cont:= cont - 1;
21     fim
22     Escreva( "O resto é"); Escreva(polinomioq[0]);
23 fim
```

2.4.2 Código em Java

```
1  package ruffine;
2  import java.util.Scanner;
3  public class briot_ruffine {
4      public briot_ruffine() {
5          // TODO Auto-generated constructor stub
6      }
7
8      public static void main(String[] args) {
9          // TODO Auto-generated method stub
10
11     int grau;
12     int poli;
13     int cont;
14     Scanner ler = new Scanner(System.in);
15     System.out.println("Escreva o Grau do polinômio ");
16     grau = ler.nextInt() ;
17     poli = grau;
18     cont = grau;
19     double[] polinomiox = new double[grau + 1];
20     double[] polinomioq = new double[grau + 1];
21     while (grau != 0) {
22       System.out.print("Escreva o termo x elevado a ");
23       System.out.println(grau);
24       polinomiox[grau] =  ler.nextInt() ;
25       grau = grau - 1;
26       }
27     System.out.println("Escreva o termo x independente ");
28     polinomiox[0] =  ler.nextInt() ;
29     System.out.println("Escreva a raiz do binômio que dividirá o
```

```
        polinômio. ex: x + a,  raiz = -a ");
30      double raiz;
31      raiz = ler.nextDouble();
32      polinomioq[poli] = polinomiox[poli] ;
33      poli = poli - 1;
34      while (poli != -1) {
35        polinomioq[poli] = polinomioq[poli + 1]*raiz + polinomiox[poli];
36        poli = poli - 1;
37        }
38      while (cont != 0) {
39        System.out.print("Termo x elevado a ");
40        System.out.print(cont -1);
41        System.out.print(" =  ");
42        System.out.println(polinomioq[cont]);
43        cont = cont -1;
44        }
45      System.out.print("Resto = ");
46      System.out.println(polinomioq[0]);
47      }
48  }
```

2.4.3 Resultados

Figura 2.19: Divisão do polinômio $3x^4 - 4x^3 + 4x + 2$ por $x + 1$.

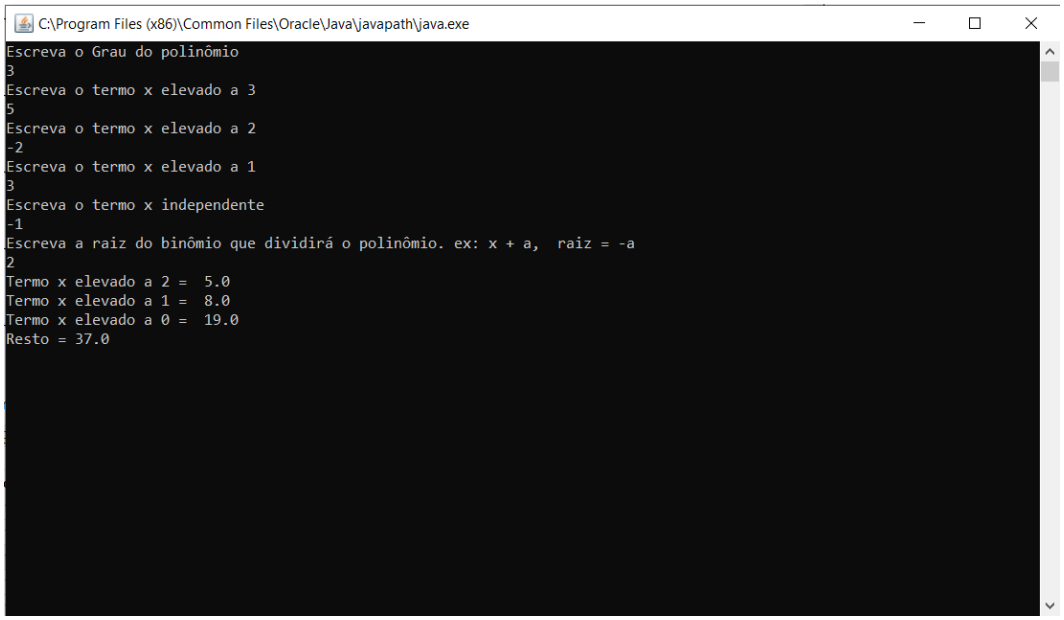


Figura 2.20: Divisão do polinômio $5x^3 - 2x^2 + 3x - 1$ por $x - 2$.

2.5 Sistema de numeração e mudança de base 10 para base 2

Usualmente utiliza-se o sistema de numeração decimal, ou seja, são dez símbolos (algarismos) para representar os números que são 0,1,2,3,4,5,6,7,8,9. Além disso, este sistema também é posicional o que implica que a posição do algarismo é importante, acompanhe o exemplo abaixo.

Número 235 (duzentos e trinta e cinco) é formado por três algarismos diferentes o 2, 3 e 5, neste sistema, a posição que o algarismo ocupa é importante pois é sabido que o 2 representa $2.100 = 200$, o 3 representa $3.10 = 30$ e 5 representa $5.1 = 5$ se trocada a ordem que os algorismos aparecem tem-se outro número, então é perceptível que a posição do algarismo é muito importante.

Mas não existe apenas o sistema decimal/posicional..., existem vários outros, e um que é bastante utilizado hoje em dia, devido ao avanço tecnológico é o sistema de numeração binário, onde existem apenas dois símbolos 0, 1. Esse sistema de numeração se encaixa perfeitamente em computadores e equipamentos eletrônicos diversos, pois em equipamentos eletrônicos só existem dois estados, "ligado" que significa que está passando energia pelo componente e "desligado" onde não está passando energia pelo componente. O símbolo "0" representa o desligado e o símbolo "1" representa ligado.

Tudo o que se vê na tela de um celular, um computador ou qualquer equipamento eletrônico é feito com operações no sistema de numeração binário. Apesar da interação com estes tipos de equipamentos acontecerem através de toque na tela, números, letras, som, entre outras, no final tudo é transformado em código binário.

Vale salientar que a lógica de funcionamento do sistema binário é igual ao do sistema decimal, cujo a base é 10. O sistema binário "base 2" , consiste em apenas dois algarismos o 1 e 0. Como dito acima, esse sistema é a base da funcionalidade dos computadores e equipamentos eletrônicos diversos e funciona da mesma forma que o sistema decimal, a única diferença entre eles é a quantidade de algarismos.

Os números binários podem ser somados, subtraídos, multiplicados e divididos da mesma

forma que os decimais, e embora o processo seja familiar, o fato de ter apenas dois algarismos pode causar um pouco de confusão. Por isso, é importante compreender como a notação posicional funciona, onde o valor que o algarismo assume depende da posição que ocupa dentro do número. Veja a seguir pequenos exemplos de como acontecem esses operações básicas na base 2 (sistema binário) para que perceba a equivalência dos algoritmos posicionais da soma, subtração, divisão e multiplicação com as mesmas operações no sistema decimal, e que conclua que esses algoritmos funcionam em qualquer outro sistema posicional de base x, onde $x > 1$.

Soma

Pequena tabuada da soma.

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 0 ("vai um" para o dígito de ordem superior)

1 + 1 + 1 = 1 ("vai um" para o dígito de ordem superior)

Soma de 101 + 011.

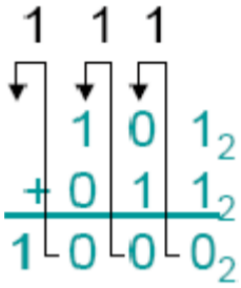


Figura 2.21: operação de soma

Subtração

Pequena tabuada da subtração.

0 - 0 = 0

0 - 1 = 1 (e "pede emprestado 1" para o dígito de ordem superior)

1 - 0 = 1

1 - 1 = 0

Subtração de 101 - 011

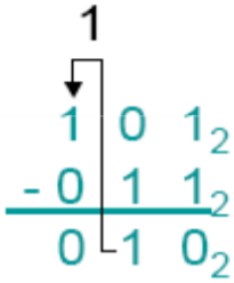


Figura 2.22: operação de subtração

Multiplicação

Pequena tabuada da multiplicação.

0 . 0 = 0

0 . 1 = 0

1 . 0 = 0

1 . 1 = 1

Multiplicação de 101 x 011

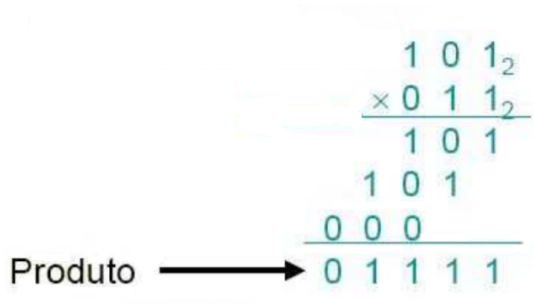


Figura 2.23: operação de multiplicação

Divisão

Idêntico ao método decimal, foram utilizados deslocamentos e subtrações.

Divisão de 101010 por 110

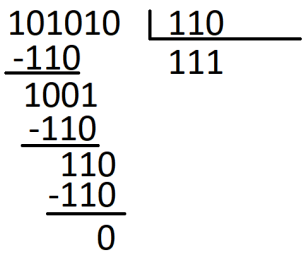


Figura 2.24: operação de divisão

No caso específico do sistema de numeração decimal, os equipamentos eletrônicos que precisam fazer esta transformação para o sistema binário usam um algoritmo matemático de mudança de base, que possibilita a mudança do sistema decimal, que tem dez símbolos, para o sistema binário, que tem apenas dois símbolos, veja como isso ocorre a seguir.

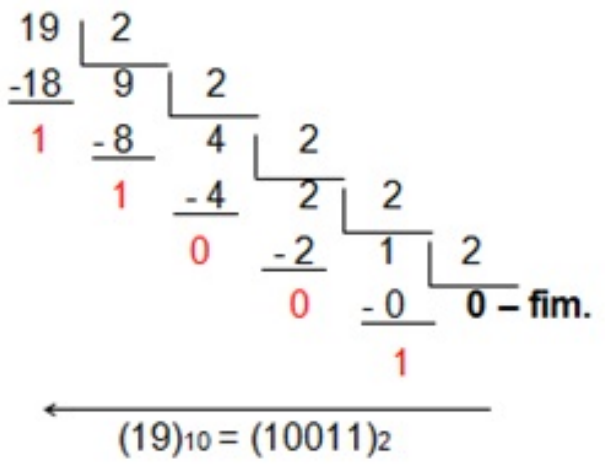


Figura 2.25: https://pt.wikipedia.org/wiki/Conversão-de-base-numérica

Ou seja, escolher um número e fazer divisões sucessivas neste número pela base 2 até zerar o quociente, logo em seguida, forma-se o número resultante na base 2 utilizando os restos das divisões de trás para a frente, assim como o exemplo acima. É interessante salientar que esse algoritmo pode ser aplicado em outras bases.

Demonstração do método da divisão

A prova é mostrada na referência[12]. Basicamente ela funciona de seguinte forma.

Suponha A_i um inteiro, A_{i+1} o quociente da divisão, r_i o resto da divisão para algum $i \geq 0$ e t a base destino. Pode-se escrever A_i como $A_i = (A_{i+1}) * t + r_i$, dividindo por t temos $\frac{(A_i)}{t} = A_{i+1} + \frac{(r_i)}{t}$

Sendo A_0 o inteiro a converter e t a base destino. Pelo método da expansão, tem-se:

$A_0 = b_m t^m + b_{m-1} t^{m-1} + ... + b_1 t^1 + b_0 t^0$

Na primeira divisão se tem:

$A_0/t = (b_m t^m + b_{m-1} t^{m-1} + ... + b_1 t^1 + b_0 t^0)/t$

$A_1 + r_0/t = b_m t^{m-1} + b_{m-1} t^{m-2} + ... + b_1 + b_0/t$

observe que $b_m t^{m-1} + b_{m-1} t^{m-2} + ... + b_1$ é inteiro, e b_0/t é uma fracão pois $b_0 < t$.

então $A_1 = b_m t^{m-1} + b_{m-1} t^{m-2} + ... + b_1$ e $r_0 = b_0$

Aplicando em A_1 o mesmo que foi feito em A_0 e a cada interação obtendo um resto $r_i = b_i$, onde b_i é o símbolo na base t correspondente à posição i do resultado. A expansão e o restante da prova podem ser vistos na referência [12].

2.5.1 Algoritmo

Algoritmo 10: algoritmo MudancaDeBase10-para-2

Entrada: n

Saída: basedois

```
1  início
2      Escrever("Escreva o Número na base decimal: ");
3      ler(n);
4      q := n; cont := 0;
5      enquanto (q <> 0) faça
6          q := n/2;
7          r := n - (q*2); n := q;
8          basedois[cont] = r;
9          cont = cont + 1;
10     fim
11     cont = cont − 1;
12     Escrever("Número na base binomial =");
13     enquanto (cont <> -1) faça
14         Escrever(" ");
15         Escrever(basedois[cont]);
16         cont = cont - 1;
17     fim
18 fim
```

2.5.2 Código em Java

```
1
2  import java.util.Scanner;
3  public class MudancaDeBase {
4      public MudancaDeBase() {
5          // TODO Auto-generated constructor stub
6      }
7      public static void main(String[] args) {
8         // TODO Auto-generated method stub
9  int r,q,n,cont;
10 int[] basedois= new int [2000];
11 Scanner ler = new Scanner(System.in);
12 System.out.print("Escreva o Número na base decimal: ");
13 n = ler.nextInt();
14 q = n;
15 cont = 0;
16 while (q != 0) {
17     q = n/2;  //quociente recebe a parte inteira da divisão  n/2
18     r = n - (q*2);
19     n = q;
20     basedois[cont] = r;
21     cont = cont + 1;
22     }
23 cont = cont - 1;
24 System.out.print("Número na base binomial =");
25 while (cont != -1) {
26     System.out.print("  ");
27     System.out.print(basedois[cont]);
28     cont = cont - 1;
29     }
```

```
30  }
31
32 }
```

2.5.3 Resultados

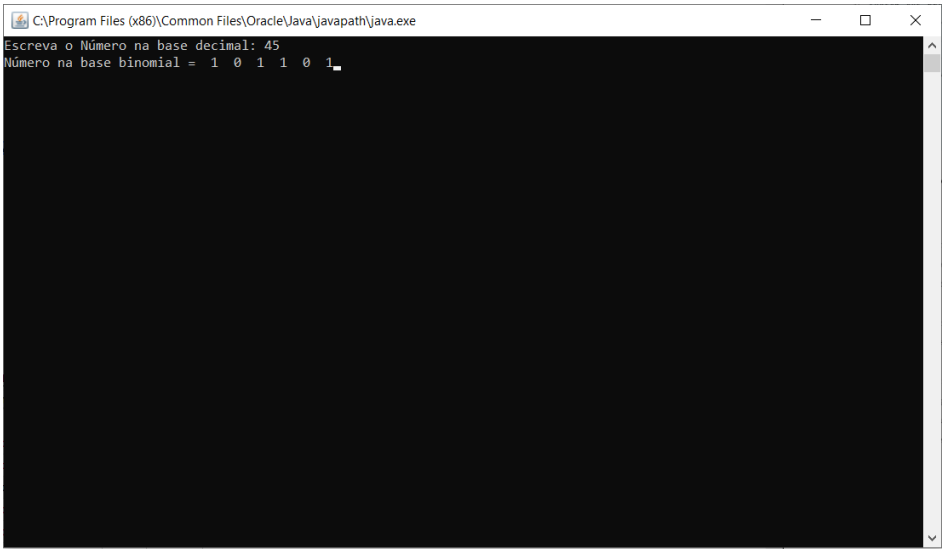


Figura 2.26: Do decimal 45 para seu correspondente na base 2.

Figura 2.27: Do decimal 1023 para seu correspondente na base 2.

Neste capítulo foram vistos alguns tópicos da matemática da educação básica que foi algoritmizada para o leitor ver que tais tópicos podem ser vistos em pseudo código e em Java. Isso foi só uma pequena demonstração, pois na educação básica existem muitos outros assuntos que podem ser transformados em códigos ou programas, tais como operações com matrizes, sistemas lineares, teorema de Pitágoras na geometria e muitos outros assuntos.

Devido a realidade a que estamos expostos, com muita tecnologia e algoritmos, é bastante interessante acrescentar uma abordagem algorítmica de tais assuntos na educação básica. Fazendo códigos e programas dos algoritmos estudados, os alunos, desde cedo, perceberam a importância da matemática aplicada a computação.

Capítulo 3

Aplicando algoritmos na sala de aula

O papel do algoritmo se torna cada vez mais essencial na nossa vida cotidiana e mesmo assim não se nota a sua importância, fato que já foi dito aqui, os algoritmos estão em todos os lugares em nosso mundo de hoje, até em uma simples receita de bolo.

O algoritmo não é uma fórmula de matemática ou um programa de computador, mas ele define uma sequência de passos finitos que podem levar a resposta de determinadas questões.

O mundo no qual se vive obriga o sujeito a lidar com novas tecnologias o tempo todo, linguagens, ferramentas e estruturas lógicas que surgem com a criação de algoritmos.

Hoje vive-se em um mundo conectado e dinâmico com ferramentas de busca, redes sociais, redes neurais, e-mails, softwares de produção, etc, e essas ferramentas trabalham com seleção das informações mais importantes para o dia a dia, um papel de grande importância para a vida conectada de hoje. Essas ferramentas atuam com grandes bases de dados e por toda a internet, fazendo o mapeamento das preferências em relação aos demais usuários, e trazendo partes novas ou esquecidas da cultura. São eles, os algoritmos, que gerenciam as interações em sites de redes sociais, enquanto destaca uma novidade de um amigo e exclui as novidades de outros.

3.1 A importância do aprendizado dos algoritmos no ensino Básico

Segundo Gomes 2015 [20],

> "Mesmo que o estudante não se torne um profissional da área de tecnologia, ele será beneficiado pelo desenvolvimento do raciocínio lógico e das demais habilidades citadas no texto, entre tantas outras. Dessa forma, se tornará um cidadão mais consciente e preparado para os desafios, os problemas e a complexidade do mundo atual"

Então ensinar programação possibilita o desenvolvimento da capacidade de abstração e do raciocínio lógico. Possibilita também o desenvolvimento de habilidades como resolução de problemas e noções de causa e efeito. Assim, aprender programação não deve ser só para quem quer seguir na área de tecnologia e sim para qualquer pessoa pois desenvolve o raciocínio e competências necessárias para realizar atividades cotidianas. O ensino da programação nas escolas é necessário para que os alunos desenvolvam sua criatividade e sua capacidade de lidar com problemas, sabendo resolvê-los passo a passo.

A codificação ou programação atualmente é considerada o centro de toda nova inovação ou criação que está presente no mundo. Alguns autores acreditam que no futuro os trabalhos serão quase todos feitos por robôs, é por isso que alguns países e seus respectivos professores estão tentando preparar seus alunos para atender às necessidades futuras e evitar que percam seus empregos para máquinas autônomas[21]. Atualmente, existem diferentes cursos, programas e até plataformas on-line que ensinam codificação para crianças, mas nem todos os países adotaram esse método ainda, muitos países ainda estão na fase inicial.

Um país pioneiro no ensino de algoritmos e programação é Israel[21]. No fim da década de 1990 Israel já incrementava o seu currículo da escola básica, incluindo nele o ensino de algoritmos e programação[21][22]. Para época, foi uma atitude arriscada por se tratar de uma Ciência relativamente nova mas hoje, sabemos que essa postura foi acertada e vários países ao

redor no mundo acrescentaram em seus currículos o estudo de algoritmos e consequentemente da programação.[23]

O Brasil deveria incluir o estude de algoritmos e programação em seu currículo assim como a Inglaterra, Israel , entre outros[21]. Talvez por ser graduado em Análise de Sistemas e em Matemática tenha uma visão mais privilegiada deste problema que algumas nações no mundo já perceberam e por meio deste trabalho pretendo dar minha pequena contribuição para o tema no Brasil.

3.2 Porque Java

Depois de falar sobre algoritmos e de demostrar alguns algoritmos matemáticos feitos em Java, era necessário falar um pouco mais sobre o Java.

Java é uma das linguagens mais utilizadas nos últimos anos[25], e sua utilização ainda vem crescendo. Além disso é umas das 5 primeiras linguagens de programação utilizada para usuários iniciantes em programação[24]. Ou seja temos a união de características, que na minha opnião, não podem ser ignoradas pois o Java é uma linguagem amplamente utilizada no mundo ao mesmo tempo que é uma das mais utilizadas para aprendizagem da programação. Isso implica que os alunos e professores terão acesso a uma grande quantidade de materiais sobre Java na internet como videos, livros, artigos, fóruns, grupos de whatsapp, manuais entre outros, inclusive uma grande quantidade destes conteúdos de maneira gratuita.

Figura 3.1: Linguagens mais utilizadas no mundo até 2020. https://olhardigital.com.br/2020/07/25/noticias/python-lidera-ranking-entre-as-10-melhores-linguagens-de-programacao-de-2020/

Outro beneficio do Java é que ele não é só uma linguagem ele é uma plataforma de desenvolvimento, com Java é possível desenvolver aplicações para desktop, celular, internet das coisas, web, televisão digital, e muito mais.

Quando se compila um arquivo em Java ele gera um bytecode que é interpretado numa JVM (maquina virtual Java) isso quer dizer que se o seu sistema operacional tiver uma JVM será possível executar o Java em seu sistema operacional e os sistemas operacionais mais utilizados no mundo, como Windows, GNU/linux, OS/2 , Mac, e outros, tem ou pode

ser instalado o JVM. O java pode ser programado até em um editor de texto comum mas existem varias IDE(ferramenta que auxilia e facilita a construção dos algoritmos em java) neste trabalho utilizo o Eclipse mas existem varias outras IDEs que também podem ser executados em vários sistemas operacionais.

Com este conjunto de características apresentados acima que encontramos em Java acredito que essa linguagem seja uma ótima escolha para este trabalho.

3.3 Um exemplo de como aplicar em sala

Nesta seção passarei um exemplo de como aplicar o estudo de algoritmo e Java utilizando Sequência Didática com o objetivo de unir o assunto da grade curricular do aluno de matemática com o ensino de algoritmos, fluxogramas e Java. Vale salientar que para conseguir programar um código, é estritamente necessário que o aluno tenha entendido o assunto, ou seja, ele entendeu o algoritmo. Na fase em que estiver fazendo o fluxograma e a programação ele estará fixando o assunto, aprendendo a programar e ainda terá o estimo de ver seus códigos executando no computador. Este é um exemplo onde o professor pode escolher outros temas do currículo pedagógico da turma em questão para aplicar a Sequência Didática assim como ele tem a liberdade de utilizar outra linguagem de programação.

3.3.1 Sequência Didática

Tema da Sequência Didática: Algoritmo da mudança de base

Objetivo da sequência didática: Fazer os alunos perceberem a importância dos algoritmos e iniciá-los em programação usando a linguagem Java.

Conteúdos a serem trabalhados: Mudança de base numérica, divisões sucessivas, Logica, linguagem Java.

Habilidades a serem desenvolvidas:

Saber e reconhecer um algoritmo

O desenvolvimento do raciocínio lógico

Identificar um algoritmo e suas estruturas

Trabalhar com comandos e estruturas da linguagem Java

Saber desenvolver um algoritmo

Saber fazer um fluxograma

Mudança de base decimal para outra qualquer

Tempo de execução da sequência didática: 10 aulas

Materiais necessários: Computador, Datashow, Celular, Quadro, Folhas de papel, Livros e manuais de Java, Manuais dos símbolos de fluxogramas padrões ISO − 9000.

Detalhamento das aulas:

Aula 1 e 2

Organização da turma: Sala de aula

Introdução:

No primeiro momento será discutido com os alunos a importância dos números em base 2, a sua principal finalidade e o porque de sua utilização através de aula expositiva/participativa dos alunos.

Desenvolvimento:

Através de aula expositiva será feito o aprofundamento do tema, que é mudança de base, explicando também que existem outras bases além da binaria(base 2) e decimal(base 10) e que o conceito para mudar de uma base x para y é sempre o mesmo. Entraremos em como trocar o nosso sistema de numeração decimal para o sistema de numeração binário. Além de fazer o contrário, ou seja , trocar de binário para decimal, fazendo a conversão para outras bases além da decimal utilizando sempre o quadro.

Conclusão:

Será solucionada as possíveis dúvidas dos alunos e depois será passada a atividade 1 de fixação de conteúdo.

Atividade 1

1) Pratique a mudança para base 2 dos números decimais abaixo:

a) 9

b) 16

c) 31

d) 234

e) 1234

f) 2048

2) Realize a conversão dos números decimais abaixo em base 2, 3 e 8 :

a) 38

b) 160

c) 265

3) Converta agora os números em base 2 abaixo em números de base 10 (decimais).

a) $1000_{(2)}$

b) $101010_{(2)}$

c) $11111_{(2)}$

d) $10100001_{(2)}$

4) Converta os números em bases diferentes em base 10

a) $234_{(8)}$

b) $122_{(3)}$

c) $42313_{(4)}$

5) Qual a importância do sistema binário na atualidade?

Aula 3 e 4

Organização da turma: sala de aula

Nesta fase, com o entendimento de como funciona a mudança de base, começamos a aprofundar nos fluxogramas, algoritmos e em java.

Introdução:

Começamos com uma pequena revisão, correção da atividade 1 e resolução de dúvidas da atividade

Desenvolvimento:

Continua-se a aula apresentado sobre os símbolos e normas dos fluxogramas segundo a ISO - 9000, explicando, exemplificando e fazendo fluxogramas de diversas situações.

Conclusão :

Será disponibilizada a atividade 2 de fixação para os alunos. Lembrando aos mesmos que eles podem utilizar materiais diversos sobre fluxograma pois a intenção é saber fazer o fluxograma e não decorar os símbolos.

Atividade 2

1) Escreva um fluxograma para cada situação abaixo:

a) Trocar uma lâmpada.

b) Trocar o pneu do carro pelo pneu step.

c) Da nota final de sua escola.

d) Ler dois valores inteiros , descobrir qual o maior e escrevê-lo.

e) Escrever todos os números ímpares até 100.

Aula 5 a 8

Organização da turma: Sala de computadores ou laboratório de informática com no máximo 2 alunos por computador.

Neste momento os alunos já tem ferramentas necessárias para programar em java , basta agora que eles tenham contato com essa linguagem de programação, para entender a estrutura geral da linguagem e conseguirem transformar o pensamento lógico que adquiriram até então em código java.

Introdução:

Começa-se tirando dúvidas e fazendo a correção da atividade 2. Começamos também a explicação, visualização e exemplificação do que é linguagem de programação utilizando e

focando sempre em Java.

Desenvolvimento:

Será apresentado toda a estrutura básica e necessária do Java que são operadores, expressões, variáveis, constantes, condicionais e estrutura de repetições. É possível que necessitemos de mais tempo aqui, é bom ressaltar que não será utilizado conceitos avançados de java como polimorfismo, criação de objetos e classes, entre outros, pois o intuito não é tornar o aluno um especialista na linguagem Java.

Conclusão:

Será realizada a atividade 3 para fixa o assunto e a correção da mesma.

Atividade 3

1) Faça um resumo de todas as estruturas vistas em sala de aula e informe para que eles servem.

2) O que são IDE?

3) Como declarar uma variável em Java?

Operadores e expressões

4) Mostre pelo menos 8 operadores e suas funções.

5) Como resolvemos expressões em Java

Condicionais

6) Crie um algoritmo que leia a nota de um aluno e mostre se ele está: aprovado (acima de 60), em recuperação (abaixo de 60) ou reprovado por média (abaixo de 20)

7) Escreva um algoritmo em Java que leia um número e informe se ele é divisível por 10, por 5 ou por 2 ou se não é divisível por nenhum deles.

Repetição

8) Escreva um algoritmo que exiba 10 vezes a mensagem “Adoro Algoritmo e Matemática”.

9) Escreva um algoritmo que calcule a soma dos números de 1 a 20.

10) Leia a idade de 5 colegas exiba a soma das idades.

11) Crie um algoritmo leia um número do usuário e exiba a sua tabuada de multiplicação.

Aula 9 e 10

Aula final, com o objetivo de avaliar o conhecimento dos alunos e de se constatar, olhando os resultados dos alunos, se o objetivo foi alcançado.

Organização da turma: Sala de computadores ou laboratório de informática com no máximo 2 alunos por computador.

Introdução:

Arrumação da sala/laboratório para aplicar a atividade avaliativa.

Desenvolvimento:

Os alunos sozinhos ou em duplas fazem a atividade avaliativa.

Conclusão:

Ao terminar o tempo o professor avalia os códigos dos alunos nos computadores.

Atividade avaliativa

Escrever o algoritmo da mudança de base: de base 10 para base 2 e o algoritmo de base 2 para base 10.

Capítulo 4

Conclusão

Os exemplos acima expostos é uma quantidade muito pequena do aparecimento, uso e aplicabilidade dos algoritmos nas vidas humanas, percebe-se então que infelizmente não é dada a importância necessária para essa ferramenta cognitiva excepcional.

O que quero dizer é que atualmente são de extrema importância os conhecimentos de algoritmos na formação de qualquer profissional, pois saber organizar os processos e ideias em passos que podem ser executados por pessoas ou computadores pode possibilitar a este profissional um diferencial no mercado de trabalho.

Para Berlinski 2002 [28],

> "Para o homem atual a formalização, aptidão, habilidade no manejo dos códigos algorítmicos são componentes fundamentais para a cultura. A trama dos algoritmos é o que tem dado ritmo ao desenvolvimento da humanidade, abrindo um novo tempo e reorganizando a sua cultura."

Em se tratando de algoritmo como um mecanismo positivo para o crescimento do conhecimento do homem, o que se deve entender é que o manusear da máquina não se limita apenas em operações algorítmicas de apertar botões.

No contexto atual há vários tipos de conhecimento e habilidades exigidos pela sociedade para que se possa exercer determinadas funções, é nesse contexto que se insere o pensamento

algorítmico, onde se considera um dos requisitos mais importantes, devido estar em todos os contextos atuais, seja no trabalho, em casa, na escola ou na vida social como um todo. De acordo com França, Silva e Silva[29], o pensamento computacional ou algorítmico está em saber utilizar o computador para aumentar o poder cognitivo e operacional humano, aumentando assim a produtividade, criatividade e inventividade.

É de grande valia inserir o pensamento algorítmico na educação básica, pois essa além de possibilitar conhecimentos na área de exatas, mostra várias formas de resolver problemas do cotidiano, possibilitando a melhoria do rendimento escolar.

Concluí-se então que o conceito básico do algoritmo é entender como funciona as coisas e os vários sistemas digitais, ou não digitais, que se usa diariamente para que seja possível resolvê-los em passos finitos, isso não só na matemática e informática, mas para todo e qualquer problema, tornando hoje o entendimento e o estudo dos algoritmos de extrema utilidade, e para isso seria ótimo introduzir o seu estudo no ensino fundamental e médio da educação básica no Brasil.

Referências Bibliográficas

[1] ALMEIDA, Regina de Cassia. *ALGORITMOS - UMA PROBLEMATIZAÇÃO DO TEMA MEDIADA PELA HISTÓRIA DA MATEMÁTICA*. Disponível em `https://www.13snhct.sbhc.org.br/resources/anais/10/1345086117_ARQUIVO_13cnhttexto.pdf`.Acesso em: 15 de novembro de 2018

[2] CRUZ, Adriano Joaquim de Oliveira (1 de janeiro de 1997). *Algoritmos*. Disponivel em `http://http://equipe.nce.ufrj.br/adriano/c/apostila/algoritmos.htm`. acesso em: 17 de novembro de 2018

[3] SANTOS, Daniel Tebaldi *O Uso de Algoritmos e Programação no Ensino de Matemática*. Campinas, SP : [s.n.], 2015

[4] D. E. Knuth *The art of Computer Programming* - volume 1 Addison-Wesley, 1973.

[5] WIKIPEDIA. *Algoritmo de Euclides*. Disponível em `https://pt.wikipedia.org/wiki/Algoritmo_de_Euclides` acesso em: 15 de marco de 2019

[6] ZARIFAN, P. Objetivo Competência: por uma nova lógica. São Paulo: Atlas, 2001.

[7] ANDRADE, J. B. de (2003). Fotogrametria. SBEE. 2 ed revista, ampliada e atualizada. Curitiba, 2003.

[8] MATEUS, M. H. M. "Tradução automática: um pouco de história". Engenharia da Linguagem. Org. Maria Helena M. Mateus e António Horta Branco. Lisboa, Edições Colibri, 1995

[9] CHOMSKY, Noam. Mídia: propaganda política e manipulação. WWF Martins Fontes, 2015.

[10] WILKE, Valéria Cristina Lopes. Informação, poder e estado: o dispositivo informacional e as políticas públicas de inclusão digital do governo brasileiro (2003-2008). 2012.

[11] PEDROSO, Hermes Antônio. *Uma breve história da equação do 2º grau*. Disponível em `https://repositorio.unesp.br/handle/11449/122614`, acesso em: 15 de fevereiro de 2020.

[12] DEIMEL, Lionel E. *Conversion of Number Representations*. Disponível em `https://deimel.org/comp_sci/conversion.htm`, acesso em: 15 de fevereiro de 2020.

[13] LEMAY, Laura; PERKINS, Charles. Aprenda Java em 21 dias - JAVA. Editora Campus, 1997.

[14] GHEZZI, Carlo; JAZAYERI, Mehdi. Conceitos de Linguagens de Programação. Editora Campus, 1987.

[15] DEITEL, H. M.; DEITEL, P. J. Java Como Programar: 4 ed. São Paulo: Bookman, 2002. 1386 p.

[16] HEFEZ, Abramo. Aritmética. 2. ed. Rio de Janeiro: SBM, 2016 (Coleção PROFMAT, v. 8). 330p.

[17] LUIZ, Robson. "Dispositivo prático de Briot-Ruffini"; Brasil Escola. Disponível em: https://brasilescola.uol.com.br/matematica/divisao-polinomios-utilizando-dispositivo-briotruffini.htm. Acesso em 15 de abril de 2020.

[18] GIRÃO, Ivna. “91 % dos estudantes do Ensino Médio não sabem matemática”; Diário do Nordeste. Disponível em: https://diariodonordeste.verdesmares.com.br/metro/91-dos-estudantes-do-ensino-medio-nao-sabem-matematica-1.212306. Acesso em 13 de junho de 2020.

[19] NOGUEIRA, Flavia. “Alunos chegam ao Médio sem saber divisão e interpretação de texto”; Nova Escola. Disponível em: https://novaescola.org.br/conteudo/12519/alunos-chegam-ao-medio-sem-saber-divisao-e-interpretacao-de-texto. Acesso em 15 de junho de 2020.

[20] GOMES, Marcos César Pires. Os benefícios do ensino de linguagem de programação no currículo regular. Disponível em: https://administradores.com.br/artigos/os-beneficios-do-ensino-de-linguagem-de-programacao-no-curriculo-regular. Acessado em: 12 de Setembro de 2020.

[21] ROBOGARDEN. Os 5 principais países em codificação e programação e Países que ensinam codificação nas escolas. Disponivel em: https://robogarden.ca/pt/blog/top-5-countries-in-coding-and-programming. Acessado em: 12 de Setembro de 2020.

[22] R. S. A. França and H. Costa, Ensino de computação na educação básica no brasil: Um mapeamento sistemático, in XXI Workshop sobre Educação em Computação, 2013.

[23] PIVA, Naiady. Inclusão da programação nos currículos escolares avança no exterior. Disponivel em: https://www.gazetadopovo.com.br/educacao/inclusao-da-programacao-nos-curriculos-escolares-avanca-no-exterior-9dlmbpvkpztvrp4vp1647l8fk/ .Gazeta do Povo. Acessado em: 12 de Setembro de 2020.

[24] MATOS, David. As 5 Melhores Linguagens de Programação Para Aprender em 2020. Disponivel em: https://www.cienciaedados.com/as-5-melhores-linguagens-de-programacao-para-aprender-em-2020/. Acessado em: 12 de Setembro de 2020.

[25] SHIMABUKURO, Igor. Python lidera ranking entre as 10 melhores linguagens de programação de 2020. Disponivel em: https://olhardigital.com.br/2020/07/25/noticias/python-lidera-ranking-entre-as-10-melhores-linguagens-de-programacao-de-2020/. Acessado em: 12 de outubro de 2020.

[26] RAMOS, Luís Eduardo. Tabela de comandos Java. Disponivel : https://www.academia.edu/7134498/Tabela_de_comando_Java. Acessado em: 20 de outubro de 2020.

[27] SILVA, Marcos Noé Pedro da. "O Surgimento da Equação do 2º Grau "; Brasil Escola. Disponível em: https://brasilescola.uol.com.br/matematica/o-surgimento-equacao-2-o-grau.htm. Acesso em 03 de abril de 2020.

[28] BERLINSKI, D. O advento do algoritmo (a Idéia Que Governa o Mundo). Globo, 2002

[29] FRANÇA, R. S.; Silva, W. C.; Silva, J. C. Ensino de Ciência da Computação na Educação Básica: Experiências, Desafios e Possibilidades. Garanhuns ? PE, Brasil, 2012.

[30] FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. Lógica de programação: a construção de algoritmos e estruturas de dados. Makron Books, 1993.

[31] ALMEIDA, Paulo Nunes de. Educação Lúdica: técnicas e jogos pedagógicos; Loyola,11ª ed. São Paulo, 2003.

[32] LANGLOIS, L.; HAMILTON, B. Assessing the Difference: Greenhouse Gas Emissions of Electricity Generating Chains.IAEA Bulletin, Vienna, v. 42, n. 2, 2003.

[33] SANTOS, P. "Tradução automática". Engenharia da Linguagem. Org. Maria Helena M. Mateus e António Horta Branco. Lisboa, Edições Colibri, 1995.

[34] BRASIL. Ministério da Educação - Secretaria da Educação Fundamental. *PCN's: parâmetros curriculares nacionais*. Brasília: MEC/SEF, 1998.

[35] LIMA, Elon Lages. "O principio da indução". Disponível em: http://www.mat.uc.pt/ mat0829/A.Peano.htm. Acesso em 14 de setembro de 2020.

Apêndice

Abaixo temos algumas tabelas com um ótimo resumo dos comandos básicos e exemplos, necessárias as demandas das atividades e sequência pedagógica.

Tipos de dados

PSEUDO CÓDIGO	LINGUAGEM JAVA
literal	String
caracter	Char
inteiro	Int
real	Float
real	Double

Declaração de variáveis

PSEUDO CÓDIGO	LINGUAGEM JAVA
idade numérico	Int idade
preço numérico	Float preco
nome literal	String nome
sexo literal	Char nome

Comentários

PSEUDO CÓDIGO	LINGUAGEM JAVA
Delimitado por /* r */ Ex: N1 numérico /*primeira nota*/	Delimitado por /* */ Ex: /* Isso é um comentário */ Se o comentário for em apenas uma linha pode-se usar // EX: //comentário

Operadores Aritméticos

PSEUDO CÓDIGO	LINGUAGEM JAVA
Adição: +	+
Subtração: -	-
Multiplicação: *	*
Divisão: /	/
Resto: %	%
Atribuição: $<-$	=
	Obs.: caso a divisão seja feita entre dois números inteiros o resultado será o quociente inteiro da divisão Ex.: RESULT = 5/2; A variável RESULT receberá 2 e não 2.5 Se a resposta esperada é a divisão exata, ou seja, 2.5 precisa-se transformar um dos números em float Ex.: RESULT = 5f/2; ou RESULT = 5.0f/2; ou RESULT = (float)5/2;

Operadores Relacionais

PSEUDO CÓDIGO	LINGUAGEM JAVA
Igual a: =	==
Diferente de: $\neq$ ou <>	!=
Maior que: >	>
Menor que: <	<
Maior ou igual a: $\geq$	>=
Menor ou igual a: $\leq$	<=

Operadores Lógicos

PSEUDO CÓDIGO	LINGUAGEM JAVA
Conjunção: e	&&
Disjunção: ou	||
Negação: não	!

Para fazer a entrada de dados através do prompt de comando o primeiro ato necessário é criar um objeto da classe Scanner, por isso recomenda-se que assim que iniciar o método main, escreva a seguinte linha de código que cria este objeto: Scanner leia = new Scanner(System.in);

Comandos de entrada de dados

PSEUDO CÓDIGO	LINGUAGEM JAVA
leia idade	idade = leia.nextInt();
leia preco	preco = leia.nextFloat();
leia salario	salario = leia.nextDouble();
leia nome	nome = leia.next();
leia sexo	sexo = leia.next().charAt(0);

Comandos de saida de dados

PSEUDO CÓDIGO	LINGUAGEM JAVA
escreva “Digite o salario:”	System.out.println(“Digite o salário:”);
escreva “salário atual:”	System.out.println(“salário atual:”+salario);
escreva “O funcionário ”, nome, “ recebe R$”, salario	System.out.println (“O funcionário ”+nome+“ recebe R$”+salario);
escreva “O aumento de 10% é ”, salario+salario*0.10	System.out.println (“O aumento de 10% é ”+(salario+salario*0.10));

Observação: O comando de saída de dados System.out.println(), mostra a mensagem e pula uma linha, caso deseje mostrar a mensagem mas não pular uma linha no final usa-se System.out.print().

Principais funções Matemáticas

PSEUDO CÓDIGO	LINGUAGEM JAVA
Math.PI	constante do valor PI
Math.abs(x)	Retorna o valor absoluto (módulo) do numero passado por parâmetro.
Math.ceil(x)	Arredonda um numero real para cima. Ex.: Math.ceil(3.7) é 4
Math.cos(x)	Calcula o cosseno de x . (x deve estar representado em radianos)
Math.exp(x)	Obtém o logaritmo natural e elevado a x
Math.floor(x)	Arredonda um número real para baixo. Ex.: Math.floor(3.7) é 3
Math.log(x)	Obtém o logaritmo natural de x.
Math.log10(x)	Obtém o logaritmo de base 10 de x.
Math.pow(x,y)	Calcula a potência de x elevado a y.
Math.sin(x)	Calcula o seno de x (x deve estar representado em radianos)

Math.cbrt(x)	Calcula a raiz cúbica de x
Math.sqrt(x)	Calcula a raiz quadrada de x
Math.tan(x)	Calcula a tangente de x (x deve estar representado em radianos)
Math.toDegress(x)	Converte a medida de x de radianos para graus.
Math.toRadians(x)	Converte a medida de x de graus para radianos.

Estreutura de decisão ou condicional - ESCOLHA

PSEUDO CÓDIGO	LINGUAGEM JAVA
escolha (op)	switch (op)
inicio	{
caso 1:	case 1:
escreva “soma : ”, a + b	System.out.println(“soma : ”+(a + b));
parar;	break;
case 2:	case 2:
escreva “subtração : ”, a - b	System.out.println(“subtração : ”+(a - b));
parar;	break;
case 3:	case 3:
escreva “multiplicação : ”, a * b	System.out.println(“multiplicação : ”+(a * b));
parar;	break;
padrão:	default:
escreva “divisão : ”, a*1.0 / b	System.out.println(“divisão : ”+(a*1.0/ b));
parar;	break;
fimescolha	}

Estreutura de decisão ou condicional - SE

PSEUDO CÓDIGO	LINGUAGEM JAVA
se (preco < 100) então	if (preco < 100) {
novopreco = preco*1.10	novopreco=preco*1.10
	}
se (sexo = 'F') então	if (sexo == 'F') {
qMulheres = qMulheres + 1	qMulheres = qMulheres + 1
	}
senão	else {
qHomens = qHomens + 1	qHomens = qHomens + 1
fimse	}
se (cor = "azul") então	if (cor.equalsIgnoreCase("azul")){
qAzul = qAzul + 1	System.out.println("A cor é azul");
	}
senão	else{
se cor = "vermelho" então	if (cor.equalsIgnoreCase("vermelho")){
qVermelho = qVermelho + 1	System.out.println("A cor é vermelha");
	}
senão	else {
qQualquer = qQualquer + 1	System.out.println("A cor não é azul e nem vermelha");
fimse	}
fimse	}

Estrutura de repetição - ENQUANTO

PSEUDO CÓDIGO	LINGUAGEM JAVA
raio, v numerico	float raio, v;
raio $<-$ 0;	raio = 0;
enquanto (raio <= 20) faça	while (raio <= 20) {
$v <- 4/3 * 3.14 * raio^3$	$v = 4/3 * (float)3.14*$ (float)pow(raio, 3);
escreva	System.out.println
(“Para raio:”, raio, “o volume é:”, v)	(“Para raio:”+ raio +“o volume é:”+v);
raio $<-$ raio + 0.5;	raio = raio + 0.5;
fimenquanto	}
	system(“pause”);
fimalgoritmo	}

Estrutura de repetição - REPITA

PSEUDO CÓDIGO	LINGUAGEM JAVA
raio, v numerico	float raio, v;
raio $<-$ 0;	raio = 0;
repita	Do{
$v <- 4/3 * 3.14 * raio^3$	$v = 4/3 * (float)3.14*$ (float)pow(raio, 3);
escreva	System.out.println
(“Para raio”, raio, “o volume é:”, v)	(“Para raio:”+ raio +“o volume é:”+v);
raio $<-$ raio + 0.5;	raio = raio + 0.5;
até (raio <= 20);	}while(raio <= 20);
	System(“pause”);
fimalgoritmo	}

Estrutura de repetição - PARA

PSEUDO CÓDIGO	LINGUAGEM JAVA

raio, cont, v numerico
raio $<-$ 0;
para cont de 0 até 40 passo 1 faça
$v <- 4/3 * 3.14 * raio^3$
	float raio, v; int cont
	raio = 0;
	for(cont=0; cont<=40; cont=cont+1) {
	$v = 4/3 * (float)3.14*$ (float)pow(raio, 3);

escreva (“Para raio:”, raio, “o volume é:”, v)	System.out.println (“Para raio:”+ raio +“o volume é:”+v);

raio $<-$ raio + 0.5;
fimpara
	raio = raio + 0.5;
	}
	System(“pause”);