

Centro Universitário Jorge Amado
Igor Takenami

INTRODUÇÃO A LÓGICA DE PROGRAMAÇÃO

Versão 2.0

Salvador
2010.1

Caro Aluno,

Este pequeno módulo visa fixar os conceitos de Lógica de Programação discutidos em nossas aulas. Seu conteúdo também servirá como material de referência, não só para a sala de aula, mas para ser utilizando quando necessário.

As problematizações apresentadas através dos estudos de caso serão laboratórios para as aplicações dos conceitos deste módulo e também deve ser guardado como material de apoio e referência.

Lembre que a leitura complementar sobre o tema é obrigatório e ajudará no seu amadurecimento profissional.

O evolução deste material é feito através das interações com os alunos, então, sua contribuição é fundamental. Qualquer dúvida ou sugestão envie um email para: itakenami@gmail.com.

Grande abraço e bom estudo!

Atenciosamente,

Igor Takenami

AVALIAÇÕES

Avaliação	Data	Peso
Observações		

Índice

PLANO DE CURSO.....	5
1. INTRODUÇÃO A LÓGICA DE PROGRAMAÇÃO	8
1.1. Algoritmos.....	8
1.2. Como criar programas de computador	8
2. CONCEITOS BÁSICOS PARA ESCREVER ALGORITMOS	9
2.1. Variáveis.....	9
2.2. Sintaxe x Semântica.....	9
3. FLUXOGRAMA.....	10
3.1. Figuras.....	10
3.2. Utilizando Fluxogramas	10
3.3. Estrutura Condicional.....	11
4. PORTUGUES ESTRUTURADO (PORTUGOL, PSEUDO-CÓDIGO)	11
4.1. Estrutura Básica.....	12
4.2. Declaração: Variáveis e Constantes.....	12
4.3. De Fluxograma para Português Estruturado	13
4.4. Operadores Relacionais.....	14
4.5. Operadores Lógico.....	14
4.6. Estruturas de Repetição.....	15
4.7. Vetores.....	16
5. HISTÓRIA DO JAVA.....	17
6. A MAQUINA VIRTUAL JAVA.....	18
7. DE PORTUGUÊS ESTRUTURADO PARA JAVA.....	20
7.1. Vetor.....	21
7.2. Matriz.....	22
8. RECURSOS DO JAVA.....	23
8.1. Formatar Valores Decimais.....	23
8.2. Números Aleatórios.....	23
8.3. Converter String em vetor de String.....	23
8.4. Converter String em vetor de char.....	23
8.5. Imprimir Caracteres ASCII.....	24
8.6. Abrindo um Arquivo.....	25
8.7. Gravando em Arquivo.....	25
9. PREPARAÇÃO PARA O INTERDISCIPLINAR.....	26
9.1. Trabalho de Jogos	26
9.2. Dicas de Apresentação.....	27
LISTA DE EXERCÍCIOS I.....	28
LISTA DE EXERCÍCIOS II.....	30
LISTA DE EXERCÍCIOS III.....	31
LISTA DE EXERCÍCIOS IV.....	32
LISTA DE EXERCÍCIOS V.....	33
ESTUDO DE CASO I.....	34
ESTUDO DE CASO II.....	35
ESTUDO DE CASO III.....	36
ESTUDO DE CASO IV.....	37
ESTUDO DE CASO V.....	38
ESTUDO DE CASO VI.....	39
ESTUDO DE CASO VII.....	40

PLANO DE CURSO

CURSO: Superior de Tecnologia em Desenvolvimento de Software	
MÓDULO: 1	CARGA HORÁRIA: 800h
HABILITAÇÃO: Programador Básico de Programadores	
PROGRAMA DA DISCIPLINA	
Disciplina: Lógica de Programação	CARGA HORÁRIA: 140 horas
Professor (a): Igor Takenami	
Semestre/Turno: 2010.1 (noturno)	
OBJETIVOS DA DISCIPLINA	
A disciplina tem por objetivo apresentar os conceitos fundamentais usados na construção de algoritmos; desenvolver no aluno a capacidade lógica de construir algoritmos para resolução de problemas; possibilitar que o aluno construa algoritmos de forma planejada e estruturada; possibilitar a codificação de programas de baixa e média complexidade, usando uma linguagem de programação; reconhecer a importância do planejamento e testes na resolução de problemas computacionais.	
Habilidades: Estimular o desenvolvimento e aprimoramento das seguintes habilidades: <ol style="list-style-type: none"> 1) Desenvolver a lógica de programação; 2) Elaborar algoritmos estruturados para a solução de problemas; 3) Verificar e corrigir algoritmos estruturados; 4) Escolher a melhor estrutura de dados e o melhor algoritmo para a solução de um determinado problema; 5) Programar de forma estruturada soluções básicas de problemas com média complexidade. Conhecimentos: Conceitos pertinentes às linguagens de programação estruturada: constantes, variáveis, vetores, funções, procedimentos, modularização; Pseudo-códigos para representação de problemas. Atitudes: <ol style="list-style-type: none"> a) Organização b) Estruturação do raciocínio c) Auto-aprendizagem d) Criatividade e) Postura pró-ativa 	
EMENTA	
Conceitos de linguagem de programação e programa de computador. O processo de desenvolvimento de algoritmos e programas de computador. Elementos básicos de construção de algoritmos e programas de computador (constantes, variáveis, funções predefinidas e expressões); comandos de entrada, saída e atribuição; estruturas sequenciais, seletivas e repetitivas. Vetores e matrizes. Introdução a Linguagem de Programação JAVA.	
CONTEÚDO	
<ol style="list-style-type: none"> 1. Definição de Programa de Computador 2. Introdução a Lógica <ol style="list-style-type: none"> Definição de lógica Utilizando lógica para resolver problemas computacionais Resolvendo problemas diversos através de estruturas lógicas Descrição narrativa 3. Construindo Fluxogramas 	

<p>Conceito de Variáveis Fluxos de Controle Algoritmos computacionais</p> <p>4. Teste de Mesa</p> <p>5. Introdução a Português Estruturado Tipos de Variáveis Comandos e Operadores Funções da Linguagem: Mod e Div Utilizando lógica em algoritmos computacionais</p> <p>6. Estruturas de Controle Decisão: Se ... Então ... Senão Repetição com teste no início: Enquanto ... Faça Repetição com teste no final: Repita ... Até Repetição com valor definido: Para..faça</p> <p>7. Vetores e Matrizes</p> <p>8. Introdução a Linguagem de Programação JAVA Estrutura básica de um programa Variáveis e seus tipos Convertendo de Português Estruturado para JAVA Comandos e Operadores em JAVA Entrada e Saída de Dados Estruturas de Controle e Repetição</p> <p>9. Vetores e Matrizes em JAVA</p> <p>10. Funções</p> <p>11. Manipulação de Arquivo Leitura Escrita</p>
METODOLOGIA DE ENSINO
<p>Conceituação: Aulas expositivas e realização de exercícios de fixação do aprendizado das técnicas de programação e exercícios avançados para o desenvolvimento lógico do aluno.</p> <p>Crítica: Através dos exercícios avançados no qual o aluno deverá interpretar o problema e resolvê-lo através de técnicas computacionais.</p> <p>Experimental: Teste de mesa. Aplicação dos exercícios no laboratório de informática .</p>
AVALIAÇÕES
<p>A avaliação será realizada no decorrer do curso através da observação da participação dos alunos nas aulas e realização de trabalhos e exercícios, além de avaliação escrita:</p> <p>APED – (Peso 1) Avaliação 01 – Prova Escrita (Peso 2) Avaliação 02 – Prova Escrita (Peso 2) Avaliação 03 – Trabalho Interdisciplinar (Peso 2)</p>
ATIVIDADES COMPLEMENTARES
<p>Acompanhamento de atividades interdisciplinares; Desafios (Exercícios).</p>

BIBLIOGRAFIA
BIBLIOGRAFIA BÁSICA
DEITEL, HARVEY; DEITEL, PAUL. Java: Como Programar , Ed. Bookman, 2002. FARRER, Harry et al. Algoritmos Estruturados . Rio de Janeiro: Campus. FORBELLONE, André, EBERSPÄCHER, Henri. Lógica de Programação: a construção de algoritmos e estrutura de dados . São Paulo: Makron Books. BERG, A. C.; FIGUEIRÓ, J. P. Lógica de Programação . 2. ed. Canoas, Ed. ULBRA, 2002.

1. INTRODUÇÃO A LÓGICA DE PROGRAMAÇÃO

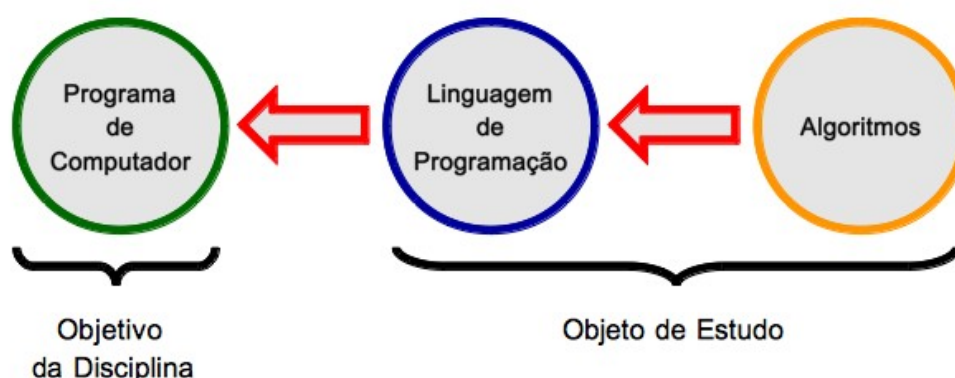
1.1. Algoritmos

Descrição de uma **seqüência lógica** de passos, que quando seguido a risca, cumpre um objetivo bem definido. Um algoritmo é como uma **receita de bolo** e da mesma forma, caso a sequência não esteja corretamente ordenada ou seus ingredientes não estejam na medida correta, o resultado pode não ser o esperado.

O papel da **lógica na programação** é usar corretamente o raciocínio ordenando o pensamento na melhor seqüência de passos com o objetivo de atingir a solução pretendida.

Um algoritmo pode ser utilizado para descrever qualquer procedimento que possua uma seqüência lógica até mesmo tarefas de nosso cotidiano.

1.2. Como criar programas de computador



Linguagem de Programação é um algoritmo escrito com algumas regras bem definidas de forma que o computador consiga entender claramente que está sendo solicitado possibilitando a sua execução e transformação em um programa de computador.

2. CONCEITOS BÁSICOS PARA ESCREVER ALGORITMOS

2.1. Variáveis

Representam um local de armazenamento de informações cujos valores podem ser modificados ao longo da execução do algoritmo. Uma variável é composta basicamente de:

- **Identificador** - nome dado à variável para possibilitar sua manipulação
- **conteúdo** - valor atual da variável

O identificador de uma variável deve obedecer as seguintes regras:

- Representar ao máximo o seu significado;
- Não pode conter espaço podendo ser substituído por “_”;
- Não pode ser formado somente por números;
- Não pode ser iniciado por números;

Ex:

```
meu_nome = "Jonny"
```

O identificador da variável é meu_nome O conteúdo da variável é "Jonny"

```
minha_idade = 30
```

O identificador da variável é minha_idade O conteúdo da variável é 30

2.2. Sintaxe x Semântica

Sintaxe - Regras que determinam como o algoritmo deve ser escrito.

Semântica - Regras que definem a forma como o algoritmo deve ser interpretado.

Ex:

```
meu_nome = João da Silva Sintaxe: Forma como é escrito
```

Semântica: Armazena na variável meu_nome o valor "João da Silva"






```
meu_nome "João da Silva" Sintaxe: Forma como é escrito
```

Semântica: Armazena na variável meu_nome o valor "João da Silva"

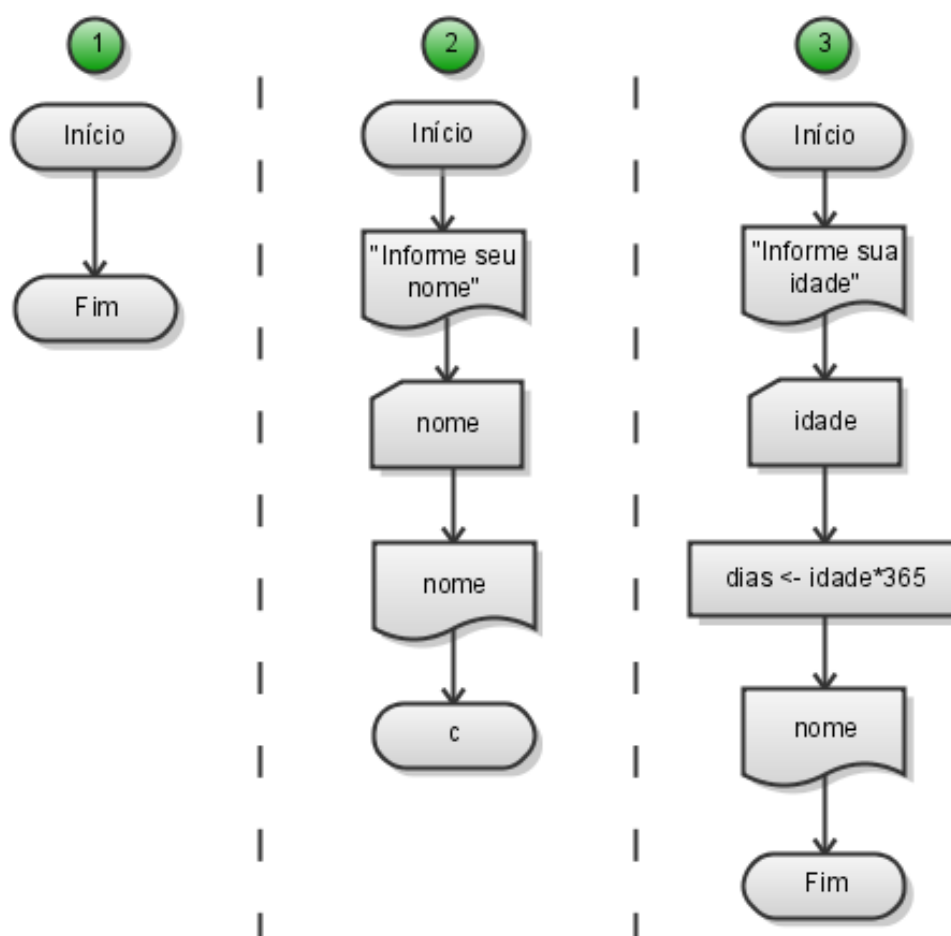
3. FLUXOGRAMA

Forma de descrever algoritmos utilizando uma **estrutura lógica** representada por **figuras geométricas**.

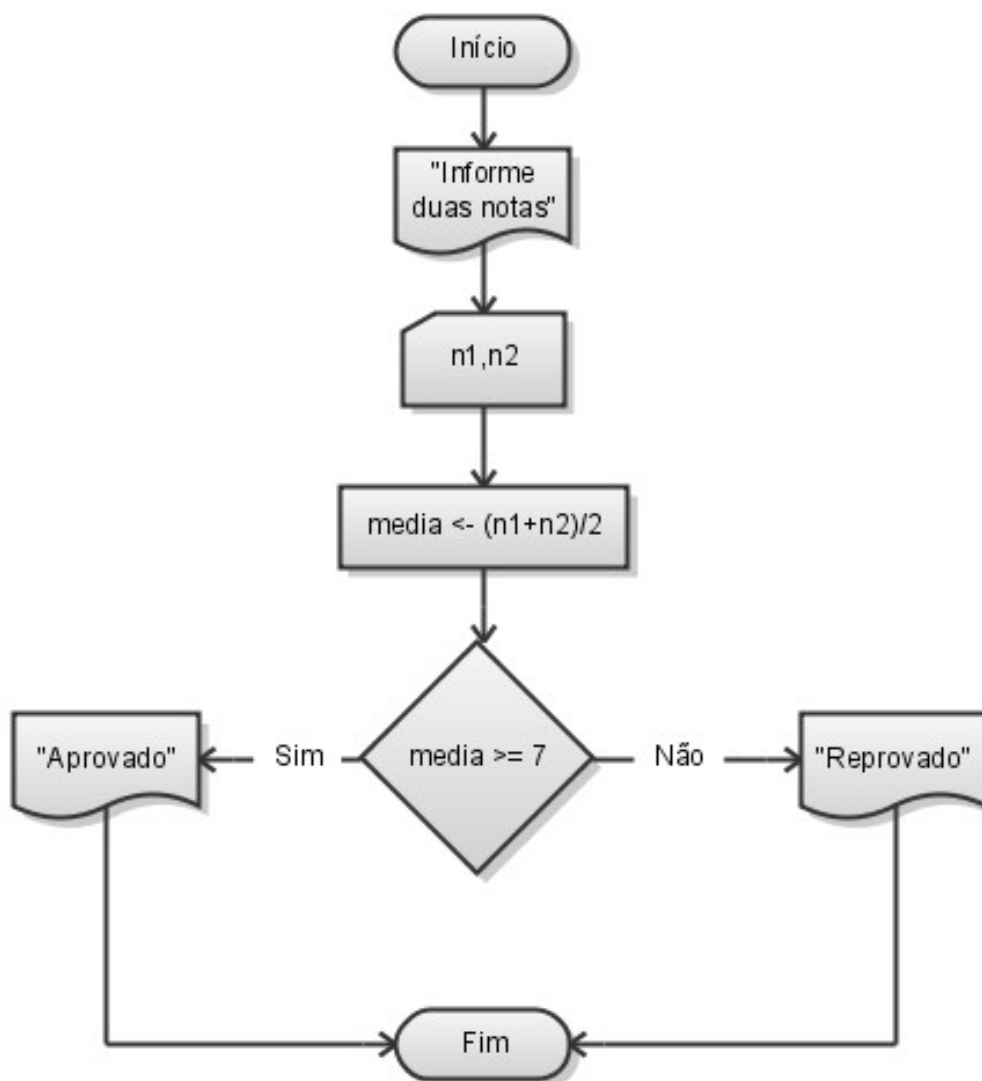
3.1. Figuras

	Início e final do Fluxograma
	Operação de entrada de dados
	Operação de Saída de dados
	Decisão
	Operação de atribuição e cálculos diversos

3.2. Utilizando Fluxogramas



3.3. Estrutura Condicional



4. PORTUGUES ESTRUTURADO (PORTUGOL, PSEUDO-CÓDIGO)

A definição de **pseudo-código** é uma alusão à uma linguagem de programação genérica na qual é possível representar um algoritmo de forma semelhante à das linguagens de programação. Esta pseudo-linguagem é escrita em português e por isto também recebe o nome de Português Estruturado ou Portugol. Apesar de existirem ferramentas para interpretar o Português Estruturado utilizando um computador, assim como nas linguagens de programação, sua sintaxe é muito variável já que não existe uma padronização definida e cada autor adota o que lhe é conveniente.

O Português Estruturado é muito utilizado no aprendizado de lógica nos cursos de computação. Sua sintaxe e semântica é muito similar às linguagens disponíveis no mercado e a sua prática serve como base para começar a escrever programas em qualquer linguagem de programação.

4.1. Estrutura Básica

```
programa soma
    ...
início
    ...
fim
```

4.2. Declaração: Variáveis e Constantes

As seguintes regras devem ser cumpridas para a declaração de variáveis e constantes:

- As variáveis e constantes só podem ser declaradas antes do início do programa;
- As regras para nomes de variáveis são as mesma definidas no item 2.1;
- As variáveis são declaradas com o comando: **var**;
- As constantes são declaradas com o comando: **const**;






Para declarar uma variável ou constante deve-se definir o tipo de informação que a mesma conterá. Ex:

- **inteiro**: 2456
- **real**: 1,34
- **caracter**: “Maria”
- **logico**: verdadeiro

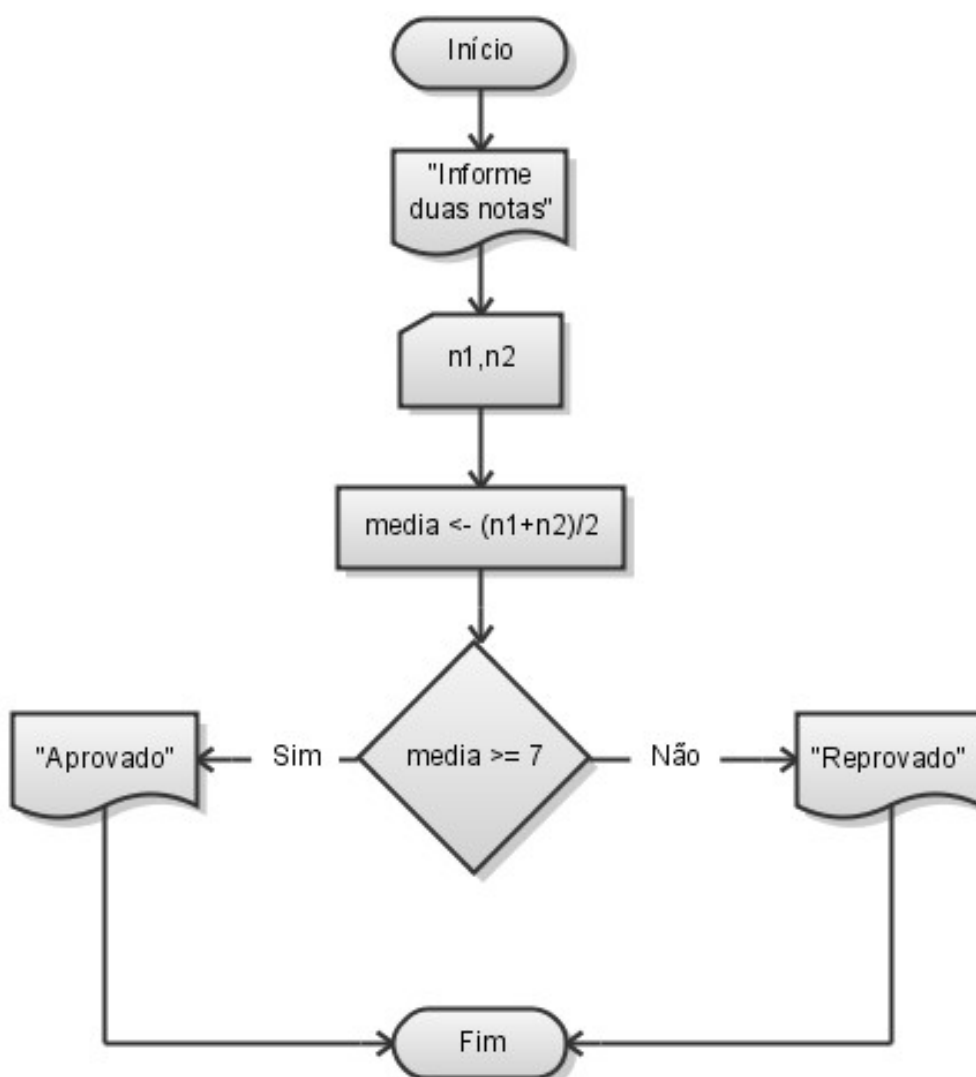
Ex:

```
programa soma
    var a:inteiro
        b,c:inteiro
    const
        pi:real
início
    ...
fim
```

4.3. De Fluxograma para Português Estruturado

	início, fim
	leia(a)
	escreva(a)
	se, fim se
	a<-10

Algoritmo Utilizando Fluxograma



Convertendo o Fluxograma acima em Português Estruturado:

```

programa media
    var n1,n2,m:real
início
    escreva("informe 2 notas")
    leia(n1,n2)
    media <- (n1+n2)/2
    se (media>=7) então
        escreva("Aprovado")
    senão
        escreva("Reprovado")
    fim se
fim
  
```

4.4. Operadores Relacionais

Operador	Descrição
=	Igualdade, A = B
<	Menor que, A < B
>	Maior que, A > B
<=	Menor ou igual, A <= B
>=	Maior ou igual, A >= B
<>	Diferente de, A <> B

4.5. Operadores Lógico

Operador	Descrição
NÃO	Nega o valor lógico, se a expressão é FALSA, se torna VERDADEIRO e vice-versa. Exemplo: NÃO(A > B)
E	A expressão somente será verdadeira se TODOS os valores operados forem verdadeiros. Nesse caso, basta um valor ser falso para que toda expressão seja falsa. Exemplo: (A > B) E (X > Y), somente será verdade se e somente se A for maior que B e X for maior que Y.
OU	A expressão somente será falsa se TODOS os valores operados forem falsos. Nesse caso, basta um valor ser verdadeiro para que toda expressão seja verdadeira. Exemplo: (A > B) OU (X > Y), será verdade se ou A for maior que B ou X for maior que Y ou os dois.

4.6. Estruturas de Repetição

```
a<-0
enquanto (a < 100) faça
    a <- a + 1
    escreva(a,"")
fim enquanto
```

```
.....
a<-0
repita
    a <- a + 1
    escreva(a,"")
até que (a = 100)
```

```
.....
para a de 1 até 100 passo 1 faça
    escreva(a,"")
fim para
```

Quando utilizar cada estrutura de repetição:

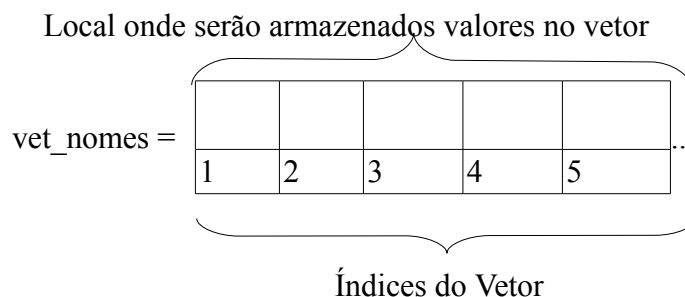
- **Enquanto** – A estrutura ENQUANTO é utilizada quando é necessário fazer um teste condicional no **início** do bloco de repetição.
- **Repita** – E estrutura REPITA é utilizada quando é necessário fazer um teste condicional no **final** do bloco de repetição. Observe que, como o teste é feito no final do bloco de repetição, este bloco **é executado pelo menos uma vez**.
- **Para** – A estrutura PARA é utilizada quando sabe-se exatamente o **quanto um bloco de repetição deve se repetir**.

4.7. Vetores

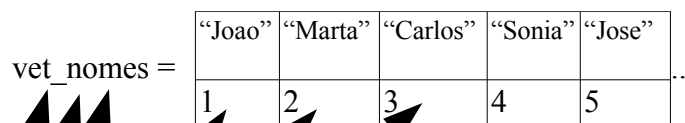
Um vetor é uma variável que possui a capacidade de armazenar **vários valores** de um **mesmo tipo**. Para declarar um vetor fazemos:

```
var vet_nomes:vetor[1..100] de caracter
```

Podemos representar graficamente o vetor declarado da seguinte forma:



Os valores do vetor são identificados pelo **nome da variável** e o **índice**, que indica a **posição onde o valor foi armazenado no vetor**. Observe o vetor abaixo:



Se fizermos:

```
escreva(vet_nomes[1]) o resultado é: João
escreva(vet_nomes[2]) o resultado é: Marta
escreva(vet_nomes[3]) o resultado é: Carlos
```

O índice de um vetor pode ser identificado por um **valor**, **variável** ou **expressão** e é utilizado muitas vezes em conjunto com as estruturas de repetição principalmente quando queremos preencher um vetor com valores informados pelo usuário. Ex:

```
programa vetores
  var vet_nomes:vetor[1..100] de caracter
início
  para x de 1 até 100 faça
    leia(vet_nomes[x])
  fim para
fim
```


5. HISTÓRIA DO JAVA

(Fonte: Adaptação da Wikipedia) Em 1991, na Sun Microsystems, foi iniciado o Green Project, o berço do Java, uma linguagem de programação orientada a objetos. Os mentores do projeto eram Patrick Naughton, Mike Sheridan, e **James Gosling**. O objetivo do projeto não era a criação de uma nova linguagem de programação, mas antecipar e planejar a “próxima onda” do mundo digital. Eles acreditavam que, em algum tempo, haveria uma convergência dos computadores com os equipamentos e eletrodomésticos comumente usados pelas pessoas no seu dia-a-dia.

Para provar a viabilidade desta idéia, 13 pessoas trabalharam arduamente durante 18 meses. No verão de 1992 eles emergiram de um escritório de Sand Hill Road no Menlo Park com uma demonstração funcional da idéia inicial. O protótipo se chamava *7 (leia-se “StarSeven”), um controle remoto com uma interface gráfica touchscreen.



Para o *7, foi criado um mascote, hoje amplamente conhecido no mundo Java, o Duke. O trabalho do Duke no *7 era ser um guia virtual ajudando e ensinando o usuário a utilizar o equipamento. O *7 tinha a habilidade de controlar diversos dispositivos e aplicações. James Gosling especificou uma nova linguagem de programação para o *7. Gosling decidiu batizá-la de “Oak”, que quer dizer carvalho, uma árvore que ele podia observar quando olhava pela sua janela.



O próximo passo era encontrar um mercado para o *7. A equipe achava que uma boa idéia seria controlar televisões e vídeo por demanda com o equipamento. Eles construíram um demo chamado MovieWood, mas infelizmente era muito cedo para que o vídeo por demanda bem como as empresas de TV a cabo pudessem viabilizar o negócio. A idéia que o *7 tentava vender, hoje já é realidade em programas interativos e também na televisão digital. Permitir ao telespectador interagir com a emissora e com a programação em uma grande rede de cabos, era algo muito visionário e estava muito longe do que as empresas de TV a cabo tinham capacidade de entender e comprar. A idéia certa, na época errada.

Entretanto, o estouro da Internet aconteceu e rapidamente uma grande rede interativa estava se estabelecendo. Era este tipo de rede interativa que a equipe do *7 estava tentando vender para as empresas de TV a cabo. E, da noite para o dia, não era mais necessário construir a infra-estrutura para a rede, ela simplesmente estava lá. Gosling foi incumbido de adaptar o Oak para a Internet e em janeiro 1995 foi lançada uma nova versão do Oak que foi rebatizada para Java. A tecnologia Java tinha sido projetada para se mover por meio das redes de dispositivos heterogêneos, redes como a Internet. Agora aplicações poderiam ser executadas dentro dos browsers nos Applets Java e tudo seria disponibilizado pela Internet instantaneamente. Foi o estático HTML dos browsers que promoveu a rápida disseminação da dinâmica tecnologia Java. A velocidade dos acontecimentos seguintes foi assustadora, o número de usuários cresceu rapidamente, grandes fornecedores de tecnologia, como a IBM anunciaram suporte para a tecnologia Java.

Desde seu lançamento, em maio de 1995, a plataforma Java foi adotada mais rapidamente do que qualquer outra linguagem de programação na história da computação. Em 2004 Java atingiu a marca de 3 milhões de desenvolvedores em todo mundo. Java continuou crescendo e hoje é uma referência no mercado de desenvolvimento de software. Java tornou-se popular pelo seu uso na Internet e hoje possui seu ambiente de execução presente em web browsers, mainframes, SOs, celulares, palmtops e cartões inteligentes, entre outros.

6. A MAQUINA VIRTUAL JAVA

(Fonte: Adaptação da Wikipedia) Programas Java não são traduzidos para a linguagem de máquina, como outras linguagens estaticamente compiladas e sim para uma representação intermediária, chamada de bytecodes.

Os bytecodes são interpretados pela máquina virtual Java (JVM - Java Virtual Machine). Muitas pessoas acreditam que por causa desse processo, o código interpretado Java tem baixo desempenho, mas isso não é verdade. Durante muito tempo esta foi uma afirmação verdadeira.

Java hoje já possui um desempenho próximo do C++. Isto é possível graças a otimizações como a compilação especulativa, que aproveita o tempo ocioso do processador para pré-compilar bytecode para código nativo. Outros mecanismos ainda mais elaborados como o HotSpot da Sun, que guarda informações disponíveis somente em tempo de execução (ex.: número de usuários, processamento usado, memória disponível), para otimizar o funcionamento da JVM, possibilitando que a JVM vá "aprendendo" e melhorando seu desempenho. Isto é uma realidade tão presente que hoje é fácil encontrar programas corporativos e de missão crítica usando tecnologia Java. No Brasil, por exemplo, a maioria dos Bancos utiliza a tecnologia Java para construir seus home banks, que são acessados por milhares de usuários diariamente. Grandes sites como o eBay utilizam Java para garantir alto desempenho.

7. DE PORTUGUÊS ESTRUTURADO PARA JAVA

Estrutura Básica	<pre> programa Exec1 inicio Código do Programa fim </pre>	<pre> import java.util.Scanner; public class Exec1 { public static void main(String[] args) { Código do Programa } } </pre>
Declaração de Variáveis	<pre> var c,x,y:inteiro nome:caracter nota:real </pre>	<pre> int c,x,y; String nome; double nota; </pre>
Atribuição	idade ← 30	idade = 30;
Igual	(a = 20)	(a == 20)
Diferente	(a <> 20)	(a != 20)
E, OU	<pre> (a > 0) E (a<11) (a == 1) OU (a == 2) </pre>	<pre> (a > 0) && (a<11) (a == 1) (a == 2) </pre>
Parte inteira da divisão	5 DIV 2	(int)5/2
Resto da Divisão	5 MOD 2	5 % 2
Entrada de Dados	leia(a)	a = (new Scanner(System.in)).nextLine();
Saída de Dados	escreva("Ola Mundo")	System.out.print("Ola Mundo");
Estrutura Condicional	<pre> se (x>10) então ... fim se ----- se (a>20) então ... senão ... fim se </pre>	<pre> if (x>10) { ... } ----- if (a>20) { ... } else { ... } </pre>
Estrutura de Repetição	<pre> Para x de 1 até 100 faça ... fim para ----- a<-0 repita a<-a+1 até que (a=10) ----- a<-0 enquanto (a<10) faça ... fim enquanto </pre>	<pre> for(x=1;x<=100;x++){ ... } ----- a=0; do { a=a+1 } while(a!=0) (observem que é o inverso) ----- a=0; while(a<10){ a=a+1; } </pre>

7.1. Vetor

Para declarar um vetor em Java basta colocar "[]" após o tipo da variável e inicializar o vetor com o seu tamanho. Ex:

```
int[] numeros = new int[10];  
String[] nomes = new String[5];
```

No primeiro caso criamos um vetor de inteiro com 10 posições enquanto que no segundo um vetor de String com 5 posições. Da mesma forma que no Português Estruturado, o Java acessa as posições dos vetores pra armazenar e ler dados. A única diferença é que no Português Estruturado o índice inicial é 1 e no Java este índice é 0. Ex:

```
numeros[0] = (new Scanner(System.in)).nextInt();  
System.out.println(numeros[0]);
```

Observem que o número que representa o tamanho do vetor pode ser um valor, variável ou expressão. Sendo assim podemos fazer:

```
int tamanho = (new Scanner(System.in)).nextInt();  
String[] nomes = new String[tamanho];
```

No exemplo acima o tamanho do vetor nomes foi definida pela variável **tamanho** informada pelo usuário. Para imprimir o conteúdo de um vetor é necessário percorrer item-a-item através de uma estrutura de repetição. Ex:

```
for(int x=0;x<nomes.length;x++){  
    System.out.println(nomes[x]);  
}
```

No exemplo acima podemos observar também a propriedade **length** da variável nomes. Ela contém o tamanho definido para o vetor e é muito utilizada nas estruturas de repetição.

7.2. Matriz

Uma matriz em Java é um vetor que armazena em cada posição um outro vetor. Quando isto acontece dizemos que o vetor possui 2 dimensões. Quando manipulamos vetores ou matrizes em Java dizemos que estamos trabalhando com **array**. Para declarar uma matriz fazemos:

```
int[][] numeros = new int[3][2];
```

Neste caso estamos dizendo que o **vetor mais externo possui tamanho 3** e que em **cada posição deste vetor será armazenado um novo vetor de tamanho 2**. A imagem que melhor representa uma matriz é a seguinte:

mat_nomes =	“Igor” “Takenami”		“Joao” “Paulo”		“Maria” “Antonieta”		..
	0	1	0	1	0	1	
	0		1		2		

As mesmas regras aplicadas aos vetores também são válidas para as matrizes. Observe que para percorrer uma matriz é necessário uma estrutura contendo um **for** dentro de um **for**. Ex:

```
for(int x=0;x<mat_nomes.length;x++){
    for(int y;y<mat_nomes[0].length;y++){
        System.out.println(mat_nomes[x][y]);
    }
}
```

8. RECURSOS DO JAVA

8.1. Formatar Valores Decimais

```
double val = 7.7777777;  
System.out.println((new DecimalFormat("#.##")).format(val));
```

Formata um número decimal seguindo a máscara "#.##". A máscara pode ser alterada de acordo com a necessidade.

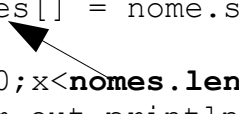
8.2. Números Aleatórios

```
int a = new Random().nextInt(4);
```

Gera um número **inteiro** aleatório variando de 0 a 3. O parâmetro passado para a função `nextInt` define o range de números aleatórios que podem ser gerados. Este parâmetro pode ser um valor, variável ou expressão.

8.3. Converter String em vetor de String

```
String nome = "Joao Silva Costa";  
  
String nomes[] = nome.split(" ");  
  
for(int x=0;x<nomes.length;x++){  
    System.out.println(nomes[x]);  
}
```



Converte uma String em um vetor de String utilizando o parâmetro passado para a função `split` como delimitador. Como o tamanho do vetor de String será definido pelo delimitador passado em `split` a propriedade **length** será fundamental para saber o tamanho do vetor gerado.

8.4. Converter String em vetor de char

```
String nome = "joao";  
char[] caracteres = nome.toCharArray();
```

Converte uma String em um vetor de char. Cada caracter da String será alocado em uma posição do vetor de char. Como o tamanho do vetor de char será definido pelo tamanho da String a propriedade **length** será fundamental para saber o tamanho do vetor gerado.

8.5. Imprimir Caracteres ASCII

```
System.out.println( (char) 178 );
```

Imprime o conteúdo de um caracter ASCII. Alguns dos caracteres ASCII que podem ajudar na construção da interface estão descritos nos valores decimais da tabela abaixo:

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	Ł	224	E0	α
129	81	ù	161	A1	í	193	C1	ł	225	E1	β
130	82	é	162	A2	ó	194	C2	Ť	226	E2	Γ
131	83	â	163	A3	ú	195	C3	ţ	227	E3	Π
132	84	ä	164	A4	ñ	196	C4	—	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	†	229	E5	σ
134	86	ã	166	A6	ª	198	C6	‡	230	E6	μ
135	87	ç	167	A7	º	199	C7	‡	231	E7	ι
136	88	ê	168	A8	¿	200	C8	ℒ	232	E8	Φ
137	89	ë	169	A9	ƒ	201	C9	ℝ	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	℔	234	EA	Ω
139	8B	ÿ	171	AB	½	203	CB	℥	235	EB	Ö
140	8C	î	172	AC	¼	204	CC	℥	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	=	237	ED	ø
142	8E	Ë	174	AE	«	206	CE	≠	238	EE	ε
143	8F	Ä	175	AF	»	207	CF	±	239	EF	∩
144	90	É	176	B0	░	208	D0	℔	240	FO	≡
145	91	æ	177	B1	▒	209	D1	℥	241	F1	±
146	92	Æ	178	B2	▓	210	D2	π	242	F2	≥
147	93	ô	179	B3		211	D3	℔	243	F3	≤
148	94	ö	180	B4	†	212	D4	℔	244	F4	[
149	95	ò	181	B5	‡	213	D5	℥	245	F5]
150	96	û	182	B6	‡	214	D6	℥	246	F6	÷
151	97	ù	183	B7	π	215	D7	≠	247	F7	≈
152	98	ÿ	184	B8	ƒ	216	D8	≠	248	F8	°
153	99	Ö	185	B9	‡	217	D9	¡	249	F9	▪
154	9A	Ü	186	BA		218	DA	ƒ	250	FA	·
155	9B	÷	187	BB	π	219	DB	■	251	FB	√
156	9C	£	188	BC	℔	220	DC	■	252	FC	¤
157	9D	¥	189	BD	℔	221	DD	■	253	FD	£
158	9E	ℒ	190	BE	℔	222	DE	■	254	FE	■
159	9F	f	191	BF	¬	223	DF	■	255	FF	□

Utilizando ASCII Art podemos melhorar a apresentação dos caracteres em tela. Um bom site para utilizar o recurso é: <http://ascii.mastervb.net/>

8.6. Abrindo um Arquivo

```
import java.io.FileReader;
import java.io.Exception;

public class LerArquivo {

    public static void main(String[] args) throws Exception {

        FileReader in = new FileReader("dados.txt");
        String conteudo = "";
        int i;

        while((i = in.read())!=-1){
            conteudo = conteudo +(char)i;
        }

        System.out.println(conteudo);
        in.close();
    }
}
```

Código para abrir um arquivo texto localizado no disco local.

8.7. Gravando em Arquivo

```
import java.io.FileWriter;
import java.io.Exception;

public class arquivo {

    public static void main(String[] args) throws Exception {

        FileWriter x = new FileWriter("dados.txt",true);
        String conteudo = "1,2,3,4\n5,6,7,8\n9,10,11,12";
        x.write(conteudo);
        x.close();

    }
}
```

Código para gravar o conteúdo de um String em um arquivo no disco local.

9. PREPARAÇÃO PARA O INTERDISCIPLINAR

9.1. Trabalho de Jogos

Jogo 1

A equipe deve desenvolver um JOGO DA VELHA utilizando a Linguagem de Programação JAVA e obedecendo aos seguintes requisitos:

- O jogo será jogado por dois jogadores que utilizarão o mesmo computador;
- “O” inicia o jogo, logo depois será a vez do “X” e assim sucessivamente;
- O computador deve identificar o final do jogo apontando o vencedor e a quantidade de jogadas necessárias para vencer o jogo;
- O jogo deve avisar caso não exista vencedor;
- Comprometimento, criatividade, interface e lógica serão critérios para avaliação.

```

X | X | X
---+---+---
X | X | X
---+---+---
X | X | X

```

Jogo 2

A equipe deve desenvolver um JOGO DA FORCA utilizando a Linguagem de Programação JAVA e obedecendo aos seguintes requisitos:

- O jogador escolherá o nível, podendo ser básico ou avançado. O nível básico possui palavras de 8 a 10 letras, o nível avançado palavras com mais de 10 letras;
- O programa deve selecionar a palavra aleatoriamente;
- O jogador pode errar no máximo 6 vezes e a cada erro uma parte do boneco será desenhado;
- O computador deve identificar se o jogador conseguiu ou não descobrir a palavra. Caso o jogador tenha acertado, o jogo deve informar quantas jogadas foram necessárias;
- Comprometimento, criatividade, interface e lógica serão critérios para avaliação.

```

  O
 / | \
/   \
/   \

```

Jogo 3

Sua equipe deve desenvolver o jogo ACERTE O NÚMERO utilizando a Linguagem de Programação JAVA e obedecendo aos seguintes requisitos:

- O jogador escolherá o nível, podendo ser básico ou avançado. No nível básico o jogador deve adivinhar 3 números e no nível avançado serão 5 números;
- Os números são selecionados pelo computador aleatoriamente podendo variar entre 1 e 100;
- A cada rodada o jogador deve palpar os números e o computador informará se cada numero informado é maior ou menor que o sorteado;
- O máximo de palpites para tentar adivinhar os números são 10;
- O computador deve identificar se o jogador conseguiu adivinhar os números e quantos palpites foram necessários;
- Comprometimento, criatividade, interface e lógica serão critérios para avaliação.

9.2. Dicas de Apresentação

- Postura profissional;
- Preparação do equipamento;
- Apresentação pessoal;
- Utilizar os recursos disponíveis para apresentação;
- Apresentação da equipe;
- Posicionamento em sala;
- Mudança de apresentador;
- Segurança;
- Cores nos slides;
- Contextualizar, Desenvolver e Concluir;
- Interdisciplinaridade;
- Surpreender a banca;
- Evitar entrar no código-fonte;
- Abordar coisas interessantes:
 - Processo de desenvolvimento;
 - Dificuldades;
 - Sucesso
- Vender o produto.

LISTA DE EXERCÍCIOS I

- a) Faça um algoritmo que solicite ao usuário sua idade e informe se o mesmo já pode tirar carteira de motorista.
- b) Faça um programa para ler a temperatura do corpo medida com um termômetro. Caso a temperatura seja maior que 37 graus o paciente esta com febre, caso contrario, sua temperatura esta normal. Após ler a temperatura imprima a mensagem apropriada.
- c) Faça um algoritmo que solicite ao usuário 2 números e imprima o maior deles.
- d) Faça um programa que leia 2 provas (peso 3 cada uma) e um trabalho (peso 1). Caso a média final seja maior ou igual a 7 imprimir a mensagem “aprovado”, caso contrário, imprimir a mensagem “reprovado”.
- e) Faça um programa que leia 2 valores e informe se os mesmo são divisores perfeitos.
- f) Faça um algoritmo que leia um valor digitado pelo usuário e informe se o numero é par ou impar.
- g) Faça um programa que leia o peso e a altura do usuário e informa se o mesmo está obeso. Lembre que para saber se uma pessoa está obesa deve-se utilizar a formula: $imc = \text{peso}/\text{altura}^2$. Caso o resultado do imc seja maior que 30 o usuário está obeso.
- h) Faça um programa que leia o valor do salário base do usuário mais suas gratificações. Aplicar as alíquotas do imposto de renda e após os devidos descontos informar o valor do salário líquido.
 - a) até 999,00 – não paga imposto
 - b) até 1499,00 – desconto de 15%
 - c) acima de 1499,00 – desconto de 27%
- i) Uma empresa concederá um aumento a seus funcionários de acordo com seus cargos. Para o cargo de “programador” o aumento será de 30%. Para todos os outros cargos o aumento é de 10%. Faça um algoritmo onde o usuário informe seu cargo e salário e imprima o novo salário de acordo com o aumento que será concedido.
- j) Faça um algoritmo onde o usuário informe sua idade em dias e imprima em **ANOS**, **MESES** e **DIAS**.
- k) O custo ao consumidor de um carro novo é a soma do custo de fabricação mais o percentagem do distribuidor e dos impostos. Supondo que o percentual do distribuidor seja de 28% e os impostos de 45%, escreva um algoritmo que leia o custo de fabricação de um carro e escreva o custo ao consumidor.
- l) Faça um programa onde o usuário informe 2 valores e a operação que deseja realizar. As seguintes operações podem ser realizadas:
 - a) “soma”
 - b) ”subtração”
 - c) ”multiplicacao”
 - d) “divisao”
- m) Os valores devem ser operados de acordo com a operação informada (soma, subtração, multiplicação e divisão) e o algoritmo deve imprimir o resultado da operação.

- n) Elabore um algoritmo que dada a idade de um nadador classifique sua categoria de acordo com a tabela abaixo. Desconsidere o nadador abaixo de 5 anos.
- a) infantil A = 5-7 anos
 - b) infantil B = 8-10 anos
 - c) juvenil A = 11-13 anos
 - d) juvenil B = 14-17 anos
 - e) adulto = maiores de 18 anos

- o) Um banco concederá um crédito especial, aos seus clientes, variável com o saldo médio dos últimos anos. Faça um algoritmo que leia o saldo médio de um cliente e calcule o valor do crédito de acordo com a tabela abaixo. Mostre uma mensagem informando o saldo médio e o valor do crédito.

de 0 a 200	Nenhum crédito
de 201 a 400	20% do valor do saldo médio
de 401 a 600	30% do valor do saldo médio
acima de 600	40% do valor do saldo médio

- p) Uma fabrica possui uma maquina para aquecimento de sua matéria prima. Nesta maquina o usuário informa o tempo necessário para aquecimento em segundos. Você deve escrever um algoritmo onde o usuário possa dizer o tempo de aquecimento, em segundos, e imprima no formato **Hora, Minutos e Segundos**.

Ex: **13230 Segundos** = 3:40:30

- q) Escrever um algoritmo que leia um valor em reais e calcule qual o menor número possível de notas de 100, 50, 10, 5 e 1 em que o valor lido pode ser decomposto. Escrever o valor lido e a relação de notas necessárias.

Ex: O valor R\$ 153,00 pode ser decomposto em:

- a. 1 de 100
- b. 1 de 50
- c. 0 de 10
- d. 0 de 5
- e. 3 de 1

LISTA DE EXERCÍCIOS II

- 1) Construir um algoritmo para solicitar dois números ao usuário e informar: a soma, a diferença e o produto.
- 2) Faça um Algoritmo que solicite ao usuário 3 (três) valores numéricos, positivos e diferentes entre si. Imprima o maior deles.
- 3) Construa um algoritmo que calcule e informe a quantidade de gasolina que será preciso colocar no carro e quanto irá custar para o seu dono ir até a sua fazenda. O usuário deverá informar a distância que será percorrida e o preço do litro da gasolina. Para o cálculo, sabe-se que o carro faz em média 12 km/litro.
- 4) Construir um algoritmo para ler a quantidade de horas trabalhadas no mês por um operário. Em seguida, calcular e informar o seu salário sabendo-se que ele ganha R\$ 10,00 por hora e, que as horas que excedem 50 valem R\$ 20,00.
- 5) Construir um algoritmo para ler o preço unitário e a quantidade de um determinado produto vendido e calcular o valor total da venda. Se o total for menor do que R\$ 500,00, solicitar do usuário o valor do frete e acrescentar.
- 6) Construir um algoritmo para lê o preço unitário e a quantidade de um determinado produto vendido e calcular o valor total da venda. Se o valor total for maior do que R\$ 500,00, informar que pode parcelar em até 5 vezes, solicitar o número de parcelas desejado e informar o valor da parcela. Caso contrário, informar que deve pagar a vista e informar o valor total.
- 7) Um analista quer comprar um computador novo que custa R\$2.000,00. Para tentar comprar, ele está prestando serviços a uma empresa durante 4 meses, recebendo um salário mensal variável entre zero e R\$800,00. Construa um algoritmo para verificar a situação do analista após trabalhar 3 meses. O algoritmo deve solicitar ao usuário o valor dos 3 primeiros salários e:
 - a) se o valor já for suficiente para comprar o computador, informar
 - b) se ainda não é suficiente, verificar se ainda pode conseguir comprar com o salário do quarto mês (considerando que o máximo que ele pode receber é R\$800,00):
 - c) se ainda for possível, informar qual o mínimo que precisa receber no quarto mês para comprar o computador.
- 8) Um pescador comprou um computador para controlar o rendimento diário de seu trabalho. Toda vez que ele traz um peso de peixes maior que o estabelecido pelo regulamento de pesca do estado da Bahia (50 quilos) deve pagar um multa de R\$ 4,00 por quilo excedente. Elabore um algoritmo onde o usuário deverá informar a quantidades de quilos e, o sistema deverá exibir o valor da multa que deverá pagar (Somente se houver excesso de peso). Caso não haja excesso de peso, deverá ser exibida a quantidade de quilos juntamente com a mensagem “A quantidade não foi excedida.”
- 9) Dados o nome do vendedor de uma empresa, seu salário fixo e o total de vendas efetuadas no mês, fazer o algoritmo para calcular o salário a ser recebido pelo vendedor sabendo-se que cada vendedor ganha uma comissão proporcional às vendas efetuadas. A comissão é de 3% sobre o total das vendas até R\$10.000,00 e de 5% se as vendas ultrapassarem este valor. Imprimir o nome do vendedor, seu salário básico, o total de vendas no mês e o salário a ser recebido.

LISTA DE EXERCÍCIOS III

- a) Para cada sequência abaixo escreva um algoritmo que imprima os valores atendendo a respectiva lógica:
- a) 1,2,3...100
 - b) 2,4,6...100
 - c) 100,99,98,97...1
 - d) 100,98,96,94...2
 - e) 99,97,95,93...1
- b) Para cada sequência abaixo escreva um algoritmo que imprima os 100 primeiros números atendendo a respectiva lógica:
- a) 1,3,4,6,7,9,10,12,13...
 - b) 1,1,2,3,5,8...
- c) Faça um algoritmo que conte de 1 a 100 e a cada múltiplo de 10 emita uma mensagem: “Múltiplo de 10”.
- d) Faça um algoritmo que solicite ao usuário para digitar 50 números entre 1 e 100. Informe o **maior** número digitado.
- e) Faça um algoritmo que solicite ao usuário para digitar 50 números entre 1 e 100. Informe o **menor** número digitado.
- f) Faça um algoritmo que leia um valor numérico fornecido pelo usuário e verifique se o valor é maior que 50. Caso seja maior que 50, solicite outro valor até que seja menor ou igual.
- g) Faça um algoritmo onde o usuário defina a quantidade de número que ele vai fornecer. Após fornecer estes números o programa deve calcular a média destes valores.
- h) Faça um algoritmo que solicite ao usuário para digitar um número. Informe quais os divisores deste número.
- i) Faça um algoritmo que solicite ao usuário para digitar um valor numérico entre 0 e 5. Caso o número digitado esteja entre 1 e 5 imprima em extenso o valor (Ex: “Um”, “Dois”, “Três”, “Quarto”, “Cinco”) e repita a operação. Caso o usuário digite 0 o programa exibe a mensagem “Saindo do Algoritmo...” e chega ao fim.
- j) Uma rainha requisitou os serviços de um monge e disse-lhe que pagaria qualquer preço. O monge, necessitando de alimentos, indagou à rainha sobre o pagamento, se poderia ser feito com grãos de trigo dispostos em um tabuleiro de xadrez, de tal forma que o primeiro quadro deveria conter apenas um grão e os quadros subsequentes, o dobro de grãos do quadro anterior. A rainha achou o trabalho barato e pediu que o serviço fosse executado, sem se dar conta de que seria impossível efetuar o pagamento. Faça um algoritmo para calcular o número de grãos que o monge esperava receber.

LISTA DE EXERCÍCIOS IV

- 1) Escreva um algoritmo que calcule a média dos números digitados pelo usuário. Termine a leitura se o usuário digitar zero.
- 2) Construa um algoritmo para solicitar ao usuário três notas de cada um dos 400 alunos do curso de Sistema de Informação, para cada aluno, calcular e informar a sua média, informando também se foi aprovado ou reprovado (a média para ser aprovado é 7). Ao final, o programa deverá ainda informar quantos alunos foram reprovados.
- 3) Em uma pesquisa realizada com 80 alunos, foram levantados os seguintes dados: IDADE e SEXO (M ou F). Construa um algoritmo que leia estes dados e informe:
 - a) A quantidade de pessoas com menos de 18 anos;
 - b) A média de idade das mulheres;
 - c) A porcentagem de homens acima de 22 anos sobre o total de homens.
- 4) Uma empresa fez uma pesquisa de mercado com 1000 entrevistados para saber se as pessoas gostaram ou não de um novo produto lançado. Para isto forneceu o sexo do entrevistado e sua resposta (sim ou não). Faça o algoritmo que calcule e imprima o seguinte:
 - a) o número de pessoas que responderam sim;
 - b) o número de pessoas que responderam não;
 - c) o percentual de pessoas do sexo feminino que responderam sim;
 - d) o percentual de pessoas do sexo masculino que responderam não.
- 5) Em uma eleição presidencial existem quatro candidatos. Os votos são informados através de códigos. Os dados utilizados para a contagem dos votos obedecem à seguinte codificação:
 - a) 1, 2, 3, 4 = voto para os respectivos candidatos;
 - b) 5 = voto em branco, qualquer outro valor considerar voto nulo

Elaborar um algoritmo que leia o código do candidato em um voto. Calcule e escreva:

- a) total de votos para cada candidato;
- b) total de votos nulos;
- c) total de votos em branco.

Como finalizador do conjunto de votos, tem-se o valor 0.

- 6) Faça um algoritmo para solicitar ao usuário 1 número que pode variar de 0 a 4. Se o usuário digitar um número maior que 4 o sistema deve exibir a mensagem “Número inválido e solicitar outro número. Caso o usuário digite 0 o programa deve finalizar. Caso o usuário digite qualquer um dos outros números (1, 2, 3 ou 4) o programa deve solicitar mais dois números e a depender da tabela abaixo deve exibir o valor da operação e repetir o procedimento. O programa só encerra quando o usuário digitar 0.

- 1 – Soma
- 2 – Subtração
- 3 – Multiplicação
- 4 – Divisão

LISTA DE EXERCÍCIOS V

- 1) Uma empresa estatística precisa fazer uma pesquisa para saber com quantos anos os adolescentes Brasileiros começam a trabalhar. Também é necessário saber se quem inicia a vida profissional antes dos 16 anos é em sua maioria homens ou mulheres. Você foi contratado para desenvolver um algoritmo que coleta **nome, sexo e idade que começou a trabalhar** de 1000 entrevistados com no máximo 20 anos. Após colher todos os dados (dos 1000 entrevistados) o algoritmo deve:
 - a) Informar qual o **nome** e com **qual idade começou a trabalhar** o entrevistado que **iniciou mais cedo sua carreira profissional**;
 - b) Informar se a **maioria** dos adolescentes que inicia a vida profissional **antes dos 16 anos é homem ou mulher**;
 - 2) Supondo que o PIB do Brasil em 2008 é de US\$ 187.274.412 e o dos EUA é US\$ 299.902.694. Sabendo-se que os EUA possuem um crescimento anual de 2% no seu PIB e que o Brasil tem crescimento anual de 4%, construa um algoritmo que determine o ano em que o PIB da economia brasileira será igual ou maior que o PIB da economia americana.
 - 3) Você foi contratado por uma empresa para construir um algoritmo capaz de gerenciar a distribuição dos tickets de vale-refeição e outras informações relacionadas aos salários dos seus 500 funcionários. Para **cada um dos funcionários**, o programa deverá solicitar o **nome, o sexo, o salário e o número de dias trabalhados no mês**. O programa deve calcular e informar, **para cada um**:
 - a) Qual deveria ser o seu aumento de salário para deixar de ter direito ao plano de saúde gratuito ou se já perdeu o direito a este benefício (considere que têm direito homens com salário até R\$1.200,00 e mulheres com salário até R\$1.500,00);
 - b) O valor total recebido em tickets por cada funcionário;
- Alem disso, **ao final**, o programa deve mostrar:
- c) O numero total de funcionários;
 - d) A média do valor total recebido em tickets pelos funcionários;
 - e) O valor total gasto pela empresa com o pagamento dos tickets.

Obs.: Funcionários com salário até R\$900,00 recebem tickets de R\$15,00; os demais recebem tickets de R\$10,00. A quantidade de tickets a ser recebida por cada funcionário corresponde ao número de dias que ele trabalhou no mês.

- 4) Em 2008 João abriu uma poupança no Banco XPTO e depositou R\$ 80.000,00. Este investimento gera um rendimento de 8% ao ano. Faça um algoritmo que escreva em qual ano João acumulará 100% do valor investido (R\$ 160.000,00) sabendo que ele não fará nenhum depósito nem saque durante este tempo.

ESTUDO DE CASO I

1) Uma loja de esportes náuticos decidiu incentivar seu novo funcionário responsável pelas vendas de jetsky e pranchas de windsurf. Neste mês, ele receberá um salário base mais uma comissão sobre as vendas, calculada com base na tabela ao lado,

Tipo	Quantidade	Comissão (por unidade)
Windsurf	Qualquer	R\$20,00
Jetsky	< 3	R\$50,00
	≥ 3	R\$80,00

compondo seu salário total. Construa um algoritmo que solicite: o salário base e a quantidade vendida de cada um dos equipamentos e informe: o salário final do funcionário e o valor total vendido por ele. Para o cálculo, considere também que se o salário total ultrapassar R\$1.800,00 haverá um desconto de 12% para o Imposto de Renda. Além disso, se ele vendeu acima de 8 equipamentos de jetsky, informar na tela que será promovido. (Preço unitário: Jetsky = R\$16.000,00 Windsurf = R\$4.000,00)

2) Uma empresa especializada em soluções para automação de pedágios possui um sistema que foi desenvolvido há muito tempo e apresenta diversos problemas. Neste sistema o operador deve informar a distância que é mantida pelo pedágio (em centímetros) e imprimir o recibo entregue aos motoristas. O problema é que a tabela de distância, fornecida aos operadores, está em centímetro e o sistema também foi desenvolvido desta forma. Os motoristas reclamam bastante e sugerem que a distância seja impressa em KM.

Você foi contratado para desenvolver um novo algoritmo onde o operador informe a distância em centímetros e imprima no formato: **Km, m e cm**. Ex: 10080090 cm = 100 **Km**, 800 **m** e 90 **cm**. Saiba que: 1 Km = 100000 cm e 1 m = 100 cm

ESTUDO DE CASO II

1) Você foi contratado por uma Universidade para desenvolver um sistema que registre: o nome, as 3 notas e a média e cada um dos 300 alunos de um dos cursos tecnológicos.

a) O sistema deve armazenar os dados em um **vetor** (de **nomes**) e uma **matriz** (de **notas** e **média**).

OBS: Observe que a média deve ser armazenado no mesmo vetor de notas.

b) Depois de armazenar as informações o sistema deve solicitar o **nome** de um aluno e buscar no vetor. Caso o aluno seja encontrado deve-se informar se o mesmo foi **aprovado**, levando em consideração que para ser aprovado a **média** deve ser maior ou igual a 7.

c) O sistema deve informar também qual foi o nome do aluno que teve a maior e a menor média.

2) Você foi contratado para desenvolver um sistema médico de cadastro e consulta por especialidade. O programa possui o seguinte menu:

MENU

=====

1 – Cadastro
2 – Consulta
3 – Sair

- Caso o usuário selecione a opção 1 o sistema deve solicitar o nome e a especialidade de um médico e cadastrar estas informações em 2 vetores.
- Na opção 2 o sistema solicita ao usuário para digitar uma especialidade. Logo após o programa deve consultar e exibir os nomes dos médicos que possuem esta especialidade. Também deve ser informado a quantidade de médicos existente e encontrados.
- A cada opção selecionada o programa deve voltar para o menu e só deve ser encerrado se o usuário selecionar a opção 3.


ESTUDO DE CASO III

- 1) Faça um programa que solicite ao usuário números inteiros e positivos para preencher uma matriz de 4x3. Após preencher a matriz o programa deve:
 - a) Informar o valor da soma dos números armazenados na matriz;
 - b) Informar a média dos valores armazenados na matriz
 - c) Imprimir os números pares da matriz na ordem contrária ao que foi armazenado.
- 2) Faça um programa que solicite o **nome** e as **3 notas** dos 100 alunos de uma turma. O nome dos alunos devem ser armazenados em um **vetor** de **String** e as notas em uma **matriz** de **double**. Agora crie uma função que **receberá** o **vetor** com os **nomes** e a **matriz** com as **notas** e retornará um **String** com o nome do aluno com a média mais alta. O programa deve chamar esta função passando os nomes e as notas.
- 3) Faça um programa que solicite ao usuário 2 frases. Caso as frases sejam diferentes o programa deve: concatenar as frases em uma variável, imprimir a quantidade de palavras e a frase concatenada de trás para frente. Caso as frases sejam iguais o programa deve imprimir a frase digitada.
- 4) Faça uma função que receba uma matriz de inteiros e retorne a soma de seus elementos.
- 5) Construir um programa para manipular um vetor de inteiros com tamanho definido pelo usuário (o usuário vai informar o tamanho do vetor). Após a criação do vetor o programa deverá solicitar ao usuário números inteiros e positivos para preencher o vetor. Após preencher o vetor:
 - a) Calcular a média dos valores e informar quantos estão abaixo desta média.
 - b) Solicitar 1 número ao usuário e informar quais as posições(índices) com conteúdo igual ao número.

ESTUDO DE CASO IV

1) Faça um programa que solicite ao usuário 2 números que vão definir o tamanho de uma matriz de inteiros. Após isto solicite ao usuário para preencher esta matriz. Após preencher a matriz com os dados fornecidos pelo usuário o programa deve ordenar, de forma crescente, cada uma das linhas da matriz. Ex:

4	7	1	6
1	5	2	3
5	7	2	8



1	4	6	7
1	2	3	5
2	5	7	8

2) Faça um programa que solicite o **nome** e as **3 notas** dos 100 alunos de uma turma. O nome dos alunos devem ser armazenados em um **vetor** de **String** e as notas em uma **matriz** de **double**. Agora crie uma função que **receberá** o **vetor** com os **nomes** e a **matriz** com as **notas** e retornará um **String** com o nome do aluno com a média mais alta. O programa deve chamar esta função passando os nomes e as notas.

ESTUDO DE CASO V

1) Faça um algoritmo onde o usuário informa um numero e o programa imprime os divisores **"PAR"** e a **quantidade** de divisores **"IMPAR"** deste numero.

2) Faça um programa no qual o usuário vai entrando sucessivamente com valores positivos entre 1 e 100 (O usuário vai digitar valores válido). **Quando o usuário entrar com o valor 0 o programa para e informa:**

A média dos valores já fornecidos;

O maior e o menor número digitado;

A quantidade de números **"PAR"** que é divisor de 3.

A quantidade de números **"IMPAR"** que não é divisor de 3.

3) A Unijorge deseja fazer um sistema para registrar o nome e a idade dos 100 alunos que ingressaram no curso de Desenvolvimento de Software. O Sistema deve solicitar o **nome** e a **idade** dos **100 alunos** e armazenar estas informações em **2 vetores: 1 vetor de caracter para o nome e 1 vetor de inteiro para a idade**.

Observe que os vetores são diferentes mas um mesmo índice corresponde as informação de um mesmo aluno. Ex: O vetor **vet_nome** de **índice 3** registra o aluno **"Carlos"** o vetor **vet_idade** de mesmo **índice 3** registra a idade de **"Carlos"**.

vet_nome =	"Joao"	"Maria"	"Carlos"	"Sonia"	"Jose"	...
	1	2	3	4	5	
vet_idade =	20	18	32	23	19	...
	1	2	3	4	5	

Depois de preencher os vetores o sistema deve solicitar que o usuário informe uma idade qualquer. A partir desta informação imprima a média de idade das das pessoas que estão abaixo da idade informada e o nome das pessoas que estão acima ou possuem esta idade.

ESTUDO DE CASO VI

Faça um programa com as seguintes funções:

- 1) Abrir um arquivo texto e retornar o seu conteúdo como String.
- 2) Receber um String como parâmetro e dividir (delimitar) o texto por “<enter>” e retornar um Vetor de String com os valores. Ex:

Entrada: “1,2,3,4 5,6,7,8”	Saída: vet[0] = “1,2,3,4”; vet[1] = “5,6,7,8”;
----------------------------------	--

- 3) Receba um vetor de String como parâmetro e retorne uma Matriz do tipo double com os valores divididos (delimitados) por “,”. Ex:

Entrada: vet[0] = “1,2,3,4”; vet[1] = “5,6,7,8”;	Saída: mat[0][0] = 1; mat[0][1] = 2; mat[0][2] = 3; mat[0][3] = 4; mat[1][0] = 5; mat[1][1] = 6; ...
--	---

ESTUDO DE CASO VII

A empresa Megalife, que trabalha com representação de produtos nutricionais, contratou os serviços da Yoopa Informática pois precisa calcular o valor do 13º de seus funcionários. A empresa também deseja fazer uma análise do quanto foi gasto com os salários durante o ano. Cada vendedor recebe por mês um percentual em cima de suas vendas. O valor do 13º salário será a média recebida no ano. A empresa gravou no arquivo c:\dados.txt as informações de nome e salários recebidos no ano.

dados.txt

```
Joao,200,800,300,1000,700,500,200,800,300,700,1200,5000
Maria,100,300,400,500,100,700,300,100,3000,1700,1200,6000
Carlos,400,500,100,100,200,300,800,1000,3000,7000,200,4000
...
```

Cada linha do arquivo representa a informação de um funcionário e as virgulas de cada linha o valor recebido por mês. Ex:

Linha1	Nome	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
	Joao	200	800	300	1000	700	500	200	800	300	700	1200	5000

Linha2	Nome	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
	Maria	100	300	400	500	100	700	300	100	3000	1700	1200	6000

Linha3	Nome	Jan	Fev	Mar	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
	Carlos	400	500	100	100	200	300	800	1000	3000	7000	200	4000

Após uma análise feita pela Yoopa Informática foi constatado que para construir este programa é preciso criar um arquivo de funções chamado: Funcoes.java. Neste arquivo deve ser implementado as seguintes funções:

- Função que abre um arquivo e retorna o seu conteúdo;
- Função para retornar um vetor de String com os nomes dos funcionários;
- Função para retornar uma matriz de double com os salários dos funcionários;
- Função que recebe os 12 salários de um funcionário e retorna o valor do seu 13º salário;
- Função que recebe os salários de todos os funcionários e informa o total gasto durante o ano (Não esqueça de levar em consideração o 13º salário);
- Função que retorna o nome do funcionário que recebeu o maior salário.

Também deve ser feito o programa principal que deve se chamar Vendas.java. É ele que utiliza as funções do arquivo Funcoes.java.

A Yoopa Informática contratou um programador muito esperto que iniciou o trabalho, mas recebeu uma proposta do Google e resolveu deixar o Brasil. Você foi contratado para finaliza o programa.

Em anexo está o código que foi produzido pelo programador.

Arquivo Vendas.java

```
=====
public class Vendas {
    public static void main(String[] args) throws Exception {
        String[] funcionario = Funcoes.getVendedores("c:/dados.txt");
        double[][] salarios = Funcoes.getSalarios("c:/dados.txt");
        for(int x=0;x<salarios.length;x++){
            System.out.println("O funcionario "+funcionario[x]+" recebeu um 13o de "+Funcoes.get13Sal(salarios[x]));
        }
        System.out.println("O total gasto em salários no ano é"+Funcoes.getTotalAno(salarios));
        System.out.println("O funcionário que recebeu o maior salario foi: "+Funcoes.nomeSalarioMaior(funcionario, salarios));
    }
}
```

Arquivo Funcoes.java

```
=====
public class Funcoes {

    //a) Função que abre um arquivo e retorna o seu conteúdo
    public static String abrir(String arquivo) throws Exception{
        FileReader in = new FileReader(arquivo);
        String texto="";
        int i;
        while((i=in.read())!=-1){
            texto = texto + (char)i;
        }
        return texto;
    }

    //b) Função para retornar um vetor de String com os nomes dos funcionários
    public static String[] getVendedores(String arquivo) throws Exception{
        String info = abrir(arquivo);//Chama a função abrir (acima) para abrir o arquivo
        String[] dados = info.split("\n");//Separa as informações de cada funcionário
        String vendedores[] = new String[dados.length];//Vetor que será preenchido com os nomes dos vendedores
        for(int x=0;x<dados.length;x++){
            String[] dados_funcionario = dados[x].split(",");//Pega a informação do funcionário
            vendedores[x]=dados_funcionario[0];//Retira do indice 0 somente o nome e desconsidera os salários
        }
        return vendedores;
    }

    //c) Função para retornar uma matriz de double com os salários dos funcionários
    public static double[][] getSalarios(String arquivo) throws Exception{}

    //d) Função que recebe os 12 salários de um funcionário e retorna o valor do seu 13o.
    public static double get13Sal(double[] sal){}

    //e) Função que recebe os salários de todos os funcionários e informa o total gasto durante o ano (Não esqueça de levar em consideração o 13o)
    public static double getTotalAno(double[][] salario){}

    //f) Função que retorna o nome do funcionário que recebeu o maior salário
    public static String nomeSalarioMaior(String[] nomes, double[][] salarios){}

}
```

