Mas a vereda dos justos é como a luz da aurora, que vai brilhando mais e mais até ser dia perfeito.
Provérbios 4:18

PROGRAMAÇÃO ORIENTADA A OBJETO

- Prof. José Antonio Gonçalves

Nestes slides

Características básicas da tecnologia Java;

Aplicação das estruturas clássicas de programação:

- laços;
- seletores;
- variáveis;
- operações matemáticas;
- operações com cadeia de caracteres (strings);
- operações com datas.

Características básicas da tecnologia Java

Características básicas da tecnologia Java

Conhecendo a Plataforma Java

Sabemos que a Orientação a Objetos é uma maneira, mais natural e simples, de se pensar, elaborar, documentar e construir aplicações e sistemas de informação.

Baseada nos conceitos de **abstração, atributos, classes, objetos**, etc. temos um novo paradigma para construção de aplicações e a tecnologia **Java** nos permite implementar todos estes conceitos, pois a linguagem é totalmente aderente às técnicas de Orientação a Objetos.

Características básicas da tecnologia Java

API's (interface de programação de aplicativos) para diferentes plataformas

Java 2 Standard Edition (J2SE): ferramentas e APIs essenciais para qualquer aplicação Java (inclusive GUI).

Java 2 Enterprise Edition (J2EE): ferramentas e APIs para o desenvolvimento de aplicações distribuídas.

Java 2 Micro Edition (J2ME): ferramentas e APIs para o desenvolvimento de aplicações para aparelhos portáteis.

Características básicas da tecnologia Java

Tipos de programas Java

Stand-Alone: Aplicação baseada na J2SE, que tem total acesso aos recursos do sistema, memória, disco, rede, dispositivos, etc. Uma estação de trabalho pode executar uma aplicação de Automação Comercial.

Java Applets: Pequenas aplicações, que não têm acesso aos recursos do hardware e depende de um navegador que suporte a J2SE para serem executados, geralmente usados para jogos, animações, teclados virtuais, etc.

Java Servlets: Programas escritos e preparados para serem executados dentro de servidores web baseados em J2EE. Geralmente usados para gerar conteúdo dinâmico de web sites.

Java Midlets: Pequenas aplicações, extremamente seguras e construídas para serem executadas dentro da J2ME, geralmente celulares, palm tops, controladores eletrônicos, computadores de bordo, etc.

Java Beans: Pequenas aplicações que seguem um padrão bem rígido de codificação e que têm o propósito de serem reaproveitados em qualquer tipo de programa Java, podendo ser chamados por aplicações Stand-Alone, Applets, Servlets e Midlets.

Características básicas da tecnologia Java

Lista de Palavras Reservadas

Nenhuma das palavras reservadas podem ser usadas com nome de objetos, classes, atributos ou métodos.

abstract	double	int	static
boolean	else	interface	super
break	extends	long	switch
byte	final	native	synchronized
case	finally	new	this
catch	float	null	throw
char	for	package	throws
class	goto*	private	transient
const*	if	protected	try
continue	implements	public	void
default	import	return	volatile
do	instanceof	short	while

* reservadas, entretanto não são usadas, pois não representam instruções da JRE;

Características básicas da tecnologia Java

Como comentado anteriormente, o Java usa as duas formas computacionais de execução de software, compilação e interpretação.

O uso das duas formas faz com que o processo de desenvolvimento do software se torne mais acelerado, pois linguagens somente compiladas, necessitam de instruções adicionais da plataforma operacional para serem executados, e as linguagens somente interpretadas geralmente são muito lentas. Por isso o Java num primeiro momento é compilado (transformar código fonte) em instruções da Java Virtual Machine (bytecode).

Como o processo de execução é divido em dois, temos a possibilidade de que o interpretador seja escrito para várias plataformas operacionais.

Ou seja, o mesmo bytecode Java não importando onde foi compilado, pode ser executado em qualquer JRE. Os fabricantes de hardware e sistemas operacionais, em conjunto com a JavaSoft, desenvolveram várias JREs para vários ambientes operacionais. Por isso, aplicações 100% Java são 100% portáveis.

Características básicas da tecnologia Java

Compilando...

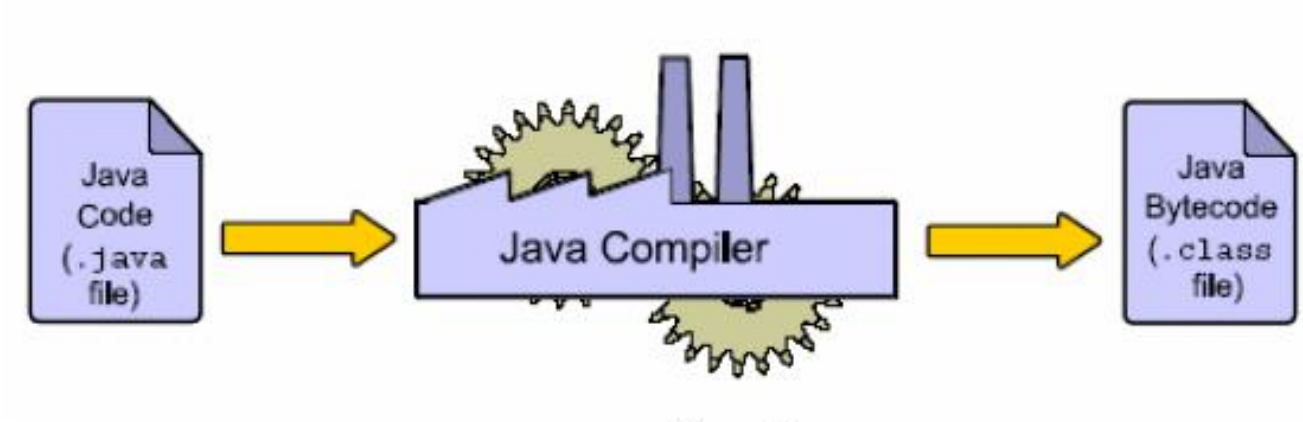


Figura 2

Características básicas da tecnologia Java

Executando...

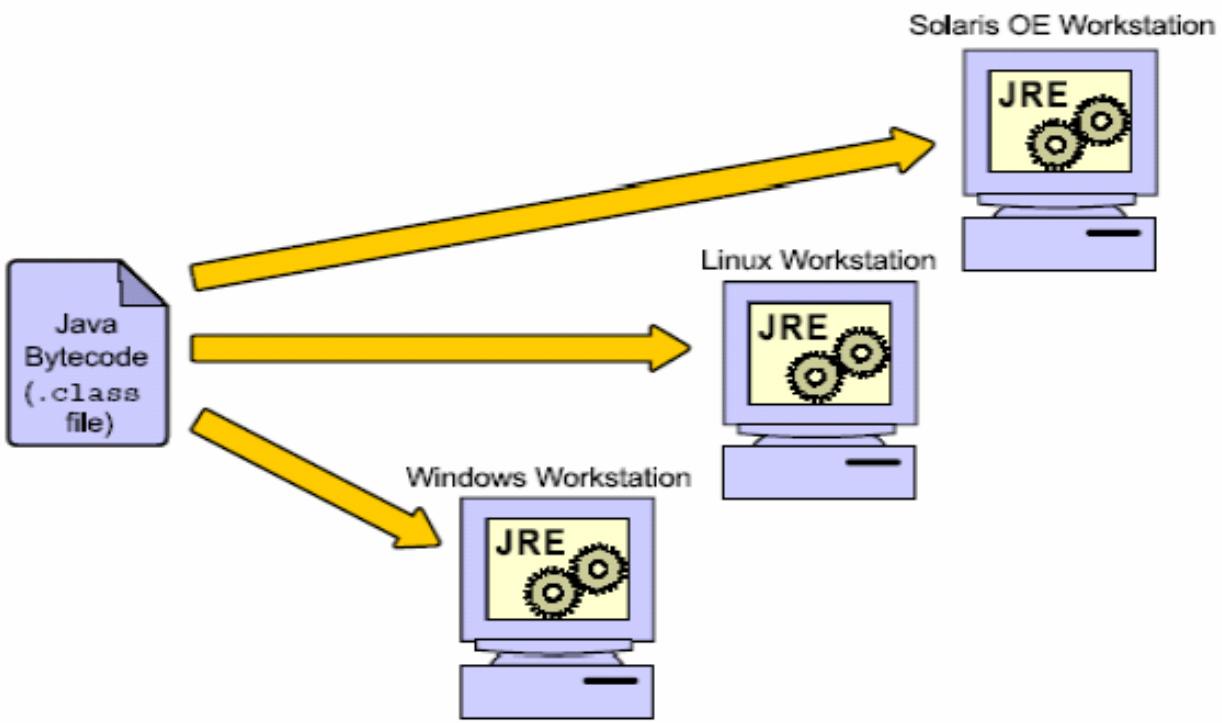
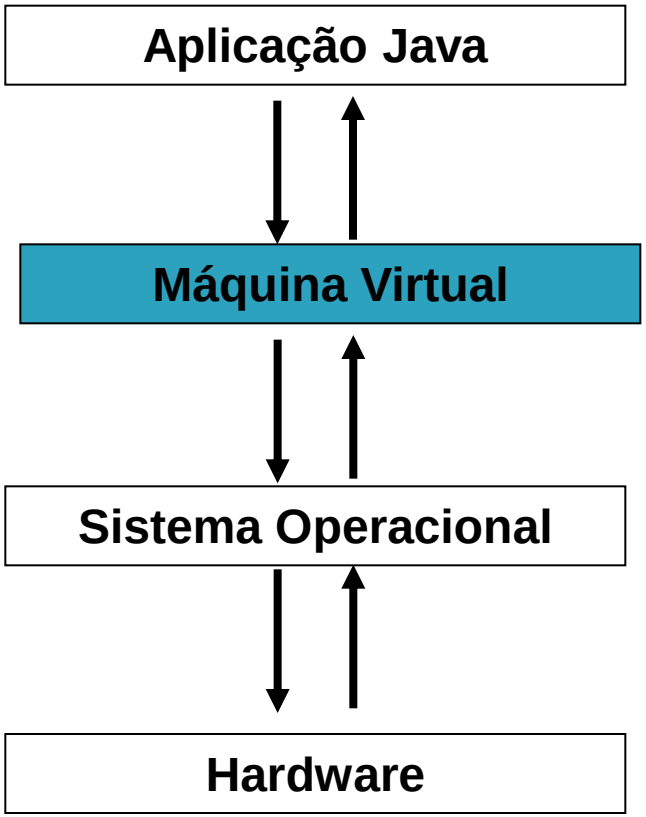


Figura 3

Características básicas da tecnologia Java

Independência de Plataforma



Características básicas da tecnologia Java

- Java reconhece letras maiúscula e minúscula (sensitive case)
- O nome do arquivo “.java” de ser o mesmo nome da classe (descrito internamente dentro do código)

Características básicas da tecnologia Java

Gerência de Memória

Garbage Collector:

- Aplicação associada a Máquina Virtual Java;

- Trata-se de um **Coletor de Lixo** que "**limpa**" da memória principal os objetos que não estão sendo mais usados. Isso acontece assim que eles perdem a **referência**;

- Este processo dinamiza ainda mais as aplicações Java.

Características básicas da tecnologia Java

Segurança

A Segurança em Java se dá em dois níveis:

- Proteção do Hardware (proteção da RAM);
- Proteção ao software (API´s).

Características básicas da tecnologia Java

Segurança: Proteção da RAM

Proteção do Hardware (proteção da RAM):

Pelo fato de Java não implementar “ponteiros”, garante a integridade no gerenciamento da memória principal. O que evita que inadvertidamente o “programador” aloque um espaço que já está sendo utilizado por outra aplicação.

Características básicas da tecnologia Java

Segurança: API's nativas ou não

Proteção ao software:

Grande quantidade de API´s (Interfaces para Programação de Aplicações). Estas API´s, fornecidas na "bibliotecas" nativas de Java, durante sua instalação foram **testadas inúmeras vezes, reduzindo assim a margem de erros durante a construção de uma aplicação**. Isso reforça também o fator de **Reusabilidade.**

Características básicas da tecnologia Java

Reusabilidade

Há a possibilidade de se reutilizar códigos (classes) que já "deram certo". Isso ocorre da mesma forma como utilizamos as classes nativas de Java.

Características básicas da tecnologia Java

Totalmente aderente aos conceitos da Orientação a Objetos

- A construção de aplicações com Java se dá totalmente através da criação de **classes** e declarações de **objetos** destas classes.

- A construção destas classes seguem os padrões, contendo:
  Atributos;
  Métodos.

- E também a aplicação dos conceitos (quando necessários) de:
  encapsulamento;
  herança;
  polimorfismo.

- Natural mapeamento do "Projeto" para "Implementação"

Características básicas da tecnologia Java

Convenções

Convenções em:

http://www.oracle.com/technetwork/java/javase/documentation/codeconvtoc-136057.html

Acessado em: 14/06/2012 às 20h

Aplicação das estruturas clássicas de programação

Aplicação das estruturas clássicas de programação

Saída de dados (na tela)

```
import java.io.*;

public class Saida {
   public static void main(String arg[]){
         System.out.println("Facim, facim!!!");
   }
}
```

Exerc: Usando o modelo anterior:

a.1)_Acrescente mais uma linha de exibição no programa anterior.

a.2)_Crie um programa que imprima todos os seus dados cadastrais

a.3)_Crie um programa que imprima a frase "Meu nome é: " que através de parâmetros passados durante a chamada (interpretação: java ...) do programa, imprima na tela esta frase + o nome passado.

Aplicação das estruturas clássicas de programação

Declaração de Variáveis e Operadores Matemáticos

```
import java.io.*;

public class Operad {

   public static void main(String arg[]){
         System.out.println("Impressão de Resultado de Cálculos:");
         System.out.println(" ");
         int a=3,b=2;
         int c=a+b;
         System.out.println("O resultado da soma de A e B é:" + c);
   }
}
```

Exerc: Usando o modelo anterior:

b.1)_ Implemente as outras operações matemática "simples" a saber

Multiplicação (*), divisão (/) e subtração (-), sendo com primeiro termo da operação a variável "a" e o segundo termo a variável "b", imprimindo seu resultado a cada operação.

Aplicação das estruturas clássicas de programação

Laços (for)

```
Import java.io.*;
public class Laco1 {
   public static void main(String arg[]){
         int a=0;
         int b=10;
         for(a=0; a<=b; a++){
                  System.out.println("O Valor de A dentro do FOR é: "+a);
         }
   }
}
```

Exerc: Usando o modelo anterior:

c.1)_ Usando o laço “for”, construa um programa que ao invés de incrementar a variável “a” até que atinja o valor de da variável “b”, faça o inverso, decremente “b” até que achegue ao valor de “a”.

Aplicação das estruturas clássicas de programação

Laços (while)

```
Import java.io.*;
public class Laco2 {
   public static void main(String arg[]){
         int a=0,
         b=15;
         while(a<=b){
                  a++;
                  System.out.println("O Valor de A dentro do while é: "+a);
         }
   }
}
```

Exerc: Usando o modelo anterior:

d.1)_ Da mesma forma que implementou o exercício c.1, implemente o d.1 usando a estrutura de repetição “while”

Aplicação das estruturas clássicas de programação

Laços (do..while)

```
Import java.io.*;
public class Laco3 {
   public static void main(String arg[]){
         int a=0,
         b=15;
         do{
                  System.out.println("O valor de A dentro do DO é: "+a);
                  a++;
         }while(a<=b);
         System.out.println("");
   }
}
```

Exerc: Usando o modelo anterior:

e.1)_ Da mesma forma que implementou o exercício c.1, implemente o e.1 usando a estrutura de repetição “do..while”

Aplicação das estruturas clássicas de programação

Seleção (if..elseif..else)

```
Import java.io.*;
public class  Selec1 {
    public static void main(String arg[]){
            int a=0, b=20;
            for(a=0; a<=b; a++){
                        if(a==(b/2)){

                        System.out.println("A é igual a metade de B, logo A vale: "+a+" e B vale: "+b);

                        }
                        else if(a!=(b/2)){
                                System.out.println("A é diferente a metade de B, logo A vale: "+a+" e B vale: "+b);
                        }
            }
    }
}
```

Exerc: Usando o modelo anterior:

f.1)_ Construa um programa que leia 2 valores inteiros e diga se o 1° é: maior, igual ou menor que o 2° valor.

Aplicação das estruturas clássicas de programação

Seleção (ternário)

```
public class Tern{
   public static void main(String args[]){
       int a=5;
       int b=2;
       int c = (a>b ? a : b);
       System.out.println(c);
       // ou → System.out.println(a>b ? a : b );
   }
}
```

Aplicação das estruturas clássicas de programação

Seleção (switch..case):

```
Import java.io.*;

public class Selec2 {
    public static void main(String arg[]){
            int valor = 1;
        switch(valor){
          case 0:
            System.out.println("Primeira Opção (Valor igual a zero) "+valor);
            break;
          case 1:
            System.out.println("Segunda Opção (Valor igual a um) "+valor);
            break;
          default:
            System.out.println("Outras Opções (Valor maior que um) "+valor);
            break;
        }
    }
}
```

Aplicação das estruturas clássicas de programação

Operadores Lógicos “e” e“ou” (&& e ||)

```
Import java.io.*;
public class Logic {
    public static void main(String arg[]){
        int a=0, c=0, valor=15;
        float b=15;
        for(a=0; a<=b; a++){
            valor--;
            c++;
            if(c<=(b/2)&& c<=valor){
                System.out.println("C eh MENOR que a metade de B 'E' MENOR IGUAL a VALOR");
                System.out.println("C vale.....: "+c);
                System.out.println("B / 2 vale.....: "+(b/2));
                System.out.println("VALOR  vale..: "+valor);
            }else if (c>=(b/2)|| (c>=valor)){
                System.out.println("C eh MAIOR que a metade de B 'OU' MAIOR IGUAL a VALOR");
                System.out.println("C vale.....: "+c);
                System.out.println("B / 2 vale.....: "+(b/2));
                System.out.println("VALOR  vale..: "+valor);


            }
        }
    }
}
```

Aplicação das estruturas clássicas de programação

Entrada de dados via console

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class  EntDados {
    public static void main(String arg[]){
            InputStreamReader c = new InputStreamReader(System.in);
            BufferedReader cd = new BufferedReader(c);
            String s = "";  String t = "";
            System.out.println("Entre com o 1º valor: ");
            try{
                    s = cd.readLine();
            }
            catch(IOException e){
                    System.out.println("Erro de entrada");
            }
            System.out.println("Entre com o 2º valor: ");
            try{
                    t = cd.readLine();
            }
            catch(IOException e){
                    System.out.println("Erro de entrada");
            }
            System.out.println("A soma dos dois valores eh : "+(s+t));
    }
}
```

Aplicação das estruturas clássicas de programação

Entrada de dados via console (convertendo "tipo"):

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class ConvertNum {
    public static void main(String arg[]){
            System.out.println("Entre com um valor: ");
            InputStreamReader c = new InputStreamReader(System.in);
            BufferedReader cd = new BufferedReader(c);
            String s = "";
            try{
                    s = cd.readLine();
            }
            catch(IOException e){
                    System.out.println("Erro de entrada");
            }
            int w = Integer.parseInt(s);
            System.out.println("O valor de entradada foi: "+w);
    }
}
```

Aplicação das estruturas clássicas de programação

Exercícios:

G1)_ Construa um programa que:

a)_ Permita o usuário entrar com 2 valores (que sejam armazenados em duas variáveis, por exemplo, “a” e “b”);

b)_ Apresente “menu de opções” com 4 opções:

Exemplo:

Menu de Opções:

1 – Somar (a+b)

2 – multiplicar (a*b)

3 – subtrair (a-b)

5 – dividir (a/b)

Escolha uma opção:

c)_ Após o usuário escolher uma opção, apareça uma linha informando o resultado do cálculo que foi escolhido:

Exemplo (sendo escolhida a 1ª opção e “a” valendo 2 e “b” valendo 1):

“A soma de a + b é: 3”

Para tanto use as estruturas de entrada de dados via console, conversão de tipos e a estrutura de seleção switch..case.

Aplicação das estruturas clássicas de programação

Alguns métodos da Classe Math.

```
public class Maths1 {
        public static void main(String arg[]){

                double f=2;
                double g=1.34;
                double h=9;

                System.out.println("\nO seno da var 'f' eh: " + Math.sin(f));
                System.out.println("\nO cosseno da var 'f' eh: " + Math.cos(f));
                System.out.println("\nA tangente da var 'f' eh: " + Math.tan(f));
                System.out.println("\nO valor mínimo entre as vars 'f' 'g' eh: " + Math.min(f,g));
                System.out.println("\nO valor máximo entre as vars 'f' 'g' eh: " + Math.max(f,g));
                System.out.println("\nO valor arredondado da var 'g'(1.434) eh: " + Math.round(g));
                System.out.println("\nO valor arredondado para cima da var 'g'(1.434) eh: " + Math.ceil(g));
                System.out.println("\nO valor arredondado para baixo da var 'g'(1.434) eh: " + Math.floor(g));

                System.out.println("\nA raiz da var 'h' eh: " + Math.sqrt(h));
                System.out.println("\nA var 'f' elevada a var 'h' eh: " + Math.pow(f,h));
                System.out.println("\nO valor absoluto da var 'g'(1.434) eh: " + Math.abs(g));
                System.out.println("\nO valor do 'PI' eh: " + Math.PI);
                System.out.println("\nGerar um valor aleatório ente 0  e 1: " + Math.random());

                System.out.println("\n\nA próxima linha não faz parte da classe Math, trata-se do operador 'Módulo'");
                System.out.println("\nO resto da divisão entre as vars 'h' e 'f' eh: "+ (f%h));
        }
}
```

Aplicação das estruturas clássicas de programação

Exercícios

h1)_ Utilizando a Linguagem de programação Java, implemente um programa que permita a entrada de um valor inteiro e informe se este valor é “par” ou “ímpar”.

Aplicação das estruturas clássicas de programação

Capturando a "data" do Sistema

```
import java.util.GregorianCalendar;

public class Data{
    public static void main(String arg[]){

            GregorianCalendar data = new GregorianCalendar(); //trabalhando com datas

            int dia = data.get(data.DAY_OF_MONTH);
            int mes = data.get(data.MONTH)+1;
            int ano = data.get(data.YEAR);
            System.out.println("\nDATA ATUAL DO SISTEMA: "+dia+"/"+mes+"/"+ano+"\n\n");
            System.out.println("\nDATA ATUAL DO SISTEMA:
    "+data.get(data.DAY_OF_MONTH)+
    "/"+data.get(data.MONTH)+"/"+data.get(data.YEAR)+"\n\n");
    }
}
```

Aplicação das estruturas clássicas de programação

Alguns métodos da Classe String

```
public class Stringe {
        public static void main(String arg[]){
                String frase="Eis-me aqui SENHOR, envia-me a miM";
                System.out.println("\t\t\t\t Eis-me aqui SENHOR, envia-me a mim");
                System.out.println("\n\n A frase anterior tem "+frase.length()+" caracteres");
                System.out.println("\n\n Transformando-a toda em MAIUSCULA:");
                System.out.print(frase.toUpperCase());
                System.out.println("\n\n TRANSFORMANDO-A TODA EM minuscula:");
                System.out.print(frase.toLowerCase());
                System.out.println("\n\n Procurando a letra da 10 posição:");
                System.out.print(frase.charAt(13));
                System.out.println("\n\n Retornando a substring dentro da frase:");
                System.out.println(frase.substring(0,11));
                System.out.println("\n\n 1ª Vez que encontrou a letra 'O' na frase:");
                System.out.println(frase.indexOf('O'));
                System.out.println("\n\n Tirando os espaços em branco:");
                System.out.println(frase.trim());
                System.out.println("\n\n Trocando as letras 'm' por 'M':");
                System.out.println(frase.replace('m','M'));

        }
}
```

Aplicação das estruturas clássicas de programação

Array unidimensional (vetor) – Entrada automática.

```
public class Vet1{
        public static void main(String arg[]){
                int larg=30;
                int vtr[]= new int[larg];
                System.out.println("O tamanho do vetor eh: "+vtr.length);
                int cont=0;
                for(int i=0;i<vtr.length;i++){
                        cont=cont+2;
                        vtr[i]=cont;
                }
                for(int i=0;i<vtr.length;i++){
                        System.out.println("Vetor no local "+i+" tem valor "+vtr[i
                }
        }
}
```

Aplicação das estruturas clássicas de programação

Array Bidimensional (Matriz) – Entrada automática.

```
public class Mat1 {
    public static void main(String arg[]){
        int lin=5, col=5, cont=1, i = 0, c = 0;
        int mtrz[][]=new int[lin][col];
        System.out.println(mtrz.length);    //tamanho da matriz
        for(i =0; i<lin; i++){
            for(c=0; c<col; c++){
                mtrz[l][c]=cont++;
             }
         }

        for(i =0; i<lin; i++){
            for(c=0; c<col; c++){
                System.out.println("Val. da "+ i +"linha eh "+ c + "coluna eh: "+ mtrz[i][c]);
            }
         }
    }
}
```

Aplicação das estruturas clássicas de programação

Exercícios (sempre utilizando a linguagem Java):

Strings

1)_ Construa um programa que permita ao usuário entrar com determinada frase, depois permita "escolher" uma letra qualquer e: caso a letra escolhida esteja na frase (seja maiúscula ou minúscula) diga quantas vezes ela apareceu e em que posição da frase. Senão, apareça uma frase informando que esta letra não existe na frase.

Matrizes

1)_ Crie um programa, no qual terá um vetor de inteiros, cujo tamanho será definido pelo valor de uma variável local, que permita ao usuário entrar com os valores. Depois, estes valores serão apresentados na ordem inversa à da entrada.

2)_ Faça o mesmo procedimento do exercício anterior, porém desta vez estará usando uma matriz bidimensional