JAVA – PARTE 2

Profº Wellington Moreira de Oliveira

ESTRUTURA CONDICIONAL

- if (condição)
- (condição)?(expressão1):(expressão2)
- switch (valor) {
  case valor1: instruções;
- break;
- default:
- break;
- }
- Problema do "else oscilante". Está ligado a qual if num aninhamento?
- goto (palavra reservada não utilizada)

ESTRUTURA DE REPETIÇÃO

- while (condição) { … }
- do { … } while(condição)
- for (inicialização;condição;incremento) { … }
- Flag ou sentinela
- break
- continue

```
class ForArray {
        public static void main(String[] args){
        int[] numeros = {1,2,3,4,5,6,7,8,9,10};
        for (int item : numeros) {
        System.out.println("Valor: " + item);
        }
      }
}
```

CONVERSÃO EXPLÍCITA E IMPLÍCITA

```
int x;
    double y;
    y = (double) x;

int num1;
    double num2;
    double total;
    total = num1 + num2;
```

MODIFICADORES DE ACESSO

- Aplicável para classes, atributos e métodos.
- public
- private
- protected

CLASSES

- class NomeClasse
- Padrão **P**ascal**C**ase
- Inicia-se sempre com maiúscula (obrigatoriamente)
- Repetir mesmo nome no arquivo: NomeClasse.java

MÉTODOS

- public void imprimirTexto(String texto)
- Padrão **c**amel**C**ase
- Não existe método sem classe
- Chamada ao método:
  - Separador Ponto: objeto.nomeMetodo()
  - N° de argumentos = n° de parâmetros
  - Regra geral: argumentos e parâmetros de mesmo tipo (nem sempre).

MÉTODO CONSTRUTOR

- Chamada automática com new
- Pode ser modificada
- Tem que ter o mesmo nome da classe (**único método que inicia com letra maiúscula**)
- Não tem retorno
- Pode ser utilizado para inicializar atributos da classe
  - Ex.:

```
public class Aluno {
    String nome;
    public Aluno(String nome){
        this.nome = nome;
    }
```

MÉTODO MAIN

- A classe que o contém é chamada de **aplicativo**
- public static void main (String[] args)
- É estático, ou seja, não preciso instanciar a classe para chamar o método
- Não retorna valor
- Permite somente um parâmetro
- Argumentos passado por linha de comando
- Benefício: alterar o comportamento do sistema sem modificar o código

MÉTODO PRINT

- Pertence à classe System do pacote java.lang
- System.out.print()
- System.out.println()
- System.out.printf ("%d\n%s\n%c\n%.2f", 26, "Alô", 'c', 47.949498);

ATRIBUTOS

- int valorTotal = 10;
- Padrão **c**amel**C**ase
- Variáveis Locais: escopo limitado pelo método que a contém
- Variáveis Estáticas (static): pertence ao escopo da classe e é acessível por todas instâncias da classe
- Variável de Instância: pertence ao escopo da classe. Cada instância da classe tem sua "instância" da variável
- Controle de Acesso (Encapsulamento)
- A localização da declaração não é pré-determinada
- Escolher bem do nome do identificador

DICAS

- Uma instrução por linha
- Recuo de 3 espaços (usar tabulação)
- Espaço antes de depois de operador
- Espaço depois de vírgula
- Comentar fim de bloco
- Nem sempre a linha mostrada contém o erro

INSTANCIANDO OBJETOS

- Criar uma classe que contenha um método
- Criar uma classe principal com o método main
- Criar um atributo do mesmo tipo da classe que será instanciada
- Instanciar o objeto usando a palavra reservada “new”
- Ex.: Aluno aluno;
- aluno = new Aluno();

GET E SET

- Troca de Mensagens entre objetos
- Para servir “clientes de um objeto”
- Atributos Privados x Métodos Públicos
- Garante o Encapsulamento

GET E SET - IMPLEMENTAÇÃO

- Set: armazenar – void – parâmetro

Ex.: public void setNome(String nome){

...

this.nome = nome;

...

}

- Get: obter – return

Ex.: public String getNome() {

...

return nome;

}

CLASSE MATH

- Faz parte do pacote java.lang

- Math.max(num1, num2...)
- Math.min(num1, num2...)
- Math.sqrt(numero)
- Math.cbrt(numero)
- Math.pow(num, exp)
- Math.log(num)
- Math.random()
- Math.round(decimal)
- Math.PI
  - Exercício 2.28
  - http://download.oracle.com/javase/1.4.2/docs/api/java/lang/Math.html

CLASSE STRING

- Faz parte do pacote java.lang

- length()
- trim(valor)
- equals(valor)
- compareTo(valor)
- compareToIgnoreCase(valor)
- toLowerCase(valor)
- toUpperCase(valor)
- endsWith(valor)
- startsWith(valor)

CLASSE STRING

- Faz parte do pacote java.lang

- valueOf(valor) – converte para String
- toCharArray(valor)
- Split(valor)
- indexOf(char)
- lastIndexOf(char)
- charAt(index)
- contains(valor)
- substring(inicio, fim)
- concat(valor1, valor2)
- replacement(valor1,valor2)

EXCEÇÕES

- throws: especifica as exceções que um método pode lançar, "lançamento de exceção"
- try: teste, "tente"
- catch: captura e trata a exceção. "Handler de Exceção"
- finally: executado sempre
- Exemplos de Exceções: ArithmeticException, InputMismatchException
- Rastreamento de Pilha
- Após a 'captura' da exceção pelo bloco catch a execução do programa não retorna onde a exceção foi encontrada mas sim para a próxima linha após este bloco
- Evitar criar tratamento desnecessários ou em substituição à instrução de controle

HIERARQUIA DE EXCEÇÕES