Universidade Federal de Campina Grande – UFCG

Centro de Engenharia Elétrica e Informática – CEEI

Unidade Acadêmica de Sistemas e Computação – UASC

Disciplina: Laboratório de Programação 2

Laboratório 01

Neste laboratório iremos experimentar a **edição** e **compilação** de programas simples em Java (explorando estruturas básicas) e preparar o ambiente de programação para as próximas aulas.

Importante:

- Antes da definição de cada classe escreva o seu nome em comentário:
- **(/*Aluno: <seu nome>*/)**

Entendendo melhor a Linguagem Java

Na primeira parte da aula, os professores conduzirão os alunos na execução de alguns passos de programação a fim de explorar: uso de **package**, **saída de dados**, **compilação** e **execução**, **case-sensitive**, tipos básicos, estruturas de desvio e um pouco de operadores em Java.

Passo1: Criação do Arquivo .java (20 minutos)

Crie um programa em Java chamado **IdadePreferencial** para **imprimir** na saída padrão (monitor) **uma frase** indicando a sua idade e se você é uma pessoa que pode usar o atendimento preferencial (acima de 60 anos). Lembre-se de nomear seu arquivo corretamente, **IdadePreferencial**.java. Por enquanto, **não adicione package no seu programa**. (Você pode usar qualquer editor de texto de sua preferência)

Passo2: Compilação, Case-Sensitive e Criação de Diretórios (10 minutos)

Agora vamos compilar e executar o programa **IdadePreferencial.java**. Inicialmente, algumas **dicas para trabalhar no Linux**:

- Para criar um diretório use o comando mkdir e para passar de um diretório para outro use o comando cd. Veja exemplo:

```
$ mkdir nomeDoDiretorio
$ cd nomeDoDiretorio
nomeDoDiretorio$ mkdir ...
```

- Para ver o conteúdo de um diretório use o comando ls.
- Para mover um arquivo de um diretório para outro:

```
$ mv arquivo nomeDoDiretorioDestino
```

Abra um Terminal (ou console) no Linux, vá para o diretório onde o programa acima foi salvo (com o comando `cd`) e olhe o que tem dentro do diretório (com a comando **`ls`**). Apenas o código fonte (IdadePreferencial.java) estará lá. Agora tente compilá-lo usando o comando *javac:* ***`javac IdadePreferencial.java`***. Se seu programa não tiver erros de compilação, você vai conseguir compilá-lo e o arquivo de *bytecodes* (**IdadePreferencial.class**) vai ser gerado. Use novamente o comando ls para se certificar que o arquivo IdadePreferencial.class existe.

IdadePreferencial.class é o arquivo que contém os *bytecodes* do programa IdadePreferencial.java. Agora, a Java Virtual Machine (JVM) pode executar o seu programa interpretando blocos de comandos para a linguagem de máquina. Tente executar o seu programa, utilizando o comando: `java IdadePreferencial`.

>> experimente mudar o nome da classe colocando todas as letras em minúsculo, sem modificar o nome do respectivo arquivo para perceber a conseqüência de Java ser **case-sensitive**.

Passo3: Criação de Pacotes e Classpath (20 minutos)

Crie o programa **IdadePreferencialInteligente.java** no **package lp2.lab01**, que tem a mesma funcionalidade do programa implementado no Passo1. Vale lembrar que agora o bytecode gerado terá um nome completo, que inicia com o nome do pacote e tem que estar numa árvore de diretórios idêntica ao que foi definido no pacote (lp2/lab01/IdadePreferencialInteligente.class).

árvore de diretórios:

>exercicio

>lp2

>lab01

Para simplificar o uso de pacotes, crie a árvore de diretório **lp2/lab01** a partir do diretório corrente. Salve o programa IdadePreferencialInteligente.java no diretório lp2/lab01. Vá para o diretório lp2/lab01 e compile novamente o programa. (Você pode tentar compilar sem ir ao diretório **lp2/lab01**, indicando apenas o caminho).

Vamos executar indicando o caminho da classe pelo seu pacote. Tente executar:

```
java lp2/lab01/IdadePreferencialInteligente
```

Note que não foi possível encontrar a classe IdadePreferencialInteligente. Para a JVM encontrar a classe navegando pelos diretório é necessário usar a notação de pacotes. Agora tente:

```
java lp2.lab01.IdadePreferencialInteligente
```

Lembre-se que pode ser necessário ensinar à Java Virtual Machine (JVM) em que diretórios ela deve procurar os seus programas (*classpath* ou `cp`). Nesse caso, a jvm deve executar o programa a partir do diretório pai do package. Assim para indicar o caminho a partir do qual a jvm vai procurar o pacote usando a opção –cp (`java –cp ../.. lp2.lab01.IdadePreferencialInteligente`).

Por exemplo, saia dos diretórios do seu projeto java até o diretório pai de 'exercicio'. Agora tente executar sua classe usando apenas:

```
java lp2.lab01.IdadePreferencialInteligente
```

ou então:

```
java exercicio.lp2.lab01.IdadePreferencialInteligente
```

Note que a JVM não consegue encontrar a classe pois não sabe qual o diretório do projeto. Vamos usar o classpath para isso. Tente executar, ainda no diretório atual o seguinte comando:

```
java -cp ./exercicio lp2.lab01.IdadePreferencialInteligente
```

Agora sim! A JVM sabe que existe uma estrutura de pacotes Java no caminho "./exercicio".

>> experimente compilar e executar a partir de diretórios diferentes para exercitar mais a opção –cp

Passo4: Definição de Variáveis (10 minutos)

Vamos modificar o programa IdadePreferencialInteligente.java para que ele **calcule sua idade a partir do ano atual e do ano de nascimento**. Nesse caso, você vai precisar criar uma **variável** para guardar o valor da idade que será resultado da diferença entre dois literais inteiros, um representando o ano atual e, o outro, o ano de nascimento.

Java é uma linguagem **estaticamente e fortemente tipada**, assim, todas as variáveis precisam ter associado um tipo em tempo de compilação. Esta associação é feita da seguinte forma:

<nome_do_tipo> <nome_da_variável>

Ex: `int x;`

No caso da idade o tipo a ser usado será inteiro (int), mas Java suporta vários outros tipos primitivos como mostrado no material disponível aqui.

Para calcular a idade será necessário usar o operador de **subtração** de Java (-). A linguagem também dispõe de outros operadores aritméticos, tais como: soma (+), multiplicação (*) e divisão (/). Ver mais informações aqui.

Agora para mostrar o valor da variável que representa a idade será necessário usar o operador de **concatenação de Java** (+). Note que é o mesmo operador da soma aritmética, assim, este operador é dito "**sobrecarregado**", pois possui dois casos de uso. Então, a concatenação ocorre da seguinte forma:

```
int x = 10;
System.out.println("texto 1" + x);
```

Nesse caso, se um dos operandos for textual, o operador + fará concatenação; se os dois operandos forem numérico, o operador + fará soma aritmética.

Faça as modificações necessárias no programa, compile-o e execute-o novamente.

Passo5: Comando de Desvio (20 minutos)

Vamos modificar o programa IdadePreferencialInteligente.java para introduzir a checagem da idade e, assim, verificar se a pessoa está apta a usar um caixa preferencial. Para tal, usaremos as **estruturas de desvio de Java**.

Um comando de desvio define uma condição em um programa, que permite que grupos de comandos sejam executados de maneira condicional, de acordo com o resultado da avaliação de um determinado teste (verdadeiro ou falso). Ou seja, programas utilizam comandos de desvio para escolher entre cursos alternativos de ações. A sintaxe do comando de seleção em Java é:

```
if (expressão){
   comando1;
}else{
   comando2;
}
```

O `comando1`, será executado se o resultado da avaliação de expressão for verdadeiro (digamos `x > 3`). Caso contrário `(else)` o `comando2` será executado. Comentários:

- Vários comandos podem ser executados na cláusula if ou na cláusula `else`, bastando para isso que sejam usados os **delimitadores de escopo { }** (abre chaves e fecha chaves).
- Comandos de seleção podem existir dentro de outros comandos de seleção (na cláusula `if` ou `else`).

A `expressão` avaliada é uma expressão condicional, podendo conter operadores relacionais (ex. >, maior; < menor) ou lógicos (ex. &&, and; ||, or). Mais operadores aqui.

Faça as modificações necessárias no programa, compile-o e execute-o novamente.

Passo6: Comando de Desvio Aninhado e Outros Tipos de Variáveis (30 minutos)

Observe que a mensagem apresentada pelo programa IdadePreferencialInteligente.java não contempla todos os casos de atendimento preferencial com exatidão, pois mulheres grávidas ou pessoas com pelo menos uma criança de colo, também tem preferência. Modifique seu programa para tratar esses casos, imprimindo qual o caso preferencial específico:

Acima de 60 anos: `"Preferencial Idoso."`

Grávida: `"Preferencial Gestante."`

Criança de colo: `"Preferencial pois está com x crianças de colo."`

(onde x é a quantidade de crianças)

Além de um novo tipo de variável (**boolean**), você vai precisar usar **comandos de desvio aninhados**. Comandos de desvios "dentro" de outros comandos de desvio. Em Java, podemos utilizar diferentes maneiras para escrever comandos aninhados. Abaixo, apresentamos a maneira mais tradicional e que permite um melhor entendimento do código com comandos de desvio aninhamos.

```
if(condicao1){
    comando1;
} else{
    if(condicao2){
        comando2;
    } else {
        comando3;
    }
}
```

OU

```
if(condicao1){
    comando1;
} else if(condicao2){
        comando2;
} else {
        comando3;
}
```

2ª Parte: Para exercitar mais

Escreva um programa em Java que imprime uma mensagem na saída padrão (monitor) indicando a qual quadrante do plano cartesiano pertence as coordenadas de um ponto cartesiano (x, y). A mensagem também deve incluir os valores das coordenadas x e y. Tais valores são definidos no seu próprio programa (não é preciso ler os valores do teclado). Note que os pontos sobre os eixos x e y (ex. 0,1 ; 10,0 ; -1,0 ; 0,-4 ; 0,0) não pertencem a nenhum quadrante; seu programa deve estar preparado para este caso também. Defina seu programa no pacote lp2.lab01.