Universidade Federal de Uberlândia
Faculdade de Computação - FACOM
FACOM32305 – Programação Orientada a Objetos
Turma 2024/1
Prof. Thiago Pirola Ribeiro

Laboratório 02 – Java básico

Crie um novo projeto no NetBeans para este exercício contendo uma classe principal para teste (sugestão: `Teste`).

No método `main()`, crie duas variáveis de cada tipo primitivo do Java: `byte`, `short`, `int`, `long`, `float`, `double`, `char` e `boolean`:

```
public static void main(String[] args) {
   java.util.Scanner sc = new java.util.Scanner(System.in);
   byte   byte1,   byte2;
   short  short1,  short2;
   int    int1,    int2;
   long   long1,   long2;
   float  float1,  float2;
   double double1, double2;
   char   char1,   char2;
   boolean A, B;
   ...
}
```

Adicione linhas para exibir cada uma das variáveis criadas usando o método `printf()` (imprime formatado), conforme descrito no exemplo do quadro ao início da próxima página.

Tente executar o código e veja que uma mensagem de erro é exibida dizendo que a variável pode não ter sido inicializada (repare que as variáveis estão sublinhadas pelo NetBeans). Algumas maneiras de se inicializar variáveis no Java são:

- Na própria declaração, adicionando um `=` e o valor inicial após o nome da variável;
- Antes da primeira utilização, atribuindo-se um valor qualquer com uma declaração do tipo `variável = <valor>;`
- Lendo um valor do console via um objeto `java.util.Scanner`, utilizando o método apropriado para cada tipo (`nextInt()`, `nextFloat()`, etc.);
- Lendo dados da interface visual usando as bibliotecas *Swing* ou *AWT*.

```
...
System.out.printf("Variaveis byte: %d e %d\n", byte1, byte2);
/*                "String           ^    ^  ", parâmetros
Máscara dos parâmetros
... Siga o exemplo para os demais tipos de acordo com a tabela:
Usando a mascara para formatar dados, sempre começando com %:
%d -- decimal inteiro          %u -- inteiro positivo
%f -- real formato xxx.yyyyyy  %E -- real científico x.yyyyyyE+zzz
%c -- caractere                %X -- %d em hexadecimal maiúsculo
%s -- String                   %x -- %d em hexadecimal minúsculo
%b -- booleano                 %o -- %d em octal

```

Tabela 1: Tipos de dados em Java

Tipo	Tam. (bits)	Valores	Valor padrão	Descrição
`boolean`	1	`false` ou `true`	`false`	
`char`	16	`'\u0000'` a `'\uFFFF'` (0 a 65535)	`'\u0000'`	(conj. caracteres Unicode ISO)
`byte`	8	-128 a 127 (-2^7 a $+2^7-1$)	0	
`short`	16	-32768 a $+32767$ (-2^{15} a $2^{15}-1$)	0	
`int`	32	$-2.147.483.648$ a $+2.147.483.647$ (-2^{31} a $2^{31}-1$)	0	
`long`	64	-2^{63} a $+2^{63}-1$	0	
`float`	32	$-3,4E{+}38$ a $-1,4E{-}45$; $+1,4E{-}45$ a $+3,4E{+}38$; $\pm$inf; NaN	0.0	ponto flutuante (IEEE754)
`double`	64	$-1,8E{+}308$ a $-4,9E{-}324$; $+4,9E{-}324$ a $+1,8E{+}308$; $\pm$inf; NaN	0.0	ponto flutuante (IEEE754)

```
Modificadores:
%[-][+][0-N][.0-9][lL][dxXuofeEgGcs]
||  |  |    |     |   |
||  |  |    |     |   +- formato (veja tabela acima)
||  |  |    |     +----- modificador long, ignorado
||  |  |    +----------- número de casas decimais após o .
||  |  +---------------- largura do campo (N caracteres)
||  +------------------- mostra '+' para números positivos
|+---------------------- alinha a esq.
+----------------------- inicia formatação
*/
```

Veja os exemplos abaixo e inicialize todas as variáveis com valores apropriados, apresentados na Tabela 1, em seguida tente executar novamente o programa.

```
short short1 = 100, short2 = 200; // na própria declaração
byte1 = -2; byte2 = 3 ;  // depois da declaração
float1 = sc.nextFloat(); // o programa aguardará entrada de dado
...
```

No mesmo programa, adicione ao final as seguintes instruções e observe os resultados ao tentar executar:

```
...
short1 = byte1; short2 = int2;
int1 = short1; int2= double2;
float1 = int1; float2 = double2;
System.out.printf("Variaveis short: %d e %d\n", short1, short2);
System.out.printf("Variaveis inteiras: %d e %d\n", int1, int2);
System.out.printf("Variaveis float: %f e %f\n", float1, float2);
```

O NetBeans avisa sobre uma possível perda de precisão, sublinhando as ocorrências.

Type casting permite alterar o tipo do resultado da operação subsequente, antes de atribuir a outra variável. Tente novamente fazendo uso do recurso *type casting*, como dado no exemplo:

```
short2 = (short)int2;
int2 = (int)double2;
float2 = (float)double2;
```

Execute as seguintes operações e exiba os resultados usando `printf()`.

```
int resultado1 = (int1+short1+byte1)*3;
System.out.printf("Resultado: %d\n", resultado1);
int resultado2 = (int)(double1+float1)%5;
System.out.printf("Resultado: %d\n", resultado2);
double resultado3 = (int1+short1+byte1)/9;
System.out.printf("Resultado: %f\n", resultado3);
```

Note que houve perda de precisão no resultado3, zerando a parte fracionária (`XX.00000`). Como corrigir o problema?

Utilize as duas variáveis boolianas, A e B, para verificar os operadores lógicos apresentados nas tabelas abaixo.

```
A = false, B= true; // Alterne os valores e verifique as tabelas
System.out.printf("%b && %b = %b\n", A, B, A && B); // A e B
System.out.printf("%b || %b = %b\n", A, B, A || B); // A ou B
System.out.printf("%b ^ %b = %b\n", A, B, A ^ B);   // A Xou B
System.out.printf("%b == %b = %b\n", A, B, A == B); // A = B ?
System.out.printf("%b != %b = %b\n", A, B, A != B); // A /= B ?
```

Finalmente vamos utilizar uma variável do tipo `String` para formatar a saída em uma tabela com largura de coluna fixa. Siga o exemplo abaixo para as demais linhas.

```
String tipo = "short";
System.out.printf("%7s: %6d %6d\n", tipo, short1, short2);
tipo = "int";
System.out.printf("%7s: %6d %6d\n", tipo, int1, int2);
tipo = "float";
System.out.printf("%7s: %6.1f %6.1f\n", tipo, float1, float2);
tipo = "double";
System.out.printf("%7s: %6.1f %6.1f\n", tipo, double1, double2);
tipo = "char";
System.out.printf("%7s: %6c %6c\n", tipo, char1, char2);
tipo = "boolean";
System.out.printf("%7s: %6b %6b\n", tipo, char1, char2);
```

Dados numéricos de diversos tipos podem adicionados a uma `String`, estes são automaticamente convertidos para texto usando o formato padrão, caso não especificado:

```
System.out.printf("\nVariáveis\n");
System.out.printf("char: %c e %c\n", char1, char2);
System.out.printf("int: %d e %d\n", int1, int2);
System.out.printf("float: %f e %f\n", float1, float2);
// Pode ser exibido alternativamente da forma
String string1 = "\nVariáveis\nchar: " + char1 + " e " + char2
               + "\nint: " + int1 + " e " + int2
               + "\nfloat: " + float1 + " e " + float2;
System.out.printf("String: %s\n", string1);
```

Para podermos definir o formato numérico a ser exibido na `String`, utilizamos o método `format()` da classe `String`, como segue:

```
String string2 = String.format("\nVariáveis" +
                               "\n%7s:%6c e %6c" +
                               "\n%7s:%6d e %6d" +
                               "\n%7s:%6.1f e %6.1f" +
                               "\n%7s:%6.1f e %6.1f",
                               "char", char1, char2,
                               "int", int1, int2,
                               "float", float1, float2,
                               "double", double1, double2 );
System.out.println(string2);
```

Exercícios

1. Utilizando seu programa, descreva o que acontece quando atribuímos valores fora dos limites apresentados na Tabela 1 para cada tipo de variável.

2. Qual a regra geral quando atribuímos o valor de uma variável de um tipo a outro? Quando devemos usar *Type cast*?

3. Ocorre arredondamento ao passarmos uma variável de precisão maior para uma de precisão menor, como de `double` para `int`?

4. Como resolver o problema de perda de precisão no resultado3?

5. O que acontece na operação `double trezQuartos = 3/4`? Por quê? Como corrigir?

6. Qual a diferença entre os operadores lógicos `^` (ou exclusivo) e `!=` (diferente) para variáveis boolianas?

7. A nota final de um aluno é composta por: duas provas com peso 3 (cada uma), um trabalho com peso 2 e duas listas de exercícios com peso 1 (cada uma). Escreva um programa para calcular a média final do aluno. Faça a entrada de dados utilizando a classe `Scanner` do Java. Adicionalmente, o programa deve apresentar na janela `Output`, as notas parciais do aluno em linhas diferentes e na última linha a seguinte mensagem `"A média do aluno é Y"` (onde Y é o resultado). Imprima o valor da média utilizando saída formatada.

8. Escreva um programa que leia dois números inteiros usando a classe `Scanner`, sendo os mesmos respectivamente base e altura, e calcule a área de um retângulo.

No resultado, o programa deve apresentar o título `"Calculo da Área de um Retângulo"`, e na linha de baixo apresentar o cálculo realizado e o resultado do valor calculado, utilizando saída formatada.