Introdução à programação em JAVA

Notas gerais sobre Java:

- Todas as instruções terminam com ponto-e-vírgula (;)
- Um bloco é formado por qualquer conjunto de instruções entre { }
- As strings são delimitadas pelo caráter aspas " e os carateres pelo caráter apóstrofe '
- A linguagem é *Case Sensitive* i.e. distingue maiúsculas de minúsculas. Por exemplo System é diferente de system
- Declaração de variáveis, cálculos com variáveis e estrutura if(…) … else semelhante à linguagem C

Comentários

Iguais aos do C e do C++:

- /*…*/ para comentários de várias linhas
- // para comentários de apenas uma linha.
- /**…*/ é lido como um comentário normal, mas é também utilizado para gerar documentação externa acerca desse código, os chamados "javadocs". Assim, quando um programador introduz comentários no seu código, desta maneira, está ao mesmo tempo a criar a sua documentação.

Nomes válidos para identificadores

- Iniciam por letras, $(cifrão) ou _(underscore). Nunca iniciam por números (não deverão incluir carateres que se possam confundir com operações aritméticas ou lógicas)
- Após a primeira letra, qualquer combinação alfanumérica e dos símbolos $ e _ é válido.
- Não possuem limite de tamanho.
- Não é permitida a utilização de palavras-chave da linguagem para identificadores (ver tabela 1 para ver todas as palavras reservadas Java).
- Identificadores são case sensitive.

Nomes válidos para identificadores:

```
int $abc | int abc | int _abc | int $ab2 | int $__$$__$$ | int $ | int _ | int _123
```

Nomes inválidos para identificadores:

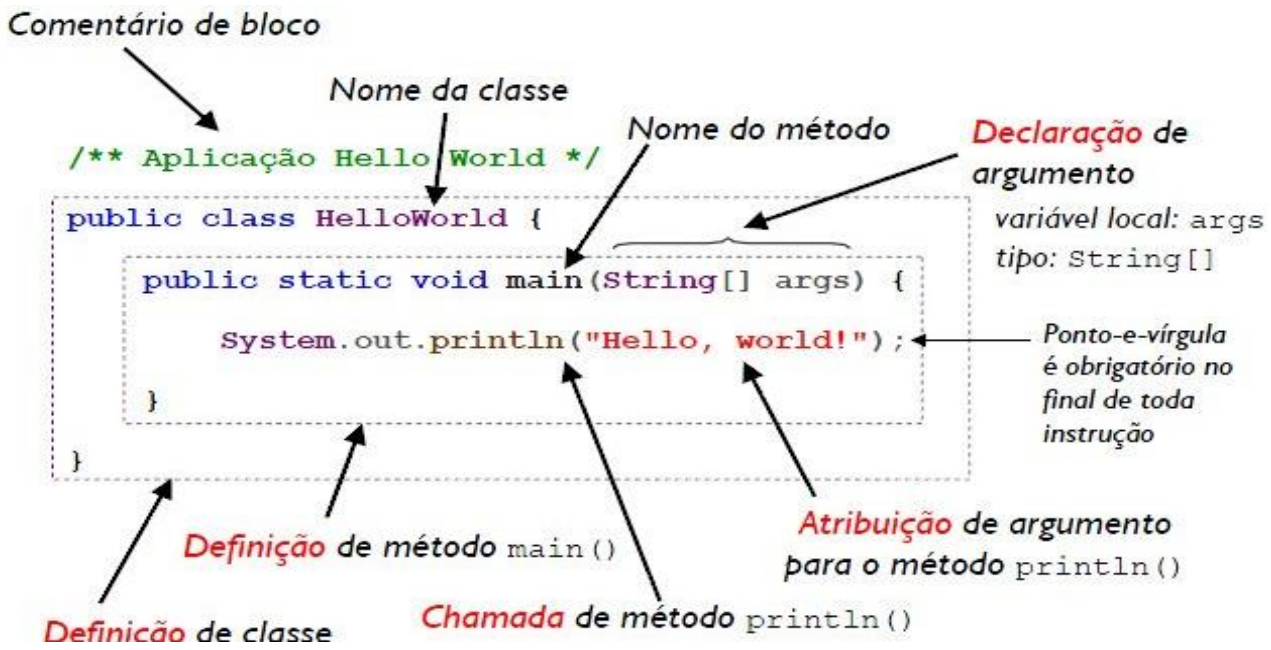
```
int 2abc | int do | int ab c | int :acdc
```

Tabela 1 (Retirada da documentação da Oracle)

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

Estrutura de um programa em Java

```
/** Aplicação Hello World */
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, world!");
    }
}
```



System.out →objeto de saída em Java

Nota: apenas no módulo na ficha 3 serão estudados os conceitos de classe e método

Tipos de dados básicos

Tipo	Descrição
void	Ausência de valor (0 bytes). É usado associado a funções que não devolvem qualquer valor.
char	N.ºs inteiros ou carateres (1 byte). Pode conter apenas 1 único caráter, e o n.º de representações possíveis são 256.
byte	N.º inteiro (1 byte). Valor min. -128, valor Max. 127.
short	N.º inteiro curto (2 bytes). Valor min. -32768, valor Max. 32767.
int	N.º inteiro (4 bytes). Valor min. -2.147.483.648, valor Max. 6482.147.483.647.
long	N.º inteiro longo (8 bytes). Valor min. -9.223.372.036.854.775.808, valor Max. 9.223.372.036.854.775.807.
float	N.ºs reais (4 bytes). Para armazenar valores numéricos com parte fracionária. Valor mín. -3,4028E + 38, valor máx. 3,4028E + 38 (6-7 dígitos)
double	N.ºs reais de dupla precisão (8 bytes). Valor mín. -1,7976E + 308 , valor máx. 1,7976E + 308 (15 dígitos)

Exemplos de declaração:

```
int a,b;
double f=3E10;
int x=03;
float a=2.14f;
```

Nota: todos os valores que são atribuídos diretamente a uma variável do tipo float são, por predefinição, consideradas double. Para que seja variável seja float é necessário colocar um f à frente do valor atribuído, caso contrário o compilador gera um erro.

Carateres

Usado para armazenar carateres simples. Trata-se de um n.º de 2 bytes, sendo utilizada uma tabela internacional denominada *Unicode* para codificar os carateres (carateres e símbolos de todos os idiomas).

Exemplo:

```
char nome_do_char = 'a';
```

Conversão de dados

Uma variável de um tipo de dados simples pode ser convertida noutro tipo de dados também básico.

As conversões sem perda de dados são as seguintes:

byte ->short, int, long, float ou double;
short ->int, long, float ou double;
char->int,long, float ou double;
int->long, float ou double,
long -> float ou double;
float -> double.

A perda de informação pode ocorrer quando uma variável de um tipo de dados com domínio maior é convertida numa variável de um tipo de dados com domínio menor, por exemplo long->int.

Para converter int em long fazer:

```
int x=2;
long y=x;
```

Para converter um long num int é necessário usar casting e do seguinte modo:

```
long x=2;
int y=(int)x;
```

Relembra a noção de casting

Sempre que temos numa variável ou expressão um valor de um determinado tipo e queremos, momentaneamente, modificar o tipo desse valor, indicamos o tipo que queremos entre parêntesis antes do identificador da variável ou expressão.

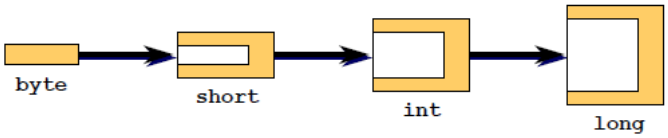
Sintaxe: (<tipo>) variável

Exemplo:

```
Public class casti {
  public static void main(String[] args) {
    int a;
    a=(int) 2.14;
    System.out.println("a= " + a);
   }
}
```

Conversões e Casts

- O Java converte automaticamente valores de um tipo numérico para outro tipo maior.



- O Java não faz automaticamente o "downcast."

Booleanos ou lógicos

Este tipo de dados permite apenas variáveis com dois valores possíveis: verdadeiro (true) e falso (false).

Exemplo:
boolean nome_bool = true;

String (cadeia de carateres)

O tipo **String**, com **S** maiúsculo, é um dos objetos mais utilizados em Java

Exemplos:

String ola;
String ola = "Olá Mundo Java ! "

Constantes

O Java oferece a palavra-chave *final* para especificar dados constantes

SINTAXE: **final < tipo > < identificador > = < valor >;**
Exemplo:

```
1. public class JavaFree {
2.       static final int someInt = 10;
3. }
```

Variável – entidade que pode assumir diferentes valores ao longo da execução dum programa.

SINTAXE: **tipo var1 [, var2, …, var n] ;**

Exemplo: int i;
char letra, l;

OPERADORES

Operadores aritméticos

Operador	Nome	Formato	Descrição
+	Soma	op1 + op2	Calcula a soma dos operandos.
-	Subtração	op1 - op2	O segundo operando é subtraído ao primeiro operando.
*	Multiplicação	op1 * op2	Calcula a multiplicação dos operandos.
/	Divisão	op1 / op2	Calcula o quociente da divisão do 1º operando pelo 2º operando.
%	Resto	op1 % op2	Calcula o resto da divisão do 1º operando pelo 2º operando.

Operadores unários ++ e - -

Estes operadores tornam-se úteis para incrementar e decrementar variáveis, reduzindo significativamente a quantidade de código escrito.

Operador	Significado	Exemplo	Equivalente a
+ +	Incremento de 1	i++, ++k	i=i+1, k=k+1
- -	Decremento de 1	i--, --k	i=i-1, k=k-1

Nota: A utilização dos operadores à esquerda (prefixo) ou à direita (postfixo) de uma variável podem, em instruções mais complexas, alterar o resultado final.

Diferença entre ++x e x++

Suponhamos que 5 é o valor de x e consideremos as seguintes expressões:

Y= x++	Y=++x
1º O valor de x é atribuído a y	1º O valor de x é incrementado
2.º O valor de x é incrementado	2º O valor de x é atribuído a y
Valor final: x vale 6 e y vale 5	Valor final: x vale 6 e y vale 6

Exemplo

```
public class decla {
  public static void main(String[] args) {
    int a,b;
    a=2; b=3;
    System.out.println("a+b= " + (++a+b));
   }
}
Qual o output do exemplo acima? _____________
```

Operadores de atribuição:

Operador	Nome	Formato	Descrição
=	Atribuição	op1 = op2	Atribui um valor a uma variável.
+=	Atribuição composta da soma	op1 += op2	O valor de op1 + op2 é calculado e atribuído a op1. Equivalente a op1 = op1 + op2.
-=	Atribuição composta da subtração	op1 -= op2	O valor de op1 - op2 é calculado e atribuído a op1. Equivalente a op1 = op1 - op2.
*=	Atribuição composta da multiplicação	op1 *= op2	O valor de op1 * op2 é calculado e atribuído a op1. Equivalente a op1 = op1 * op2.
/=	Atribuição composta da divisão	op1 /= op2	O valor de op1 / op2 é calculado e atribuído a op1. Equivalente a op1 = op1 / op2.
%=	Atribuição composta do resto	op1 %= op2	O valor de op1 % op2 é calculado e atribuído a op1. Equivalente a op1 = op1 % op2.

Exemplo:

```
Se distancia = 4, velocidade = 2
    distancia = velocidade  resultado: 2
    distancia += velocidade  resultado: 6
    distancia -= velocidade  resultado: 2
    distancia *= velocidade  resultado: 8
    distancia /= velocidade  resultado: 2
    distancia %= velocidade  resultado: 0
```

Operadores Relacionais

Operador	Significado	Exemplo
==	Igualdade	a= = b
>	Maior que	a>b
>=	Maior ou igual que	a>=b
<	Menor que	a<b
<=	Menor ou igual que	a<=b
! =	Diferente de	a! = b
?:	Operador ternário: se a condição for verdade o resultado é 4 caso contrário o resultado é 2	a>b? 4:2
Instanceof	Devolve true se o objeto da esquerda for uma instância do objeto da direita	Ref **instanceof** Ident verifica se a referência Ref é do tipo Ident

Operadores Lógicos

Operador	Significado	Exemplo	Resultado se x for 2
&&	AND (E lógico)	(x>=3 && x<=19)	false
||	OR (OU lógico)	(x == 1 || x== 2)	true
!	NOT (negação lógica)	! (x>1)	false
^	não ou (Disjunção exclusiva)	(x>=3 ^x<=19)	true

^ produz um valor verdadeiro apenas se a quantidade de operadores verdadeiros for ímpar.

Nota: Relembra as tabelas de verdade.

Precedência de Operadores

Order	Operadores	Comments	Assoc.
1	++ -- + - ~ !(tipo)	Unary operadores	R
2	* / %	Multiply, divide, remainder	L
3	+ - +	Add, subtract, add string	L
4	<< >> >>>	Shift (>>> is zero-fill shift)	L
5	< > <= >= instanceof	Relational, tipo compare	L
6	== !=	Equality	L
7	&	Bit/logical e	L
8	^	Bit/logical exclusive OR	L
9	|	Bit/logical inclusive OR	L
10	&&	Logical e	L
11	||	Logical OR	L
12	?:	Conditional operador	R
13	= op=	Assignment operadores	R

EXERCÍCIOS:

1. As variáveis a e b têm os valores 1 e 7, respetivamente e as expressões um valor lógico. Que valores têm a, b e as expressões lógicas após a execução (considera false 0 e true qualquer valor diferente de zero)?

a. b = ++ a

b. (a == b || a!= b)

c. true && false

d. 5 == 7 && a==5

e. b == 7 && a< 5

f. (a == 1 || b >= -1 && !(true))

2. Qual o resultado das variáveis das expressões seguintes, tendo a o valor 5.

a. 5>7 ? false : true ? 7 : 2

b. (a==3) < 4 ? 3 : 2

3. Seja x=1, y=3 e z= 0. Calcula o resultado das expressões seguintes.

a. y* = z++ + 2/++x

b. x* = y-- *y - ++x

c. y- = x * - -Y/+ + z

d. y% = ++x + x- - *++z

e. x+ = y++ * ++z

---

4. De que tipo é a linguagem JAVA?

5. Indica duas caraterísticas da linguagem java.

6. Completa com os termos indicados (poderão ser utilizados ou não):

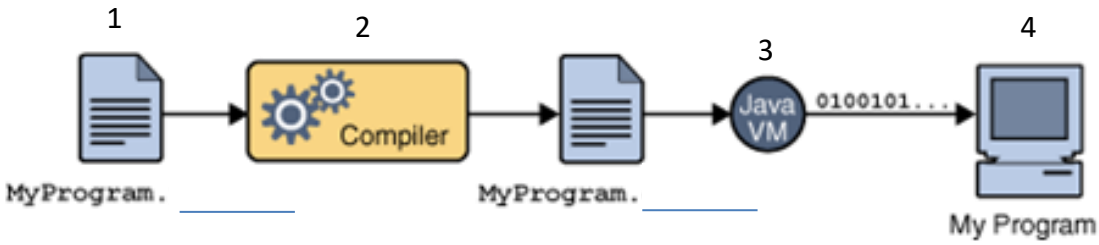
Compilação, interpretação, codificação, Bytecode, executável, fonte, Unicode, é, não é, Javax, Linux, Javac

A máquina virtual do java, o J.V.M. faz a _____ de programas em _____, estes últimos resultam da _____ do programa _____, escrito em Java.

Na compilação tradicional _____ necessário criar um executável para cada plataforma enquanto que o compilador de JAVA, o chamado _____, cria um programa especial que poderá vir a ser usado em qualquer plataforma.

Em Java para cada Sistema Operativo _____ necessário instalar uma J.V.M.

7. Observa a figura



Procede à descrição de cada uma das fases de criação de um programa em Java e completa a legenda da figura, indicando a extensão do MyProgram em cada fase:

1	
2	
3	
4	