



Curso de Java

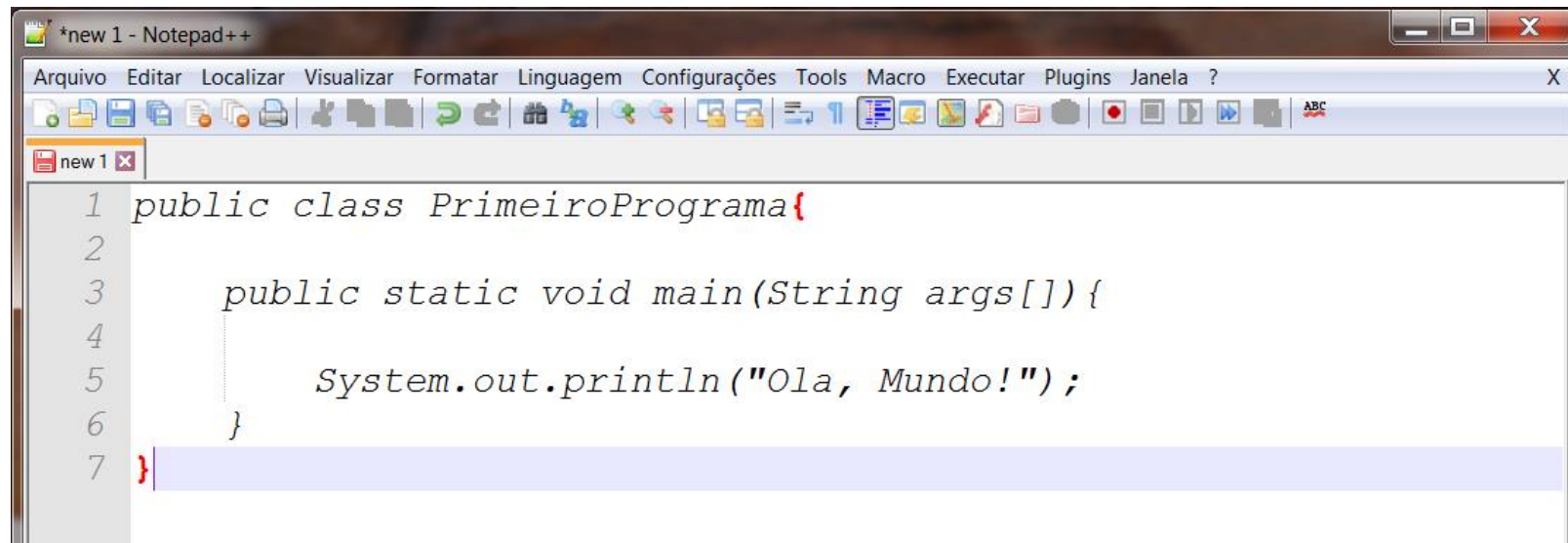
aula 02

Prof. Anderson Henrique



Criando nosso primeiro programa

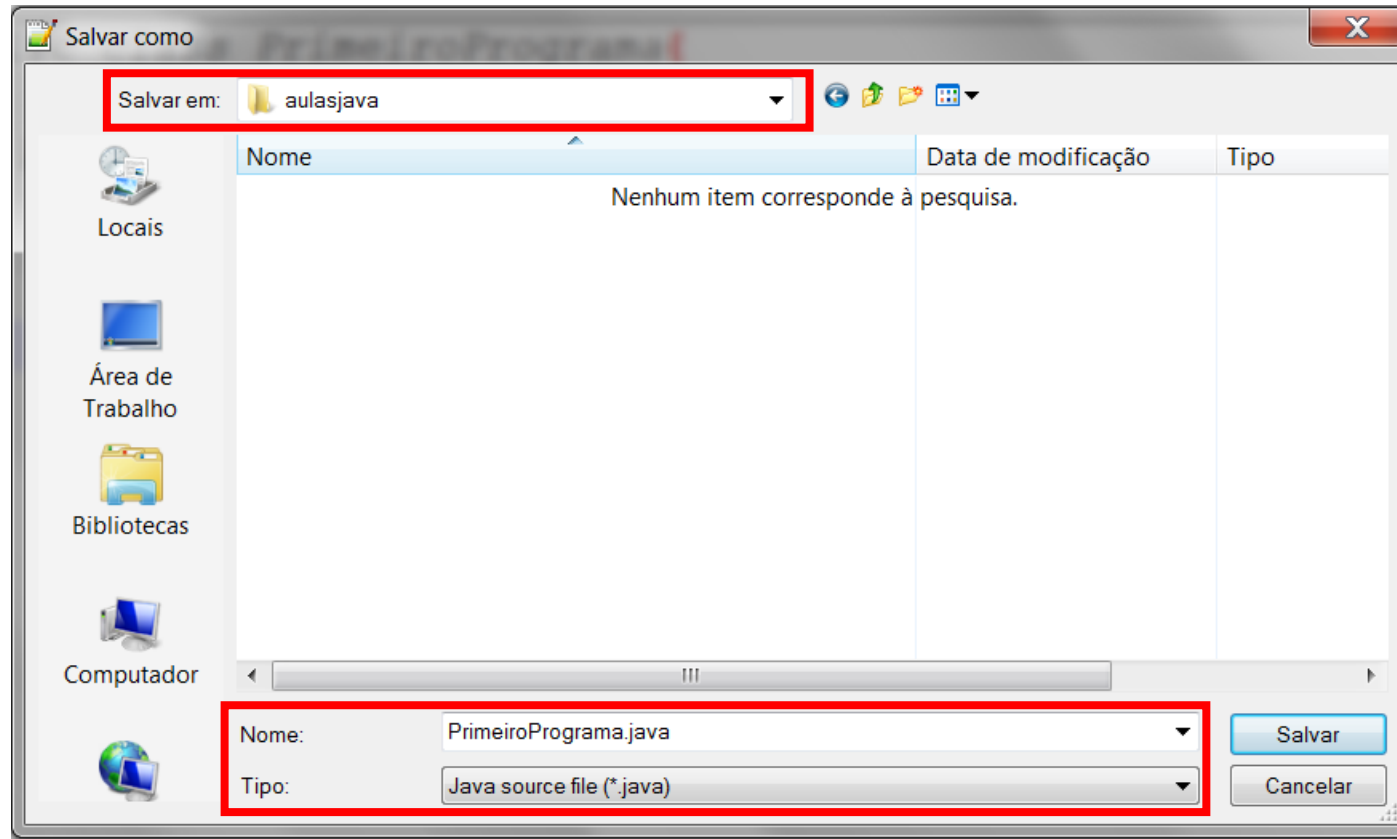
No editor notepad++, vamos escrever o nosso programa inicial...



```
*new 1 - Notepad++
Arquivo  Editar  Localizar  Visualizar  Formatar  Linguagem  Configurações  Tools  Macro  Executar  Plugins  Janela  ?
new 1
1 public class PrimeiroPrograma{
2
3     public static void main(String args[]){
4
5         System.out.println("Ola, Mundo!");
6     }
7 }
```

Após escrever, precisamos salvar o arquivo com o mesmo nome que foi dado a classe na linha 1, e o tipo de extensão de arquivo é .java (Java Source File)...

Vamos criar uma pasta, no diretório C:\ > chamada aulasjava e salvá-lo neste diretório...

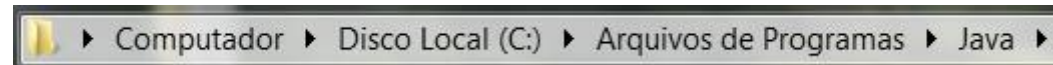


Será criado um arquivo .java, o arquivo pré-compilado, ainda não compreensível pela JVM... Precisamos gerar o *bytecode*, utilizando o `javac` (Java Compiler).

Precisamos configurar o ambiente para que esteja disponível o acesso a todos os programas utilizados pelo Java.



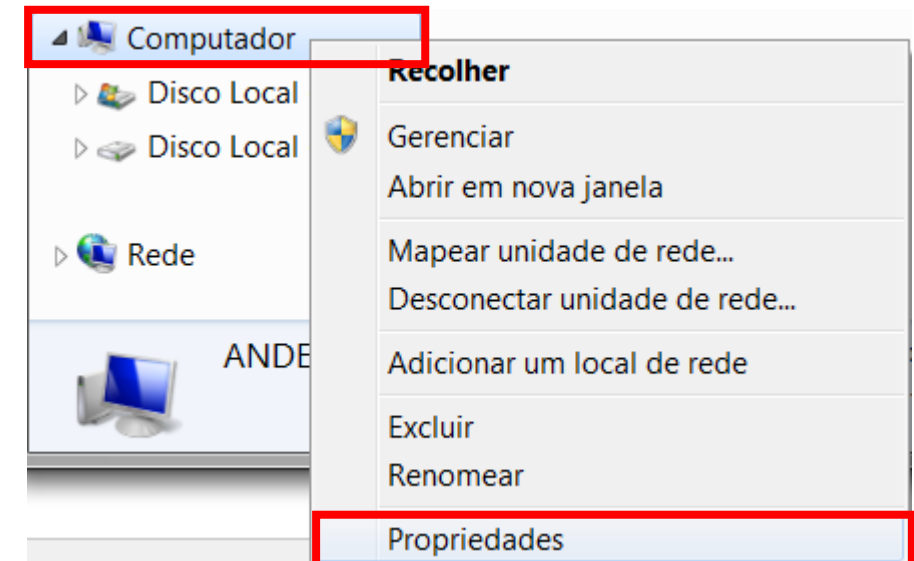
Como vimos, na aula anterior, os programas ficam localizados no seguinte caminho em nosso sistema operacional:



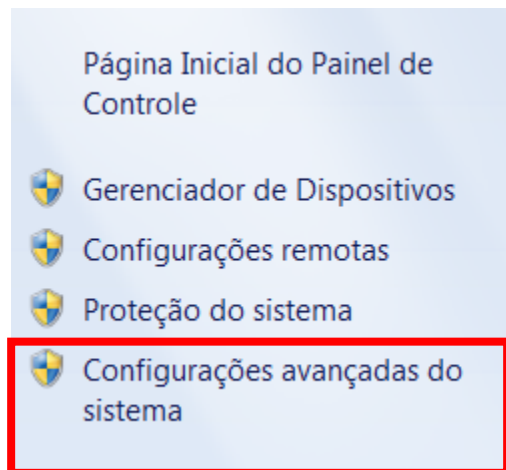
Clique na pasta amarela, você terá o caminho necessário para acessar os recursos que o Java oferece:



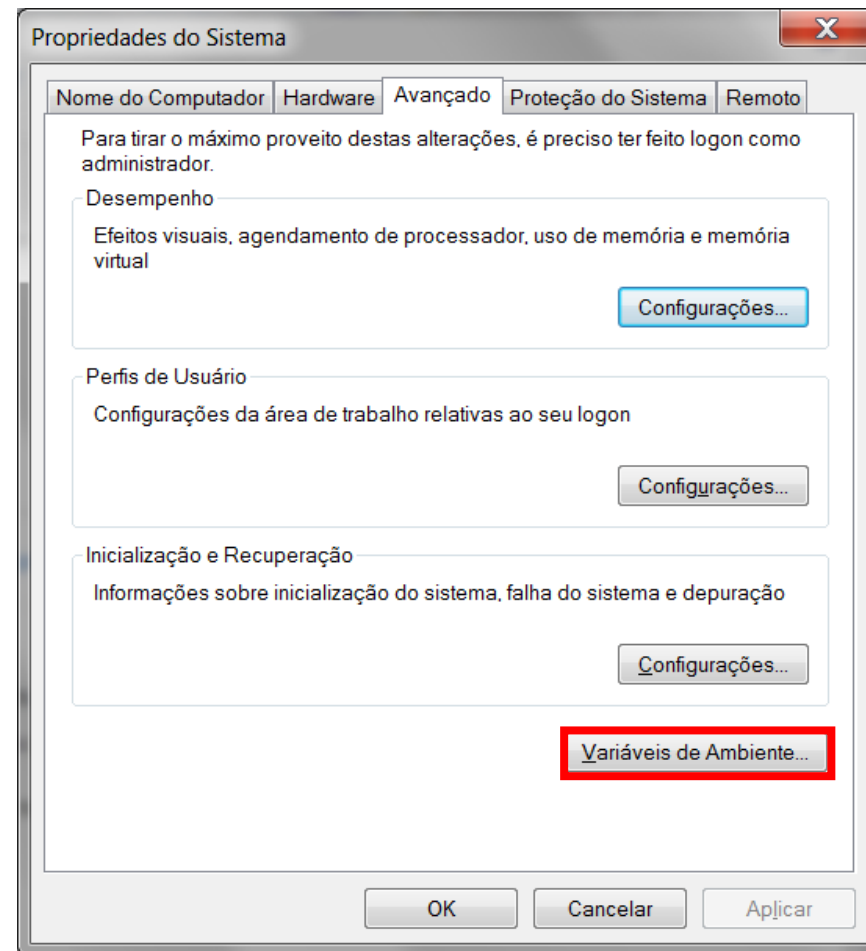
Copie este caminho... Agora vá na janela do windows explorer, com o botão auxiliar do mouse, clique em... Computador > propriedades...



Na próxima janela, acesse configurações avançadas do sistema...



clique na opção Variáveis de Ambiente...



Na próxima janela, procure em variáveis do sistema a opção path > e clique em Editar...

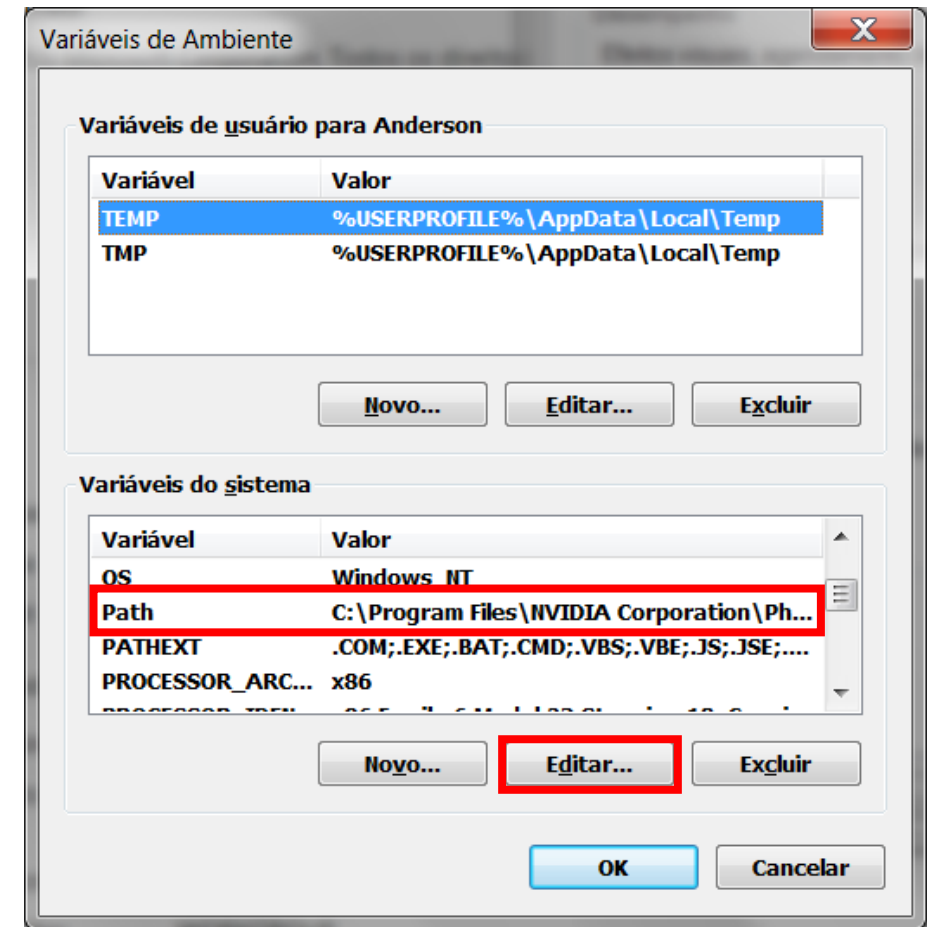
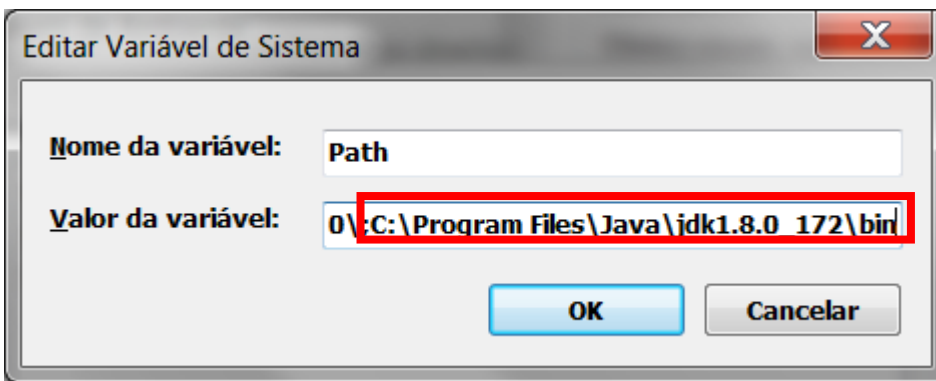
Na próxima janela, teremos dois campos:

Nome da variável: Path

Valor da variável:

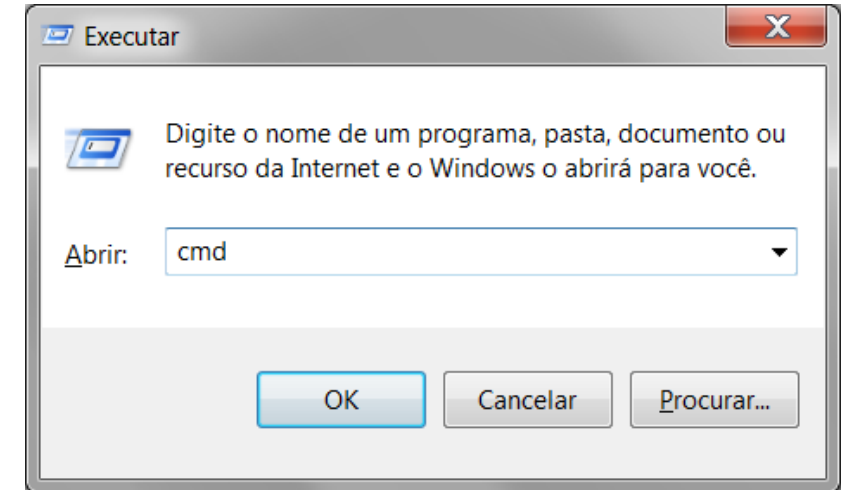
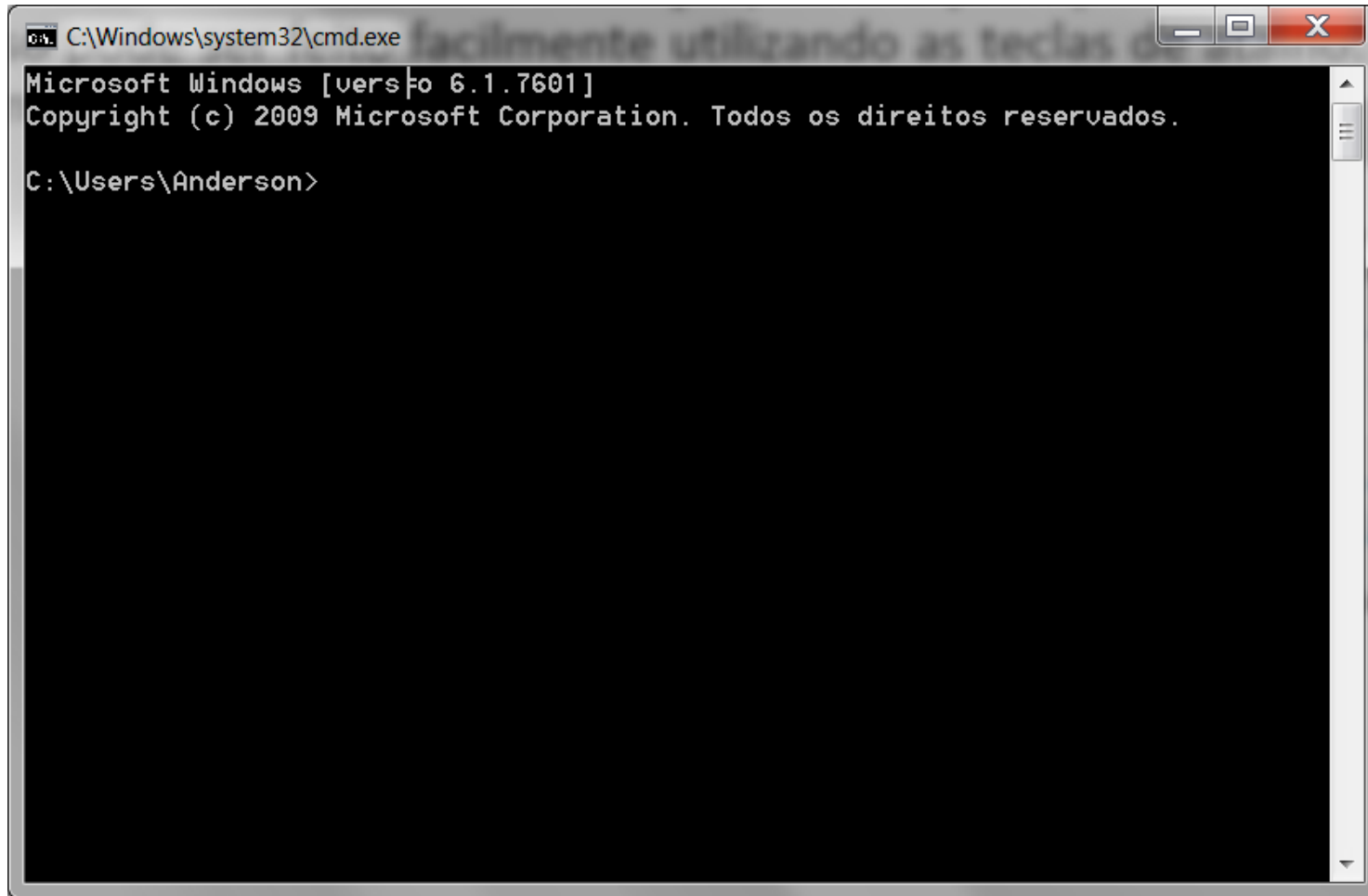
Em valor da variável, no final da linha, cole o caminho dos programas java...

C:\Program Files\Java\jdk1.8.0_172\bin



Confirme a alteração, clicando em ok em todas as janelas.

Após ter sido feita essa alteração, abra o prompt de comando (cmd) do Windows. Isso pode ser feito facilmente utilizando as teclas de atalho: windows + r, digitando cmd e clicando no botão ok...





Estes são os principais comandos do cmd:

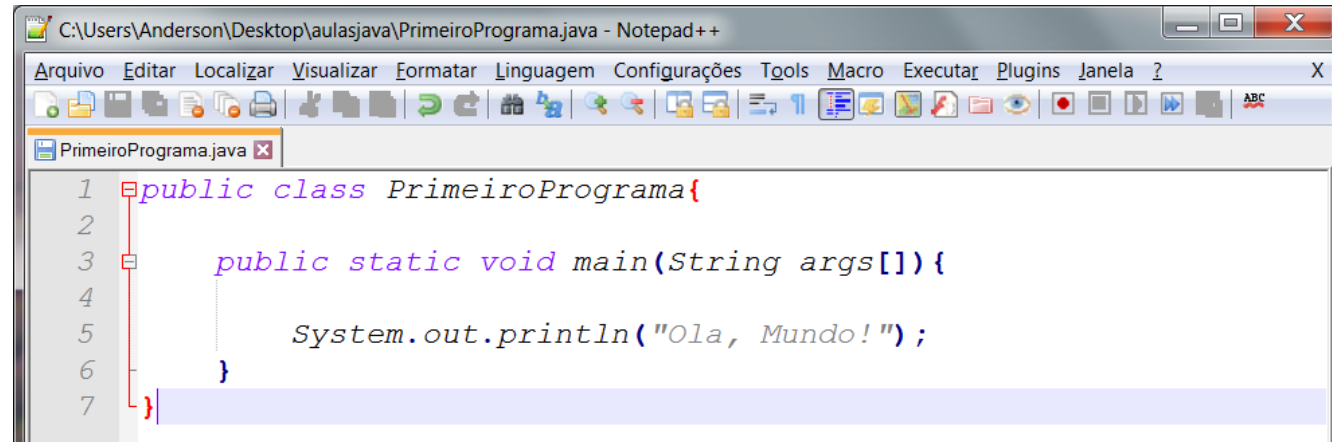
Comando	Para que serve
cls	Limpar a tela
cd..	Sair de um diretório
dir	Exibe todo o conteúdo do diretório
cd diretório	Entrar em um diretório (Ex.: cd aulasjava)
javac	Compilar um código-fonte com a extensão .java
java nomeClasse	Rodar um programa, através do pseudocódigo (<i>bytecode</i>)

```
C:\Windows\system32\cmd.exe
C:\>cd aulasjava
```

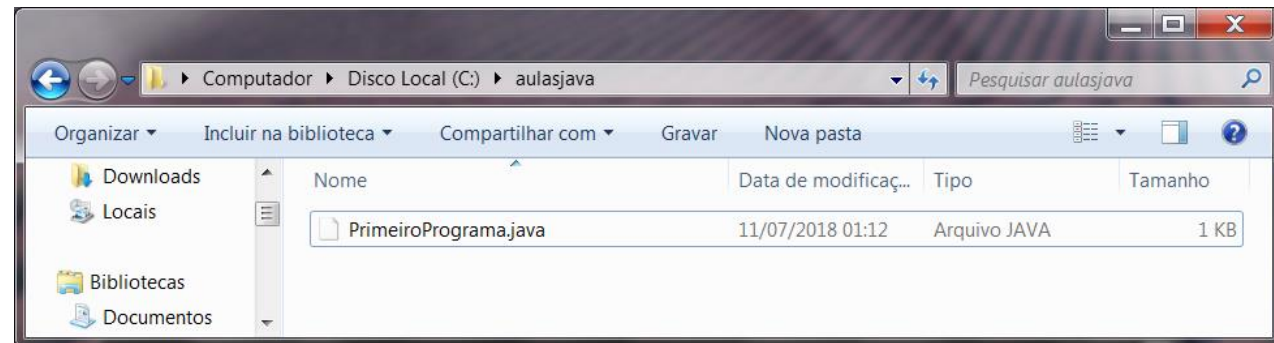
```
C:\Windows\system32\cmd.exe
C:\>javac PrimeiroPrograma
```

```
C:\Windows\system32\cmd.exe
C:\>java PrimeiroPrograma
```

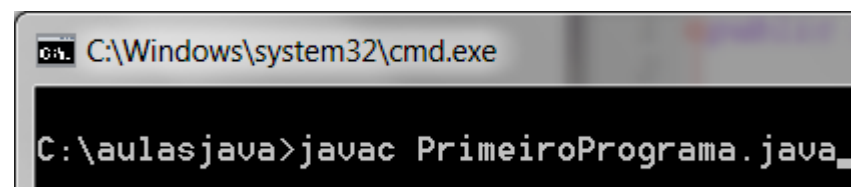

Vamos compilar o arquivo que fora salvo anteriormente no nosso diretório que se encontra em C:\aulasjava.

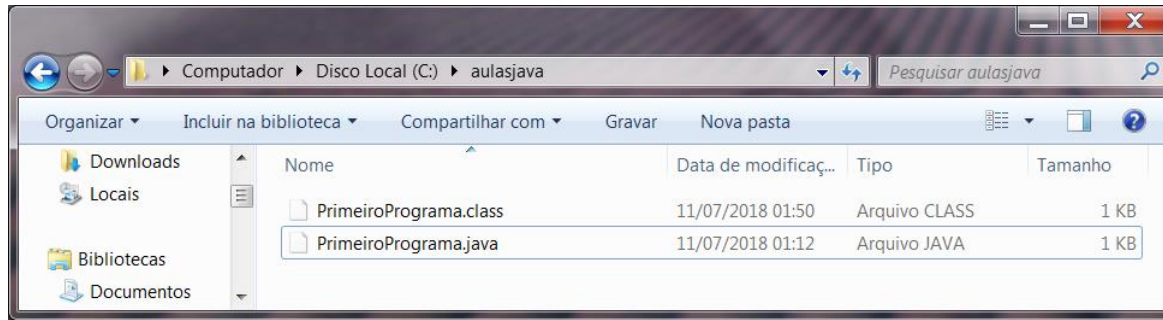


```
1 public class PrimeiroPrograma{
2
3     public static void main(String args[]){
4
5         System.out.println("Ola, Mundo!");
6     }
7 }
```

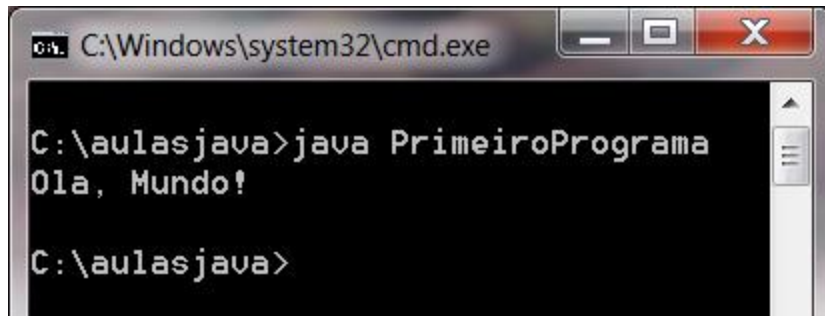


Após a compilação será gerado um outro arquivo com a extensão .class, esse é o nosso *bytecode*.





Agora podemos pedir para que a nossa JVM (interpretador), venha executar o nosso primeiro programa: ele escreverá na tela: “Ola, Mundo!”



Após a execução do programa, a tarefa será finalizada... Vamos compreender cada linha de código digitada no documento PrimeiroPrograma.java



Variáveis

Assim como ocorre na linguagem C++, em Java todas as variáveis devem ser necessariamente declaradas antes de seu uso efetivo. Nessa declaração, fornecemos ao compilador o nome da variável e o tipo de dado que ela deve armazenar.

Uma dica interessante é utilizar as três primeiras letras iniciais para identificar o tipo de dado. Por exemplo, as variáveis **intNumero** e **intValor** são facilmente identificadas como sendo do tipo **int** (abreviação de *integer*) porque se iniciam com a cadeia de caracteres “int”. Para variáveis do tipo **double**, pode-se utilizar “dbl”.

O tipo de dado é importante para que o compilador possa determinar quantas posições de memória devem ser alocadas, de acordo com tamanho em bytes do tipo de dado especificado. Por exemplo, uma variável do tipo **byte** somente ocupa um byte, ao passo que uma variável do tipo **int** (inteiro) precisa de 4 bytes para ser armazenada.

Essa especificação de tipos também define a faixa de valores que podem ser armazenados na variável. No exemplo da variável do tipo **byte**, somente são aceitos valores entre -128 e 127. já com o tipo **int** a variável pode armazenar valores entre -2147483648 e 2147483647.

Veja a tabela a seguir, com mais detalhes:

Tipos Primitivos



Família	Tipo Primitivo	Classe Invólucro	Tamanho	Exemplo
Lógico	boolean	Boolean	1 bit	true
Literais	char	Character	1 byte	'A'
	-	String	1 byte/cada	"JAVA"
Inteiros	byte	Byte	1 byte	127
	short	Short	2 bytes	32 767
	int	Integer	4 bytes	2 147 483
	long	Long	8 bytes	2^{63}
Reais	float	Float	4 bytes	$3.4e^{+38}$
	double	Double	8 bytes	$1.8e^{+308}$

Palavras reservadas

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

Além dos tipos de dados primitivos, encontramos também em Java os chamados tipos de referência. Uma variável desse tipo contém na verdade um valor que representa o endereço de memória em que se encontra armazenada a informação ou o conjunto de valores.

Uma referência às vezes também é chamada de ponteiro, muito embora Java não ofereça suporte para esse conceito, como ocorre em C++. Um exemplo prático de referência é um vetor.

Ainda em relação às variáveis, um assunto muito importante que se deve considerar é o escopo. O escopo de uma variável define tanto o seu tempo de vida quanto sua abrangência. O primeiro conceito se refere ao tempo em que a variável permanece ativa (ou seja, é visível pelo programa). Por exemplo, uma variável pode ser declarada de maneira que exista enquanto uma determinada classe está em ação ou durante toda a execução do programa. Já a abrangência diz respeito às áreas do código que podem ter acesso à variável e seu conteúdo.



Operadores

Para podermos manipular os dados armazenados nas variáveis de memória, dispomos dos operadores. Temos basicamente quatro tipos de operadores: aritméticos, que permitem a execução de cálculos; relacionais, para efetuarmos comparações entre variáveis e valores; lógicos, que permitem a construção de expressões lógicas dentro de comandos de decisões; operadores de atribuição de valores.

Veja as tabelas a seguir:



Operadores Aritméticos

Operador	Descrição
+	(Adição)
-	(Subtração)
*	(Multiplicação)
/	(Divisão)
%	(Resto/Módulo)



Operadores Incremento e Decremento

Valor inicial de x	Expressão	Valor final de y	Valor final de x
5	$y = x++;$	5	6
5	$y = ++x;$	6	6
5	$y = x--;$	5	4
5	$y = --x;$	4	4



Operadores Relacionais

Símbolo	Nome do Operador	Exemplo	Significado
>	Maior que	$x > y$	x é maior que y?
>=	Maior ou igual	$x \geq y$	x é maior ou igual a y ?
<	Menor que	$x < y$	x é menor que y?
<=	Menor ou igual	$x \leq y$	x é menor ou igual a y ?
==	Igualdade	$x == y$	x é igual a y?
!=	Diferente de	$x != y$	x é diferente de y?



Operadores Lógicos

Operador	Ação
&&	And (E)
	Or (Ou)
!	Not (Não)



Tabela verdade

A	B	A E B	A OU B	NÃO A	NÃO B
Verdadeiro	Verdadeiro	Verdadeiro	Verdadeiro	Falso	Falso
Verdadeiro	Falso	Falso	Verdadeiro	Falso	Verdadeiro
Falso	Verdadeiro	Falso	Verdadeiro	Verdadeiro	Falso
Falso	Falso	Falso	Falso	Verdadeiro	Verdadeiro

Prosseguiremos no próximo slide...



Professor: Anderson Henrique

Programador nas Linguagens Java e PHP

