

ARA0066 - PROGRAMAÇÃO ORIENTADA A OBJETOS EM JAVA



ARA0075

Aula - 01





Formato: 3 horas de aulas
no formato presencial.

Carga horária: 80



Nossa agenda de hoje.

- Saudação
- Apresentação prof. Fábio Cardozo
- Apresentação dos alunos
- Apresentação da disciplina
- Módulo 1-> Tema 1



Sejam bem vindos.



Prof. Fábio Cardozo

- Téc. em Tecnologia da Informação
- Graduado em Análise e desenvolvimento de Sistemas
- Pós-graduado em Rede de Computadores
- Pós-graduado em Tecnologias de Educação a Distância
- Mestre em Modelagem Matemática e Computacional
- Doutorando em Tecnologias em Desenvolvimento Agropecuário

Prof. Fábio Cardozo - OUTROS FATOS



- Sou casado
- Tenho 2 filhas
- E sim, zerei Street Fighter II no Hard sem dar continue (Gravei o evento em VHS)...

Apresentação dos alunos

- Nome
- Idade
- Por que de estarem nesse curso?
- Área de interesse profissional na TI
- Hobbie



Ementa

Classes e objetos

Encapsulamento

Herança e Polimorfismo

Agrupamento de objetos

Ambiente Java

Hierarquia e Herança

Métodos

Interfaces (Java)

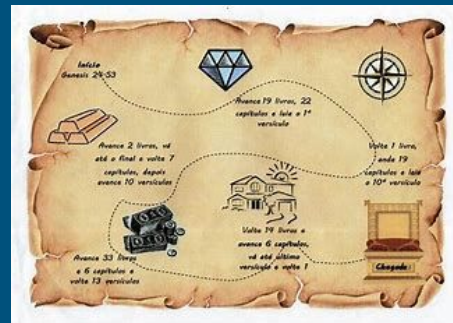
Exception

Threads

Ponteiros e alocação de memória.

API JDBC

Subprogramas



Objetivos



Após o curso o aluno deve ser capaz de compreender os conceitos de POO, sendo capaz de desenvolver sistemas baseados neste paradigma.

Além dos conceitos, o aluno ainda deverá ser capaz de implementar sistemas em JAVA utilizando POO.

Ambiente de desenvolvimento

Utilização de Terminal + Sublime Text

Download em: <https://download.sublimetext.com/Sublime%20Text%20Build%203211%20x64%20Setup.exe>

Java Development Kit - JDK

Download em: <https://jdk.java.net/19/>

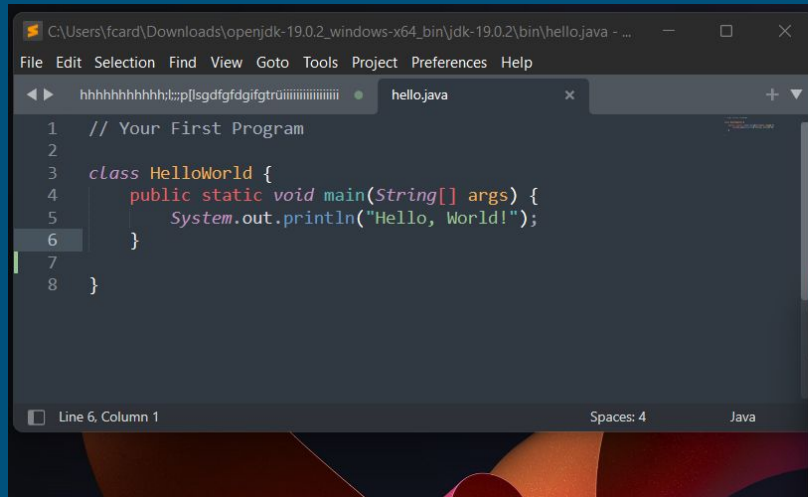
IntelliJ - Community

Download em: <https://www.jetbrains.com/idea/download/?section=windows>



Ambiente de desenvolvimento

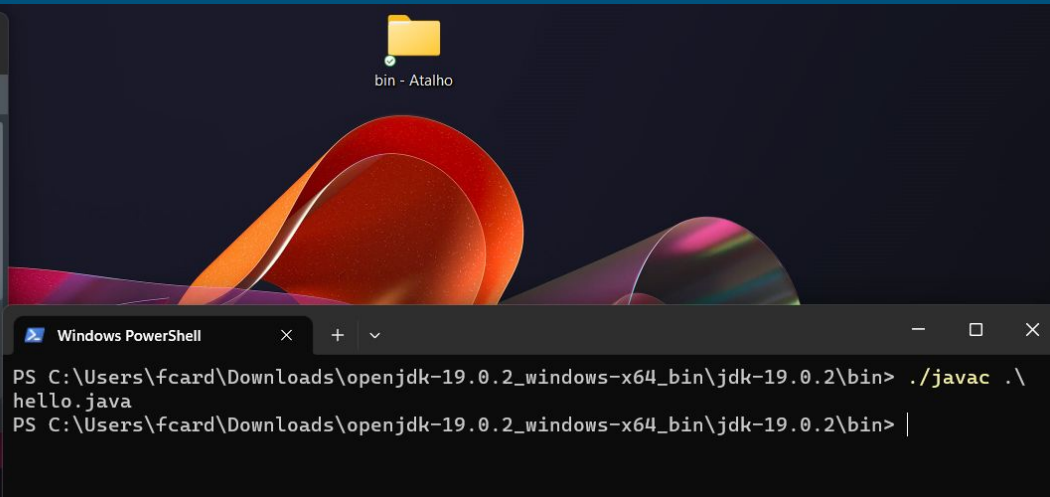
Antes de utilizarmos uma IDE (Integrated development environment). Utilizaremos o compilador “Javac” em linha de comando no terminal.



The screenshot shows an IDE window with the title bar 'C:\Users\fc card\Downloads\openjdk-19.0.2_windows-x64_bin\jdk-19.0.2\bin\hello.java - ...'. The menu bar includes File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, and Help. The editor displays the following code:

```
1 // Your First Program
2
3 class HelloWorld {
4     public static void main(String[] args) {
5         System.out.println("Hello, World!");
6     }
7
8 }
```

The status bar at the bottom indicates 'Line 6, Column 1', 'Spaces: 4', and 'Java'.



The screenshot shows a Windows desktop with a folder icon labeled 'bin - Atalho'. In the foreground, a 'Windows PowerShell' terminal window is open. The command prompt shows the following commands and output:

```
PS C:\Users\fc card\Downloads\openjdk-19.0.2_windows-x64_bin\jdk-19.0.2\bin> ./javac .\hello.java
PS C:\Users\fc card\Downloads\openjdk-19.0.2_windows-x64_bin\jdk-19.0.2\bin> |
```

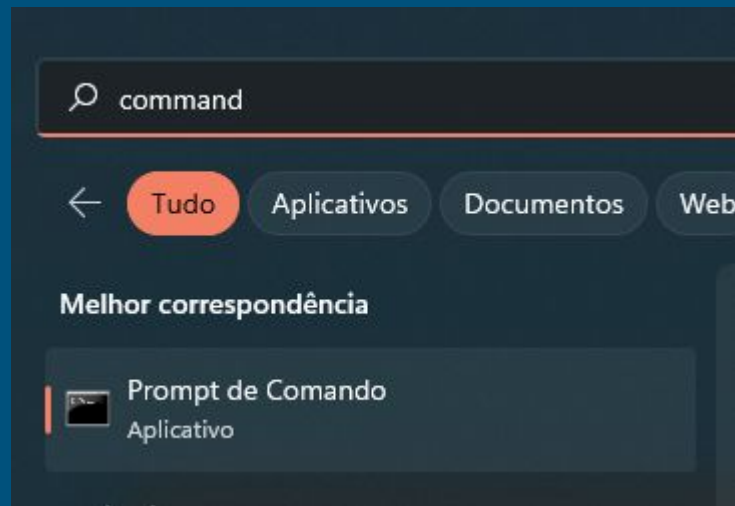
Ambiente de desenvolvimento

Primeiro código:

```
1  class HelloWorld {  
2      public static void main(String[] args) {  
3          System.out.println("Hello, World!");  
4      }  
5  }
```

Ambiente de desenvolvimento - Atividade 1

- Acesse o Terminal (linha de comando).
- Verifique a existência, dos comando “javac” e “java”



Ambiente de desenvolvimento - Atividade 2

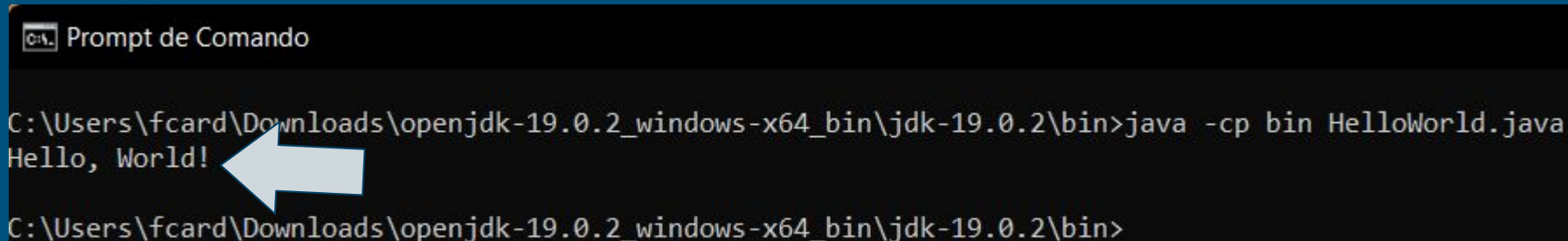
- Utilizando o código do slide (primeiro código), realize a compilação e execução da classe HelloWorld.

```
1 | javac -d bin HelloWorld.java
```

```
java -cp bin HelloWorld.java
```

Ambiente de desenvolvimento - Atividade 2

Resultado esperado.



The screenshot shows a Windows Command Prompt window with the title "C:\> Prompt de Comando". The command prompt displays the following text:

```
C:\Users\fc card\Downloads\openjdk-19.0.2_windows-x64_bin\jdk-19.0.2\bin>java -cp bin HelloWorld.java  
Hello, World!
```

A large white arrow points to the output "Hello, World!". Below the output, the command prompt shows the next prompt:

```
C:\Users\fc card\Downloads\openjdk-19.0.2_windows-x64_bin\jdk-19.0.2\bin>
```

Tema 1: Conceito básico de POO.

O que é Programação Orientada a Objetos?



classe



objeto

Classe:

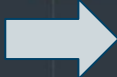
Código base, para a criação da classe Aluno.

```
1  class Aluno {  
2      //Atributo  
3      private String Nome;  
4  
5      //Métodos  
6      public void inserirNome ( ) {  
7          Nome = System.in.toString ();  
8      }  
9      public String recuperarNome ( ) {  
10         return Nome;  
11     }  
12 }
```

Classe:

Atributo da classe Aluno.

```
1  class Aluno {  
2      //Atributo  
3      private String Nome;  
4  
5      //Métodos  
6      public void inserirNome ( ) {  
7          Nome = System.in.toString ();  
8      }  
9      public String recuperarNome ( ) {  
10         return Nome;  
11     }  
12 }
```



Classe:

Métodos da classe Aluno.

```
1  class Aluno {  
2      //Atributo  
3      private String Nome;  
4  
5      //Métodos  
6      public void inserirNome ( ) {  
7          Nome = System.in.toString ();  
8      }  
9      public String recuperarNome ( ) {  
10         return Nome;  
11     }  
12 }
```

Instância de classe:

Etapas da criação de uma instância.

Primeiramente, uma variável é declarada como sendo do tipo de alguma classe.

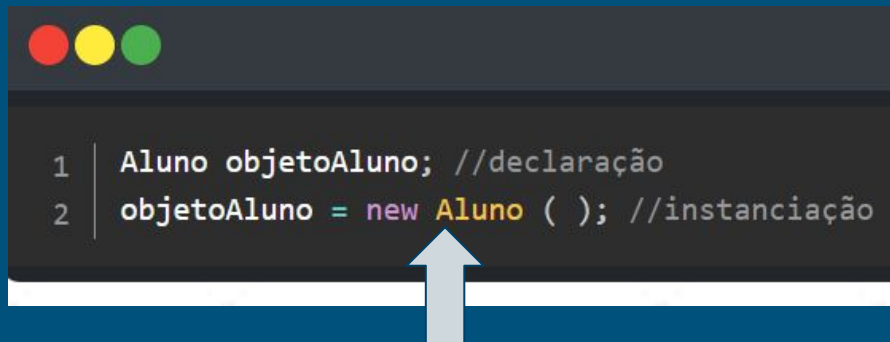


A seguir, o compilador é instruído a gerar um objeto a partir daquela classe, que será rotulado com o identificador que nomeia a variável.

```
1 | Aluno objetoAluno; //declaração
2 | objetoAluno = new Aluno ( ); //instanciação
```

Instância de classe:

Constructor



```
1 | Aluno objetoAluno; //declaração
2 | objetoAluno = new Aluno ( ); //instanciação
```

A light blue arrow points from the word `new` in line 2 to the text box on the right.

O processo de criação do objeto começa com a alocação do espaço em memória.



E prossegue com a execução do código de construção do objeto.

Esse código, obrigatoriamente, deve ser implementado num método especial chamado **construtor**.

Instância de classe: Construtor

O método construtor é sempre executado quando da instanciação de um objeto e obrigatoriamente deve ter nome idêntico ao da classe.

Além disso, ele pode ter um modificador, mas não pode ter tipo de retorno. A instrução *"new"* é sempre seguida da chamada ao construtor da classe. Finalmente, a atribuição (*"="*) inicializa a variável com o retorno (referência para o objeto) de *"new"*.

Reverendo a instanciação, identificamos que ela pode ser decomposta da seguinte forma:



Instância de classe: Observação

Diferentemente de linguagens como a C e a C++, Java não possui arquivos de cabeçalho, portanto, a implementação dos métodos ocorre junto de sua declaração. Além disso, em Java cada classe pública deve estar num arquivo com o mesmo nome da classe e extensão *"java"*. Logo, a classe do Código 1 deve ser salva num arquivo de nome **"Aluno.java"**.

Próximo assunto: Encapsulamento:

Fim