Universidade Cruzeiro do Sul

Laboratório de Desenvolvimento de Algoritmos

Conteúdo:
- Breve Histórico do Java
- Características da Linguagem
- Processo de execução (compilador e interpretador)
- Primeiro exemplo

Prof. Alcides / Profa. Ana Paula / Profa. Cristiane

Ultima aula – Características Java

- Orientada a Objetos
- Independência quanto a plataformas
- Ausência de Ponteiros
- Multithreading
- Acesso remoto a Banco de dados
- Segurança

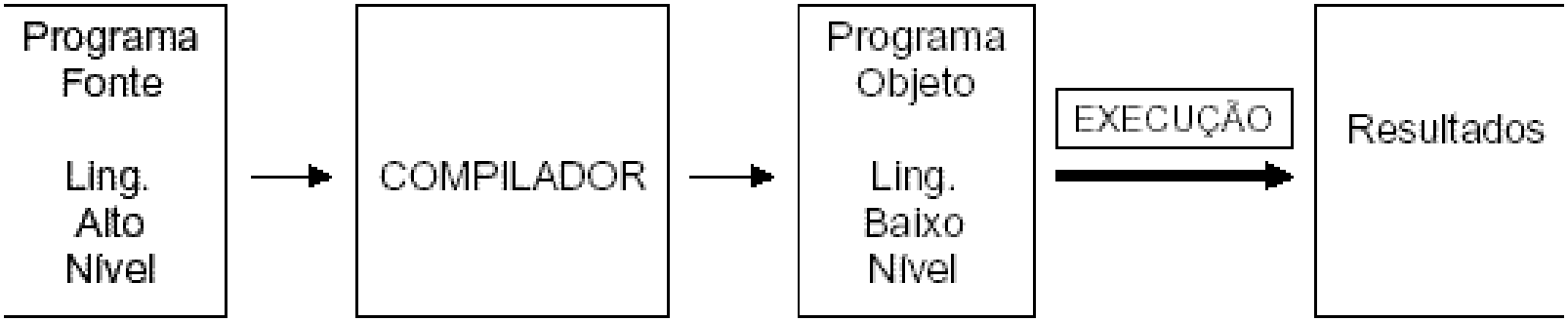
Como a máquina entende os códigos?

- Para que o computador "entenda" um programa escrito em uma linguagem (de alto nível) é necessário um meio de tradução entre a linguagem de alto nível utilizada no programa e a linguagem de máquina.

- Para essa tarefa temos basicamente dois métodos:
  - Compilador
  - Interpretador

Compilador

- Traduz o programa escrito em uma linguagem de programação para um programa equivalente escrito em linguagem de máquina (programa-objeto).



Vantagens:

Velocidade de execução
Oculta o código fonte

Desvantagem:

A cada alteração no programa fonte é necessário gerar novamente o programa-objeto

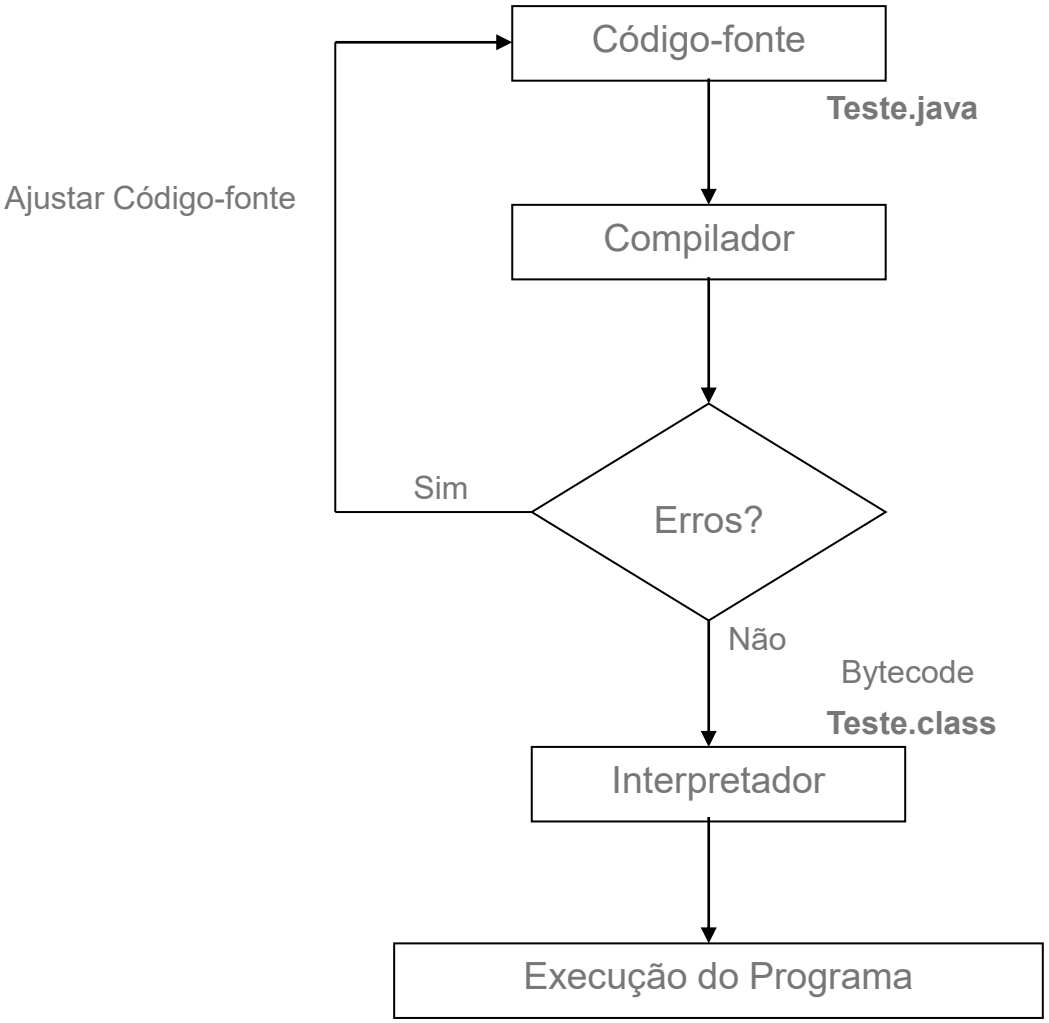
Interpretador

- Traduz e faz a checagem da sintaxe e envia para execução, instrução por instrução.
- Precisa estar presente todas as vezes que vamos executar o programa e o processo acima é repetido.

Vantagem: consome menos memória
Desvantagem: execução mais lenta

Exemplo: Uma página html é interpretada pelo Navegador.

Sequência de criação/execução em JAVA



Na fase de execução é necessário que haja a Máquina Virtual Java (MVJ também conhecida como JVM)

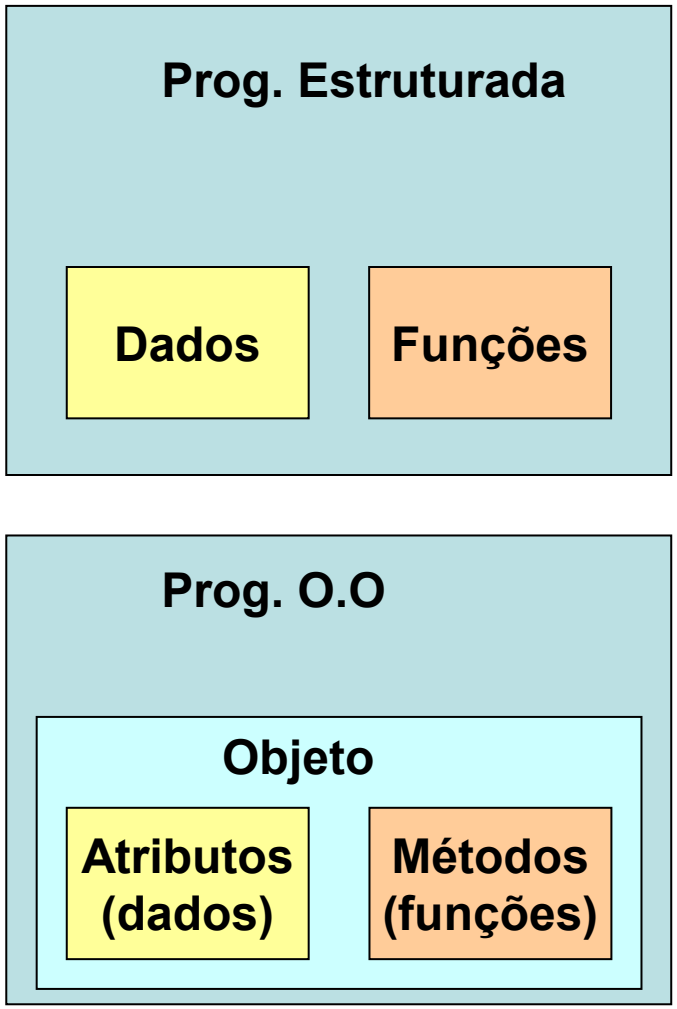
A MVJ interpreta os bytecodes gerados pelo compilador.

O objetivo da MVJ é permitir que qualquer sistema operacional possa executar uma aplicação Java

Seqüência de criação/execução

- Digitação do programa ⇨ código-fonte ⇨ Obrigatoriamente salvo com a extensão **.java.**
- O código-fonte deve então ser passado pelo compilador **javac**, que o transforma em um arquivo bytecode Java, com extensão **.class**.
- Finalmente, o bytecode pode ser executado pelo Interpretador (**java**), em qualquer plataforma que disponha de uma máquina virtual Java.
- Para desenvolver e executar programas Java, precisamos de um compilador (para traduzir seu programa para o "bytecode") e a "JVM". Ambos encontramos no Java Development Kit (JDK) o qual pode ser baixado do site da Oracle.

Paradigmas de Programação



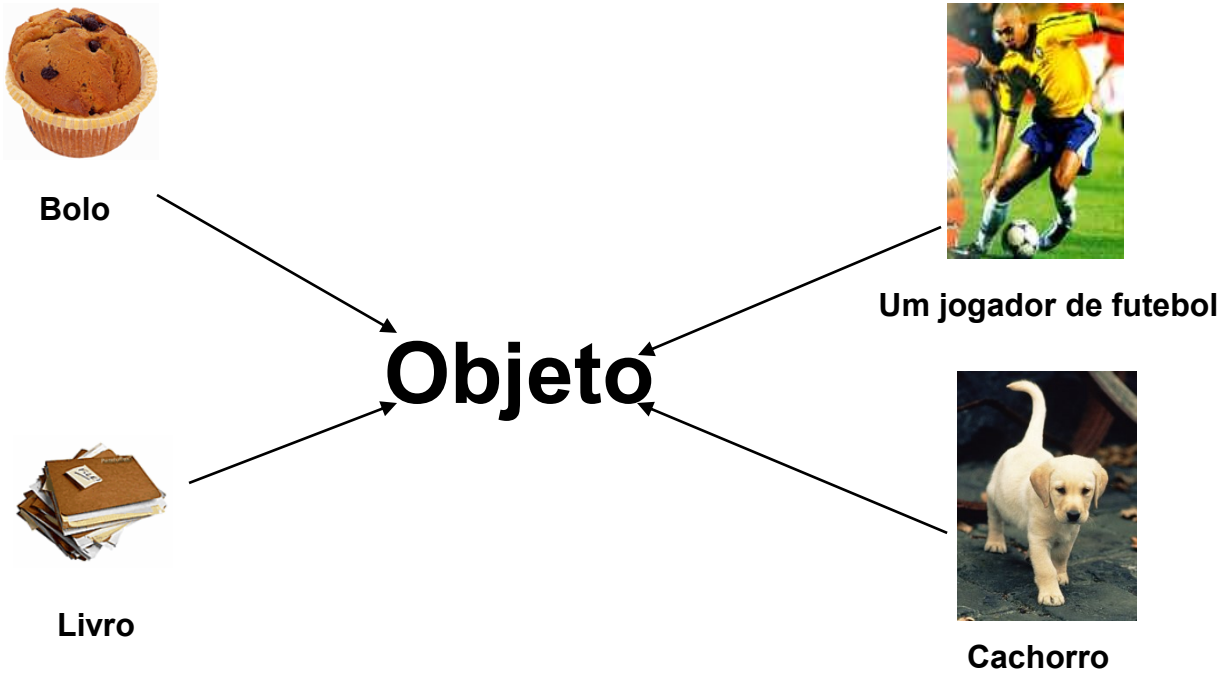
Estruturado:
Ênfase nos procedimentos, implementados em blocos estruturados, com comunicação entre procedimentos por passagem de dados;

Orientado a Objetos:
Dados e procedimentos fazem parte de um só elemento básico (objeto). Os elementos básicos comunicam-se entre si por mensagens e tem ênfase nos dados e no agrupamentos dos mesmos.

Objetos: O que são?

São quaisquer coisas na natureza que possuam propriedades (características) e comportamentos (operações).

Exemplos de Objetos:



Atributos

- São características presentes nos objetos;
- Os valores de todos os atributos é chamado de estado do objeto;
- Somente atributos que são de interesse do sistema devem ser considerados.

atributos

nome = Marina
cpf = 022.200.708-12
idade = 27

Para criarmos objetos em um sistema precisamos abstraí-lo criando uma classe que o represente.

Abstração em Orientação a Objetos

- **É o conjunto de operações realizadas com a finalidade de representar objetos do mundo real em forma de classes.**

- **As classes não devem incluir todos os atributos e nem todas as ações (métodos) dos objetos do mundo real, somente o que for pertinente ao papel a ser desempenhado dentro do programa.**

Classe - Estrutura de um objeto

Um objeto tem dados e comportamento.

Atributos

Métodos

nome = Marina
cpf = 022.200.708-12
idade = 27

corre
dorme

nome = Felipe
cpf = 039.217.908-22
idade = 42

corre
dorme

Classe

- As **Classes** e os **Objetos** possuem uma relação de dependência, pois não existe objeto sem a classe;

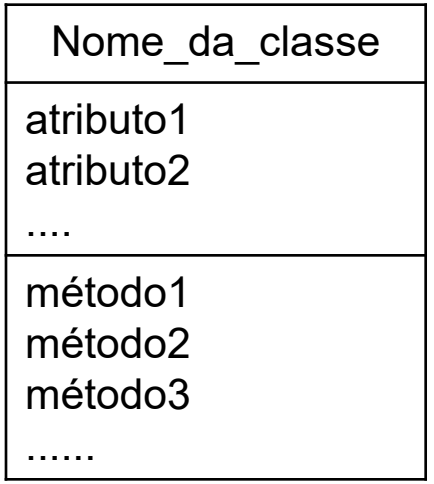
- A **Classe** representa a **abstração das características comuns** mais relevantes (atributos e métodos) de um conjunto de objetos. A classe passa a ser um molde para se criar objetos do mesmo tipo.

Classe

"Uma classe é uma entidade que descreve um conjunto de objetos com propriedades (atributos) e comportamentos (métodos) semelhantes e com relacionamentos comuns com outros objetos"

As classes são as partes mais importantes de qualquer sistema orientado a objetos.

Essas classes podem incluir abstrações que são parte do *domínio* do problema. Graficamente, as classes são representadas por retângulos incluindo nome, atributos e métodos.
Devem receber nomes representativos.



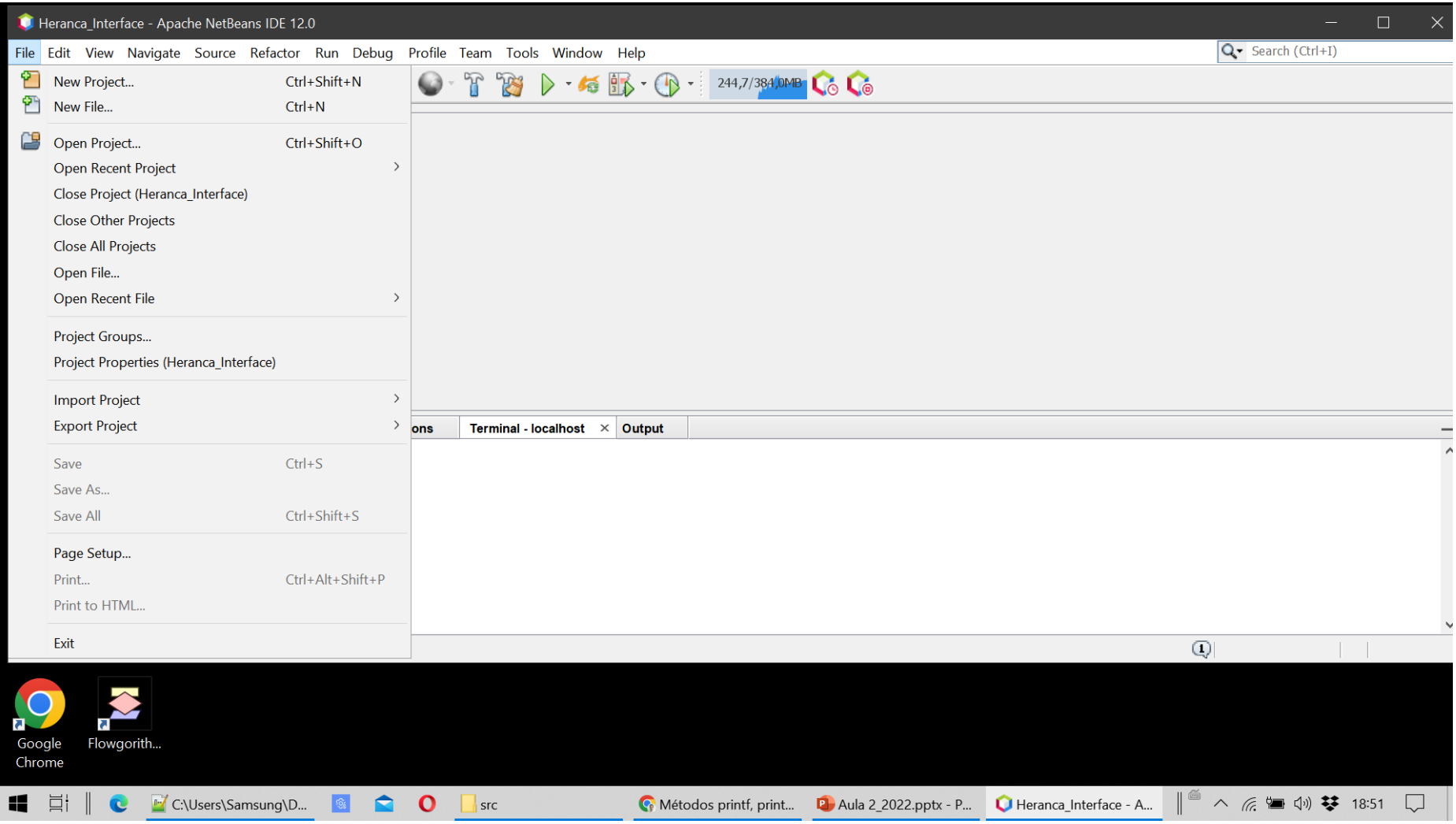
Representação em Diagrama de Classe UML (*Unified Modelling Language*)

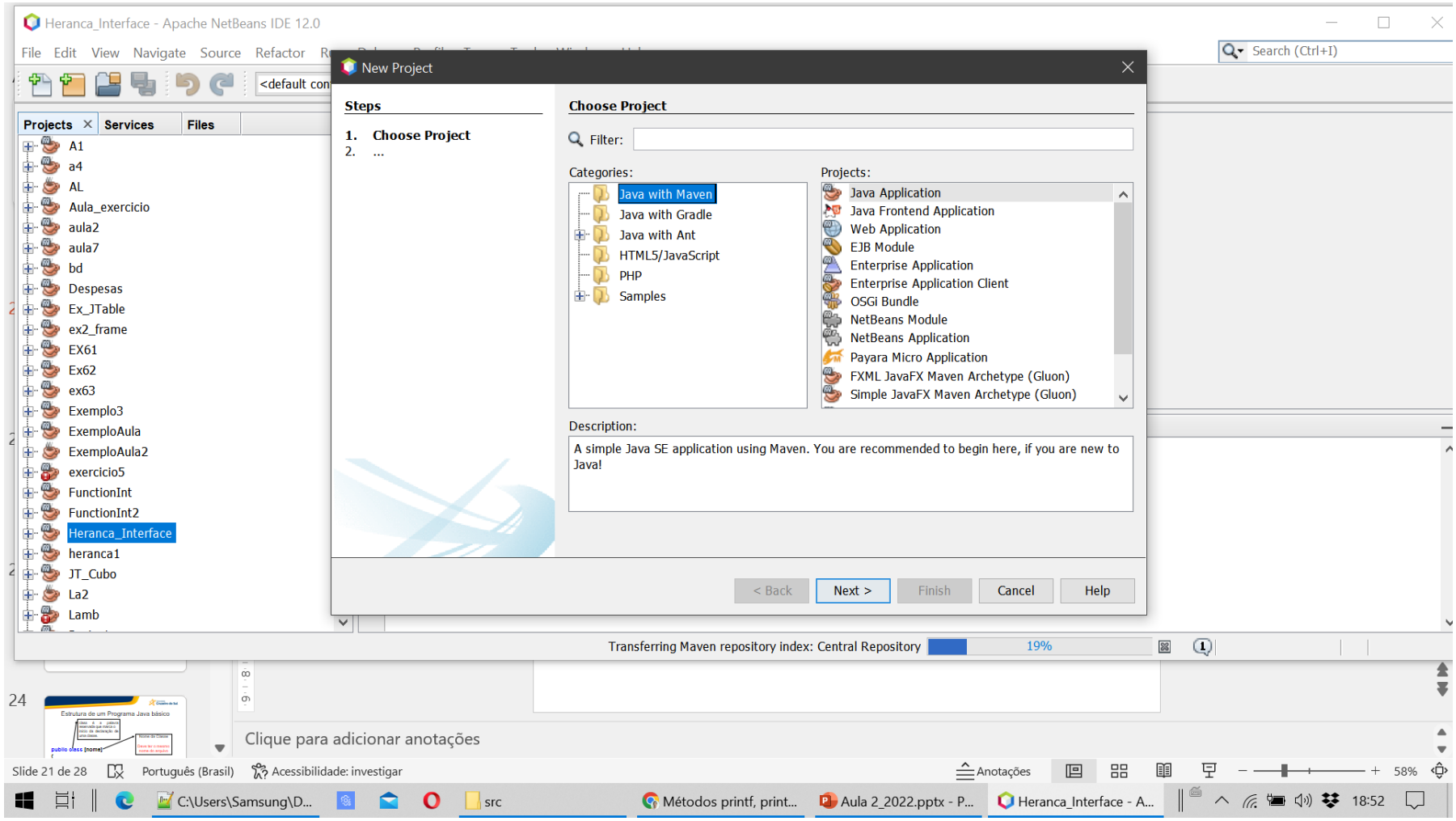
Exercício

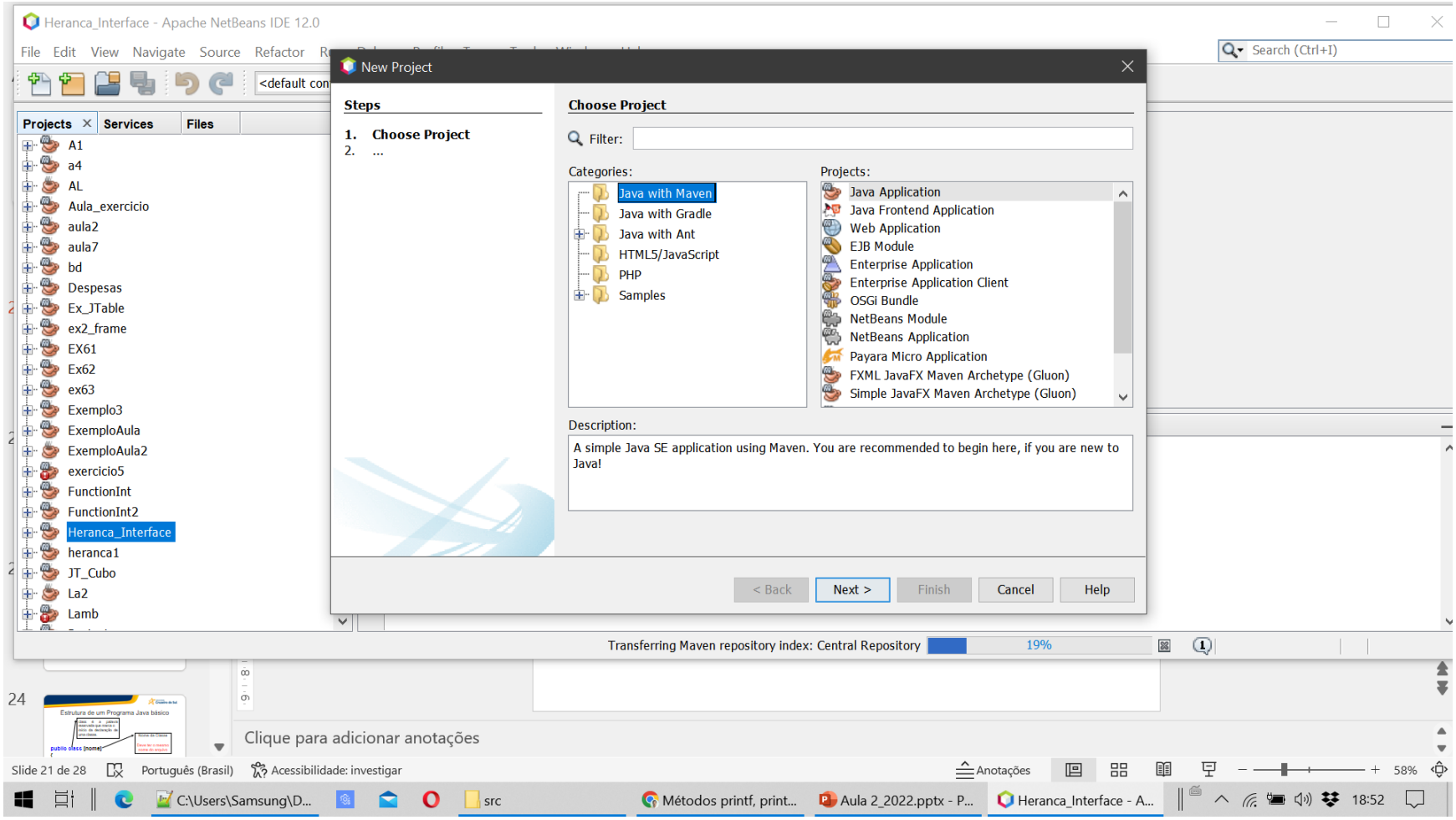
Descreva uma classe para representar:

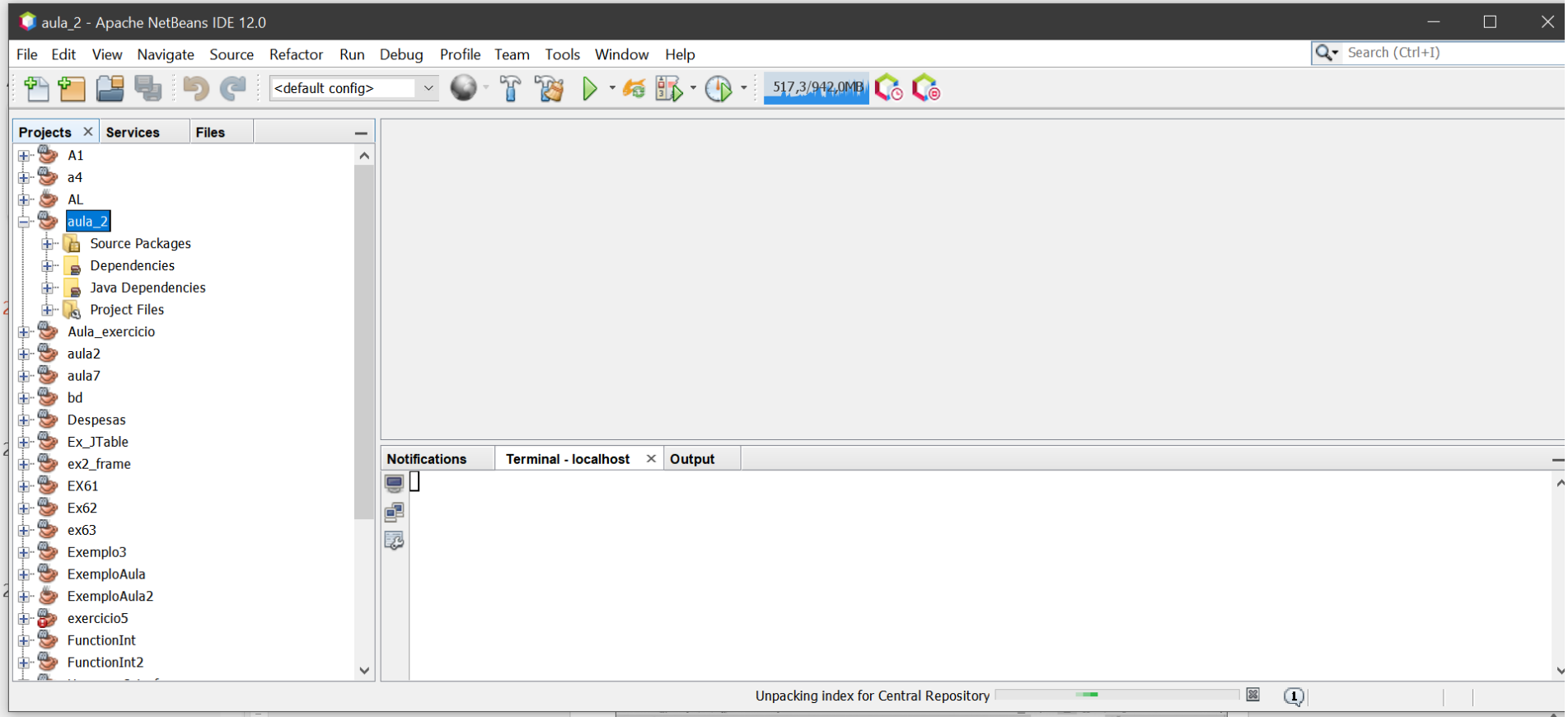
Janela

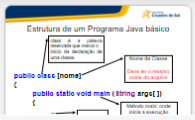
Quais atributos?
Quais métodos (comportamento)?


Heranca_Interface - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
New Project... Ctrl+Shift+N
New File... Ctrl+N
Open Project... Ctrl+Shift+O
Open Recent Project
Close Project (Heranca_Interface)
Close Other Projects
Close All Projects
Open File...
Open Recent File
Project Groups...
Project Properties (Heranca_Interface)
Import Project
Export Project
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Page Setup...
Print... Ctrl+Alt+Shift+P
Print to HTML...
Exit
244,7/384,0MB
Terminal - localhost
Output
Google Chrome
Flowgorith...
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
Heranca_Interface - A...
18:51

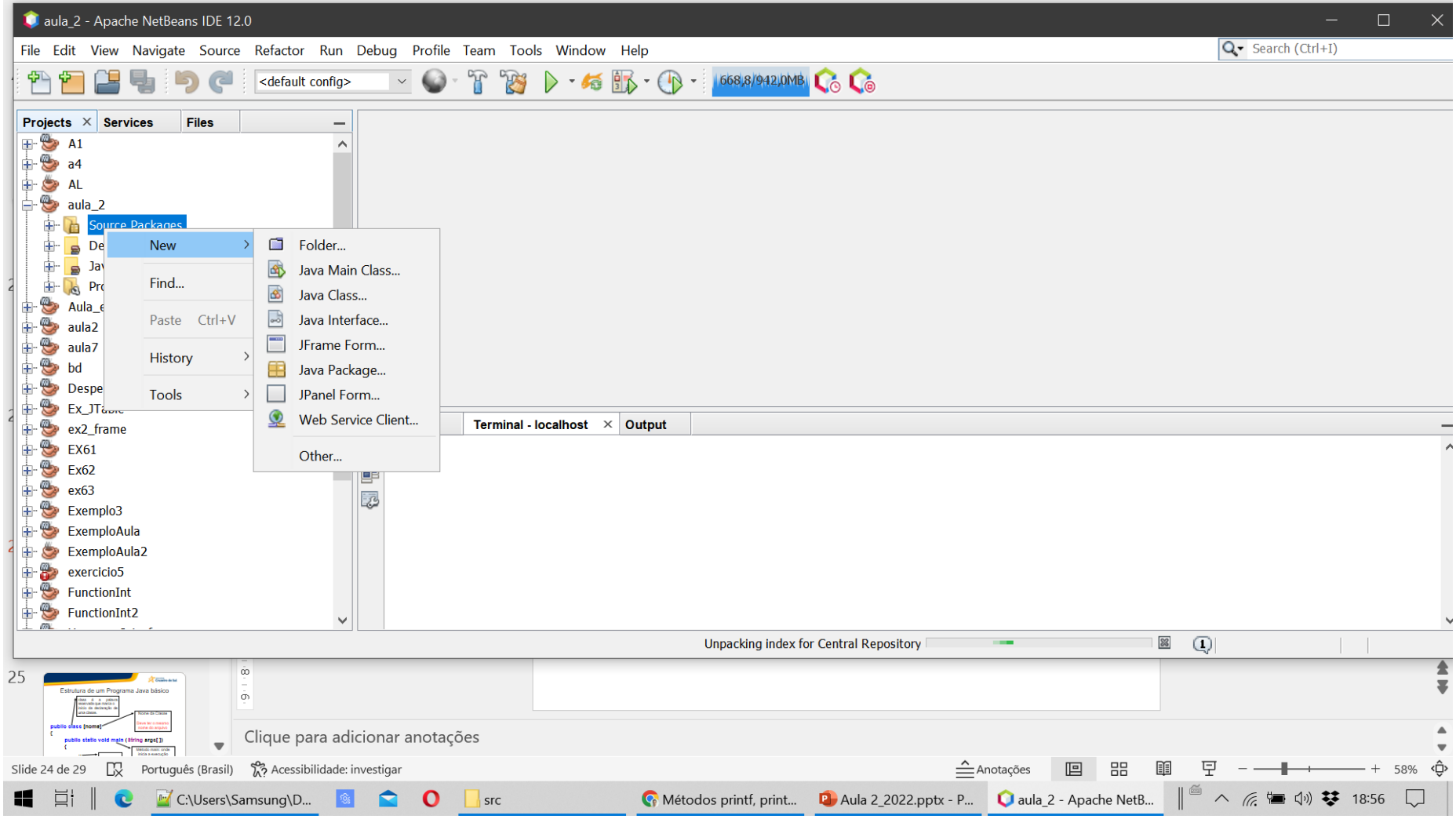

Heranca_Interface - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor
Search (Ctrl+I)
Projects
Services
Files
A1
a4
AL
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
La2
Lamb
New Project
Steps
1. Choose Project
2. ...
Choose Project
Filter:
Categories:
Java with Maven
Java with Gradle
Java with Ant
HTML5/JavaScript
PHP
Samples
Projects:
Java Application
Java Frontend Application
Web Application
EJB Module
Enterprise Application
Enterprise Application Client
OSGi Bundle
NetBeans Module
NetBeans Application
Payara Micro Application
FXML JavaFX Maven Archetype (Gluon)
Simple JavaFX Maven Archetype (Gluon)
Description:
A simple Java SE application using Maven. You are recommended to begin here, if you are new to Java!
< Back
Next >
Finish
Cancel
Help
Transferring Maven repository index: Central Repository
19%
24
Clique para adicionar anotações
Slide 21 de 28
Português (Brasil)
Acessibilidade: investigar
Anotações
58%
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
Heranca_Interface - A...
18:52

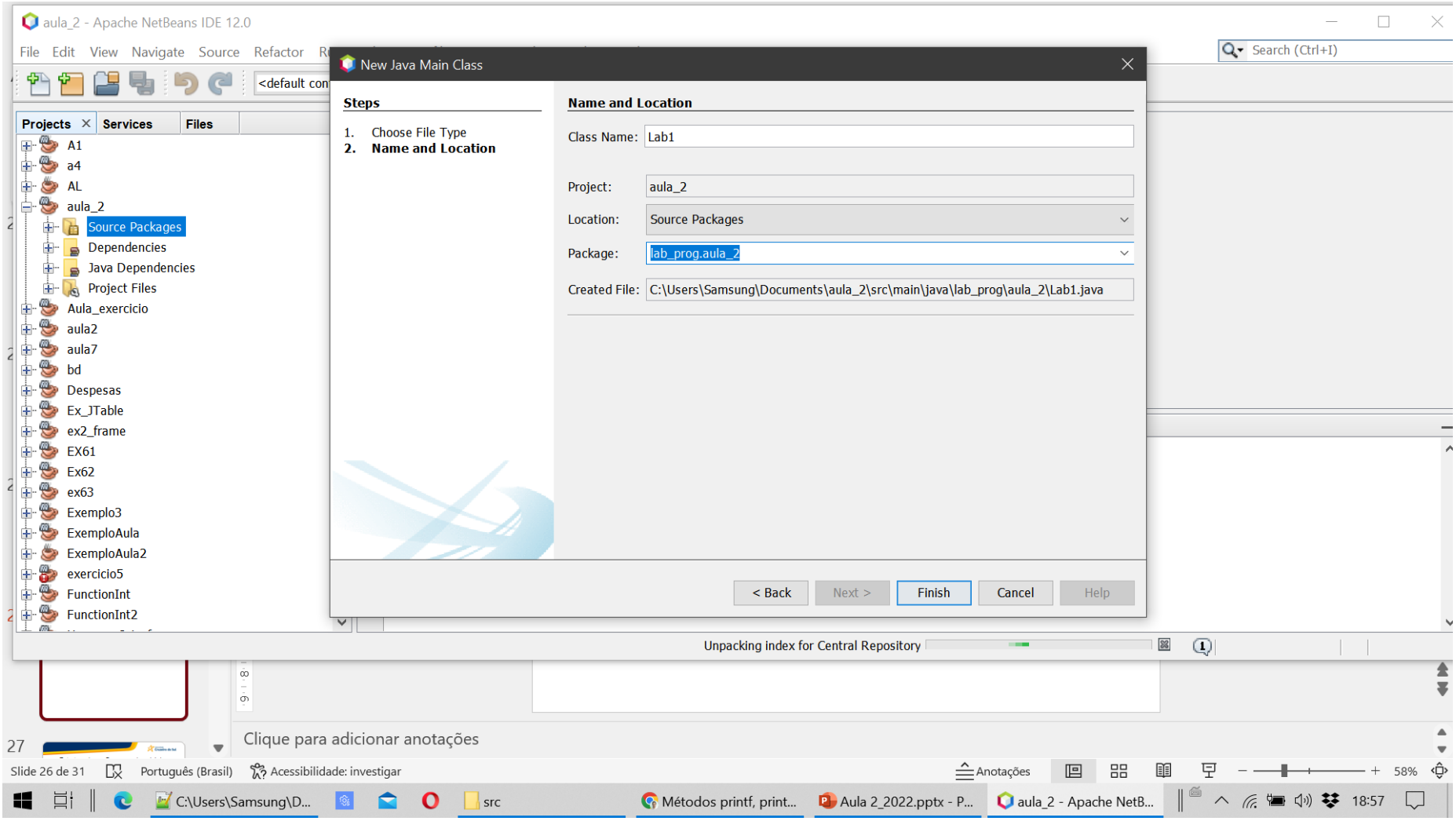

Heranca_Interface - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor
Search (Ctrl+I)
Projects
Services
Files
A1
a4
AL
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
La2
Lamb
New Project
Steps
1. Choose Project
2. ...
Choose Project
Filter:
Categories:
Java with Maven
Java with Gradle
Java with Ant
HTML5/JavaScript
PHP
Samples
Projects:
Java Application
Java Frontend Application
Web Application
EJB Module
Enterprise Application
Enterprise Application Client
OSGi Bundle
NetBeans Module
NetBeans Application
Payara Micro Application
FXML JavaFX Maven Archetype (Gluon)
Simple JavaFX Maven Archetype (Gluon)
Description:
A simple Java SE application using Maven. You are recommended to begin here, if you are new to Java!
< Back
Next >
Finish
Cancel
Help
Transferring Maven repository index: Central Repository
19%
Clique para adicionar anotações
Slide 21 de 28
Português (Brasil)
Acessibilidade: investigar
Anotações
58%
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
Heranca_Interface - A...
18:52

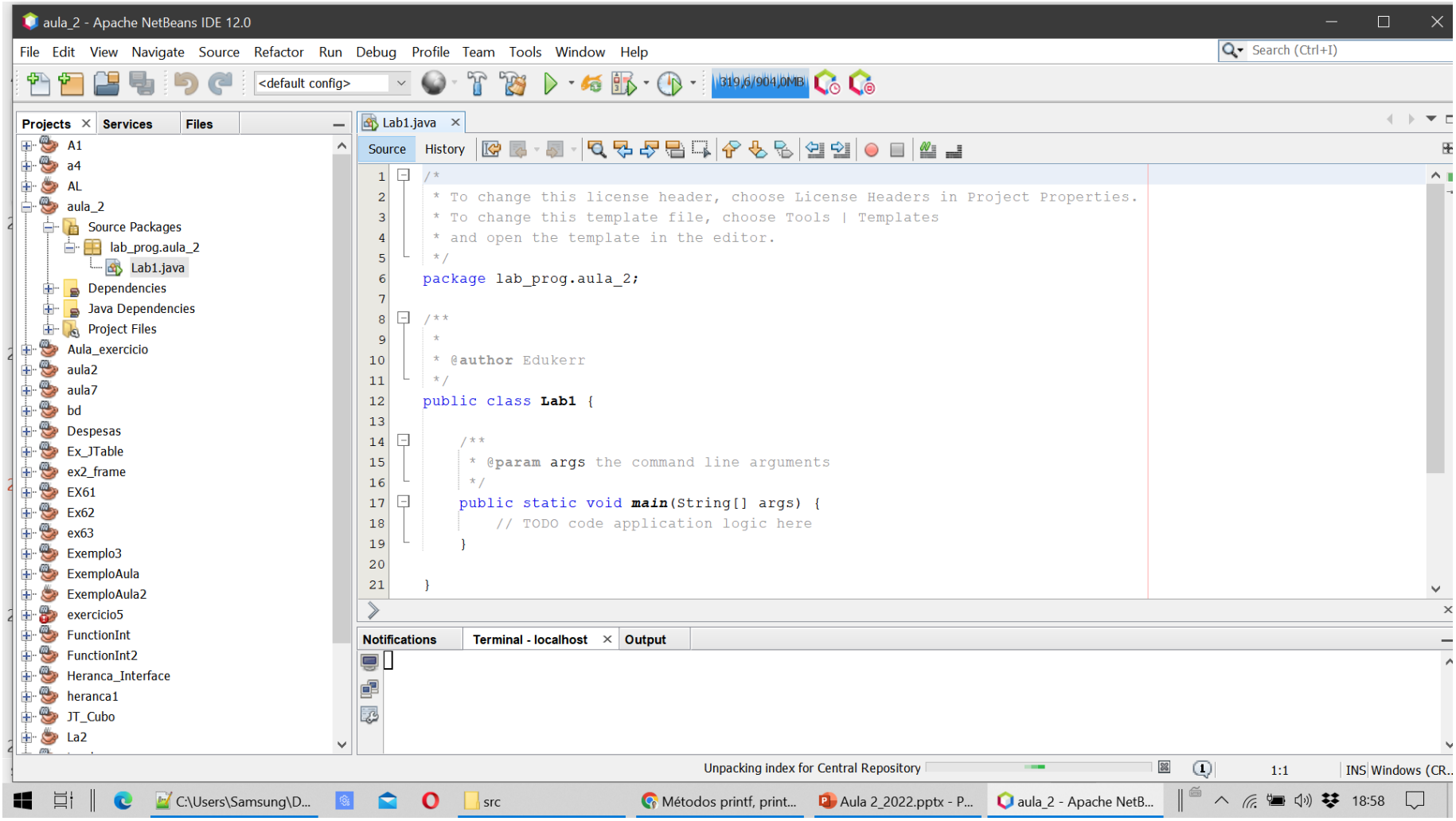


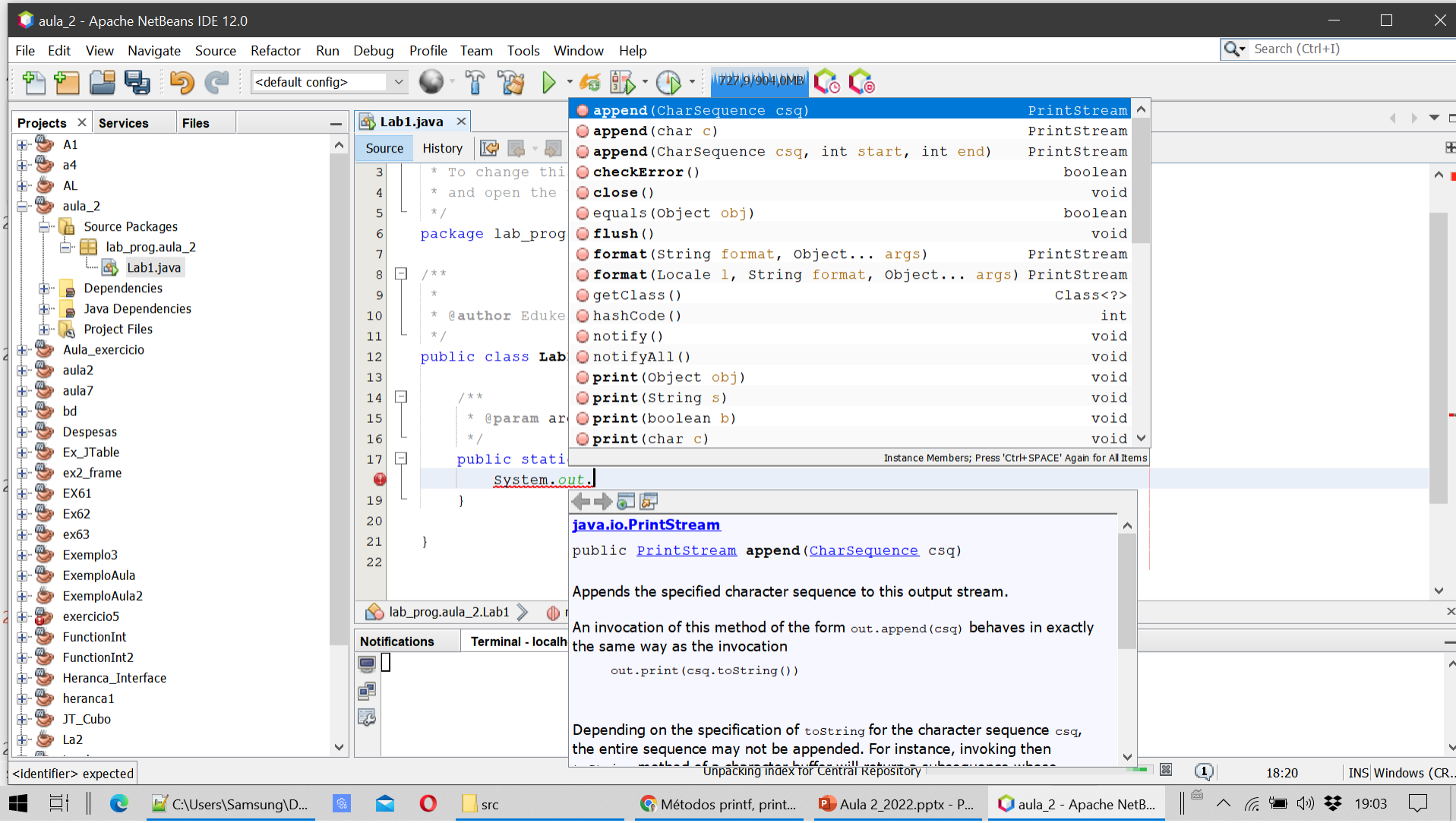


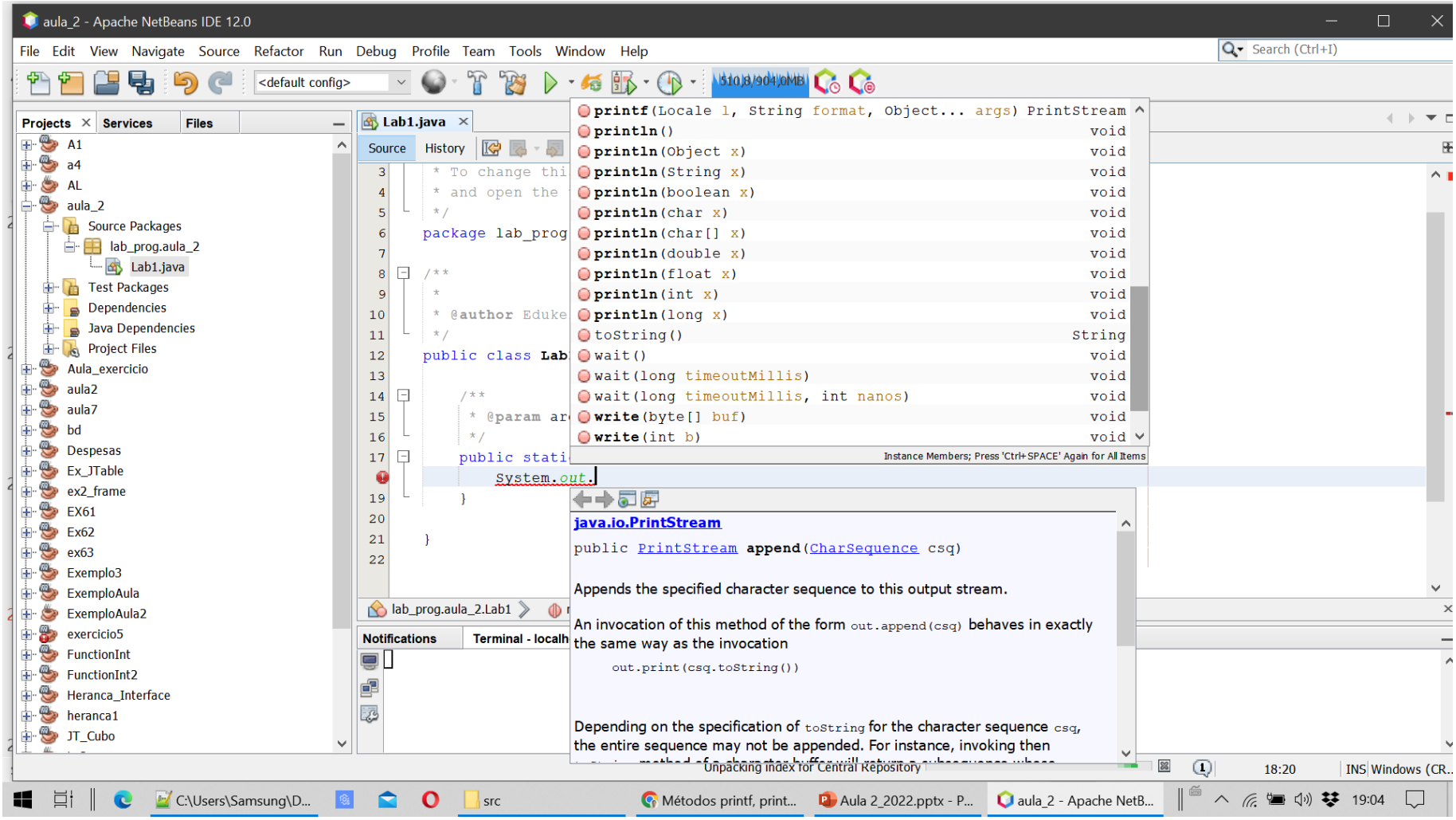
Clique para adicionar anotações

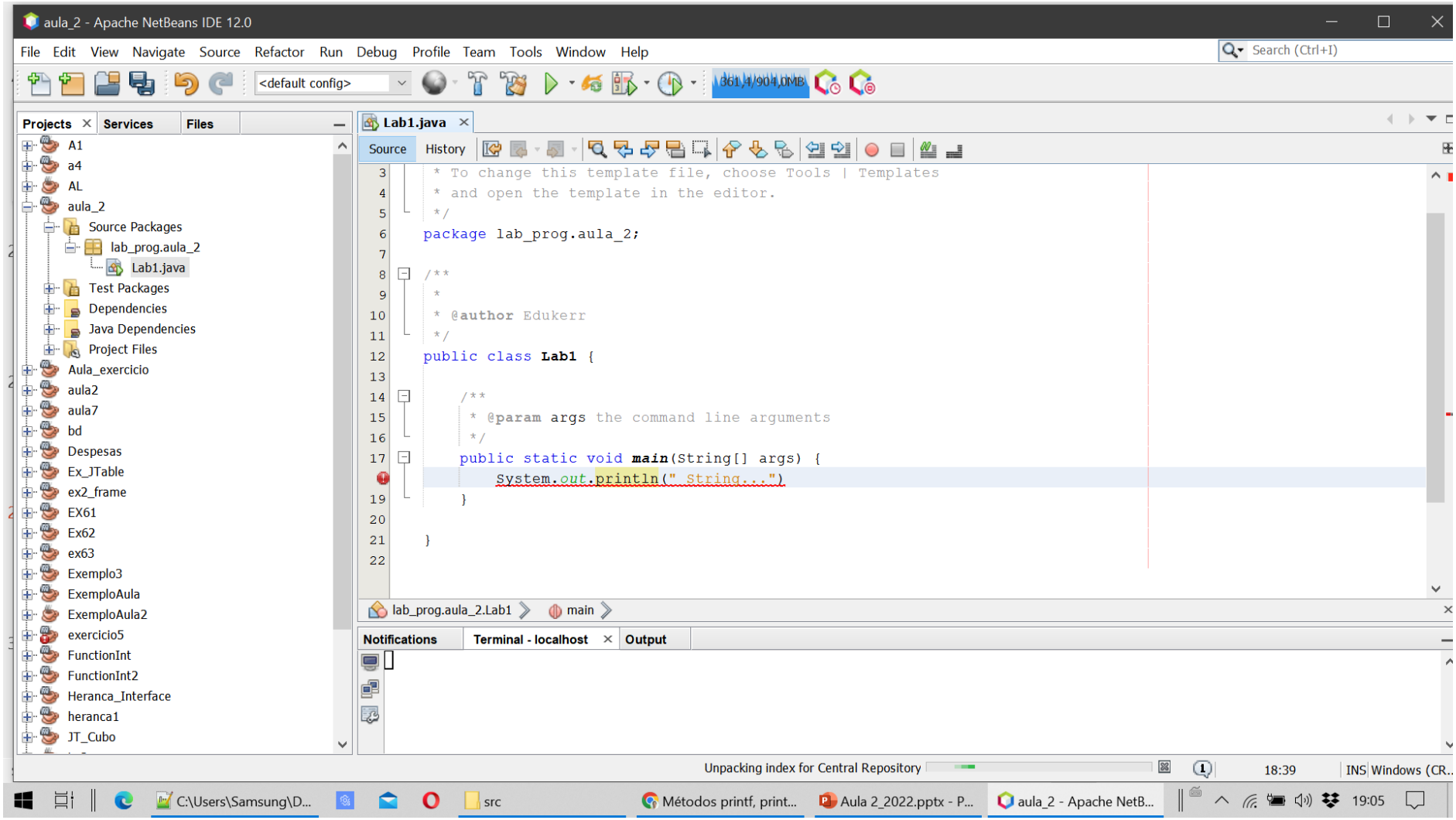

aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
668,8/942,0MB
Projects
Services
Files
A1
a4
AL
aula_2
Source Packages
New
Find...
Paste Ctrl+V
History
Tools
Folder...
Java Main Class...
Java Class...
Java Interface...
JFrame Form...
Java Package...
JPanel Form...
Web Service Client...
Other...
Aula_e
aula2
aula7
bd
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Terminal - localhost
Output
Unpacking index for Central Repository
Clique para adicionar anotações
Slide 24 de 29
Português (Brasil)
Acessibilidade: investigar
Anotações
58%
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
18:56


aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor
Search (Ctrl+I)
Projects Services Files
A1
a4
AL
aula_2
Source Packages
Dependencies
Java Dependencies
Project Files
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
New Java Main Class
Steps
1. Choose File Type
2. Name and Location
Name and Location
Class Name: Lab1
Project: aula_2
Location: Source Packages
Package: lab_prog.aula_2
Created File: C:\Users\Samsung\Documents\aula_2\src\main\java\lab_prog\aula_2\Lab1.java
< Back
Next >
Finish
Cancel
Help
Unpacking index for Central Repository
Clique para adicionar anotações
Slide 26 de 31
Português (Brasil)
Acessibilidade: investigar
Anotações
58%
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
18:57

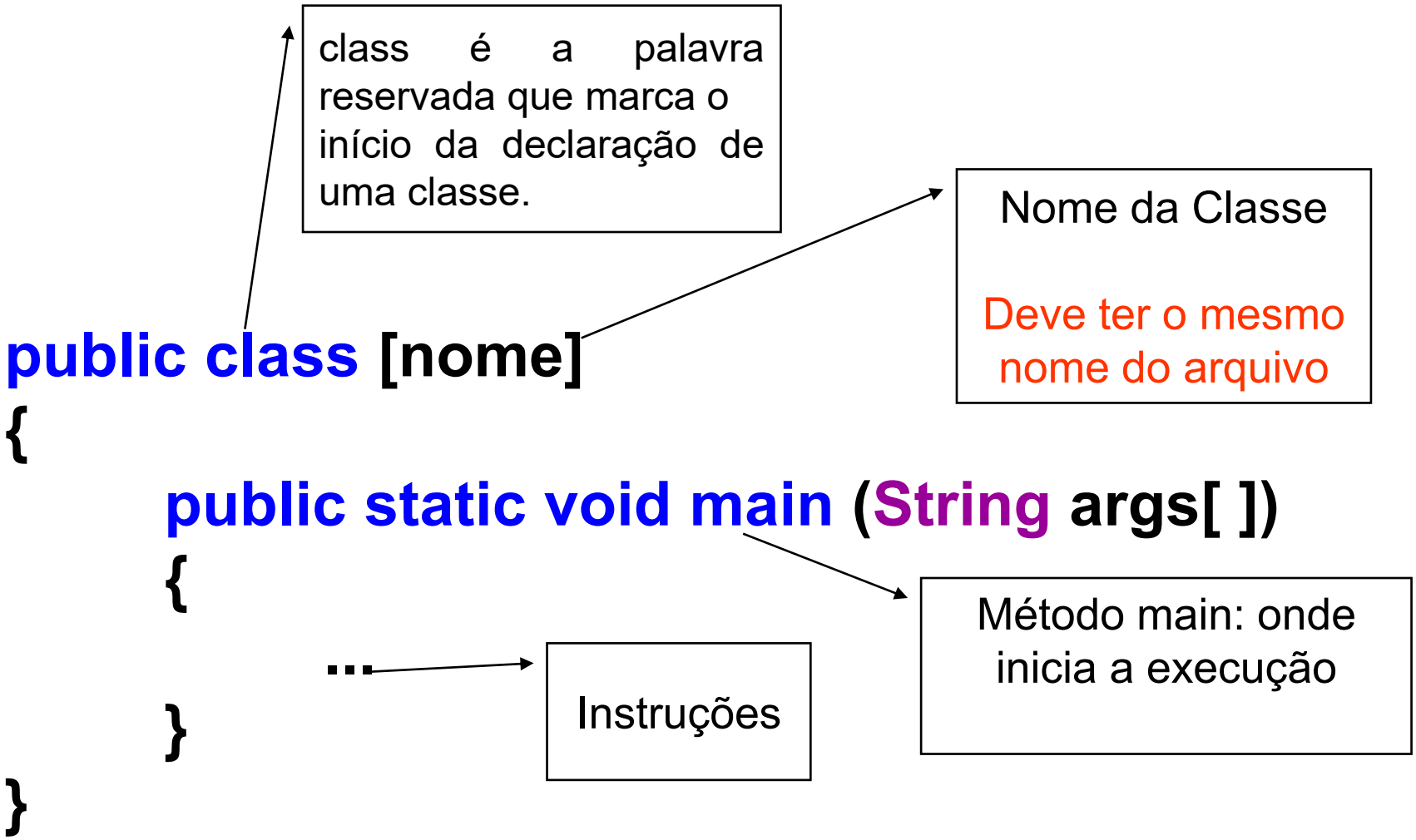

aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects Services Files
A1
a4
AL
aula_2
Source Packages
lab_prog.aula_2
Lab1.java
Dependencies
Java Dependencies
Project Files
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
La2
Lab1.java
Source History
1 /*
2 * To change this license header, choose License Headers in Project Properties.
3 * To change this template file, choose Tools | Templates
4 * and open the template in the editor.
5 */
6 package lab_prog.aula_2;
7
8 /**
9 *
10 * @author Edukerr
11 */
12 public class Lab1 {
13
14 /**
15 * @param args the command line arguments
16 */
17 public static void main(String[] args) {
18 // TODO code application logic here
19 }
20
21 }
Notifications
Terminal - localhost
Output
Unpacking index for Central Repository
1:1
INS Windows (CR...
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
18:58


aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
727,9/904,0MB
Projects Services Files
A1
a4
AL
aula_2
Source Packages
lab_prog.aula_2
Lab1.java
Dependencies
Java Dependencies
Project Files
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
La2
Lab1.java
Source History
3 * To change thi
4 * and open the
5 */
6 package lab_prog
7
8 /**
9 *
10 * @author Eduke
11 */
12 public class Lab
13
14 /**
15 * @param ar
16 */
17 public stati
System.out.
19 }
20
21 }
22
append(CharSequence csq) PrintStream
append(char c) PrintStream
append(CharSequence csq, int start, int end) PrintStream
checkError() boolean
close() void
equals(Object obj) boolean
flush() void
format(String format, Object... args) PrintStream
format(Locale l, String format, Object... args) PrintStream
getClass() Class<?>
hashCode() int
notify() void
notifyAll() void
print(Object obj) void
print(String s) void
print(boolean b) void
print(char c) void
Instance Members; Press 'Ctrl+SPACE' Again for All Items
java.io.PrintStream
public PrintStream append(CharSequence csq)
Appends the specified character sequence to this output stream.
An invocation of this method of the form out.append(csq) behaves in exactly the same way as the invocation
out.print(csq.toString())
Depending on the specification of toString for the character sequence csq, the entire sequence may not be appended. For instance, invoking then
lab_prog.aula_2.Lab1
Notifications
Terminal - localh
<identifier> expected
Unpacking index for Central Repository
18:20
INS Windows (CR...
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
19:03


aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Projects Services Files
A1
a4
AL
aula_2
Source Packages
lab_prog.aula_2
Lab1.java
Test Packages
Dependencies
Java Dependencies
Project Files
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
Lab1.java
Source History
* To change thi
* and open the
*/
package lab_prog
/**
*
* @author Eduke
*/
public class Lab
/**
* @param ar
*/
public stati
System.out.
}
}
printf(Locale l, String format, Object... args) PrintStream
println() void
println(Object x) void
println(String x) void
println(boolean x) void
println(char x) void
println(char[] x) void
println(double x) void
println(float x) void
println(int x) void
println(long x) void
toString() String
wait() void
wait(long timeoutMillis) void
wait(long timeoutMillis, int nanos) void
write(byte[] buf) void
write(int b) void
Instance Members; Press 'Ctrl+SPACE' Again for All Items
java.io.PrintStream
public PrintStream append(CharSequence csq)
Appends the specified character sequence to this output stream.
An invocation of this method of the form out.append(csq) behaves in exactly the same way as the invocation
out.print(csq.toString())
Depending on the specification of toString for the character sequence csq, the entire sequence may not be appended. For instance, invoking then
lab_prog.aula_2.Lab1
Notifications
Terminal - localh
Unpacking index for Central Repository
18:20
INS Windows (CR...
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
19:04


aula_2 - Apache NetBeans IDE 12.0
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
361,4/904,0MB
Projects
Services
Files
A1
a4
AL
aula_2
Source Packages
lab_prog.aula_2
Lab1.java
Test Packages
Dependencies
Java Dependencies
Project Files
Aula_exercicio
aula2
aula7
bd
Despesas
Ex_JTable
ex2_frame
EX61
Ex62
ex63
Exemplo3
ExemploAula
ExemploAula2
exercicio5
FunctionInt
FunctionInt2
Heranca_Interface
heranca1
JT_Cubo
Lab1.java
Source
History
3 * To change this template file, choose Tools | Templates
4 * and open the template in the editor.
5 */
6 package lab_prog.aula_2;
7
8 /**
9 *
10 * @author Edukerr
11 */
12 public class Lab1 {
13
14 /**
15 * @param args the command line arguments
16 */
17 public static void main(String[] args) {
System.out.println(" String...")
19 }
20
21 }
22
lab_prog.aula_2.Lab1
main
Notifications
Terminal - localhost
Output
Unpacking index for Central Repository
18:39
INS
Windows (CR...
C:\Users\Samsung\D...
src
Métodos printf, print...
Aula 2_2022.pptx - P...
aula_2 - Apache NetB...
19:05

Estrutura de um Programa Java básico



Essa estrutura estará em todos os programa desenvolvidos em java

Comentando/Documentando em Java

Qualquer informação especificada entre os caracteres de comentário será ignorada pelo compilador. Os tipos de comentários são os seguintes:

- Comentário de bloco /* texto */ Todo o texto é ignorado. Este tipo de comentário pode ocupar várias linhas.
- Comentário de Linha // texto Todo o texto depois de // até o final da linha será ignorado.
- Comentário do JavaDoc /** usado para documentação */

Este ultimo é um tipo especial de comentário, funciona com diretiva para a geração automática da documentação das classes, utilizando o JAVADOC.

Comentando/Documentando em Java

A regras que tratam dos comentários são:

- Comentários não podem ser aninhados
- Não podem ocorrer dentro de Strings ou literais
- As notações /* e */ não tem significado especial nos comentários iniciados por //.
- A notação // não tem significado especial dentro dos comentários de bloco (/* e /**.

Um pequeno exemplo

```
/*****************************************************
 Autor: Nome do Aluno
 RGM: rgm do aluno
*****************************************************/
public class Exemplo1 {
    public static void main(String[] args) {
        System.out.println("Meu primeiro código");
        System.out.println("Impressão simples");
    }
}
```

Comentários no código:

- Uma única linha: /*
- Várias linhas: Inicia-se com o delimitador /* e termina com */.

É ignorado pelo compilador.

Observações

- Termine todas as linhas de instrução com **;**
- Sempre compile o programa antes de executar
- Quando ocorrer um erro de compilação, dê um duplo clique sobre a mensagem de erro para destacar o comando errado no programa e verifique também a linha anterior, que pode ser a responsável pelo erro, especialmente se faltar o ;
- Como convenção, todos os nomes de classe (identificador) em Java iniciam com uma letra maiúscula.
- Use comentários, iniciados por // ou /* ... */
- Java é Case Sensitive, logo diferencia letras maiúsculas de letras minúsculas, por exemplo: Exemplo1 é diferente de exemplo1.