

# Exercícios sobre Polimorfismo

*Programação de Sistemas II - 1º semestre de 2022*

Prof. Joaquim Pessôa Filho

Prof. Tomaz Mikio Sasaki

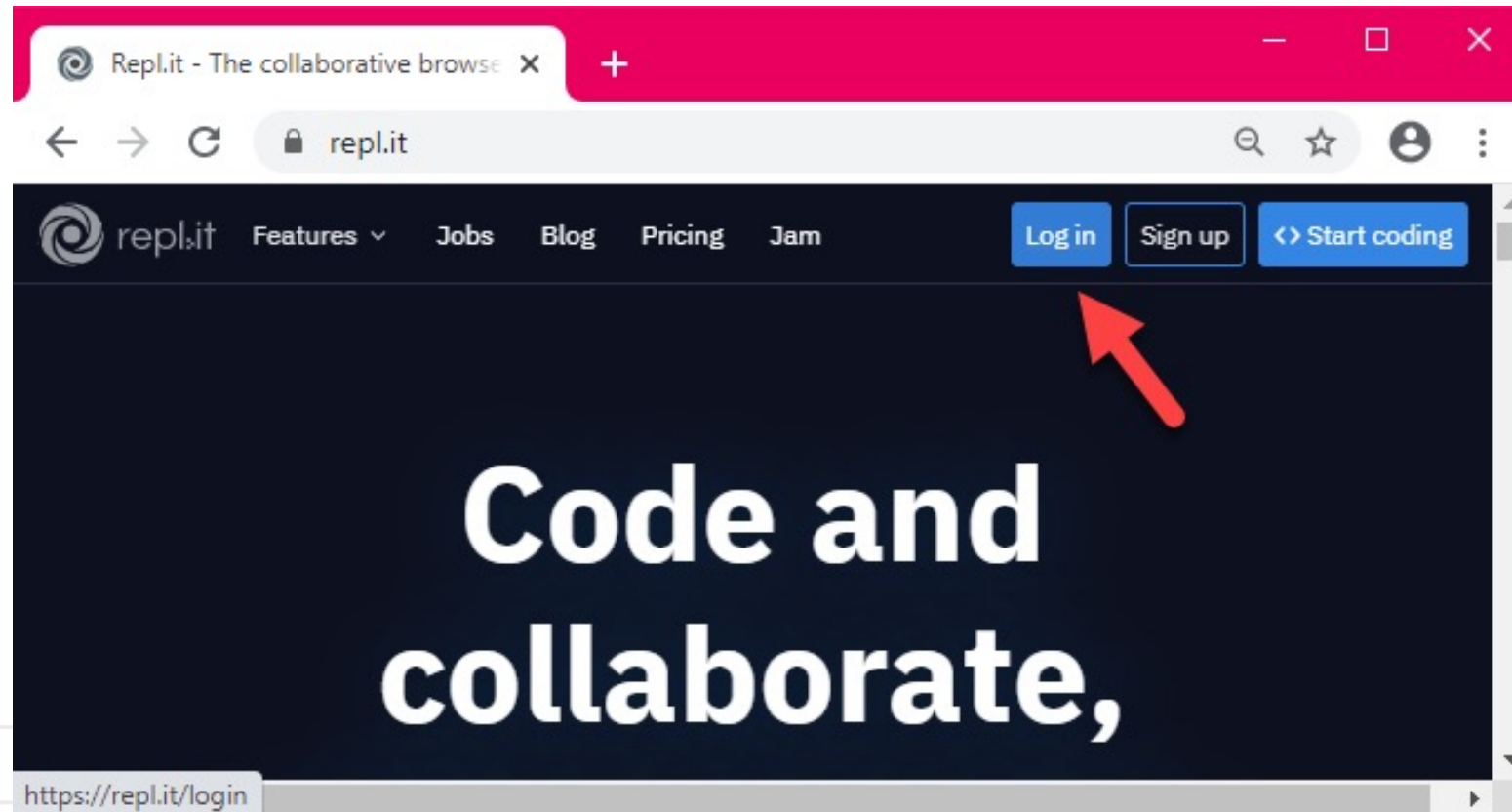


# Ambiente de desenvolvimento

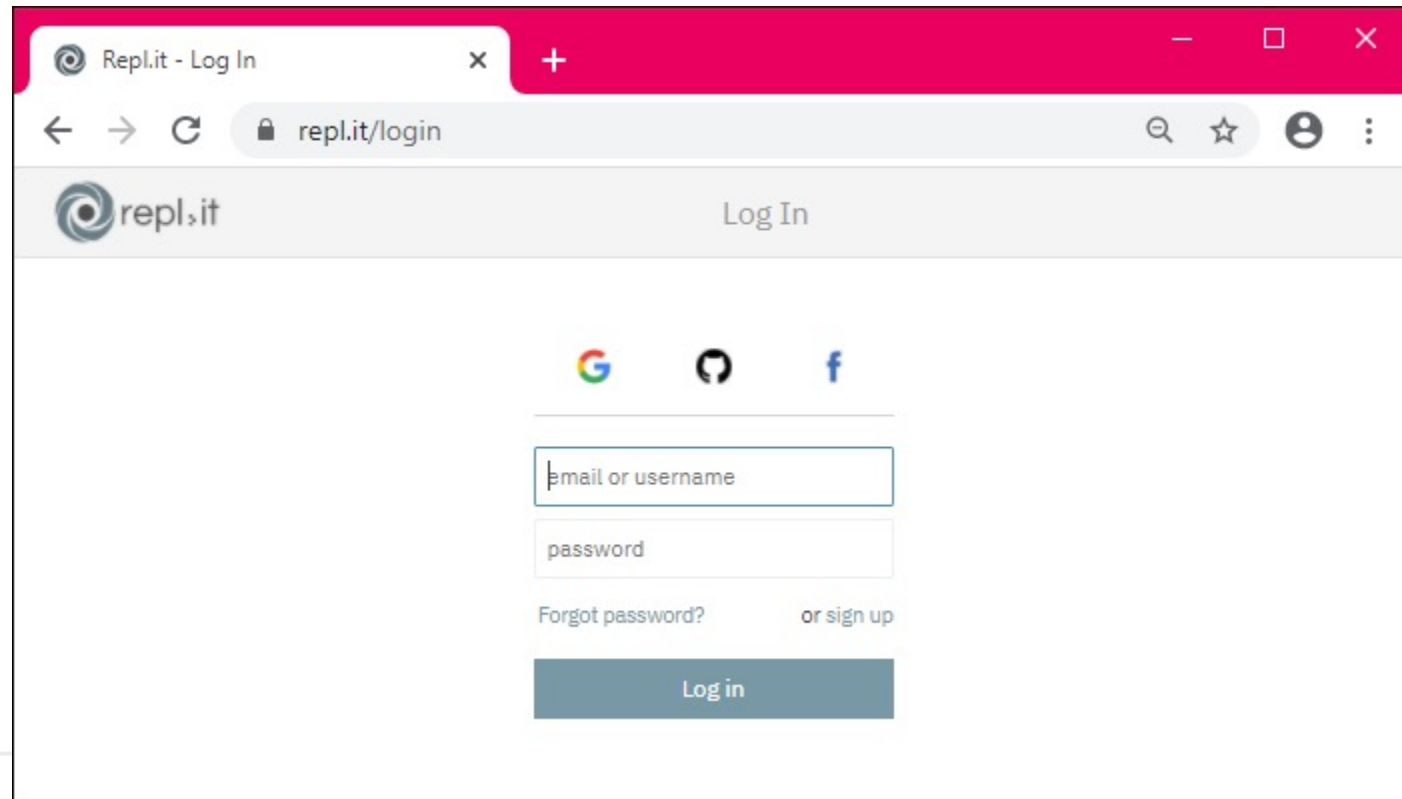
---

- Estas instruções mostram como realizar o exercício utilizando o ambiente online repl.it (<https://repl.it>).
- Caso prefira utilizar outro ambiente de desenvolvimento, você pode obter o código inicial em <https://github.com/joaquimp/ps2-LAB1b>

Acesse <https://repl.it> e efetue o login na aplicação.



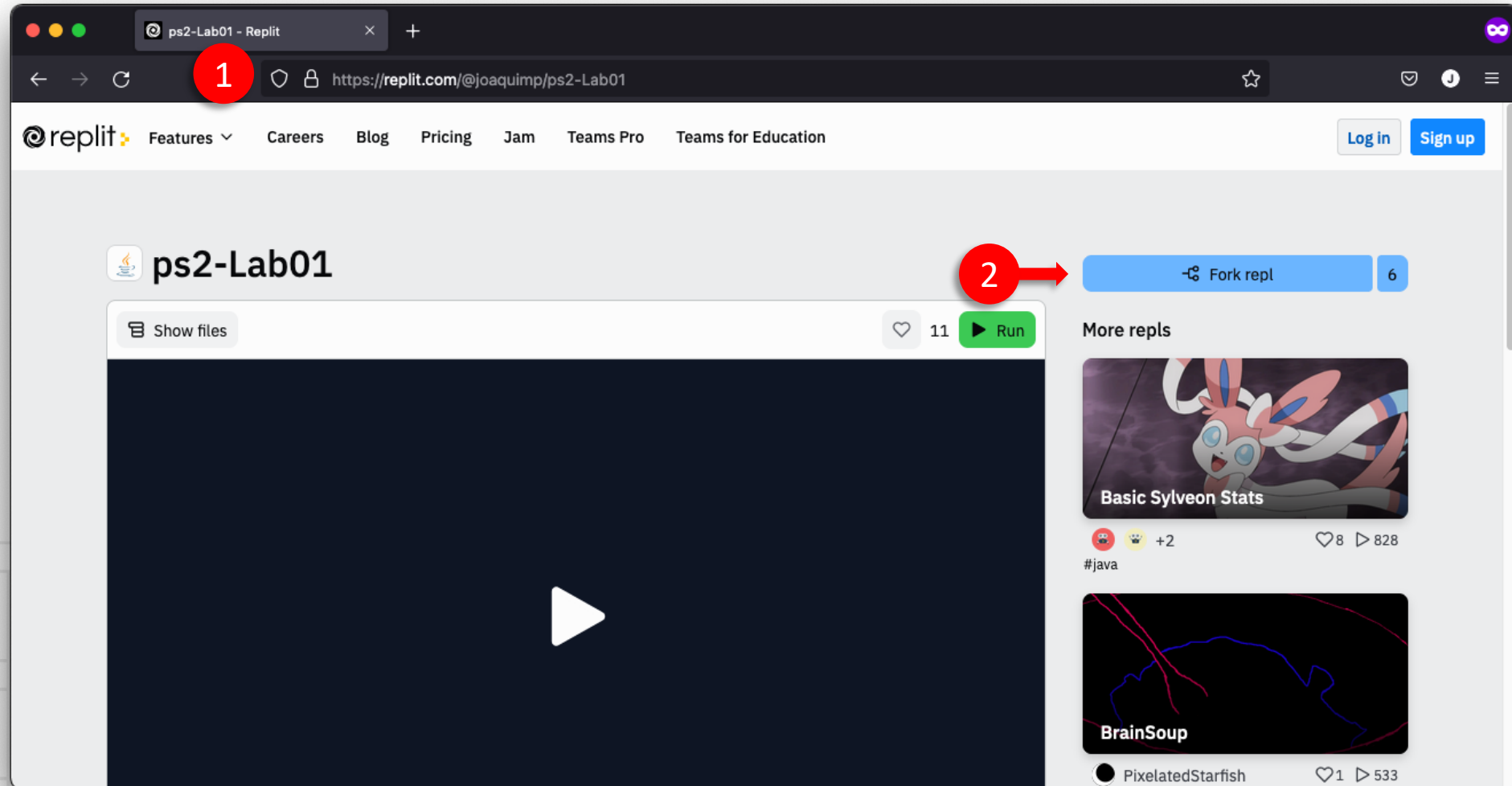
Efetue o login utilizando uma das contas sugeridas (Google, GitHub, Facebook), ou crie uma nova **conta gratuita** na aplicação.



The image shows a web browser window with the title "Repl.it - Log In". The address bar displays "repl.it/login". The page header features the Repl.it logo and the text "Log In". Below the header, there are three social login options: Google, GitHub, and Facebook. Underneath these, there are two input fields: "email or username" and "password". Below the password field, there are two links: "Forgot password?" and "or sign up". At the bottom, there is a blue "Log in" button.

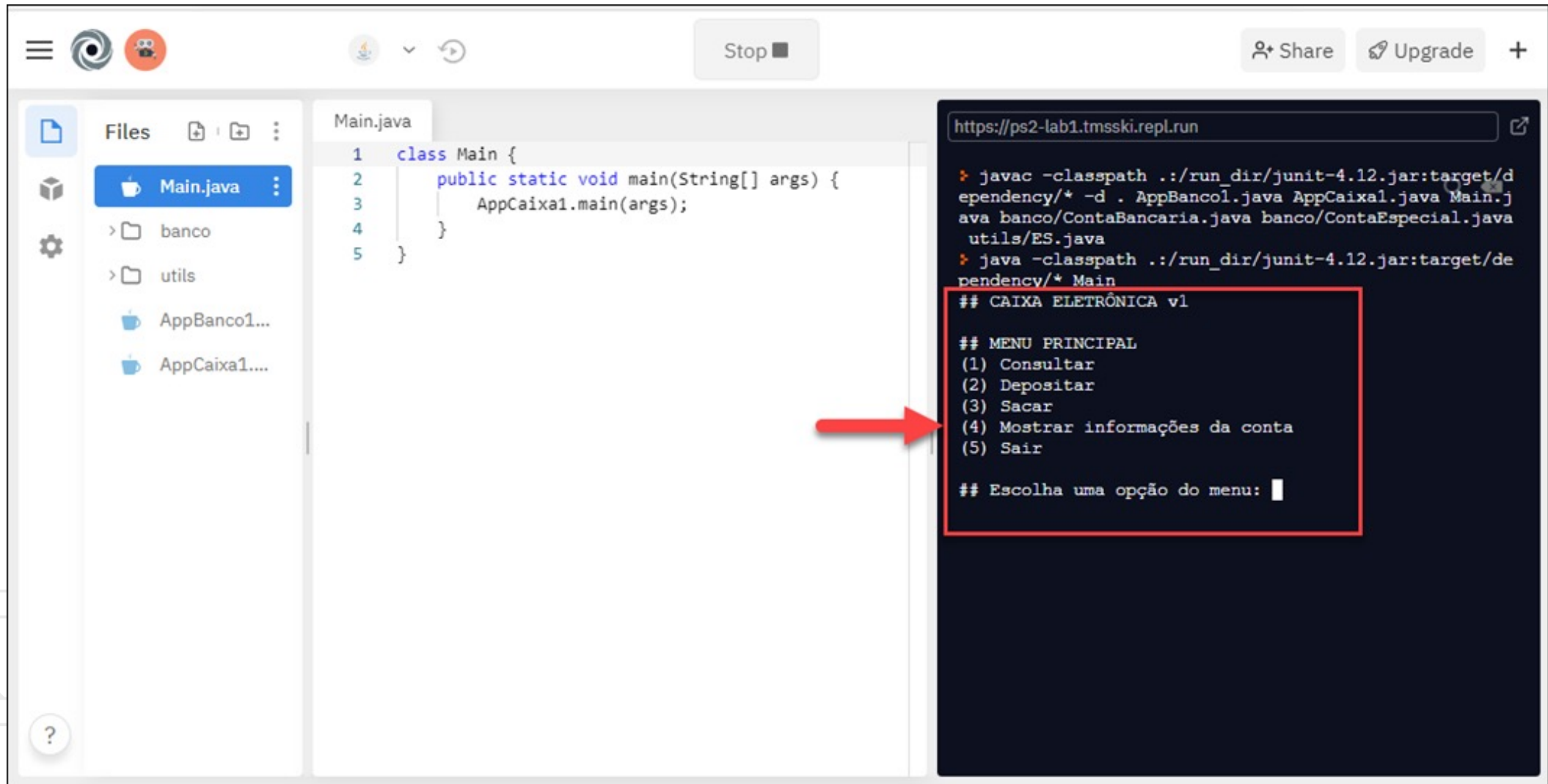
Após efetuar o login, acesse a URL <https://replit.com/@joaquimp/ps2-Lab01> e pressione o botão **Fork**.

Isto criará uma cópia do projeto para que você possa alterar.



# Exercício 1

Pressione o botão **Run** e verifique o funcionamento de cada opção oferecida pelo programa **AppCaixa1**.



```
1 class Main {
2     public static void main(String[] args) {
3         AppCaixa1.main(args);
4     }
5 }
```

```
https://ps2-lab1.tmsski.repl.run
> javac -classpath ../run_dir/junit-4.12.jar:target/de
ependency/* -d . AppBanco1.java AppCaixa1.java Main.j
ava banco/ContaBancaria.java banco/ContaEspecial.java
utils/ES.java
> java -classpath ../run_dir/junit-4.12.jar:target/de
pendency/* Main
## CAIXA ELETRÔNICA v1

## MENU PRINCIPAL
(1) Consultar
(2) Depositar
(3) Sacar
(4) Mostrar informações da conta
(5) Sair

## Escolha uma opção do menu: |
```

Examine o código da classe **ContaBancaria**. Note que é semelhante à classe que utilizamos nos exemplos da última aula.

The screenshot displays a Java IDE interface. On the left, a file explorer shows a project structure with files like Main.java, ContaBancaria.java, and ContaEspecial.java. The main editor window shows the source code for `banco/ContaBancaria.java`. The code is as follows:

```
1 package banco;
2 public class ContaBancaria {
3     private String nomeCorrentista;
4     protected double saldo;
5     public ContaBancaria(String nome, double saldo) {
6         nomeCorrentista = nome;
7         this.saldo = saldo;
8     }
9     public double consultar() {
10         return saldo;
11     }
12     public void depositar(double valor) {
13         saldo += valor;
14     }
15     public boolean sacar(double valor) {
16         if (valor <= saldo) {
17             saldo -= valor;
18             return true;
19         }
20         return false;
21     }
22     public String getNomeCorrentista() {
23         return nomeCorrentista;
24     }
25     public String toString() {
26         return "Conta de " + nomeCorrentista;
```

On the right side, a terminal window shows the output of the Java runtime environment:

```
https://ps2-lab1.tmsski.repl.run
OpenJDK Runtime Environment (build
11.0.6+10-post-Ubuntu-18.04.1)
>
```



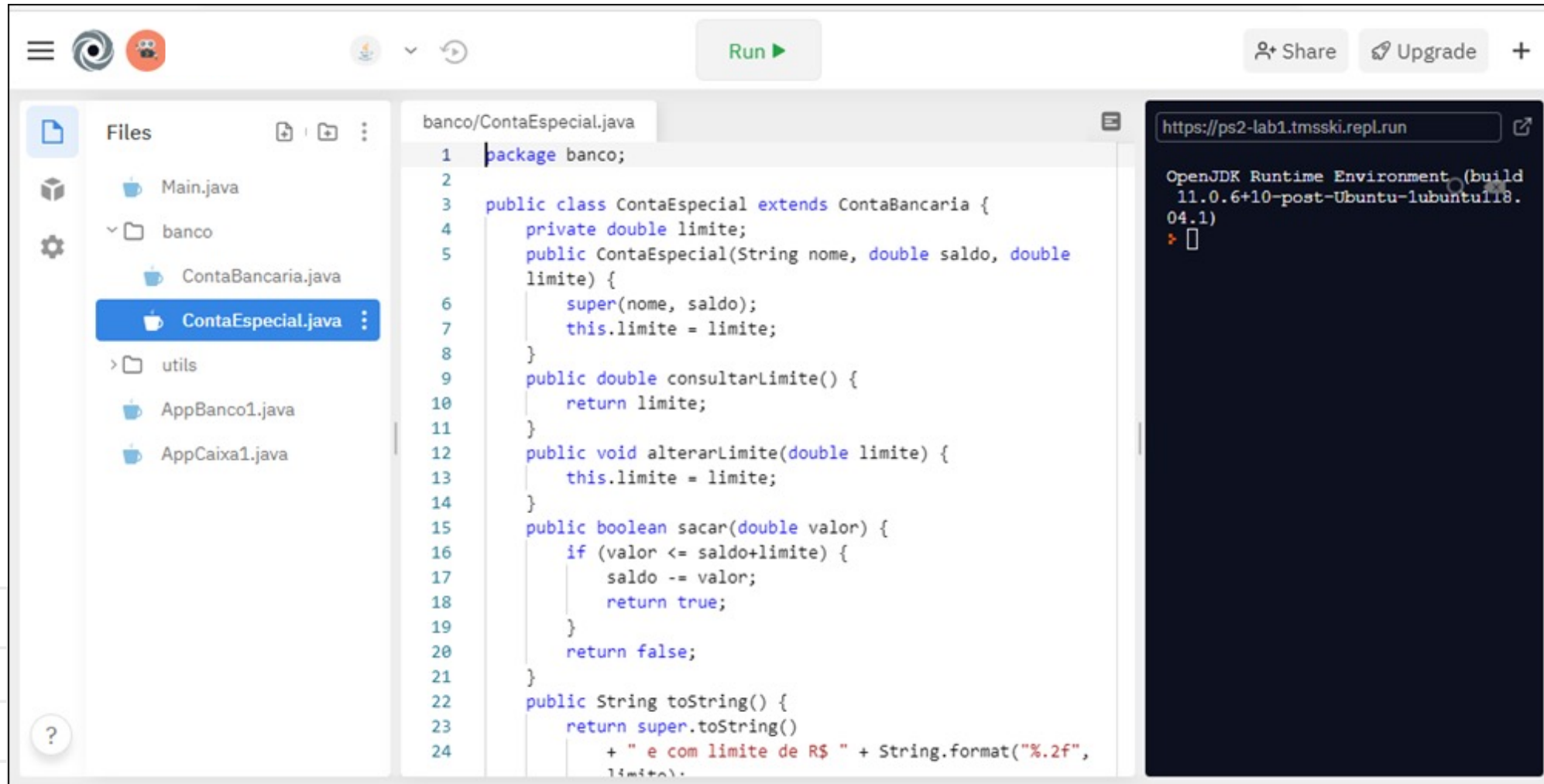
Examine o código da classe **AppCaixa1**. Entenda como cada opção do programa foi implementada.

The screenshot displays a Java IDE interface. On the left, a file explorer shows a project structure with files: Main.java, ContaBancaria.java, ContaEspecial.java, AppBanco1.java, and AppCaixa1.java (selected). The main editor shows the code for AppCaixa1.java, which includes imports for utils.ES.\* and banco.ContaBancaria, a static ContaBancaria object named conta, and a main method with a menu-driven loop.

```
1 import static utils.ES.*;
2 import banco.ContaBancaria;
3
4 public class AppCaixa1 {
5     private static ContaBancaria conta = new ContaBancaria
6         ("Bob", 100.00);
7
8     public static void main(String[] args) {
9         print("## CAIXA ELETRÔNICA v1");
10        boolean sair = false;
11        while (!sair) {
12            print("\n## MENU PRINCIPAL");
13            print("(1) Consultar");
14            print("(2) Depositar");
15            print("(3) Sacar");
16            print("(4) Mostrar informações da conta");
17            print("(5) Sair");
18            int opcao = inputInt("\n## Escolha uma opção do
19                menu:");
20            if (opcao == 1) {
21                opcaoConsultar();
22            }
23            else if (opcao == 2) {
24                opcaoDepositar();
25            }
26            else if (opcao == 3) {
```

On the right, a terminal window shows the output of the program, indicating the OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1) and a prompt character.

Examine o código da classe **ContaEspecial**. Note que é semelhante à classe-filha que utilizamos nos exemplos da última aula.



```
1 package banco;
2
3 public class ContaEspecial extends ContaBancaria {
4     private double limite;
5     public ContaEspecial(String nome, double saldo, double
6         limite) {
7         super(nome, saldo);
8         this.limite = limite;
9     }
10    public double consultarLimite() {
11        return limite;
12    }
13    public void alterarLimite(double limite) {
14        this.limite = limite;
15    }
16    public boolean sacar(double valor) {
17        if (valor <= saldo+limite) {
18            saldo -= valor;
19            return true;
20        }
21        return false;
22    }
23    public String toString() {
24        return super.toString()
25            + " e com limite de R$ " + String.format("%.2f",
26                limite);
```

OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-1ubuntu118.04.1)

# Exercício 1

Usando como exemplo a classe **AppCaixa1**, desenvolva um novo programa (em uma classe chamada **AppCaixa2**) para gerenciar uma conta especial.

Seu programa deverá gerenciar uma conta especial do correntista "Carlos", criada com saldo inicial de R\$ 250,00 e limite de R\$ 150,00.

Seu programa deverá oferecer as seguintes opções no menu principal:

```
## MENU PRINCIPAL
(1) Consultar saldo
(2) Depositar
(3) Sacar
(4) Consultar limite
(5) Alterar limite
(6) Mostrar informações da conta
(7) Sair

## Escolha uma opção do menu:
```

Caso o usuário escolha a opção "Consultar limite", seu programa deverá apresentar o valor do limite atual da conta:

```
## Limite da conta: R$ 150.0
```

Caso o usuário escolha a opção "Alterar limite", seu programa deverá solicitar o valor do novo limite:

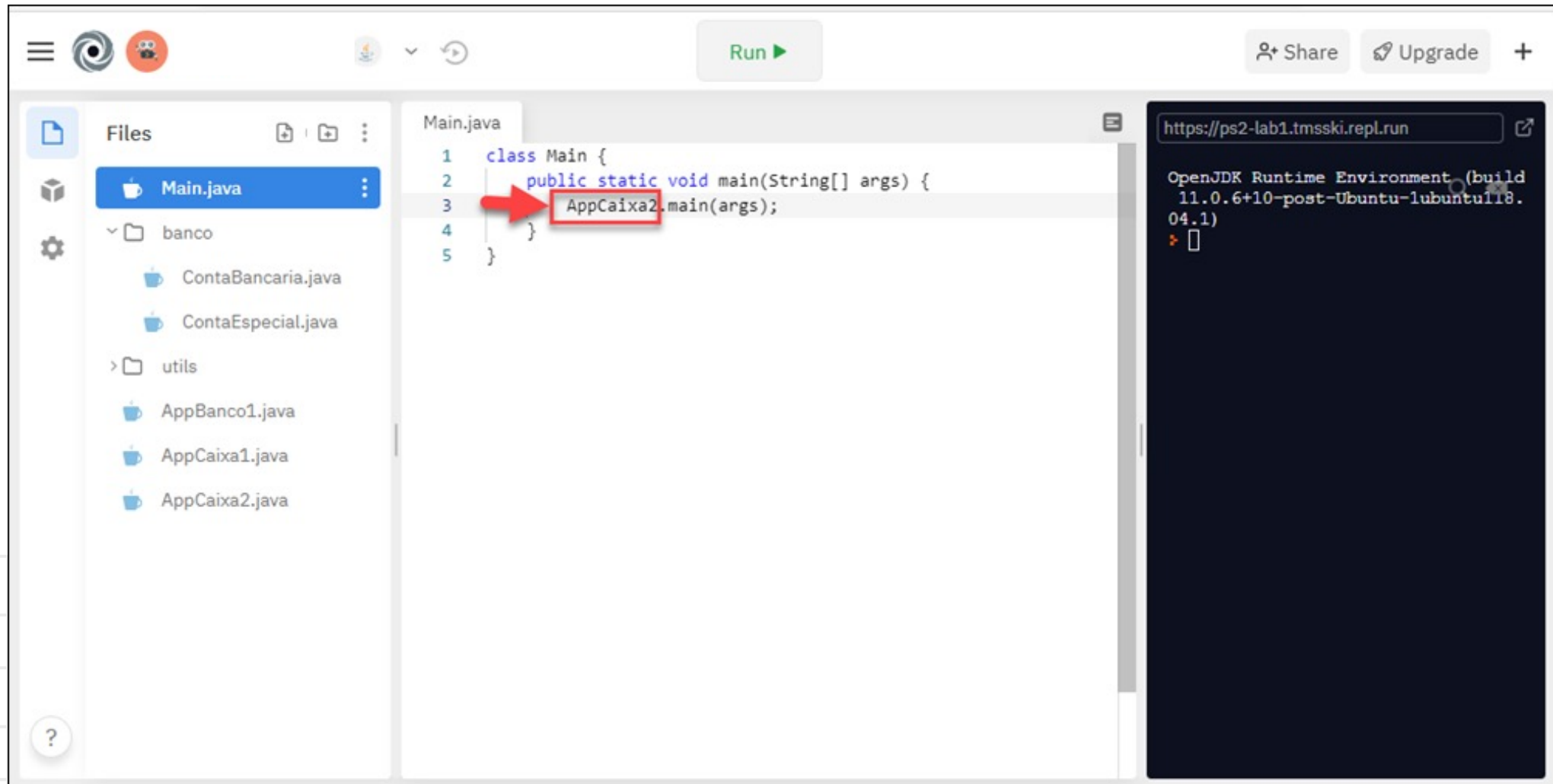
```
## Valor para limite:
```

Caso o usuário escolha a opção "Mostrar informações da conta", seu programa deverá apresentar uma mensagem no formato:

```
## Informações da conta:
Conta de Carlos com saldo de R$ 200,00 e com limite de R$ 300,00
```

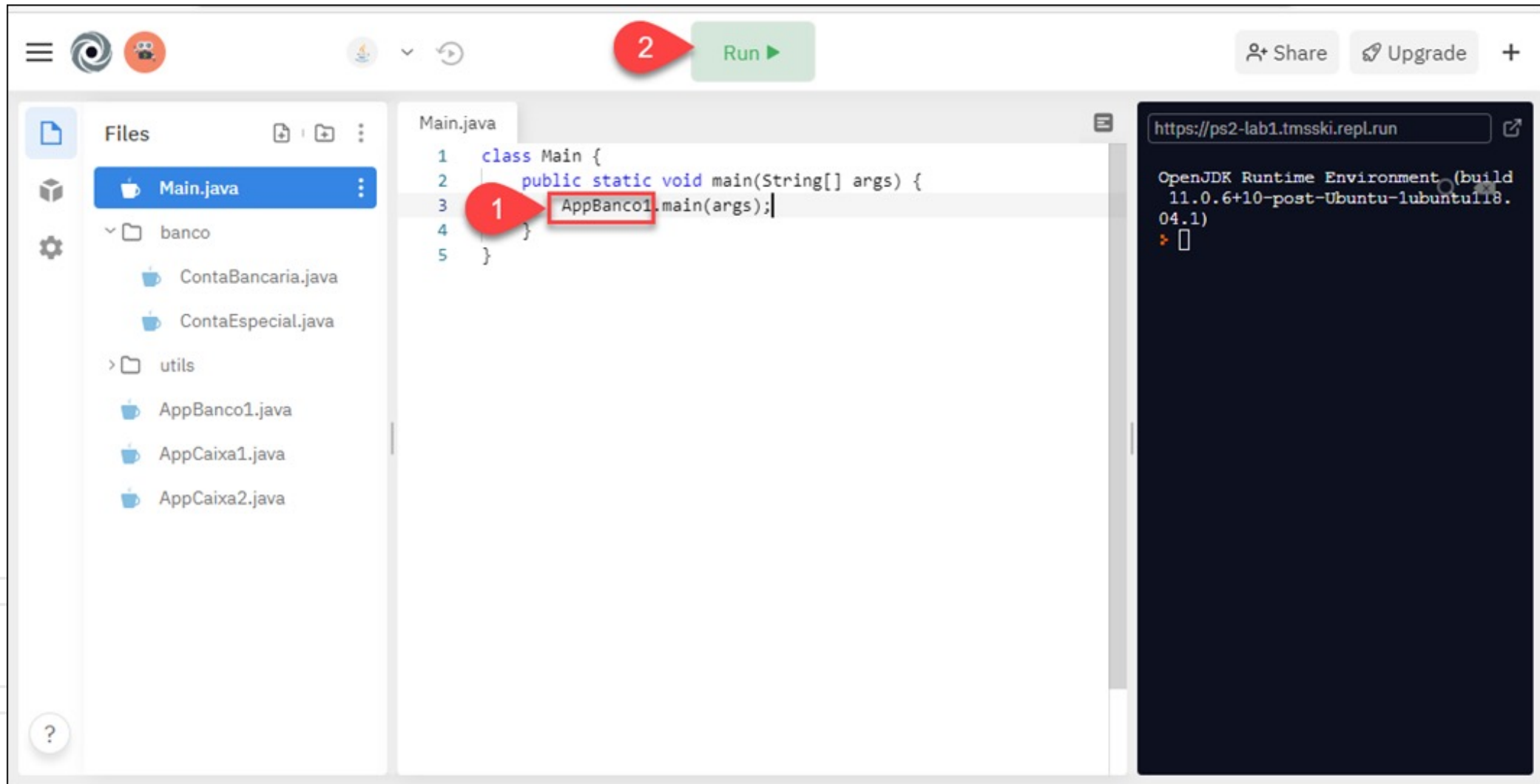
As demais opções devem ter uma função semelhante à da aplicação **AppCaixa1**, só que realizando as operações na conta especial.

Para testar o funcionamento da nova classe **AppCaixa2**, altere o nome da classe a ser executada no arquivo **Main.java** para **AppCaixa2**.



## Exercício 2

Altere o nome da classe a ser executada no arquivo **Main.java** para **AppBanco1** e pressione o botão **Run**.



Verifique o funcionamento de cada opção oferecida pelo programa **AppBanco1**.

```
1 class Main {
2     public static void main(String[] args) {
3         AppBanco1.main(args);
4     }
5 }
```

```
https://ps2-lab1.tmsski.repl.run

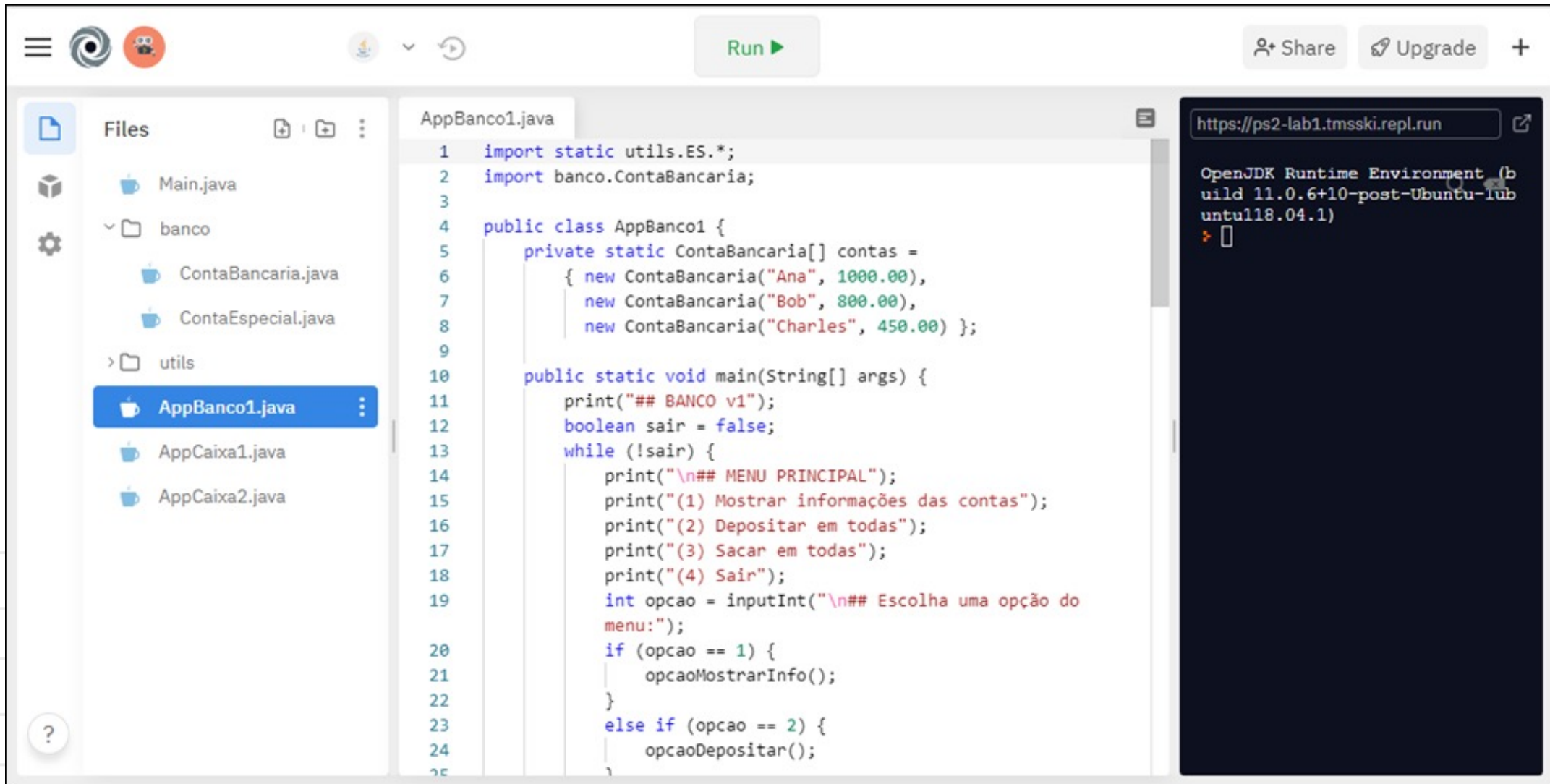
❖ javac -classpath ./run_dir/junit-4.12.jar:target/dependency/* -d . AppBanco1.java AppCaixa1.java AppCaixa2.java Main.java banco/ContaBancaria.java banco/ContaEspecial.java utils/ES.java
❖ java -classpath ./run_dir/junit-4.12.jar:target/dependency/* Main
## BANCO v1

## MENU PRINCIPAL
(1) Mostrar informações das contas
(2) Depositar em todas
(3) Sacar em todas
(4) Sair

## Escolha uma opção do menu: █
```



Examine o código da classe **AppBanco1**. Entenda como cada opção do programa foi implementada.



The screenshot displays a Java IDE interface. On the left, a file explorer shows a project structure with files: Main.java, ContaBancaria.java, ContaEspecial.java, AppBanco1.java (selected), AppCaixa1.java, and AppCaixa2.java. The main editor shows the code for AppBanco1.java:

```
1 import static utils.ES.*;
2 import banco.ContaBancaria;
3
4 public class AppBanco1 {
5     private static ContaBancaria[] contas =
6         { new ContaBancaria("Ana", 1000.00),
7           new ContaBancaria("Bob", 800.00),
8           new ContaBancaria("Charles", 450.00) };
9
10    public static void main(String[] args) {
11        print("## BANCO v1");
12        boolean sair = false;
13        while (!sair) {
14            print("\n## MENU PRINCIPAL");
15            print("(1) Mostrar informações das contas");
16            print("(2) Depositar em todas");
17            print("(3) Sacar em todas");
18            print("(4) Sair");
19            int opcao = inputInt("\n## Escolha uma opção do menu:");
20            if (opcao == 1) {
21                opcaoMostrarInfo();
22            }
23            else if (opcao == 2) {
24                opcaoDepositar();
25            }
26        }
27    }
28 }
```

On the right, a terminal window shows the command to run the application: `https://ps2-lab1.tmsski.repl.run` and the output: `OpenJDK Runtime Environment (build 11.0.6+10-post-Ubuntu-18.04.1)`.



# Exercício 2

Usando como exemplo a classe **AppBanco1**, desenvolva um novo programa (em uma classe chamada **AppBanco2**) para gerenciar um conjunto de contas especiais.

Seu programa deverá gerenciar 3 contas especiais (os dados iniciais de cada conta podem ser definidos a seu critério).

Seu programa deverá oferecer as seguintes opções no menu principal:

```
## MENU PRINCIPAL
(1) Mostrar informações das contas
(2) Depositar em todas
(3) Sacar em todas
(4) Aumentar limite de todas
(5) Sair

## Escolha uma opção do menu: |
```

Caso o usuário escolha a opção "Mostrar informações da conta", seu programa deverá apresentar uma mensagem no formato:

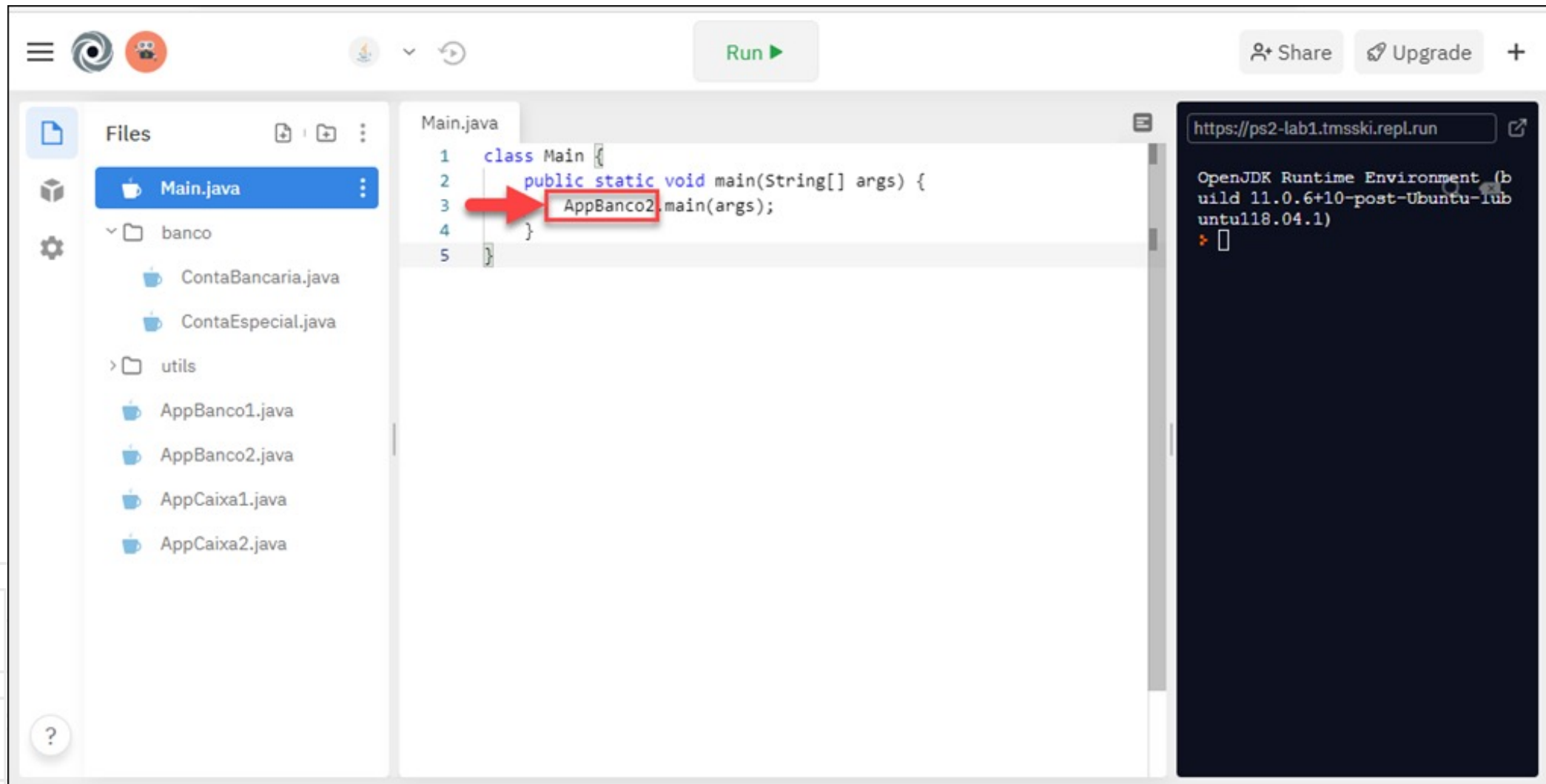
```
## Informações das contas:
Conta de Ana com saldo de R$ 1000,00 e com limite de R$ 800,00
Conta de Bob com saldo de R$ 800,00 e com limite de R$ 550,00
Conta de Charles com saldo de R$ 450,00 e com limite de R$ 400,00
```

Caso o usuário escolha a opção "Alterar limite de todas", seu programa deverá solicitar o valor a ser acrescentado no limite das contas:

```
## Valor a ser adicionado ao limite de todas:
```

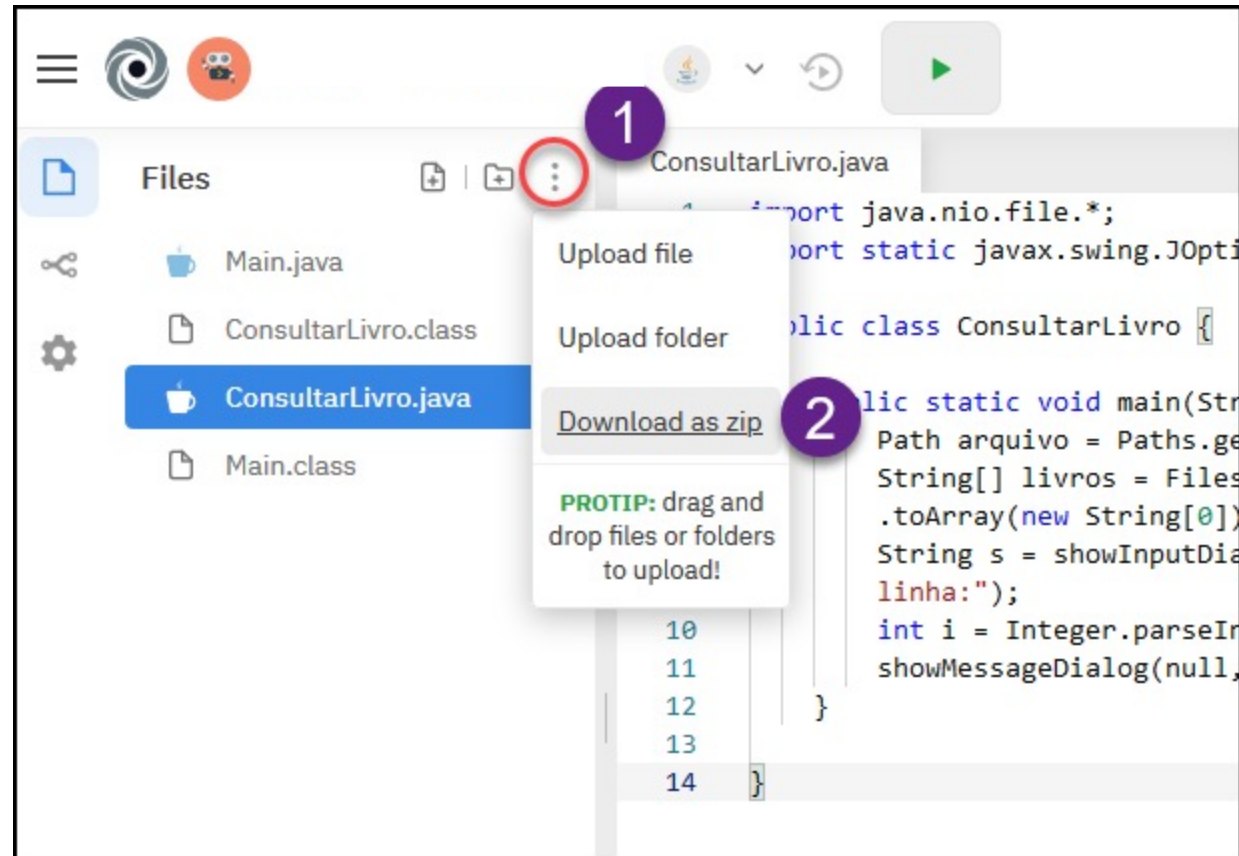
As demais opções devem ter uma função semelhante à da aplicação **AppBanco1**, só que realizando as operações nas contas especiais.

Para testar o funcionamento da nova classe **AppBanco2**, altere o nome da classe a ser executada no arquivo **Main.java** para **AppBanco2**.



# Entrega

Quando terminar todos os exercícios, siga os passos abaixo para baixar todo o conteúdo em um arquivo compactado. A seguir, faça a entrega do arquivo compactado na tarefa do Moodle.





Universidade Presbiteriana  
**Mackenzie**



1952 – 2022



Faculdade de  
**Computação e Informática**

