
Newton
Quem se prepara, não para.

Programação Orientada a Objetos
Prática 4

3º período

Professora: Michelle Hanne

Orientações

1) Criar uma conta no GitHub - https://github.com/
2) Criar um repositório Público no GitHub com o nome **Prática4_OO**
3) Subir os arquivos para o repositório criado
4) Enviar o link do repositório na tarefa do Canvas.

OBS: O envio deverá ser realizado individualmente

Questão 1 – Conta Bancária (Guiada)

Newton
Quem se prepara, não para.

- Criar um Projeto com o nome **ContaAbstract**
- Criar uma **classe abstrata** com o nome **Conta**
  - Atributo: private double saldo;
  - Métodos setSaldo e getSaldo públicos
  - Método: public abstract void imprimeExtrato(); //sem conteúdo
- Criar uma classe com o nome **ContaPoupança que extende Conta**
  - public class ContaPoupança extends Conta
  - Importar duas bibliotecas de data e hora
    - import java.text.SimpleDateFormat;
    - import java.util.Date;

Questão 1 – Conta Bancária (Guiada)

Newton
Quem se prepara, não para.

- Criar um Projeto com o nome **ContaAbstract**
- Criar uma **classe abstrata** com o nome **Conta**
  - Atributo: private double saldo;
  - Métodos setSaldo e getSaldo públicos
  - Método: public abstract void imprimeExtrato(); //sem conteúdo
- Criar uma classe com o nome **ContaPoupança que extende Conta**
  - public class ContaPoupança extends Conta
  - Importar duas bibliotecas de data e hora
    - import java.text.SimpleDateFormat;
    - import java.util.Date;

Questão 1 – Conta Bancária (Guiada)

Newton
Quem se prepara, não para.

- Criar o método ImprimeExtrato, conforme abaixo:

```
public void imprimeExtrato() {
        System.out.println("### Extrato da Conta ###");

        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/aaaa
    HH:mm:ss");
        Date date = new Date();

        System.out.println("Saldo: "+this.getSaldo());
        System.out.println("Data: "+sdf.format(date));

        }
```

Questão 1 – Conta Bancária (Guiada)

Newton
Quem se prepara, não para.

Criar no main:

```
Conta cp = new ContaPoupanca();
 cp.setSaldo(2121);
cp.imprimeExtrato();
```

Questão 2 – Conta Bancária Interface(Guiada)

- Criar um Projeto com o nome **ContaInteface**
- Criar uma **Inteface** com o nome **Conta**
  - void depositar(double valor);
  - void sacar(double valor);
  - double getSaldo();

Questão 2 – Conta Bancária Interface(Guiada)

- Criar a Classe **ContaCorrente que implementa Conta:**

```
public class ContaCorrente implements Conta{
        private double saldo;
    private double taxaOperacao = 0.45;
@Override
    public void depositar(double valor) {
            this.saldo += valor - taxaOperacao;
    }
@Override
    public double getSaldo() {
        return this.saldo;
    }
@Override
    public void sacar(double valor) {
        this.saldo -= valor + taxaOperacao;
    }
```

Questão 2 – Conta Bancária Interface(Guiada)

- Criar a Classe **ContaPoupança que implementa Conta:**

```
public class ContaPoupança implements Conta{
        private double saldo;
@Override
public void depositar(double valor) {
 this.saldo += valor;
}
@Override
public double getSaldo() {
 return this.saldo;
}
@Override
public void sacar(double valor) {
 this.saldo -= valor;
}
```

Questão 2 – Conta Bancária Interface(Guiada)

- Criar a Classe Pública **GeradorExtratos:**

```
public class GeradorExtratos {

 public void geradorConta(Conta conta){
 System.out.println("Saldo Atual: "+conta.getSaldo());
}
}
```

Questão 2 – Conta Bancária Interface(Guiada)

- Agora faça os testes no main:

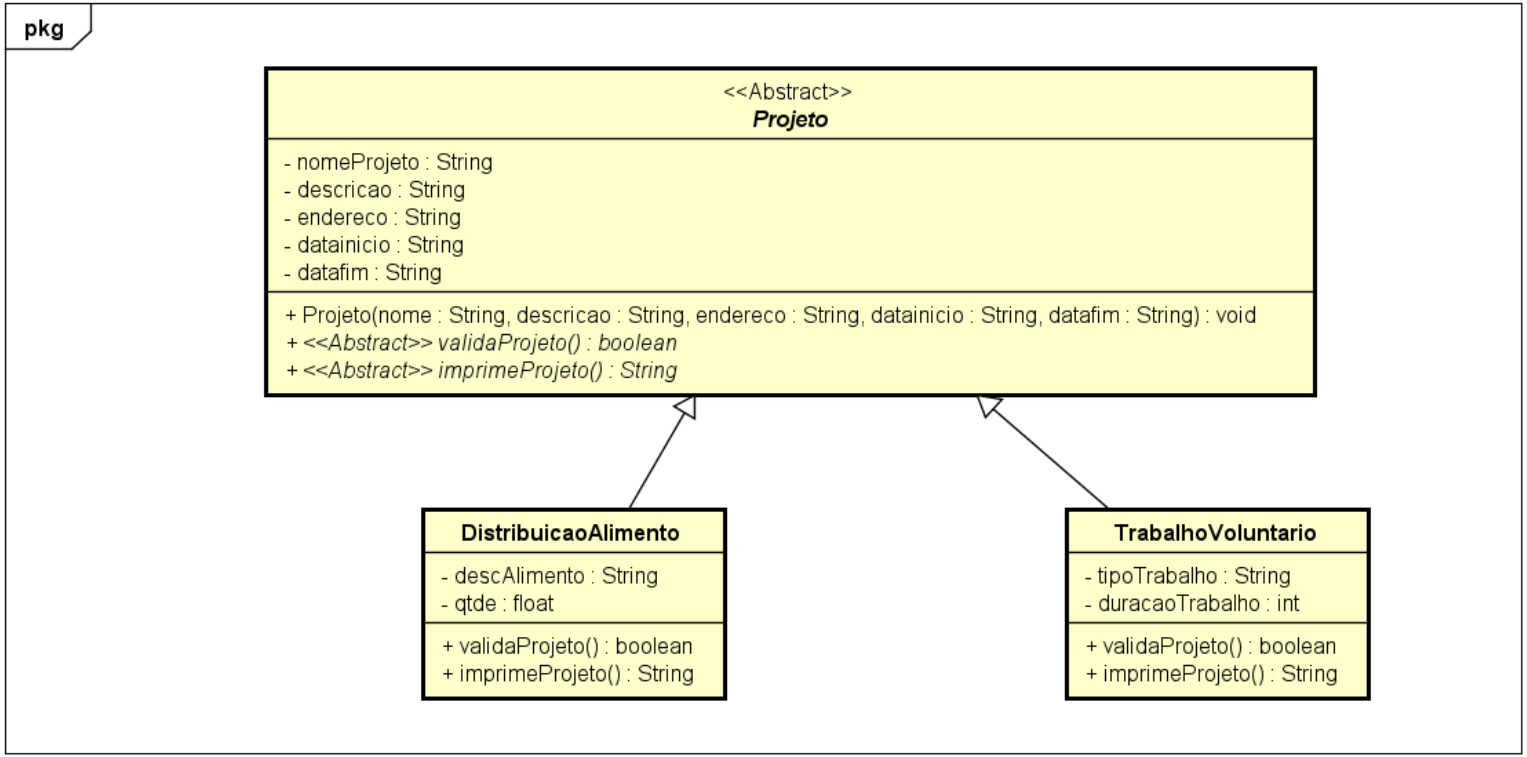
```
ContaCorrente cc = new ContaCorrente();
cc.depositar(1200.20);
cc.sacar(300);

ContaPoupanca cp = new ContaPoupanca();
cp.depositar(500.50);
cp.sacar(25);

GeradorExtratos gerador = new GeradorExtratos();
gerador.geradorConta(cc);
gerador.geradorConta(cp);                                    }
```

Questão 3 – ProjetoSocial

Newton
Quem se prepara, não para.



Questão 3 – ProjetoSocial

Newton
Quem se prepara, não para.

1. Implemente as classes em Java, conforme o diagrama. Crie os construtores para cada classe, mesmo que não esteja nos Diagramas, utilize o comando **super().**
2. Crie os métodos *Setters* e *Getters* de todos os atributos.
3. Crie o método **validaProjeto** na classe **DistribuicaoAlimento**. Este método deverá retornar ***true*** se a **datafim** do Projeto estiver vazia, caso contrário retornará ***false***.
4. Crie o método **imprimeProjeto** na classe **DistribuicaoAlimento**. Este método deverá retornar uma String com os seguintes atributos: *nomProjeto, descricao, datainicio, datafim, descAlimento e qtde.*
5. Crie o método **validaProjeto** na classe **TrabalhoVoluntario**. Este método deverá retornar ***true*** se a **duracao** for superior a 2, caso contrário retornará ***false***.

Questão 3 – ProjetoSocial

Newton
Quem se prepara, não para.

6. Crie o método **imprimeProjeto** na classe **TrabalhoVoluntario**. Este método deverá retornar uma String com os seguintes atributos: *nomProjeto, descricao, datainicio, datafim, tipoTrabalho e duracaoTrabalho.*
7. Crie a classe main() que deve possuir um menu para o usuário escolher a opção que deseja se voluntariar:

    <1> Cadastrar Projeto Distribuir Alimentos
    <2> Cadastrar Projeto Trabalho Voluntário
    <3> Sair

O Sistema deverá solicitar a entrada de dados. Use a classe Scanner ou JOptionPane
A opção <1> deverá criar um objeto do tipo **DistribuicaoAlimento**, executar o método **validaProjeto()** e o **imprimeProjeto()**
A opção <2> deverá criar um objeto do tipo **TrabalhoVoluntario**, executar o método **validaProjeto()** e o **imprimeProjeto()**
O programa deverá ficar em loop e só será encerrado se o usuário digitar o número 3.

Questão 4 – Celular

Newton
Quem se prepara, não para.

Analise o contexto abaixo, identifique as classes do tipo interface, abstrata, herança e polimorfismo, se houver. Faça a implementação do projeto:

- A interface de uso de um celular é composta por recursos como ligar e desligar o celular, câmera, acessa ao fone de ouvido e controle de volume.
Todos os recursos são do tipo SIM ou Não (verdadeiro ou Falso).

- Existem atributos comuns aos vários tipos de celulares, o que permite que cada fabricante escolha se terá ou não esse recurso, são eles: enviar mensagem, acessar a Internet, e-mails, rádio, tv, verificar o sistema de arquivos.
Todos os recursos são do tipo SIM ou Não (verdadeiro ou Falso). Também há o atributo preço que é particular para cada tipo de fabricante.

Questão 4 – Celular

Newton
Quem se prepara, não para.

- Existe dois tipos de fabricantes de celular: Celular Fabricante A e Celular Fabricante B, cada um com atributos específicos, porém todos implementam a interface

Crie um main() para “mocar” os dados e testar os objetos das classes. Exemplo:

- O Fabricante celular A possui todos os recursos comuns aos celulares, porém não tem TV. O seu valor é R$ 6800.00
- O Fabricante celular B possui todos os recursos comuns aos celulares, porém não tem rádio. O seu valor é R$ 5200,00.