

TEMA PROJETO: Cadeia de Restaurantes(Bob's ou Subway)

Discente: João Batista Araújo de Lima Filho.

Classe Acompanhamento

```
package atividade8;

// I - Crie a Classe1 com um nome relacionado ao seu projeto:
//a) Deve ser uma classe abstrata;
//b) Deve ter método abstrato.

    "Aqui foi criado a classe abstrata Acompanhamento"

public abstract class Acompanhamento {

    private String nome;
    private double preco;
    private String tamanho;

    public Acompanhamento(String nome, String tamanho){
        setNome(nome);
        setTamanho(tamanho);
        if(nome.equals("Batata Palito")){
            setPreco(3.99);
        } else{
            setPreco(9.99);
        }
        if(getTamanho().equals("Pequeno")) {
            this.preco += 2.99;
        } else if (getTamanho().equals("Medio")) {
            this.preco += 4.99;
        } else {
            setPreco(getPreco() + 6.99);
        }
    }

    "Esse é o método abstrato adicional"

    public abstract void adicional(); // Metodo abstrato

    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }

    public double getPreco() {
        return preco;
    }

    public void setPreco(double preco) {
        this.preco = preco;
    }
}
```

```

}

public String getTamanho() {
    return tamanho;
}

public void setTamanho(String tamanho) {
    this.tamanho = tamanho;
}
}

```

Classe Nuggets que extends Acompanhamento

```
package atividade8;
```

```
// II - Crie a Classe2 (com um nome relacionado ao seu projeto), que implementa a
//Classe1.
```

“Essa classe Nuggets implementa a classe abstrata Acompanhamento”

```
public class Nuggets extends Acompanhamento{

    public Nuggets(String nome, String tamanho) {
        super(nome,tamanho);
    }

```

“Sobrescrita do método abstrato adicional() da classe abstrata Acompanhamento. Aqui esse método coloca um adicional ao Acompanhamento de acordo com o acompanhamento e atualiza o seu preço”

```

@Override
public void adicional() {
    if(getNome().equals("Nuggets de Frango")) {
        setNome(getNome() + " + Molho Caipira");
        setPreco(getPreco() + 3.99);
    } else {
        setNome(getNome() + " + Molho Barbecue");
        setPreco(getPreco() + 3.99);
    }
}
}

```

Classe Batata que extends Acompanhamento

```
package atividade8;
```

```
// III - Crie a Classe3 (com um nome relacionado ao seu projeto), que implementa a
//Classe1. A implementação do método abstrato deve funcionar diferente daquela realizada na
//Classe2.
```

“Essa classe Batata implementa a classe abstrata Acompanhamento”

```
public class Batata extends Acompanhamento{
```

```
public Batata(String nome, String tamanho) {
    super(nome,tamanho);
}
```

“Sobrescrita do método abstrato adicional() da classe abstrata Acompanhamento. Aqui esse método coloca um adicional ao Acompanhamento de acordo com o acompanhamento e atualiza o seu preço”

```
@Override
public void adicional() {
    if(getNome().equals("Batata Palito")) {
        setNome(getNome() + " ,Adicional: Cheddar com Bacon");
        setPreco(getPreco() + 4.99);
    } else {
        setNome(getNome() + " ,Adicional: Tirinhas de Frango Empanado");
        setPreco(getPreco() + 7.99);
    }
}
```

Classe Main

```
package atividade8;

import java.util.ArrayList;

public class Main {
    public static void main(String[] args) {

        // IV - Crie um ArrayList do tipo Classe1:
        //a) Preencha ele com elementos tanto da Classe2 como da Classe3.
        //b) Mostre, através da saída do console, o resultado da chamada do método
        //abstrato para cada elemento do ArrayList.

        “Aqui temos um ArrayList do tipo Acompanhamento”

        ArrayList<Acompanhamento> acompanhamentos = new ArrayList<Acompanhamento>();
        System.out.println("Repositorio Acompanhamentos\n");

        “Aqui preenchemos o ArrayList tanto com elementos do tipo Nuggets quanto do tipo
        Batata”

        acompanhamentos.add(new Batata("Batata Palito","Pequeno"));
        acompanhamentos.add(new Nuggets("Nuggets de Frango","Medio"));
        acompanhamentos.add(new Batata("Batata Rustica","Grande"));
        acompanhamentos.add(new Nuggets("Nuggets de Queijo","Pequeno"));

        “Esse primeiro laço mostra todos os elemento do ArrayList sem o método abstrato”

        for (Acompanhamento i: acompanhamentos) {
            System.out.println("Nome: "+i.getNome()+" , Tamanho: "+i.getTamanho()+" , Preço:
            "+i.getPreco());
        }

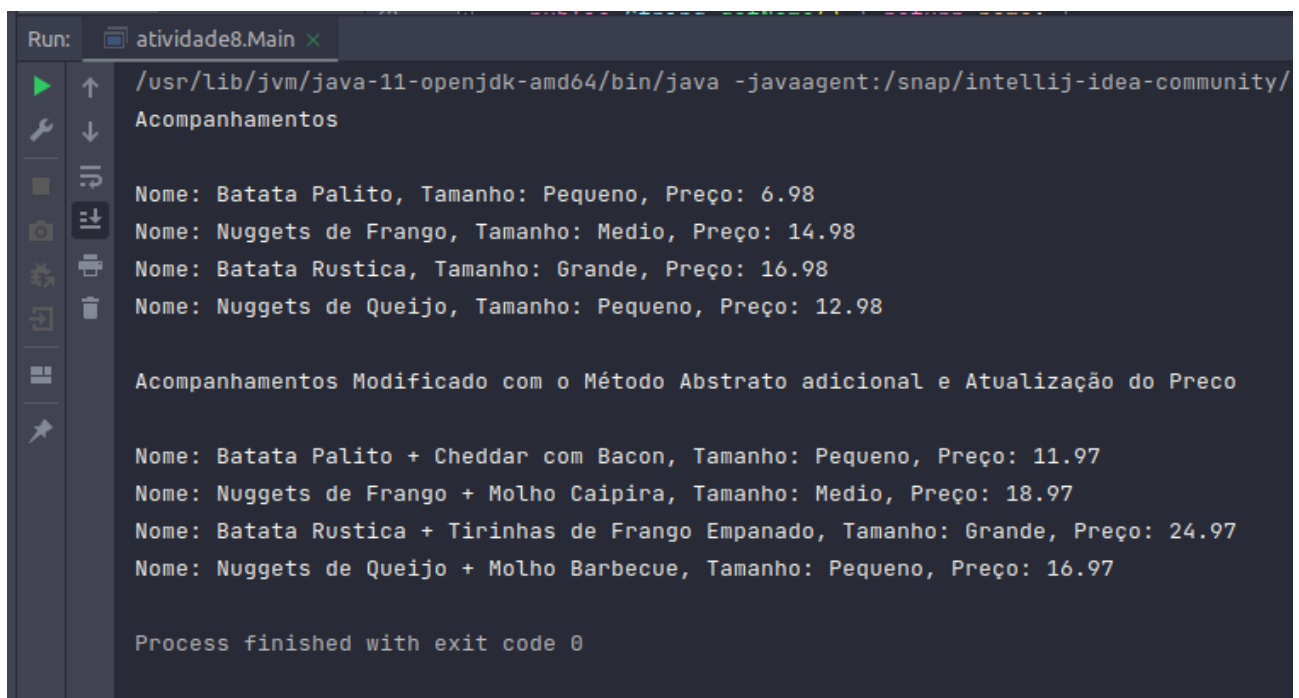
        System.out.println("\nRepositorio Acompanhamentos Modificado com o Adicional e
```

```
Atualização do Preço\n");
```

“Esse segundo laço mostra o resultado da chamada do método abstrato para cada elemento do ArrayList”

```
for (Acompanhamento i: acompanhamentos) {  
    i.adicional();  
    System.out.println("Nome: "+i.getNome()+"", Tamanho: "+i.getTamanho()+"", Preço:  
"+i.getPreco());  
}  
  
}  
}
```

IMAGENS DAS SAÍDAS NO CONSOLE



```
Run: atividade8.Main x  
/usr/lib/jvm/java-11-openjdk-amd64/bin/java -javaagent:/snap/intellij-idea-community/  
Acompanhamentos  
  
Nome: Batata Palito, Tamanho: Pequeno, Preço: 6.98  
Nome: Nuggets de Frango, Tamanho: Medio, Preço: 14.98  
Nome: Batata Rustica, Tamanho: Grande, Preço: 16.98  
Nome: Nuggets de Queijo, Tamanho: Pequeno, Preço: 12.98  
  
Acompanhamentos Modificado com o Método Abstrato adicional e Atualização do Preço  
  
Nome: Batata Palito + Cheddar com Bacon, Tamanho: Pequeno, Preço: 11.97  
Nome: Nuggets de Frango + Molho Caipira, Tamanho: Medio, Preço: 18.97  
Nome: Batata Rustica + Tirinhas de Frango Empanado, Tamanho: Grande, Preço: 24.97  
Nome: Nuggets de Queijo + Molho Barbecue, Tamanho: Pequeno, Preço: 16.97  
  
Process finished with exit code 0
```