

## Atividade Prática Supervisionada – POO

Prof. Me Renato Alves Ferreira

### Integrantes

Albérico Lucas Bispo Ferreira – 1186701

Ariel Bulgari Peixoto – 2845792

Cainã Molinas Zampini - 2072186

Felipe Ferreira – 2993620

Gabriel Eigenmann - 2970007

Geovanna Telles Nogueira - 2565671

Giovane Lisboa Rodrigues - 2234414

Henrique Trajano- 2976320

Mayara Lupeti Turbiani - 2558151

## To Do List

### Descrição do trabalho (o que é / para quê / finalidade / aplicabilidade)

Espera-se do trabalho,

Desenvolvimento de uma atividade prática supervisionada de um projeto em uma grande equipe de desenvolvimento (entre 4 e 8 integrantes) que contemple os principais conceitos da POO, como: Classes concretas e abstratas, métodos comuns e estáticos, herança, polimorfismo, interfaces, encapsulamento, coleção, etc.

*-Os códigos deverão ser bem documentados e visíveis, assim como as imagens com os prints de execução do projeto e programas;*

*-Usem o projeto desenvolvido em aula como referência;*

*-Distribuam as tarefas entre os integrantes do grupo e aponte o responsável por cada item, mas todos devem participarem do todo.*

### Parte 1 – “Apresentação do projeto”

#### CLASSE TASK

```
package                                model;

import                                java.time.LocalDate;

public class Task { /*Representação da tarefa que irá ser realizada*/

    private                                static                                int                                contador;

    private                                int                                idTask;
    private                                String                                description;
    private                                boolean                                isDone;
    private                                LocalDate                                deadLine;

    public Task(String description, LocalDate deadLine){ /*Construtor
da                                     Tarefa*/
        Task.contador++;
        this.idTask = Task.contador;
        this.description = description;
```

```

        this.isDone = false;
        this.deadLine = deadLine;
    }
    /*Getters e Setters*/
    public int getIdTask() {
        return idTask;
    }

    public String getDescription() {
        return description;
    }

    public boolean isDone() {
        return isDone;
    }

    public LocalDate getDeadLine() {
        return deadLine;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public void setDeadLine(LocalDate deadLine) {
        this.deadLine = deadLine;
    }

    public void doneTask() {
        this.isDone = true;
    }

    @Override
    public String toString() { /*Conversão das informações para String
dentro do Objeto para fácil extração*/
        return "Task{"
            + "idTask=" + idTask
            + ", description='" + description + '\''
            + ", isDone=" + isDone
            + ", deadLine=" + deadLine
            + '}';
    }
}

```

### CLASSE TASKCONTROLLER

```

package model;

import java.time.LocalDate;

public class Task { /*Representação da tarefa que irá ser realizada*/

    private static int contador;

    private int idTask;
    private String description;
}

```

```

private boolean isDone;
private LocalDate deadLine;

da public Task(String description, LocalDate deadLine){ /*Construtor
Tarefa*/
    Task.contador++;
    this.idTask = Task.contador;
    this.description = description;
    this.isDone = false;
    this.deadLine = deadLine;
}
/*Getters e Setters*/
public int getIdTask() {
    return idTask;
}

public String getDescription() {
    return description;
}

public boolean isDone() {
    return isDone;
}

public LocalDate getDeadLine() {
    return deadLine;
}

public void setDescription(String description) {
    this.description = description;
}

public void setDeadLine(LocalDate deadLine) {
    this.deadLine = deadLine;
}

public void doneTask(){
    this.isDone = true;
}

@Override
dentro do Objeto para fácil extração*/
public String toString() {
    return "Task{"
        + "idTask=" + idTask
        + ", description='" + description + '\''
        + ", isDone=" + isDone
        + ", deadLine=" + deadLine
        + '}';
}
}

```

### CLASSE MAIN

```

import controller.TaskController;
import model.Task;

```

```

import java.time.LocalDate;
import java.util.List;
import java.util.Scanner;

public class Main { /*Classe principal que rodará o programa*/

    private static TaskController controller = new TaskController();

    public static void main(String[] args) { /*Método principal,
referente ao contato com o usuário.*/
        Scanner scanner = new Scanner(System.in);
        int choose = 0;
        while (choose != 7) {
            System.out.println("===== TO DO LIST
=====
\n"
            "=====
\n"
            "==== 1. Criar uma tarefa =====
\n"
            "==== 2. Listar as tarefas =====
\n"
            "==== 3. Procurar uma tarefa =====
\n"
            "==== 4. Editar uma tarefa =====
\n"
            "==== 5. Concluir uma tarefa =====
\n"
            "==== 6. Deletar uma tarefa =====
\n"
            "==== 7. Fechar o programa =====
\n");

            while (!scanner.hasNext())
                choose = scanner.nextInt();
            switch (choose) {
                case 1:
                    Main.createTask(scanner);
                    break;
                case 2:
                    Main.listTask(scanner);
                    break;
                case 3:
                    Main.findTask(scanner);
                    break;
                case 4:
                    Main.updateTask(scanner);
                    break;
                case 5:
                    Main.doneTask(scanner);
                    break;
                case 6:
                    Main.deleteTask(scanner);
                    break;
            }
        }
    }
}

```

```

        private static void createTask(Scanner scanner){ /*Método que irá
criar                                uma                                tarefa*/
            scanner.nextLine();
            System.out.println("====    Insira a descrição da tarefa:
====");
            String description = scanner.nextLine();
            System.out.println("====    Insira a data limite(AAAA-MM-DD):
====");
            String limitDate = scanner.next();
            Task newTask = Main.controller.createTask(description,
LocalDate.parse(limitDate));
            System.out.println(newTask);
        }

        private static void listTask(Scanner scanner){ /*Método que irá
exibir                                a                                Lista                                de                                Tarefa*/
            List<Task> list = Main.controller.listTask();
            list.forEach(task -> System.out.println(task));
        }

        private static void findTask(Scanner scanner){ /*Método que irá
procurar                                a                                tarefa                                de                                acordo                                com                                o                                ID                                digitado*/
            System.out.println("====    Insira o ID da tarefa:
====");
            int id = scanner.nextInt();
            Task task = Main.controller.findTask(id);
            System.out.println(task);
        }

        private static void updateTask(Scanner scanner){ /*Método que irá
atualizar/editar                                a                                tarefa*/
            System.out.println("====    Insira o ID da tarefa:
====");
            int updateId = scanner.nextInt();
            scanner.nextLine();
            System.out.println("====    Insira a descrição da tarefa:
====");
            String description = scanner.nextLine();
            System.out.println("====    Insira a data limite(AAAA-MM-DD):
====");
            String limitDate = scanner.next();
            Task updateTask = Main.controller.updateTask(description,
limitDate, updateId);
            System.out.println(updateTask);
        }

        private static void doneTask(Scanner scanner){ /*Método que irá
retornar                                ao                                usuário                                se                                a                                tarefa                                foi                                concluída                                ou                                não*/
            System.out.println("====    Insira o ID da tarefa:
====");
            int id = scanner.nextInt();
            Main.controller.doneTask(id);
            System.out.println("====    Tarefa Finalizada!!
====");
        }

        private static void deleteTask(Scanner scanner){ /*Método que irá
excluir                                a                                tarefa*/

```

```

====");
    int id = scanner.nextInt();
    Main.controller.deleteTask(id);
    System.out.println("====");
}
}

```

Insira o ID da tarefa:

Tarefa Deletada!!

## Parte 2 – “Prints das telas/execução”

```

"C:\Program Files\Eclipse Adoptium\jdk-8.0.322.6-hotspot\bin\java.exe" ...
===== TO DO LIST =====
=====
==== 1. Criar uma tarefa      ====
==== 2. Listar as tarefas    ====
==== 3. Procurar uma tarefa  ====
==== 4. Editar uma tarefa    ====
==== 5. Concluir uma tarefa  ====
==== 6. Deletar uma tarefa   ====
==== 7. Fechar o programa    ====
=====

```

1. Exibição do menu principal.

```

"C:\Program Files\Eclipse Adoptium\jdk-8.0.322.6-hotspot\bin\java.exe" ...
===== TO DO LIST =====
=====
==== 1. Criar uma tarefa      ====
==== 2. Listar as tarefas    ====
==== 3. Procurar uma tarefa   ====
==== 4. Editar uma tarefa     ====
==== 5. Concluir uma tarefa   ====
==== 6. Deletar uma tarefa    ====
==== 7. Fechar o programa     ====
=====

1
==== Insira a descrição da tarefa:  ====
Teste 1
==== Insira a data limite(AAAA-MM-DD): ====
2022-03-19
Task{idTask=1, description='Teste 1', isDone=false, deadLine=2022-03-19}

```

2. Adição de tarefa..

```

===== TO DO LIST =====
=====
==== 1. Criar uma tarefa      ====
==== 2. Listar as tarefas    ====
==== 3. Procurar uma tarefa   ====
==== 4. Editar uma tarefa     ====
==== 5. Concluir uma tarefa   ====
==== 6. Deletar uma tarefa    ====
==== 7. Fechar o programa     ====
=====

2
Task{idTask=1, description='Teste 1', isDone=false, deadLine=2022-03-19}
===== TO DO LIST =====

```

3. Listagem de tarefas.

```

===== TO DO LIST =====
=====
==== 1. Criar uma tarefa      ====
==== 2. Listar as tarefas    ====
==== 3. Procurar uma tarefa   ====
==== 4. Editar uma tarefa     ====
==== 5. Concluir uma tarefa   ====
==== 6. Deletar uma tarefa    ====
==== 7. Fechar o programa    ====
=====

3
==== Insira o ID da tarefa:    ====
1
Task{idTask=1, description='Teste 1', isDone=false, deadLine=2022-03-19}

```

4.Procura de tarefas por Id.

```

Task{idTask=1, description='Teste 1', isDone=false, deadLine=2022-03-19}
===== TO DO LIST =====
=====
==== 1. Criar uma tarefa      ====
==== 2. Listar as tarefas    ====
==== 3. Procurar uma tarefa   ====
==== 4. Editar uma tarefa     ====
==== 5. Concluir uma tarefa   ====
==== 6. Deletar uma tarefa    ====
==== 7. Fechar o programa    ====
=====

4
==== Insira o ID da tarefa:    ====
1
==== Insira a descrição da tarefa:  ====
Comer batata
==== Insira a data limite(AAAA-MM-DD): ====
2022-04-14
Task{idTask=1, description='Comer batata', isDone=false, deadLine=2022-04-14}

```

5. Edição de tarefa.



```
===== TO DO LIST =====  
=====
```

====	1. Criar uma tarefa	====
====	2. Listar as tarefas	====
====	3. Procurar uma tarefa	====
====	4. Editar uma tarefa	====
====	5. Concluir uma tarefa	====
====	6. Deletar uma tarefa	====
====	7. Fechar o programa	====

```
=====
```

5

====	Insira o ID da tarefa:	====
------	------------------------	------

1

====	Tarefa Finalizada!!	====
------	---------------------	------

6.Concluir tarefa.

```

===== Tarefa Finalizada!! =====
===== TO DO LIST =====
=====
===== 1. Criar uma tarefa =====
===== 2. Listar as tarefas =====
===== 3. Procurar uma tarefa =====
===== 4. Editar uma tarefa =====
===== 5. Concluir uma tarefa =====
===== 6. Deletar uma tarefa =====
===== 7. Fechar o programa =====
=====

6
===== Insira o ID da tarefa: =====
1
===== Tarefa Deletada!! =====

```

7.Excluir tarefa.

```
====          Tarefa Deletada!!          ====
===== TO DO LIST =====
=====
====      1. Criar uma tarefa      ====
====      2. Listar as tarefas      ====
====      3. Procurar uma tarefa    ====
====      4. Editar uma tarefa      ====
====      5. Concluir uma tarefa    ====
====      6. Deletar uma tarefa     ====
====      7. Fechar o programa     ====
=====

7

Process finished with exit code 0
|
```

8.

Tarefa finalizada.

#### REFERENCIAS -

DEITEL, Paul.DEITEL, Harvey; Java: Programar, 10. ed. São Paulo, 2017

PUGA, Sandra. Lógica de Programação e estrutura de dados com aplicações em Java. 2. ed. São Paulo. 2009

Felipe Ferreira (2993620)