INSTITUTO SUPERIOR MANUEL TEIXEIRA GOMES


# Trabalho Prático 1 - Programação Orientada a Objetos
# Teste Americano

**Relatório de Programação Orientada a Objetos**

**Leandro Fonseca, n.º 22001805**

Docente:
Prof. Doutor Tiago Candeias

2021

# Resumo

Este trabalho foi realizado com o objectivo de implementar os conceitos de encapsulamento, herança e polimorfismo, estudados no âmbito do paradigma de programação orientada a objetos.
Para implementar os conhecimentos, foi pedido para relaizar um programa que ajuda-se um professor de matemática a avaliar os testes dos alunos. Sendo que o teste iria ser ao estilo americano misturando perguntas abertas e perguntas fechadas.

# Conteúdo

# 1 Introdução

## 1.1 Descrição do Problema

O professor de matemática exausto de avaliar questões, decidiu formular os seus testes, ao bom estilo americano, com respostas de escolha múltipla. Depois de pensar melhor, verificou que deveria misturar questões abertas e questões fechadas. Para automatizar a tarefa de cálculo das cotações, pediu ajuda aos informáticos.

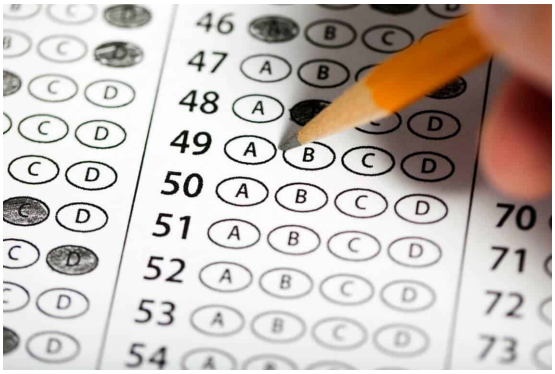


**Figura 1.1** – Teste Americano. Fonte Desconhecida **Link**

Um teste é composto por várias perguntas abertas ou fechadas (questões de escolha múltipla). Cada pergunta fechada apresenta várias opções e cada uma dessas opções tem uma cotação. Por exemplo, numa questão fechada com 4 opções, a resposta certa à pergunta, vale 1 valor e as opções erradas valem -0.25, cada uma. As cotações para as várias questões abertas e opções das questões fechadas, são atribuídas na criação do teste. Cada pergunta aberta, pode ser respondida livremente e tem a cotação na linha posterior à questão. Cada aluno da turma responde às várias questões do teste e depois de calculados todos os testes é apresentada a classificação. Note que, se um aluno não responder a uma questão, não pontua. O resultado final apresenta o nome e nota, com ordenação decrescente pela nota final. No caso de dois estudantes terem a mesma classificação, a ordenação deve considerar o nome, por ordem crescente.

## 1.2 Objetivo

Dado o número de perguntas, o conjunto de perguntas abertas ou fechadas com opção e respetiva cotação, o número de alunos e as respostas de cada aluno às questões. Considere que depois do número da questão, aparece o tipo de questão, aberta ou fechada (A ou F). Note que a falta de resposta a uma pergunta é marcada pelo carácter -". Após a execução do programa é obtida a pauta final do teste com a avaliação ordenada pelos critérios especificados.

**Input:**
4
1 F "Quanto é 10+10?"
A 10 -0.25
B 20 1
C 30 -0.25
D 40 -0.25
2 F "Quanto vale 10-5?"
A 1 -0.25
B 5 1
C 10 -0.25
D 15 -0.25
3 F "Quanto vale 5-10?"
A 1 -0.25
B -5 1
C 10 -0.25
D 15 -0.25
4 A "Prove que 15*3 = 15+15+15"
2
5
Amilcar
1 B
2 C
3 D
4 2
Beatriz
1 A
2 B
3 C
4 -
Carina
1 -
2 C
3 D
4 1
Dinis
1 D
2 B
3 A
4 2
Eurico
1 C
2 A
3 -
4 -

**Output:**
Amilcar 2.5
Dinis 2.5
Carina 1.5
Beatriz 0.5
Eurico -0.5

**Figura 1.2** – Exemplo de Input e Outputn

# 2 Planeamento

Antes de começar a desenvolver o programa foi feito primeiro o diagram de classes com o que iria ser preciso implementar.

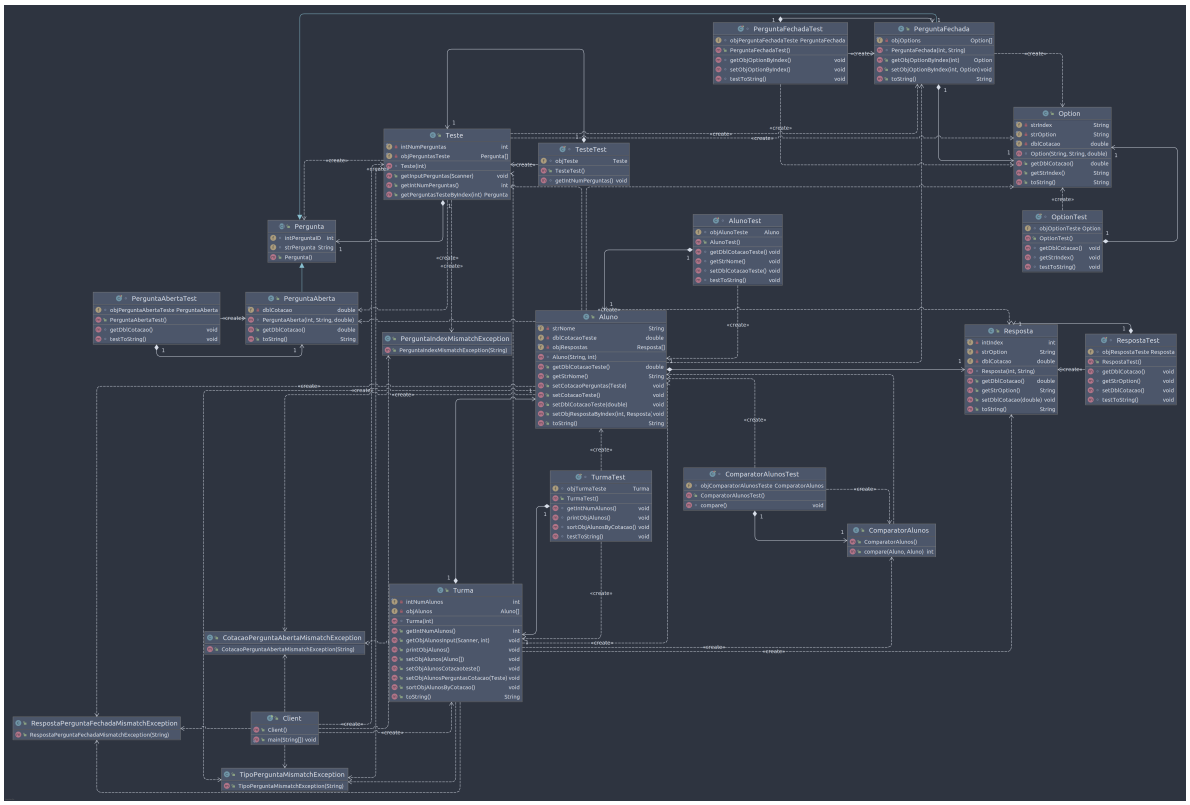

**Figura 2.1** – Diagrama de Classes desenhado no IntelliJ IDEA Ultimate

Para implementar este programa será preciso as seguintes classes:

- Classe Main que vai ser onde o programa vai executar.
- Classe Turma que vai reunir todos os alunos.
- Classe Aluno que vai representar cada aluno.
- Classe Teste que vai conter as informações do Teste.
- Classe Abstrata Pergunta que vai ter os atributos de pergunta.
- Classe PerguntaAberta que vai extender a classe Pergunta e representa as perguntas abertas.
- Classe PerguntaFechada que vai extender a classe Pergunta e representa as perguntas fechadas com as suas opções.
- Classe Option que vai implementar a opção de uma pergunta fechada.
- Classe Resposta que contém as informações da resposta de um aluno a uma pergunta.
- Classe ComparatorAlunos que implementa a Classe Comparator do *java.util* para comparar dois alunos para poder ordená-los.

# Testes Unitários

3

Na parte de desenvolvimento foi seguida a metodologia TDD(Test-Driven Development), ou seja baseado em testes. Foi utilizada a framework JUnit5 para a realizar os testes unitários, para puder validar os métodos das classes implementadas.

### 3.0.1 AlunoTest

A classe AlunoTest é a classe que implementa os testes unitários para os seguintes métodos da classe Aluno:

- public double getDblCotacaoTeste();
- public String getStrNome();
- public void setDblCotacaoTeste(double dblCotacaoTeste);
- public String toString();

### 3.0.2 ComparatorAlunosTest

A classe ComparatorAlunosTest é a classe que implementa os testes unitários para os seguintes métodos da classe ComparatorAlunos:

- public int compare(Aluno objAluno1, Aluno objAluno2);

### 3.0.3 OptionTest

A classe OptionTest é a classe que implementa os testes unitários para os seguintes métodos da classe Option:

- public double getDblCotacao();
- public String getStrIndex();
- public String toString();

### 3.0.4 PerguntaAbertaTest

A classe PerguntaAbertaTest é a classe que implementa os testes unitários para os seguintes métodos da classe PerguntaAberta:

- public double getDblCotacao();
- public String toString();

### 3.0.5 PerguntaFechadaTest

A classe PerguntaFechadaTest é a classe que implementa os testes unitários para os seguintes métodos da classe PerguntaFechada:

- public Option getObjOptionByIndex(int intIndex);
- public public void setObjOptionByIndex(int intIndex, Option option);
- public String toString();

### 3.0.6 RespostaTest

A classe RespostaTest é a classe que implementa os testes unitários para os seguintes métodos da classe Resposta:

- public String getStrOption();
- public double getDblCotacao();
- public setDblCotacao(double dblCotacao);
- public String toString();

### 3.0.7 TesteTest

A classe TesteTest é a classe que implementa os testes unitários para os seguintes métodos da classe Teste:

- public int getIntNumPerguntas();

### 3.0.8 TurmaTest

A classe TurmaTest é a classe que implementa os testes unitários para os seguintes métodos da classe Turma:

- public int getIntNumAlunos();
- public void sortObjAlunosByCotacao();
- public void printObjAlunos();
- public String toString();

# 4 Exceções

Para evitar que o programa termina-se subitamente devido a erros durante a execução foram implementadas classes que extendem a classe Exception para puder dar catch dos erros. As Exceções implementadas neste programa foram:

- FileNotFoundException que é provocado quando não é encontrado o ficheiro de input.
- PerguntaIndexMismatchException que é provocado quando o número das perguntas não segue a ordem sequencial começando no 1 e inteiros.
- TipoPerguntaMismatchException que é provocado quando o tipo de pergunta introduzido pelo input não é nem aberta nem fechada.
- CotacaoPerguntaAbertaMismatchException que é provocado quando a cotação da pergunta aberta é menor que 0 e maior que o máximo indicado no teste.
- RespostaPerguntaFechadaMismatchException que é provocado quando a opção respondida pelo Aluno não corresponde a nenhuma opção disponível no Teste.
- InputMismatchException que é provocado quando o input dado no ficheiro não corresponde com o expectado.

# 5 Conclusão

Concluío com este trabalho, que através do planeamento através do desenho do diagrama de classes, a implementação das classes, testes unitários e exceções foi conseguido aplicar todo o conhecimento adquirido nas aulas Teórico-práticas da disciplina de Programação Orientada a Objetos [1].

Foi um trabalho com algum grau de dificuldade e que foi necessário pensar muito bem na sua implementação e concepção, pudendo implementar encapsulamento, herança, polimorfismo, testes unitários, exceções e JavaDoc.

O trabalho foi implementado no IDE IntelliJ IDEA Ultimate utilizando o JDK na versão 15 no sistema Operativo Linux baseado em Ubuntu e pode ser acedido através do zip enviado ou pelo Repositório no GitHub.

# Bibliografia

[1] Prof. Doutor Tiago Candeias, 2021/2022, aulas Teórico-práticas da disciplina de Programação Orientada a Objetos, 2º ano, 1º semestre da licenciatura em Engenharia Informática do Instituto Superior Manuel Teixeira Gomes.

[2] K. Berry. Tex live documentation. TeX Live has been developed since 1996 by collaboration between the TeX user groups. [Online]. Available: https://tug.org/texlive/