FACULDADE DE CIÊNCIAS E TECNOLOGIAS
DA UNIVERSIDADE DE COIMBRA - FCTUC
DEPARTAMENTO DE ENGENHARIA INFORMÁTICA - DEI

Programação Orientada aos Objetos

Relatório
Projeto - POOTrivia

GONÇALO GAIO
Nº 2022224905

2023

INTRODUÇÃO:

No presente relatório serão relatadas todas as medidas e decisões de implementação em algoritmos mais complexos de forma a alcançar o resultado final desejado, cumprindo com todos os requisitos e usufruindo de todos os conceitos e conhecimentos adquiridos ao longo do semestre, na cadeira de POAO, para uma boa utilização do paradigma e princípios da programação orientada aos objetos.

Ideia Central:

Desenvolver uma classe responsável pela GUI (interface gráfica), a qual, irá usufruir da classe que gerencia o jogo em si, que, por sua vez, será implementada com base em packages que trabalham e gerenciam as perguntas de cada tipo.

Desta forma, foi decidido começar com o mais básico inicialmente, de forma a atacar problemas menores de cada vez, ou seja, inicialmente foi realizada a criação da package 'Questions' e foi feita a sua divisão em outras 3 packages: 'Art', 'Science' e 'Sport'.

Package Questions:

Nesta package, para além das demais packages, é encontrada a classe abstrata 'Question', a qual tem como objetivo representar todas as questões independente do seu tipo específico, assim, é nesta classe que o algoritmo mais complexo relativamente à manipulação das questões é feito e trabalhado, a leitura da questão, pontos base, escolha correta e todas as demais escolhas. Passo a explicar:

→ setQuestion(String type):
Algoritmo baseado no facto que todas as questões se encontram presentes no ficheiro de texto 'pootrivia.txt', assim, dado o tipo da questão e com o número da questão (número de 0 a 4 representando a posição da questão relativamente ao seu tipo, presente no ficheiro), previamente armazenado no próprio objeto, é percorrido todo o ficheiro até encontrar a linha com a questão pretendida, da qual é retirada o texto da questão, os pontos base e a(s) resposta(s) correta(s) e só por fim são adicionadas todas as possíveis opções de resposta a lista de opções. É importante ressaltar que para os géneros de 'Ski' e 'Natação' as respostas são apenas verdadeiro e falso, logo, ao encontrar a pergunta correta é lido, na mesma zona da resposta correta, um 1 que corresponde a V ou 0 que corresponde a F, sendo imediatamente após adicionadas as duas possíveis opções à lista de opções, escusando assim, de ler e armazenar mais linhas no ficheiro, já as questões do tipo Soccer, uma vez que apresentam duas respostas corretas dependendo da dificuldade, inicialmente, são armazenadas juntas, e separadas apenas quando a questão for criada com um determinado nível.

Nota: Em todos os casos onde as respostas não são de verdadeiro ou falso, as respostas são escolhidas e apresentadas de forma aleatória contendo obrigatoriamente sempre a resposta correta.

Package Questions.Art:

Aqui é presente a classe ‘ArtQuestion’, derivada da classe abstrata ‘Question’, que tem dois principais métodos:

→ getPoints(), onde é calculada a pontuação da pergunta, baseada nos pontos base da pergunta * a majoração das perguntas de arte (vezes 10).

→ setChoicesTo(short questionLevel), onde são designadas o número de opções de acordo com o quão avançado no jogo o jogador se encontra (passado para a função através de questionLevel), ou seja, se o jogador estiver na pergunta 1 ou 2 (antes da 3ª) são apresentadas apenas 3 possíveis escolhas, caso esteja na pergunta 3 ou 4, são apresentadas 4 escolhas, caso seja a 5 e última pergunta, então são apresentadas todas as 6 opções de resposta.

Package Questions.Science:

Aqui é presente a classe ‘ArtQuestion’, derivada da classe abstrata ‘Question’, que, diferente das demais questões, apresenta duas listas de resposta possíveis, porém com apenas uma resposta correta igual nas duas listas. Estas listas são identificadas e formadas no próprio construtor. Tal como as demais, esta classe também apresenta dois principais métodos:

→ getPoints(), onde é calculada a pontuação da pergunta, baseada nos pontos base da pergunta + a majoração das perguntas de ciências (mais 5).

→ setChoicesTo(short questionLevel), onde é designada a lista de opções de acordo com o quão avançado no jogo o jogador se encontra (passado para a função através de questionLevel), ou seja, se o jogador estiver na pergunta 1 ou 2 (antes da 3ª) são apresentadas as opções da lista das opções fáceis, caso contrário, são apresentadas as opções da lista das opções difíceis.

Package Questions.Sport:

Aqui são apresentadas as classes ‘SkiQuestion’, ‘SwimQuestion’, ‘SoccerQuestion’ e a classe abstrata ‘SportQuestion’. A classe abstracta ‘Sport’ é derivada da classe abstrata Question e por sua vez as demais, são derivadas da classe ‘SportQuestion’, construindo assim, um novo ramo de herança.

A classe abstrata ‘SportQuestion’ é responsável apenas por armazenar a sua majoração intrínseca (+3 à pontuação da pergunta) e permitir a subcategorização das perguntas acerca das três diferentes modalidades do desporto.

Quanto à classe ‘SkiQuestion’ e ‘SwimQuestion’, estas apresentam apenas um método de relevância: getPoints() o qual calcula os pontos totais da questão conforme a majoração indicada, acrescentando antes a majoração de desporto, e, uma vez que estas questões são de verdadeiro e falso, não são alteradas conforme a dificuldade, não sendo necessária a alteração das opções através do método setChoicesTo(short questionLevel).

Já quanto à classe ‘SoccerQuestion’, o mesmo não é verdade, esta apresenta novamente 2 métodos de relevância (getPoints() e setChoicesTo(short questionLevel)) e uma divisão de opções fáceis e difíceis, porém, à diferença das questões de ciência, estas perguntas apresentam respostas diferentes consoante a dificuldade, esta divisão de listas é efetuada no construtor, sendo apenas decidido mais tarde com o método setChoicesTo(short questionLevel).
Como referido esta classe apresenta novamente dois métodos de interesse, passo a explicar:

→ getPoints(), onde é calculada a pontuação da pergunta, baseada nos pontos base da pergunta + a majoração das perguntas de desporto (mais 5) + a majoração das perguntas de futebol (mais 1).

→ setChoicesTo(short questionLevel), onde é designada a lista de opções de acordo com o quão avançado no jogo o jogador se encontra (passado para a função através de questionLevel), ou seja, se o jogador estiver na pergunta 1 ou 2 (antes da 3ª) são apresentadas as opções da lista das opções fáceis e a resposta correta para essa mesma lista, caso contrário, são apresentadas as opções da lista das opções difíceis, assim como a resposta correta para a mesma.

Classe POOTrivia:

Esta é a classe que representa o jogo em si, onde as questões são manipuladas de forma a serem guardadas e escolhidas. Aqui são armazenados os dados referentes ao jogo como as questões que o jogador respondeu assim como as que acertou e errou, e é guardado ainda o nickname do jogador assim como a data e hora do término do seu jogo, quanto aos métodos, existem 3 mais importantes:

→ getTotalPoints(), onde através da lista de perguntas corretas é calculado o valor total dos pontos do jogo.

→ nextQuestion(List<Question> allQuestions), onde dada uma lista de questões possíveis, de forma aleatória, é escolhida uma nova questão (que ainda não tenha sido escolhida) para ser a próxima a ser apresentada ao jogador.

→ compareTo(POOTrivia game), onde dado um outro objeto de jogo, os dois são comparados conforme a sua pontuação total (utilizado para poder comparar e formar o top 3).

Classe GameGUI:

Esta classe é responsável pela interação gráfica do jogo, nela, é utilizada apenas uma frame (JFrame) na qual são colocados e retirados panels (JPanel's) de forma a atualizar o estado do jogo.

Inicialmente é apresentado ao jogador um 'Menu Inicial' onde o mesmo pode começar um novo jogo, ver o atual top 3 ou, ainda, ver os créditos do jogo.

Ao começar um novo jogo, um novo objeto POOTrivia é inicializado apresentando de forma interativa as questões através de label's (JLabel's) e as possíveis opções através de botões, que possuem 'ActionListeners' para indicarem se o jogador acertou ou errou a pergunta e agirem conforme. Caso acerte, é apresentado o total dos pontos que ganhou ao acertar, atualizando o contador total de pontos, caso contrário apenas é indicado que o jogador errou e aparece um botão para poder passar para a próxima pergunta do jogo, ou, se assim for o caso, para o final do jogo. Assim que a 5ª e última pergunta for respondida, ao clicar em 'Next', é perguntado o nome completo do jogador e, em seguida é gravado o jogo (em um ficheiro de objeto) e de acordo com os jogos guardados na pasta designada, é apresentado um top 3 assim como dois botões, um para permitir ao jogador voltar ao menu, outro para permitir jogar novamente.

Clicando em Top 3, no Menu, são lidos todos os ficheiros contendo os jogos na pasta designada e é realizada uma lista dos 3 melhores conforme as pontuações obtidas, esta é apresentada ao jogador juntamente com um botão que permite retornar ao Menu. Nota: Caso não haja jogos suficientes para preencher o top 3, (ex.: apenas existem 2 jogos gravados), então é apresentada uma linha para representar um espaço em branco naquela posição.

Por fim, mas não menos importante, ao clicar em créditos um breve texto feito através de labels (JLabel's) é apresentado ao jogador de forma a apresentar o autor e medidas/objetivos do jogo, assim como um botão 'Menu' que permite ao jogador retornar ao menu do jogo.