1 2 9 0

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D COIMBRA

RELATÓRIO

PooTrivia

Alexandre Rodrigues
nº2022249408
Rui Machado
Nº2022210627

Introdução

Este relatório documenta o desenvolvimento de um jogo de trivia em Java, aplicando os princípios da Programação Orientada a Objetos (POO). As classes essenciais, como LeitorFicheiro, Pergunta, P_Frame e Jogador, foram implementadas para criar a estrutura do jogo. O código abrange desde a eficiente leitura de perguntas até a lógica de pontuação, interação com o utilizador e a utilização de bibliotecas Java, como Swing e AWT, para criar uma interface gráfica interativa. A seguir, apresentamos uma visão geral das etapas-chave e decisões de design do projeto.

Metodologia

Para o desenvolvimento do código, foi necessário importar utilizadas várias bibliotecas e módulos, cada um desempenhando funções específicas nas classes:

- LeitorFicheiro:

    1. **import java.io.BufferedReader**: Leitura eficiente de caracteres de um fluxo de entrada (usado para ler arquivos de texto).
    2. **import java.io.FileReader;:** Leitura de caracteres de um arquivo de texto.
    3. **import java.io.IOException:** Tratamento de exceções para erros de entr ada/saída.
    4. **import java.util.ArrayList:** Implementação de uma lista dinâmica para adição e remoção eficientes de elementos.
    5. **import java.util.List:** Representação de uma sequência ordenada de elementos (usado em conjunto com ArrayList).
- Pergunta:

    1. **import java.io.Serializable:** Indica que uma classe pode ser convertida em uma sequência de bytes, facilitando operações como salvar e carregar objetos em arquivos ou transmiti-los pela rede.

- P_Frame:
    1. **import javax.swing.*:** Facilita a construção de interfaces gráficas usando o conjunto de bibliotecas Swing em Java.
    2. **import java.awt.*:** Fornece classes para criar elementos básicos de interfaces gráficas em Java através do Abstract Window Toolkit (AWT).
    3. **import java.awt.event.*:** Contém classes para lidar com eventos de interface gráfica, como cliques de mouse e pressionamentos de teclas.
    4. **import java.util.List**.

- projeto:
    1. **import javax.swing.***
    2. **import java.util.Collections:** Oferece métodos estáticos para operações de coleções em Java, incluindo ordenação, reversão e embaralhamento.
    3. **import java.util.List**
    4. **import java.awt.event.***

- Jogador:
    1. **import java.io.*:** Importa todas as classes do pacote java.io, que oferece suporte a operações de entrada e saída (I/O), como leitura e gravação de arquivos.
    2. **import java.text.SimpleDateFormat:** Importa a classe SimpleDateFormat do pacote java.text, que é utilizada para formatar datas e horas conforme um padrão específico.
    3. **import java.util.ArrayList**
    4. **import java.util.Collections;**
    5. **import java.util.Comparator:** Possibilita a utilização da interface Comparator do pacote java.util, que é empregada para definir a ordem de classificação de objetos, frequentemente usada em operações de ordenação de coleções.
    6. **import java.util.Date:** Importa a classe Date do pacote java.util, que representa um ponto específico no tempo. Essa classe é frequentemente usada para trabalhar com datas e horas em Java.
    7. **import java.util.List;**

Implementação

Como solicitado criamos o nosso ficheiro UML inicialmente o que nos ajudou a planificar as nossas ideias e a arranjar um ponto de partida e uma ideia da estrutura do nosso trabalho.A implementação do código deu-se em várias etapas:

- Começamos por definir as classes que iremos necessitar para o nosso projeto e chegamos à conclusão que seriam necessárias pelo menos oito sendo as principais as classes 'Pergunta' e 'P_Frame',a classe Pergunta é a nossa super class onde dela demos extend a outras 3 classes denominadas de Arte, Ciências e Desporto. contudo, para ir ao encontro com os requisitos do enunciado criamos mais classes que davam extend à desporto sendo estas Futebol, Natação e Ski.
- Em seguida criamos o ficheiro com perguntas dos temas dados devidamente formatado para depois lermos e separarmos para uma mais fácil utilização: "futebol; F; Qual o número do Haaland?; 10, 7, 9, 24; 2; 5"
- Na nossa super class começamos por definir as variáveis que precisávamos e em seguida criamos o nosso construtor para inicializar o objeto Pergunta
- Temos três classes que dão extend à nossa super como referimos previamente, sendo elas implementadas de forma que o construtor adicione pontos à pontuação associada à pergunta. No caso de ciências adiciona 5 pontos, desporto adiciona 3 e arte multiplica por 10.
- Como a parte do desporto tem vários tópicos diferentes foi necessário criar mais 3 classes que davam extend à desporto. Foram elas também implementadas de forma que o construtor adicione pontos à pontuação associada à pergunta. No caso de natação adiciona 10 pontos, futebol adiciona 1 e ski multiplica por 2.
- Criamos uma class para ler o ficheiro criado inicialmente com o intuito de ler as perguntas e as respostas. Começamos por iniciar uma lista e através de "try (BufferedReader br = new BufferedReader(new FileReader(nomeFicheiro))) "conseguimos ler o ficheiro. Depois criamos um loop e separamos pelos ";" .em seguida, através de um if verificamos se o tamanho dos dados era 6 (Ajuste para 6, porque há 5 ponto e vírgula) e caso fosse formatamos os dados. Ainda dentro do loop e do if convertemos strings para números e criamos as perguntas adicionando à lista. E finalmente através do "catch" tratos de exceções na conversão e fora do das condições anteriores outro para erro a ler o ficheiro.Criamos um método para criação de perguntas através do "Switch"e também criamos uma instância para perguntas adequadas.

- O construtor da classe P_Frame inicializa a interface gráfica do jogo de perguntas, recebendo como parâmetros a lista de perguntas e uma instância do jogador. Durante a inicialização, configura os componentes gráficos, como rótulos e botões, definindo o layout da janela. Adiciona um ouvinte para salvar o estado do jogo ao fechar a janela. A função exibirTop3() calcula e apresenta

o top 3 de jogadores por meio da classe JOptionPane. O método initComponents() configura os componentes gráficos iniciais da janela, preparando a interface para a interação com o utilizador. A função processarResposta() é central para a lógica do jogo, avaliando se há mais perguntas, verificando a resposta do jogador, atualizando a pontuação e avançando para a próxima pergunta. O método exibirPerguntaAtual() mostra a pergunta na interface, atualizando os textos dos botões com as opções possíveis e controlando a visibilidade com base no estado do jogo.

- Inicialmente, definimos os atributos essenciais para a implementação do código e criamos um construtor para inicializar as propriedades do jogador, incluindo nome, pontuação, data e hora, além das listas de perguntas acertadas e erradas. Em seguida, implementamos métodos como "obterPontos()", "obterDataHoraAtual()" e "obterNome()" para acessar informações específicas do jogador.
  As funções "acertouPergunta()" e "errouPergunta()" foram desenvolvidas para atualizar a pontuação e gerenciar as listas de perguntas acertadas e erradas. Introduzimos ainda métodos para salvar e carregar o estado do jogo, utilizando serialização para persistência de dados. Além disso, criamos funcionalidades como "obterIniciaisNome()" para extrair as iniciais do nome do jogador.
  A implementação inclui ainda métodos para imprimir informações do jogador e listas de perguntas. A função "carregarTodosEstadosJogo()" lê os estados existentes no diretório atual, enquanto "calcularTop3()" ordena e imprime os três melhores jogadores com base nas pontuações.

- O método principal "main" é o ponto de entrada do programa, iniciando por definir o nome do ficheiro contendo as perguntas do jogo. Utiliza a classe LeitorFicheiro para carregar essas perguntas a partir do ficheiro para uma lista. Em seguida, solicita ao jogador que insira o seu nome por meio de uma caixa de diálogo. Com o nome do jogador obtido, cria-se uma instância da classe Jogador. Finalmente, o jogo é inicializado chamando o método "iniciarJogo" com a lista de perguntas e o jogador recém-criado.
  O método "iniciarJogo" realiza duas ações principais. Inicialmente, embaralha a lista de perguntas utilizando a funcionalidade shuffle da classe Collections, garantindo que as perguntas apareçam de forma aleatória durante o jogo. Posteriormente, inicia a interface gráfica do jogo com a classe P_Frame, herdada de JFrame. A janela é configurada com um tamanho específico, centralizada no ecrã e tornada visível para o jogador. Adicionalmente, é adicionado um ouvinte de eventos de janela para imprimir as informações do jogador quando a janela é fechada.

Manual de Utilizador

Instruções Gerais:

Iniciar o Jogo:

Execute o programa e forneça o seu nome quando solicitado.
O jogo apresentará uma série de perguntas de diferentes categorias.

Responder às Perguntas:

Para cada pergunta exibida, selecione a opção que você acredita ser a resposta correta.
Clique no botão "Próxima Pergunta" para avançar para a próxima pergunta.

Pontuação:

Ganhe pontos acertando as perguntas. A pontuação varia com base na dificuldade e categoria da pergunta.
Erros não afetam sua pontuação, mas tente acertar o máximo possível para obter a pontuação máxima!

Final do Jogo:

O jogo consiste em responder a 5 perguntas. Ao final, sua pontuação total será exibida.
As informações do jogador serão salvas, e você terá a chance de visualizar o TOP 3 dos jogadores.

Dicas Importantes:

Dificuldades:

As perguntas têm diferentes níveis de dificuldade: Fácil, Médio e Difícil.

Categorias:

As perguntas estão divididas em categorias, como Desporto, Ciência, Arte, entre outras.

Estatísticas do Jogador:

Você pode verificar suas estatísticas, incluindo perguntas acertadas e erradas, após o jogo.

Salvar o Progresso:

Se desejar interromper o jogo e continuar posteriormente, feche a janela. Seu progresso será automaticamente salvo.

Comandos:

Selecionar Resposta:

Clique na opção desejada para selecionar sua resposta.

Próxima Pergunta:

Clique no botão "Próxima Pergunta" para avançar.

Finalizar Jogo:

O jogo será concluído após 5 perguntas. Siga as instruções finais exibidas na tela.

Visualizar TOP 3:

Durante a exibição das informações finais, clique em "OK" para visualizar o TOP 3 dos jogadores.

Conclusão

O projeto de trivia em Java, baseado em Programação Orientada a Objetos (POO), culminou num jogo bastante interativo. A implementação de classes como LeitorFicheiro, Pergunta, P_Frame e Jogador proporcionou uma estrutura organizada e modular. A utilização de bibliotecas Java, como Swing e AWT, contribuiu para uma

interface gráfica intuitiva. A lógica de pontuação e interação foi integrada de forma coesa, oferecendo uma experiência consistente aos utilizadores. A leitura eficiente de perguntas de ficheiro evidenciou a adaptabilidade do sistema.
Em suma, o projeto não só aprimorou habilidades em POO, como também demonstrou a aplicação versátil dessa abordagem.