

Paradigmas da programação

Trabalho prático: “Realzação de testes com qualificação automática”

Introdução

Trabalho realizado por:

1090674 Tiago neves, turma 1de

O presente relatório foi realizado no âmbito do Trabalho Prático “Realização de Testes com Qualificação Autónoma” da disciplina de Paradigmas de Programação.

O objectivo deste relatório é o de ajudar na compreensão do código e na explicação das decisões tomadas pelo programador ao longo da construção do programa.

O programa tem como objectivo a criação de uma interface gráfica que permita ao utilizador a realização de testes, apresentando no fim a sua classificação.

Estrutura do Programa

O programa é composto essencialmente por 2 partes: a parte gráfica e a parte do sistema.

A parte gráfica é constituída principalmente por 3 interfaces, a interface do menu, a interface do teste e a interface do resultado.

A parte do sistema possui principalmente a classe de leitura das perguntas do ficheiro, a criação dos objectos do tipo “pergunta” e a criação do ArrayList do tipo de objectos “pergunta” que vai guardar as perguntas criadas durante o tempo de execução, onde as 10 perguntas irão ser escolhidas para fazerem parte do teste.

No menu inicial do programa o utilizador escolhe os temas dos quais pretende incluir perguntas no teste. O utilizador dispõe de 5 temas, sendo eles: Matemática, História, Geografia, Informática e Ciências, podendo indicar se preferir o número de perguntas por cada tema.

Após a selecção dos temas são escolhidas aleatoriamente as perguntas de cada tema do ArrayList, sendo criada a interface Teste, que corresponde à interface onde serão questionadas as perguntas ao utilizador e onde ele as pode responder. Depois de concluído o teste é apresentada a interface final, onde podem ser encontrados os resultados do teste, avaliando assim a sua prestação mesmo.

O programa é composto por 14 classes, sendo feita de seguida uma pormenorização de cada uma, apresentando os seus atributos e os seus métodos.

Começando pela classe **ArrayPerguntas**, é uma classe que estende a classe ArrayList<Pergunta> e que tem como objectivo a instanciação do ArrayList de Perguntas que irá conter todas as Perguntas. Ela possui os seguintes métodos:

* Baralhar(): este método faz o “shuffle” do ArrayPerguntas recorrendo a classe Collections disponibilizada pelo Java para tal.
* escolherPerguntasTeste(): este método escolhe as 10 perguntas para o teste do ArrayPerguntas principal que contém todas as perguntas e adiciona-as a um novo ArrayPerguntas, que apenas irá conter as 10 perguntas do teste, recebendo por parâmetros um array de Strings com os temas escolhidos pelo utilizador e também um array de inteiros com o número de perguntas pretendido pelo utilizador para cada tema, estando os índices destes 2 arrays sincronizados. Para tal este método usa o método escolherPerguntasTema que irá ser descrito em seguida.
* escolherPerguntasTema(): este método adiciona ao novo ArrayPerguntas o número de perguntas escolhido pelo utilizador de apenas 1 tema, sendo chamado pelo método escolherPerguntasTeste() tantas vezes quanto o número de temas escolhidos pelo utilizador.

A classe **Ficheiro** permite a leitura do ficheiro que contém as perguntas, a criação dos objectos do tipo Pergunta e o seu armazenamento no ArrayPerguntas principal, dispondo de apenas um método:

* Ler Ficheiro(): este método lê o ficheiro “perguntas.txt” linha a linha. Para cada tipo de Pergunta, diferenciados por um inteiro, o método começa por ler o tema da pergunta e caso a linha seguinte se inicie com o número 1, o programa sabe que a Pergunta que irá ser lida de seguida é uma pergunta do tipo RespostaUnica. Lê de seguida o título da pergunta e um inteiro. Este inteiro indica o número de respostas que o utilizador vai ter á sua escolha, variando entre 3 a 5 respostas. De seguida efectua a leitura de tantas linhas quanto as indicadas pelo último inteiro referido, cada linha correspondendo a uma resposta. Por fim irá ler mais um inteiro que representa qual das respostas está certa e assim com estes dados instancia um objecto do tipo RespostaUnica e adiciona-o ao ArrayPerguntas.

Caso a linha se inicie com o número 2, o programa sabe que a Pergunta que irá ser lida de seguida é uma pergunta do tipo RespostaMultipla. O método de leitura é muito semelhante, apenas mudando a resposta certa, que neste caso pode ser mais que uma. Para isso é lida uma linha que contém 0’s e 1’s separados por vírgulas, representado os 1’s uma resposta certa e os 0’s uma resposta errada. Estes valores serão transformados em true ou false (1= true, 0= false) sendo adicionados a um array de booleans usando o método split() da Classe String para tal, array este que será enviado para o construtor da RespostaMultipla, junto da pergunta e das respostas.

Caso a linha se inicie com o número 3, o programa sabe que a Pergunta que irá ser lida de seguida é uma pergunta do tipo RespostaExtenso, sendo lida a pergunta e uma linha com as várias respostas possíveis, separadas por “;”, que irão ser tratadas de maneira análoga às respostas do tipo RespostaMultipla acima referidas.

Após criar um desses 3 tipos de perguntas, é lida a linha seguinte do ficheiro, que se trata de uma linha em branco criada apenas por aspectos visuais do ficheiro de texto, sendo todo este processo repetido até todas as perguntas do ficheiro se encontraram instanciadas e adicionadas ao ArrayPerguntas.

A classe **Pergunta** é uma classe abstracta, responsável pelo modelo genérico de constituição de uma pergunta. Ela tem como atributos a pergunta e o tópico de cada Pergunta, dispondo de um construtor para instanciar um objecto do tipo Pergunta com esses 2 atributos referidos. Dispõe dos seguintes métodos:

* getPergunta(): é um método abstracto e tem como objectivo retornar a String com a pergunta, a ser implementado nas suas subclasses.
* getNumeroRespostas():é um método abstracto e tem como objectivo retornar um inteiro com o nº de respostas que essa Pergunta contém, a ser implementado nas suas subclasses.
* getTopico():retorna a String com o tópico dessa Pergunta.

A classe **RespostaMultipla** permite instanciar Perguntas de escolha múltipla com mais que 1 resposta certa e é uma subclasse da classe Pergunta e possui 2 atributos. Um array de booleans resp\_certas com o tamanho do número de repostas que essa Pergunta tiver, que contém os valores true ou false consoante a resposta estiver correcta ou errada, respectivamente e também um array de Strings que contém os textos das respostas possíveis da Pergunta. Dispõe do construtor com os 2 atributos referidos passados por parâmetro, dos vários métodos get() que retornam os seus atributos e a implementação também dos métodos abstractos da sua superclasse, dispondo também do método ValidaResposta(), que recebendo um array de booleans que contém as respostas dadas pelo utilizador, vai comparar com o array que contém as respostas certas dessa Pergunta, retornando um inteiro (-1,0 ou 1) consoante a resposta esteja errada, o utilizador não tenha respondido (estando assim todas as posições do array que o utilizador enviou a “false”) ou a resposta esteja certa (todas as posições dos 2 arrays estão iguais), respectivamente.

A classe **RespostaUnica** permite instanciar Perguntas de escolha múltipla com apenas 1 resposta certa e é uma subclasse da classe Pergunta e possui 2 atributos. Um inteiro resp\_certa com o número da reposta certa dessa Pergunta e também um array de Strings que contém os textos das respostas possíveis da Pergunta. Dispõe do construtor com os 2 atributos referidos passados por parâmetro, dos vários métodos get() que retornam os seus atributos e a implementação também dos métodos abstractos da sua superclasse, dispondo também do método ValidaResposta(), que recebendo um inteiro, vai comparar com o inteiro que indica a resposta certa dessa Pergunta, retornando um inteiro (-1,0 ou 1) consoante a resposta esteja errada, o utilizador não tenha respondido (estando assim o inteiro que o utilizador enviou a 0) ou a resposta esteja certa (os 2 inteiros são iguais), respectivamente.

A classe **RespostaExtenso** permite instanciar Perguntas de resposta aberta com e é uma subclasse da classe Pergunta e possui 1 atributo. Um array de Strngs respostas\_certa que contém os as respostas possíveis a ser dadas pelo utilizador que estão correctas. Dispõe do construtor com o atributo referido passado por parâmetro, dos vários métodos get() que retornam o seu atributo e a implementação também dos métodos abstractos da sua superclasse, dispondo também do método ValidaResposta(), que recebendo uma String, vai comparar essa String com todas as Strings do array respostas\_certas que contém as respostas certas dessa Pergunta, retornando um inteiro (-1,0 ou 1) consoante a resposta esteja errada (a String dada é diferente de todas as Strings do array), o utilizador não tenha respondido (estando assim a String que o utilizador enviou vazia) ou a resposta esteja certa (a String dada é igual a pelo menos uma das Strings existentes no array), respectivamente.

A classe **PainelMultipla** é uma subclasse da classe JPanel que permite criar 1 painel que contém representada de uma forma visual 1 objecto RespostaMultipla, que irá ser usada na classe Teste para mostrar as perguntas ao utilizador e permitir que o mesmo as responda. Contém 2 atributos, um array de objectos JComponent (onde irão ser colocados as JLabels e as JCheckBoxes) e um objecto do tipo RespostaMultipla, que é a resposta a ser representada no teste. Dispõe de um construtor que recebe como parâmetros a RespostaMultipla referida e um inteiro, que irá representar o nº da pergunta no teste e também dos métodos get() para retornar os seus atributos.

A classe **PainelUnica** é uma subclasse da classe JPanel que permite criar 1 painel que contém representada de uma forma visual 1 objecto RespostaUnica, que irá ser usada na classe Teste para mostrar as perguntas ao utilizador e permitir que o mesmo as responda. Contém 2 atributos, um array de objectos JComponent (onde irão ser colocados as JLabels e os JRadioButtons) e um objecto do tipo RespostaUnica, que é a resposta a ser representada no teste. Dispõe de um construtor que recebe como parâmetros a RespostaUnica referida e um inteiro, que irá representar o nº da pergunta no teste, dos métodos get() para retornar os seus atributos e também de uma classe interior chamada TrataRadioButtons que contém o ActionListener para o tratamento da selecção dos JRadioButtons, apenas permitindo que 1 deles esteja seleccionado e também que seja possível desseleccionar todos os JRadioButtons para o caso de o utilizador estiver indeciso e tiver seleccionado uma resposta, mas preferir não responder à pergunta para não perder pontos caso erre.

A classe **PainelExtenso** é uma subclasse da classe JPanel que permite criar 1 painel que contém representada de uma forma visual 1 objecto RespostaExtenso, que irá ser usada na classe Teste para mostrar as perguntas ao utilizador e permitir que o mesmo as responda. Contém 2 atributos, um JTextField (caixa de texto onde o utilizador irá digitar a sua resposta) e um objecto do tipo RespostaExtenso, que é a resposta a ser representada no teste. Dispõe de um construtor que recebe como parâmetros a RespostaExtenso referida e um inteiro, que irá representar o nº da pergunta no teste e também dos métodos get() para retornar os seus atributos.

A classe **Menu** é uma subclasse da classe JFrame que permite criar 1 janela que contém representada de uma forma visual o menú principal, que contém as indicações para o teste e os 5 temas possíveis para as perguntas do teste listados, acompanhados de JCheckBoxes que servem para o utilizador indicar se pretende esse tema incluído no teste ou não, e de JTextFields para o utilizador introduzir o número de perguntas que pretende para esse tópico. Contém 5 atributos, um JButton (botão para fechar o menu e gerar o teste), um array de objectos do tipo JComponent (que irá guardar as JCheckBoxes, JLabels e JTextFields para cada tópico), um array de Strings que contém os 5 temas disponíveis para utilizador escolher, um array de inteiros (que contém o número de perguntas por tema introduzido pelo utilizador) e um array de Strings que irá conter apenas os temas que o utilizador seleccionou para o teste. Dispõe de um construtor que recebe como parâmetros uma String título para colocar na barra da janela criada e o ArrayPerguntas principal (que irá ser usado para gerar o teste, mais a frente) e também 2 classes interiores, uma TrataCaixasTexto que possui o ActionListener para as JCheckBoxes (apenas deixa no estado activo as JTextFields dos temas cuja JCheckBox estiver seleccionada) e uma TrataBotaoEnd que contém o ActionListener para o botão que fecha a janela e gera o teste, que verifica se as informações dadas pelo utilizador são coerentes, avisando-o caso contrário, guarda os temas seleccionados pelo utilizador bem como o nº de perguntas por tema, caso o utilizador o tenha indicado, fecha a janela do menu, escolhe as perguntas para o teste segundo as indicações do utilizador e gera a interface Teste onde o utilizador vai realizar o referido teste.

A classe **Relógio** é uma classe adaptada de um dos relógios disponibilizados no Moodle que irá criar um relógio apenas com os ponteiros dos segundos e minutos para ser utilizado na interface Teste. A única alteração adicional foi a adição da chamada do método terminarTeste() (que irá ser explicado mais à frente) ao ActionListener da classe interior MostradorListener, que quando o relógio chega ao tempo limite de realização do teste (10 minutos), pára o relógio e invoca esse referido método, forçando assim o fim do teste.

A classe **Teste** é uma subclasse da classe JFrame que permite criar 1 janela que contém representada de uma forma visual o teste, que contém num CardLayout as 10 Perguntas do teste com as opções de resposta dependendo do tipo de pergunta. Contém 6 atributos, 3 JButtons (1 botão para fechar o teste e gerar o resultado e 2 botões para navegar pelo CardLayout), um JPanel (que irá conter o CardLayout com as Perguntas), um JFrame que se refere ao próprio teste que serve para utilizar nos ActionListeners que mais a frente serão referidos e um double que é o valor da classificação final do teste. Dispõe de um construtor que recebe como parâmetros uma String título para colocar na barra da janela criada e um ArrayPerguntas com as 10 perguntas seleccionadas para o teste, dipões também de 2 classes interiores, uma TrataDeslocamento que possui o ActionListener para os JButtons de navegação pelo CardLayout e uma TrataBotaoEnd que contém o ActionListener para o botão que fecha a janela e gera o resultado, que invoca o método terminarTeste() para tal.

Este método tanto pode ser invocado pelo botão de terminar teste como pelo relógio, possuindo por essa razão uma variável boolean que é passada por parâmetro, que permite ao método saber se o teste vai ser terminado por ordem do utilizador ou pelo tempo ter terminado, sendo esta distinção apenas utilizada para a criação de uma String que irá ser apresentada na janela do resultado final junto com a classificação do teste. Este método invoca outro método da classe Teste, de nome corrigirTeste(), que vai verificar as respostas dadas pelo utilizador, invocando os métodos validaResposta() dos 3 tipos de Perguntas, somando ou subtraindo cotação à nota final do teste consoante a resposta esteja correcta ou errada; fecha a janela do teste, cria a referida String e instancia uma JanelaResult (que vai ser referida seguidamente) que contém o resultado do teste.

A classe **JanelaResult** é uma subclasse da classe JFrame que permite criar 1 janela que mostra ao utilizador a classificação obtida, finalizando assim o teste e o programa.

Para terminar existe a classe **Main** que se limita a criar o ArrayPerguntas principal, invocar o método LerFicheiro() da classe Ficheiro para preencher o ArrayPerguntas criado e instanciar a janela do Menu, sendo que as outras 2 janelas são chamadas nos ActionListeners dos seus botões ou componentes.

AMbiente de desenvolvmento

O presente programa foi desenvolvido num Toshiba Satellite L40-18L, com um processador Intel® Pentium® Dual-Core T2330 1.60 Ghz com 2GB de memória RAM DDR2 a 667 MHz. O sistema operativo é o Windows 7 Professional Edition 32-bits e o compilador utilizado foi: Java 1.6.0\_17; Java HotSpot(TM) Client VM 14.3-b01.

CoNclusões

Após a realização deste trabalho pode-se concluir que os objectivos mínimos pedidos pelo enunciado estão presentes no programa e em funcionamento correcto. Um aspecto a realçar é o facto de não terem sido adicionadas praticamente nenhumas funcionalidades extras, o que teria melhorado a qualidade do programa. No planeamento do trabalho estava planeado que o programa disponibilizasse a interface de acrescentar perguntas criadas pelo utilizador às perguntas já existentes, mas pelo facto de o grupo ser apenas de 1 elemento e terem sido encontrados alguns contratempos durante a programação levou a que, já muito em cima do prazo de entrega do trabalho, não houvesse tempo para a criação da referida interface.

Manual do utilizador

Na interface Menu o programa pode apresentar 4 diferentes janelas JOptionPane a informar o utilizador de erros na introdução dos dados.

* Se o utilizador não seleccionar nenhum tema e pressionar o botão para gerar o teste, o programa mostrará uma janela com a seguinte mensagem: “Tem que seleccionar pelo menos um tema!”.

O utilizador deve seleccionar pelo menos um tema para poder gerar um teste.

* Se o utilizador tentar introduzir 1 valor que não seja do tipo inteiro numa das caixas de texto para o número de perguntas de 1 dos tópicos, o programa mostrará uma janela com a seguinte mensagem: “O número de perguntas por tema deve ser do tipo inteiro.”.

O utilizador deve verificar os valores que introduziu nas caixas de maneira a que todos sejam números do tipo inteiro.

* Se o utilizador tentar introduzir 1 valor inteiro mas que seja maior que 10 ou menos que 0 numa das caixas de texto para o número de perguntas de 1 dos tópicos, o programa mostrará uma janela com a seguinte mensagem: “O número de perguntas por tema deve ser positivo e não maior que 10.”.

O utilizador deve verificar os valores que introduziu nas caixas de maneira a que todos sejam números do tipo inteiro positivos e menores que 10.

* Se nenhum dos problemas acima for encontrado o programa vai somar os números de perguntas por tema introduzidos e verificar se a soma é igual a 10, para poder gerar o teste com as 10 perguntas nas quantidades por temas pretendidas pelo utilizador. Se a soma for diferente de 10, o programa mostrará uma janela com a seguinte mensagem: “O número total de perguntas deve ser igual a 10.”.

O utilizador deve rectificar os valores que introduziu nas caixas de texto de modo a que a soma seja igual a 10.