

Programmation Orientée Objet Examen du 1^{er} septembre 2003

Durée 2h

Seuls documents autorisés : documents distribués et notes personnelles de cours

I

(Les questions 2 et 3 peuvent être traitées même si la question 1 ne l'a pas été.)

1. Ecrire un programme java qui permet de traiter des données saisies par l'utilisateur dans une interface graphique ainsi que l'on peut voir sur les copies d'écran suivantes. De plus, un click sur le bouton 'quitter' ou sur la case de fermeture doit provoquer l'arrêt du programme. Les classes seront nommées `bonjour.Fenetre`, `bonjour.GestionnaireExit` et `bonjour.Principale`.



2. On suppose que les fichiers sources `.java` se trouvent dans un répertoire `src` et les pseudo-code `.class` dans un répertoire `classes`. Les répertoires `src` et `classes` sont dans le répertoire `sept03`.

Donner les commandes Unix permettant de compiler et exécuter ce programme. On précisera dans quel répertoire on se trouve pour exécuter ces commandes.

(Cette question, ainsi que la suivante, peut être traitée même si la question 1 ne l'a pas été.)

3. On suppose maintenant que l'on utilise un gestionnaire de fermeture, faisant partie d'un paquetage `gestionnaires`, déjà compilé dans `utilitaires/classes/gestionnaires/GestionnaireExit.class`. Indiquer les modifications à faire pour les questions précédentes.

II

Cette partie généralise la partie I de plusieurs façons, ainsi que l'on peut voir sur la copie d'écran suivante.

The screenshot shows a Java Swing window titled "licence". At the top, there are input fields for "nom" (Lepetit), "prenom" (Vincent), and "année de naissance" (1983). Below these are fields for "physique" (15), "math" (14,5), "algèbre" (16), "chimie" (5), and "langues" (10). The "langues" field has a dropdown menu with options: anglais, allemand, and espagnol. At the bottom of the form are four buttons: "valider", "enregistrer", "terminer", and "quitter".

Below the form, there is a text area displaying the results of three actions:

- Après click sur le bouton 'valider' (question 1):** The text area shows the student's name, prenom, age (20 ans), and a list of notes: physique: 15, algèbre: 16, chimie: 5, anglais: 10. Below this, it shows the final note (average) of 11,5.
- Après click sur le bouton 'enregistrer' (question 2):** The text area shows the student's name, prenom, and age, followed by a message: "données mémorisées et relevé de notes enregistré dans le fichier Lepetit_Vincent".
- Après click sur le bouton 'terminer' (question 3):** The text area shows a summary of the final notes for the group, including the average of 11,72 for the 4 students and the average of 12,25 for the 4 subjects.

- L'interface graphique doit pouvoir gérer un ensemble quelconque de matières définies par le programme principal. Les données peuvent concerner successivement plusieurs étudiants appartenant à un groupe donné dans le programme principal. La fenêtre a pour titre le nom du groupe. (1) Au click sur le bouton 'valider', les données sont saisies et le relevé de note affiché dans la zone texte. (2) Au click sur le bouton 'enregistrer', les données sont enregistrées en mémoire (pour chaque étudiant et pour chaque matière). Le relevé de notes est enregistré dans un fichier dont le nom est celui de l'étudiant (Nom_Prenom). (3) Un click sur le bouton 'terminer' provoque l'affichage des notes finales de tous les étudiants, la moyenne de ces notes, ainsi que la moyenne des notes dans chaque matière.

- Outre la classe Fenetre, les classes utilisées pour ce travail sont :

(1) La classe principale Gestion, les classes Matiere et sa sous-classe MatiereOptionnelle, la classe PanelMatiere.

(2) Les classes Note, Etudiant et Groupe. Les notes sont des triplets (étudiant, matière, valeur). Les données relatives à un étudiant incluent son relevé de notes. Les données relatives à une matière incluent les notes obtenues par tous les étudiants à cette matière.

- La classe principale, la documentation des classes et certains attributs sont donnés en annexe.

Questions

1. click sur 'valider'.

Compléter le programme concernant la déclaration et l'assemblage des composants (déclaration des composants, constructeur de la classe Fenetre, classes Matiere, MatiereOptionnelle et PanelMatiere), et la saisie des données au click sur le bouton 'valider'.

A cette étape il n'y a pas de création d'instance de la classe Etudiant ni d'instance de la classe Note, les valeurs des notes sont seulement rangées dans le tableau de double notes, notes[i] étant la note dans la matière tabMatiere[i]. On supposera que toutes les données ont été correctement saisies.

2. Click sur 'enregistrer'.

Cette question peut-être traitée même si la précédente ne l'a pas été, il suffit de savoir que les données ont été enregistrées dans les attributs de la classe Fenetre, en particulier que les valeurs de notes données dans les champs adéquats ont été enregistrées dans le tableau notes.

Compléter le programme réagissant au click sur le bouton 'enregistrer' créant ou modifiant l'instance de la classe Etudiant et enregistrant les données de l'étudiant et les notes des matières, ainsi qu'il est indiqué dans la documentation de la méthode actionPerformed de la classe Fenetre.

3. Click sur 'terminer'.

Compléter le programme terminant l'affichage au click sur le bouton 'terminer'.

ANNEXE

```

/** programme principal permettant la gestion d'un groupe d'étudiants
 * - les matières, qui peuvent être optionnelles sont données une fois pour
 * toutes dans ce programme principal
 * - les données relatives aux étudiants sont gérées à partir d'une fenetre */
public class Gestion {
    public static void main(String[] args) {
        Groupe gr = new Groupe("licence");
        Matiere[] tabMatières = new Matiere[4];
        tabMatières[0] = new Matiere("physique");
        String[] tabOptions0 = {"algèbre", "géométrie"};
        tabMatières[1] = new MatiereOptionnelle("math", tabOptions0);
        tabMatières[2] = new Matiere("chimie");
        String[] tabOptions1 = {"anglais", "allemand", "espagnol"};
        tabMatières[3] = new MatiereOptionnelle("langues", tabOptions1);
        Fenetre f = new Fenetre(gr, tabMatières);
        f.pack();
        f.show();
    } }

-----
/** Fenêtre permettant, pour un ensemble de matières donné, de
 * - saisir les données relatives à un étudiant (identité, groupe, notes)
 * - générer son relevé de notes
 * - mémoriser ces données
 * - calculer des moyennes, par étudiant et par matière */
public class Fenetre extends Frame implements ActionListener {
    // identité
    ... (question 1)
    // matières et notes
    private Matiere[] tabMatières;
    private PanelMatiere[] tabPanel;
    private TextField[] tfNotes;
    private double[] notes;
    private boolean complet; // sera égal à true ssi toutes les données
    // (sauf éventuellement le prénom) ont été correctement remplies (non demandé dans l'exercice)
    private String releve;
    // boutons
    ... (question 1)
    // zone de texte
    ... (question 1)
    /** crée une fenêtre de titre le nom du 'groupe' pour la lecture des données
     * relatives aux étudiants de ce groupe et à leurs notes dans les matières de
     * 'tabMatières' */
    public Fenetre(Groupe groupe, Matiere[] tabMatières) {... (question 1)}
    /** au click sur le bouton 'valider'
     * - lit les données correspondant aux différents champs, construit et
     * affiche le relevé de notes dans la zone de texte (question 1)
     * - pour chaque donnée (sauf prénom) incomplète ou incorrecte,
     * affiche un message adéquat et continue (question 4)
     * (on ne crée pas d'instance d'étudiant à cette étape)
     * au click sur le bouton 'enregistrer'
     * - si les données sont complètes, enregistre l'étudiant et les notes, enregistre le relevé de
     * notes dans un fichier dont le nom est constitué du nom et du prénom (vide si absent) de
     * l'étudiant et affiche un message dans la zone de texte (question 2)
     * - si les données sont incomplètes, affiche seulement un message dans la
     * zone de texte (non demandé dans l'exercice)
     * au click sur le bouton 'terminer'
     * - affiche dans la zone de texte les notes finales de tous les étudiants du groupe et la
     * moyenne de ces notes, ainsi que les moyennes des notes de chaque matière (question 3)
     * au click sur le bouton 'quitter' arrête le programme
     */
    public void actionPerformed(ActionEvent e) {...} }

-----
/** classe implémente une matière et les notes obtenues à cette matière */
public class Matiere {
    private String nom;
    private Vector vecNotes ;
    /** crée une matière de nom 'nom' */
    public Matiere(String nom) {...}
    /** renvoie la moyenne des notes obtenues à cette matière */
    public double moyenne() {...}
    /** renvoie le nom de cette matière */
    public String getNom() {...}
    /** renvoie le vecteur des notes obtenues à cette matière */
    public Vector getVecNotes() {...} }

-----
/** sous-classe de Matiere implémentant les matières optionnelles */
public class MatiereOptionnelle extends Matiere {
    private String[] options;

```

```

/** crée une matière optionnelle de nom 'nom' et d'options 'options' */
public MatiereOptionnelle(String nom, String[] options) {...}
/** renvoie les options de cette matière optionnelle */
public String[] getOptions() {...}    }

-----

/** sous-classe de Panel comprenant les composants relatifs à une matière
 * optionnelle ou non */
public class PanelMatiere extends Panel{
    private TextField tfnote = new TextField(10);
    private CheckboxGroup cbg = new CheckboxGroup();
    /** crée un Panel comportant
     * - le nom de cette matière
     * - les options à cocher dans le cas d'une matière optionnelle
     * - un champ de lecture pour la valeur de la note
     */
    public PanelMatiere(Matiere matiere) {...}
    /** renvoie le champ de lecture de ce Panel */
    public TextField getTfnote() {...}
    /** renvoie le CheckboxGroup de ce Panel */
    public CheckboxGroup getCbg() {...}    }

-----

/** classe implémentant des notes constituées de triplets (étudiant, matière, valeur) */
public class Note {
    private Etudiant etu;
    private Matiere mat;
    private double val;
    /** crée une note de valeur 'val' pour l'étudiant 'etu' et la matière 'mat' */
    public Note(Etudiant etu, Matiere mat, double val) {...}
    /** renvoie la matière de cette note */
    public Matiere getMat() {...}
    /** renvoie la valeur de cette note */
    public double getVal() {...}    }

-----

/** classe implémentant les données relatives à un étudiant (identité, groupe,
 * notes, relevé de notes) */
public class Etudiant {
    private Groupe groupe;
    private String nom;
    private String prenom;
    private int age;
    private Note[] tabNotes; // tableau de triplets (étudiant, matière, valeur)
    private String releve;
    /** construit un étudiant de nom 'nom' et prénom "prenom" */
    public Etudiant(String nom, String prenom) {...}
    /** S'il n'y a pas d'étudiant 'nom' 'prenom'
     * 1) crée cet étudiant avec son 'groupe', son 'age', ses notes 'valNotes',
     * et son relevé de notes 'relevé'
     * (Attention :
     * 'valNotes' est un tableau de double,
     * 'this.tabNotes' est un tableau d'instances de la classe Note,
     * c'est-à-dire de triplets (etudiant, matière, valeur))
     * 2) enregistre les notes dans chaque matière de 'tabMatiere'.
     ** S'il y a déjà un étudiant 'nom' 'prenom'
     * 1) enregistre les nouvelles valeurs de toutes les autres données
     * 2) met à jour les notes correspondantes dans les matières
     */
    public static Etudiant enregistrer(Groupe groupe, String nom, String prenom,
        int age, Matiere[] tabMatiere, double[] valNotes, String releve) {...}
    /** renvoie la moyenne des notes (note finale) de cet étudiant */
    public double moyenne() {...}
    /** renvoie le nom de cet étudiant */
    public String getNom() {...}
    /** renvoie le prénom de cet étudiant */
    public String getPrenom() {...}    }

-----

/** classe implémentant un groupe d'étudiants */
public class Groupe {
    private String nom;
    private Vector vecEtudiants = new Vector();
    /** crée un groupe de nom 'nom' */
    public Groupe(String nom) {...}
    /** calcule la moyenne des notes finales des étudiants de ce groupe */
    public double moyenne() {...}
    /** renvoie le nom de ce groupe */
    public String getNom() {...}
    /** renvoie le vecteur des étudiants de ce groupe */
    public Vector getVecEtudiants() {...}
    /** renvoie une représentation sous forme de chaîne de caractères de ce groupe comportant la liste
     * des étudiants avec leur note finale et la moyenne des notes finales des étudiants */
    public String toString() {...}

```