Examen (1h45, avec documents)

Nom : **Prénom :**

— En cas de doute sur le sujet, notez vos choix sur la copie. Aucune réponse ne sera donnée par le surveillant.
— Il est conseillé de lire complètement le sujet avant de commencer à y répondre !
— Barème indicatif :

Exercice	1	2	3	4
Points	3	9	4	4

Exercice 1 : Exceptions

On considère le programme Java du listing 1. Il compile sans erreur.

1.1 Indiquer ce qui s'affiche après l'exécution de chacune des commandes suivantes :

```
java Main -5
java Main 7
java Main 43
java Main dix
```

1.2 On veut remplacer les deux RuntimeException par Exception. Indiquer, en annotant le listing 1 du sujet, les modifications à apporter pour que le programme compile. On ne changera pas le comportement du programme.

Listing 1 – La classe Main

```
class A extends RuntimeException {}
class B extends RuntimeException {}
class Main {
   static public void main(String[] args) {
      int i = Integer.parseInt(args[0]);
      verifier1(i);
      try {
         verifier2(i);
      } catch (A e) {
         System.out.println("A");
      } catch (B e) {
         System.out.println("B");
      } finally {
         System.out.println("F");
   }  }

   static void verifier1(int i) {
      if (i < 0) {
         throw new A();
   }  }

   static void verifier2(int i) {
      if (i > 10) {
         throw new B();
}  }  }
```

1 Autour d'un annuaire

Exercice 2 : Contact
On considère un contact spécifié par l'interface ci-après.

```
1  /** Définition d'un contact. */
2  public interface Contact {
3     /** Obtenir le nom de ce contact. */
4     String getNom();
5
6     /** Obtenir le téléphone de ce contact. */
7     String getTelephone();
8
9     /** Changer le nom de ce contact. */
10    void setNom(String nom);
11
12    /** Changer le téléphone de ce contact. */
13    void setNumero(String numero);
14 }
```

2.1 *Documentation.*

2.1.1 Les commentaires de documentation sont importants. Expliquer pourquoi ils sont essentiels pour une interface.

2.1.2 Indiquer ce qui manque dans les commentaires de documentation de l'interface Contact.

2.2 *Réalisation.* Écrire une réalisation de l'interface Contact, appelée ContactSimple, en respectant les contraintes suivantes :

1. Le constructeur prend en paramètre le nom et le téléphone du contact.
2. Le constructeur et les méthodes setNom et setNumero lèveront l'exception hors contrôle IllegalArgumentException si la chaîne en paramètre n'a pas au moins un caractère.
3. L'égalité logique telle que définie dans la classe Object (méthode equals(Object)) devra être redéfinie de manière à ce que deux contacts soient égaux s'ils ont même nom et même numéro de téléphone.

2.3 *Test.* Écrire une classe de test JUnit qui comprend les tests suivants :

— Si on change le téléphone du contact (XC, 2186) en 2187, alors le téléphone de ce contact est 2187 et son nom est toujours XC.
— Le contact (XC, 2186) est logiquement égal à un nouveau contact (XC, 2186).
— Changer le nom du contact (XC, 2186) avec une chaîne vide conduit à l'exception IllegalArgumentException.

2.4 *Utilisation.* On considère une méthode m1 dont la signature est la suivante :

```
1     void m1(Contact m);
```

et le code suivant :

```
1     public static void main(String args) {
2        // Création d'un contact c1 (XC, 2186)
3        Contact c1 = new Contact("XC", "2186");
4
5        // Appel de m1 avec c1
6        ...
7     }
```

Indiquer au moins deux techniques qui peuvent être utilisées pour être sûr que l'objet c1 de la méthode `main` ne sera pas modifié par l'appel de m1 (quelque soit le code de m1).

Exercice 3 : Annuaire simplifié

En s'appuyant sur la notion de contact définie dans l'exercice précédent, on souhaite maintenant définir un annuaire dont l'interface est donnée au listing 2. L'annuaire doit respecter les contraintes suivantes :

— Pour un numéro de téléphone donné, il ne peut y avoir qu'un seul contact (un numéro de téléphone ne peut correspondre qu'à un seul nom).
— Pour un nom, il peut y avoir plusieurs contacts (un même nom peut être associé à plusieurs numéros de téléphone).
— Si on enregistre un contact dont le nom ou le numéro de téléphone est déjà présent dans l'agenda, alors les informations de l'annuaire sont mises à jour en respectant les règles précédentes.

Listing 2 – L'interface Annuaire

```
1  import java.util.*;
2
3  public interface Annuaire extends Iterable<Contact> {
4
5     /** Enregistrer dans cet annuaire. */
6     void enregistrer(Contact c);
7
8     /** Supprimer le contact c de cet annuaire. */
9     void supprimer(Contact c);
10
11    /** Obtenir tous les contacts de cet annuaire qui portent ce nom. */
12    Set<Contact> fromNom(String nom);
13
14    /** Obtenir le contact de cet annuaire qui a ce numéro de téléphone. */
15    Contact fromTelephone(String telephone);
16 }
```

3.1 *Implantation.* Dans cette question on supposera qu'un Contact est immuable (pas de méthode de modification, donc pas de `setNom` ou de `setTelephone`).

3.1.1 Expliquer pourquoi cette hypothèse simplifie la réalisation de Annuaire.

3.1.2 Indiquer les structures de données de l'API collection utiles pour réaliser cet Annuaire.

3.1.3 Écrire la classe AnnuaireCollection correspondante.

3.2 On veut pouvoir écrire le code suivant parcourant tous les contacts de l'annuaire.

```
1  public class AnnuaireUtil {
2     static public void afficher(Annuaire annuaire) {
3        for (Contact c : annuaire) {
4           System.out.println(c);
5        }
6     }
7  }
```

Indiquer et faire les modifications nécessaires à Annuaire et AnnuaireCollection.

Exercice 4 : IHM Swing

Compléter le code du listing 3 pour obtenir l'IHM de la figure 1. On rendra actifs les trois boutons de l'interface.

Listing 3 – Squelette de la classe AnnuaireSwing

```
1  import javax.swing.*;
2  import java.awt.*;
3  import java.awt.event.*;
4
5  public class AnnuaireSwing {
6
7     final private JFrame fenetre = new JFrame("Annuaire");
8     final private JTextField nom = new JTextField(10);
9     final private JTextField tel = new JTextField(10);
10    final private JButton quitter = new JButton("Quitter");
11    final private JButton supprimer = new JButton("Supprimer");
12    final private JButton enregistrer = new JButton("Enregistrer");
13    final private Annuaire annuaire;
14
15    public AnnuaireSwing(final Annuaire annuaire) {
16       this.annuaire = annuaire;
17
18       fenetre.pack();
19       fenetre.setVisible(true);
20    }
21
22    public static void main(String[] args) {
23       new AnnuaireSwing(new AnnuaireCollection());
24    }
25
26 }
```

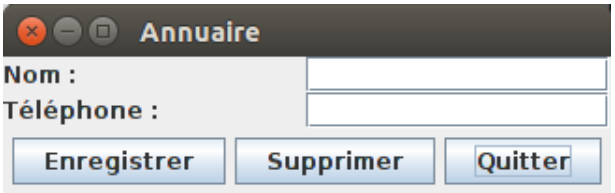

FIGURE 1 – IHM de l'annuaire