Université de Nice–Sophia Antipolis — Vendredi 28 mai 2004
E.S.I.N.S.A. 1ère Année

Contrôle de TP n°2a

Documents non autorisés — Durée : 2h

Attention : la qualité des commentaires (avec notamment, la présence des antécédents et conséquents avant chaque méthode), les noms donnés aux variables, l'utilisation à bon escient des majuscules et la bonne indentation rentreront pour une part **importante** dans l'appréciation du travail.
Les exercices ne sont pas tous indépendants. Cependant, si vous n'avez pas réussi à écrire la méthode demandée à l'un des exercices, vous pouvez faire les autres exercices en considérant que vous disposez quand même de ladite méthode. Le sujet comporte un verso.

Avant de commencer

Vous mettrez votre solution dans votre compte sur `djinn` dans le répertoire `Calculatrice`. Ce répertoire contient aussi les fichiers `Nombre.java` et `Operation.java` qui vous seront nécessaires pour la suite.

Nombres entiers relatifs

Les nombres que vous manipulez usuellement disposent tous des quatre opérations arithmétiques classiques (addition, soustraction, multiplication et division). Pour exprimer cela, les classes représentant des nombres vont implémenter une interface `Nombre` (*cf.* le fichier `Nombre.java`) qui déclare ces quatre opérations. De plus, pour simplifier, vous considérerez que ces opérations ne sont définies qu'entre nombres du même type : par exemple, bien qu'il puisse exister des classes implémentant `Nombre` pour les réels ou les rationnels, un entier ne pourra être divisé que par un autre entier et le résultat sera un entier.

Les exercices qui suivent vont vous guider dans la réalisation d'une classe pour les nombres entiers relatifs. En parallèle, vous testerez vos méthodes à partir d'une classe `Calculatrice` contenant la méthode `main`.

La classe Entier

1) Définissez la classe publique `Entier` qui contient un attribut privé `valeur` de type `int`. Ajoutez-lui deux constructeurs publics :

1.a) Un constructeur à un paramètre de type entier. Ce paramètre sera la valeur du nouvel objet `Entier`.

1.b) Un constructeur à un paramètre de type chaîne de caractères. Vous supposerez que cette chaîne est la représentation correcte d'un entier qui sera la valeur du nouvel objet `Entier`. Vous penserez à utiliser les méthodes disponibles dans la classe `Integer` de l'API Java.

2) Ajoutez une méthode publique `valeur` qui retourne la valeur de l'`Entier` courant.

3) Redéfinissez la méthode `toString` pour que la représentation de l'`Entier` courant soit de la forme : `valeur`.

4) Faites maintenant les modifications nécessaires de la classe `Entier` pour qu'elle implémente l'interface `Nombre`.

4.a) Pour toutes les méthodes de l'interface vous supposerez que le `Nombre` passé en paramètre est un objet de type `Entier` (autrement dit, vous ne vous soucierez pas des exceptions lors de conversions explicites de type). Le résultat sera un nouvel objet `Entier`. Enfin, lors d'une division par zéro vous émettrez une exception du type `ArithmeticException` avec pour message : Erreur, division par zéro !

4.b) Dans la classe `Calculatrice`, créez deux objets `Entier` de valeur 35 et 0 en utilisant chacun des constructeurs disponibles. Ensuite, écrivez l'instruction qui permettrait d'afficher le résultat de la division de 35 par 0 grâce à la méthode `sur`. Vous ferez en sorte que cette instruction affiche un message d'erreur *mais ne provoque pas l'arrêt du programme.*

5) Les nombres de Harshad sont les entiers naturels divisibles par la somme de leurs chiffres. Par exemple, 18 (divisible par $8+1$) est un nombre de Harshad , mais pas 11 (non divisible par $1+1$).

5.a) Ajoutez dans `Entier` la méthode `public boolean harshad ()` qui retourne vrai si la valeur de l'`Entier` courant est un nombre de Harshad et faux sinon.

5.b) Testez la méthode `harshad` dans `Calculatrice`.

Calculatrice graphique

L'objectif de cette section est de programmer une calculatrice à notation polonaise inverse sur une pile de deux éléments. Contrairement à ce que laisse supposer son nom, la notation polonaise inverse est une méthode de calcul simple : au lieu de noter par exemple l'addition `3 + 2`, on l'écrit `3 2 +`. L'intérêt dans une calculatrice est que vous n'avez pas besoin de vous souvenir de l'opération désirée pendant que l'utilisateur saisit la deuxième opérande.

Les captures d'écran des figures 1 à 5 illustrent le comportement désiré de la calculatrice.

La classe Calculatrice

La première étape de l'implémentation est de compléter la classe `Calculatrice`. Cette classe comprend deux champs privés `nombre` et `saisie` de type `Nombre`.

6) Écrivez la méthode d'initialisation `private void initialise ()` qui met `nombre` à `null` et `saisie` à `0`.

Écrivez ensuite un constructeur sans paramètre qui se contente d'appeler `initialise`.

7) On souhaite maintenant permettre la saisie de chiffres et d'opérations dans cette calculatrice.

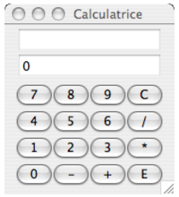

FIG. 1 – La calculatrice initialisée.

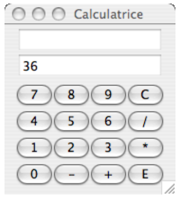

FIG. 2 – Après avoir pressé sur les boutons '3' et '6' successivement.

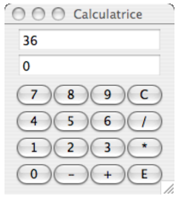

FIG. 3 – Après avoir pressé sur le bouton 'E'.

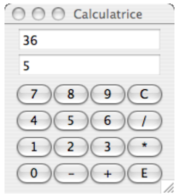

FIG. 4 – Après avoir pressé sur le bouton '5'.

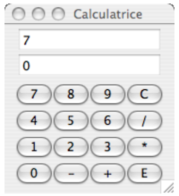

FIG. 5 – Après avoir pressé sur le bouton '/'.

7.a) Écrivez la méthode `public void saisie (int c)` qui augmente d'un chiffre le nombre placé dans le champ `saisie`. Par exemple, si `saisie` vaut `3` et le paramètre `c` vaut `6`, alors à l'issue de cette saisie le champ `saisie` doit valoir `36`.

7.b) La classe `Operation` fournie définit sept champs statiques publics :
- la constante `Operation.INIT` indique qu'il faut réinitialiser la calculatrice ;
- la constante `Operation.ENTREE` indique qu'il faut faire passer le contenu du champ `saisie` dans le champ `nombre` et mettre `0` dans `saisie` ;
- les constantes `Operation.PLUS`, `Operation.MOINS`,`Operation.FOIS` et `Operation.SUR` peuvent indiquer deux comportements :
  - soit `nombre == null`, et alors elles se contentent de faire passer le contenu du champ `saisie` dans le champ `nombre` et de mettre `0` dans `saisie` comme le ferait `Operation.ENTREE`,
  - soit elles font effectuer l'opération voulue entre `nombre` et `saisie`, mettent le résultat dans `nombre` et mettent `saisie` à `0`.

Écrivez la méthode `public void saisie (Operation operation)` qui réalise une opération.

8) Deux méthodes restent à écrire pour permettre l'affichage du contenu des champs `nombre` et `saisie` : les méthodes `nombre ()` et `saisie ()` qui retournent chacune une chaîne de

caractères correspondant aux nombres présents dans ces champs.

La classe Interface

La classe `Interface` définit la fenêtre qui permet d'interagir avec la calculatrice. Elle comprendra donc un champ `calc` de type `Calculatrice`.

Par ailleurs, elle devra permettre un affichage respectant les captures d'écran des figures 1 à 5. Celui-ci a été obtenu à l'aide de deux objets `java.awt.TextField` pour l'affichage des champs `calc.nombre` et `calc.saisie`. Un bouton a été crée pour chaque chiffre possible, et un pour chaque action possible. Tous les boutons sont regroupés dans un `java.awt.Panel` utilisant un `java.awt.GridLayout`.

9) Écrivez le constructeur de la classe `Interface`.

Il doit créer des champs de texte de type `java.awt.TextField` non éditables grâce à la méthode `setEditable` de `TextField`.

Il doit aussi créer la totalité des boutons. Les textes à donner aux boutons des opérations peuvent être obtenus par des appels à la méthode `toString` : par exemple, `Operation.INIT.toString()` retourne `"C"`.

Pour finir, le constructeur affiche la fenêtre.

10) Associez des actions à chacun des boutons :
- les boutons des chiffres appellent la méthode `saisie` de la classe `Calculatrice` avec le chiffre correspondant ;
- les boutons d'opérations le font avec la constante de la classe `Operation` correspondante.

La classe `Interface` peut faire office de *listener* pour les boutons.

Dans tous les cas, il faut mettre à jour l'affichage des deux `TextField`s *via* la méthode `setText` après une pression sur un bouton.

Écriture du `main`

Il ne vous reste plus qu'à écrire la méthode `main` pour pouvoir utiliser votre calculatrice.