

Université Paris-Est Créteil  
IUT de Sénart-Fontainebleau  
Département informatique  
IS3  
Algorithmique Programmation  
2013–2014

Partiel  
Durée : 2 heures

Seuls sont autorisés, à titre de documents, les listings comprenant explicitement le nom de l'étudiant (et celui-ci seulement) et les notes manuscrites.

Exercice 1.- (Visualisation du tri à bulle)

*Dans l'entreprise dans laquelle vous faites votre stage, nn vous demande de faire un exposé sur les différentes méthodes de tri. Pour cela, vous voulez visualiser les étapes grâce à une animation.*

Écrire un applet Java qui initialise de façon aléatoire un tableau de 20 entiers compris entre 1 et 100, qui affiche un bouton étiqueté 'Étape suivante' et qui, à chaque étape du tri à bulle pour ce tableau, visualise le contenu de ce tableau par 20 segments verticaux, le segment "actif" étant rouge.

Exercice 2.- (Menu déroulant)

Écrire une application Java qui affiche un cadre (fermant) de titre « Cadre avec menus déroulants », dans lequel apparaissent deux menus déroulants « Langage » et « Couleur », les items du premier étant « English » et « French », ceux du second étant « rouge », « vert » et « bleu ».

Au départ il n'y a rien dans la partie active du cadre. Lorsqu'on choisit un langage, « Hello! » ou « Bonjour » s'affiche (en noir si aucune couleur n'a été sélectionnée auparavant). Lorsqu'on choisit une couleur, le texte sera affiché dans cette couleur.

[On pourra venir l'objet panneau d'une méthode `change(String, Color)` pour faciliter la mise en œuvre.]

Exercice 3.- (Chiffrement de Vigenère)

*Lorsqu'on veut transmettre un message (une chaîne de caractères, dit **message en clair**) à une personne sans qu'un tiers puisse en comprendre le sens, on **chiffre** celui-ci, en utilisant une méthode et un paramètre (la **clé**) pour obtenir un **message chiffré**. Le destinataire doit connaître la méthode de chiffrement mais surtout la clé pour le **déchiffrer**, c'est-à-dire retrouver le message en clair.*

*Dans le **chiffrement de Vigenère**, le paramètre est une (petite) chaîne de caractères, par exemple 'essai'. Pour chiffrer 'patrick' sur l'alphabet des lettres minuscules, par exemple :*

*- Pour la première lettre du message, 'e' étant la cinquième lettre de l'alphabet, 'p' est remplacé par la quatrième lettre qui suit 'p', c'est-à-dire 't'. On remarque que 'a' donne 'e'.*

- Pour la seconde lettre, on fait de même mais avec un décalage dicté par la seconde lettre de la clé. Ici 's' étant la dix-neuvième lettre de l'alphabet, 'a' est remplacé par la dix-huitième lettre après 'a', soit 's'.

- La lettre 't' est de même remplacée par la dix-huitième lettre qui suit 't', c'est-à-dire 'l', le suivant de 'z' étant 'a'.

- 'a' étant la première lettre de l'alphabet, 'r' est remplacé par elle-même, c'est-à-dire 'r'.

- Le 'i' de la clé étant la neuvième lettre de l'alphabet, le 'i' du message (c'est par hasard que ce sont les mêmes) est remplacé par la huitième lettre qui suit 'i', c'est-à-dire 'q'.

- Pour la lettre suivante du message, on a épuisé toutes les lettres de la clé, on revient alors à la première lettre de la clé. Comme nous l'avons déjà dit, 'e' étant la cinquième lettre de l'alphabet, 'c' est remplacé par la quatrième lettre qui suit 'c', c'est-à-dire 'g'.

- Enfin 's' étant la dix-neuvième lettre de l'alphabet, 'k' est remplacé par 'c'.

Le chiffrement de 'patrick' est donc 'tslrqgc'.

Dans toute la suite on considère l'alphabet de 26 caractères { 'a', 'b', ..., 'z' }.

- 1°) Le principe du déchiffrement est simple (à condition de posséder la clé).

Les principes de chiffrement et de déchiffrement sont simples à comprendre mais leur exécution à la main est pénible. On se propose donc d'utiliser un programme Java pour nous aider.

Quel est le message en clair correspondant à 'fgfjwyj' en utilisant la même clé que ci-dessus.

- 2°) En se rappelant que les 26 lettres de 'a' à 'z' sont codées par un certain code (par exemple ASCII ou Unicode), qui est un entier naturel tel que 'b' soit le code de 'a' plus un et ainsi de suite, donner une formule mathématique exprimant le chiffrement et une autre le déchiffrement.

- 3°) Écrire une méthode :

String chiffre(String, String)

prenant en argument un message en clair et une clé et renvoyant le message codé.

- 4°) Écrire une méthode :

String dechiffre(String, String)

prenant en argument un message chiffré et une clé et renvoyant le message en clair correspondant.

[Faites attention au fait qu'en Java, on a :  $-27 \% 26 = -1$  et non 19 comme on pourrait s'y attendre !]

- 5°) Écrire un programme Java qui demande une clé, un message, s'il est en clair ou crypté (la réponse étant un caractère), et affiche le message (respectivement crypté ou en clair) correspondant.

Une session doit être :

```
$ java Vigenere
Clé = essai
En clair (Y) ou chiffré (N) ? Y
```

```
Message en clair = patrick
Message chiffré = 'tslrqgc'
$
```

# Documentation

## 1 Nombres aléatoires

Rappelons que la méthode statique :

```
random()
```

de la classe `Math` permet d'obtenir un nombre réel de l'intervalle  $[0,1[$  de façon pseudo-aléatoire.

## 2 Menu déroulant

Pour pouvoir utiliser des menus déroulants, on doit ajouter une **barre de menus** au cadre, objet de la classe :

```
MenuBar
```

qui s'ajoute grâce à la méthode :

```
setMenuBar(MenuBar)
```

Un **menu déroulant** est un objet de la classe :

```
Menu
```

dont le principal constructeur est :

```
Menu(String)
```

Il s'ajoute à la barre des menus grâce à la méthode :

```
add(Menu)
```

de la classe `MenuBar`.

Un **item** d'un menu est un objet de la classe :

```
MenuItem
```

dont le principal constructeur est :

```
MenuItem(String)
```

Il s'ajoute à un menu grâce à la méthode :

```
add(MenuItem)
```

de la classe `Menu`.

Pour pouvoir détecter le choix d'un item, il faut implémenter l'interface `ActionListener` dont seuls les items ont besoin d'être à son écoute.

## 3 Les chaînes de caractères

Rappelons que la méthode de `String` :

```
public char charAt(int)
```

renvoie le  $i$ -ième caractère de la chaîne de caractères, les caractères étant numérotés à partir de 0.