Séance 2 de travaux pratiques

1 Échauffement

Bonjour monde ! (⋆) String [q1]

Voici l'exercice incontournable de programmation !

Compléter la procédure `showHelloWorld` qui affiche `"Bonjour monde "` ! (suivi d'un retour à la ligne). Attention aux espaces !

▷ `HelloWorld.java`

Bonjour nom ! (⋆) String [q2]

Compléter la procédure `showHelloName` qui attend en argument une chaîne de caractères `nom` et qui affiche `"Bonjour nom "` ! (où `nom` est bien sûr remplacé par l'argument de la fonction). Attention aux espaces !

▷ `HelloName.java`

Bonjour nom ! (⋆) String [q3]

Compléter la fonction `getHelloName` qui attend en argument une chaîne de caractères `nom` et qui retourne la chaîne "Bonjour nom !" (où `nom` est bien sûr remplacé par l'argument de la fonction). Attention aux espaces !

▷ `HelloNameReturn.java`

Animations au zoo (⋆) String [q4]

Le propriétaire d'un zoo vous demande de l'aider à programmer le panneau d'informations afin qu'il puisse afficher l'horaire et le nom du prochain spectacle.

Pour cela, complétez la procédure `showPerformance` qui attend un entier `hour` (que l'on supposera compris entre `0` et `23`), ainsi qu'une chaîne de caractères `performanceName` et affiche :

```
A [hour]h aura lieu le spectacle [performanceName]. Venez nombreux !
```

puis retourne à la ligne.

Un exemple de sortie serait :

```
A 16h aura lieu le spectacle Otaries jongleuses. Venez nombreux !
```

▷ `Zoo.java`

2 Boucle

Comment lire une boucle ? (⋆) boucleavecdépendance [q5]

Quelles sont les assertions vraies pour la boucle suivante ?

```
for (int i = 0; i < 3; i++) {
  showInt(i);
}
```

☐ Elle répète une action trois fois
☐ Elle ne s'arrête jamais.
☐ Elle affiche la valeur de la variable i à chaque itération.
☐ Elle contient une erreur de syntaxe.

De l'autre côté du miroir (⋆⋆) String boucleavecdépendance [q6]

Compléter la fonction `mirror`, de façon à ce qu'étant donnée une chaîne `s`, elle retourne la chaîne miroir (c'est-à-dire à l'envers). Par exemple, elle associe à `"foo"` la chaîne `"oof"`.

▷ `Mirror.java`

Séparateur textuel ajustable (⋆⋆) String conditionnelledansboucle boucleavecdépendance [q7]

Le but de l'exercice est d'afficher une (jolie) barre textuelle horizontale, de la taille spécifiée par l'argument de la procédure. Complétez la procédure `showSeparator` qui attend en argument un entier représentant la taille de la barre, puis affichez la en alternant les séparateurs `sep1`, `sep2` et `sep3` de sorte que la barre ondule. On pensera à retourner à la ligne. En cas d'erreur, une ligne vide est affichée.

Par exemple, pour `sepSize = 11`, on obtiendra ce résultat : `.-°-.-°-.-°`.

▷ `SizeableSeparator.java`

Séparateur textuel ajustable à partir d'un modèle (⋆⋆⋆) String boucleavecdépendance conditionnellecomplexe modulo [q8]

Le but de l'exercice est d'afficher une barre horizontale, dont la taille sera spécifiée par l'argument `sepSize` de la procédure. Complétez la procédure `showSeparator` qui attend en argument un entier représentant la taille de la barre, puis affichez la en répétant le motif `pattern` passé en argument. On pensera à retourner à la ligne. En cas d'erreur, une ligne vide est affichée.

Par exemple, pour `sepSize = 11` et `pattern = ".-°-"`, on obtiendra ce résultat : `.-°-.-°-.-°`.

▷ `GenericSizeableSeparator.java`

Retour sur les tables de l'école (⋆⋆) String boucleavecdépendance [q9]

Compléter la fonction `showTable` qui attend un entier `n` et renvoie la table de multiplication de cet entier sous forme de chaîne de caractères, suivant le motif `k * n = m` sur chaque ligne, où `k` va de 1 à 10 et `m` est le résultat de la multiplication. (Cette chaîne finit par un retour à la ligne.)

▷ `Table.java`

3 Conditionnelle

Expressions booléennes (⋆) conditionnellecomplexe parenthésage [q10]

Parmi les conditions suivantes, lesquelles sont vraies ?

☐ ! 1 == 1 && (1 == 2 || (1 != 1 && 2 == 2))
☐ ! 1 == 1 && (1 == 2 || 1 != 1) && 2 == 2
☐ ! (1 == 1 && 1 == 2) || (1 != 1 && 2 == 2)
☐ ! ((1 == 1 && 1 == 2) || 1 != 1 && 2 == 2)

Chaînes modulo 2 (⋆) int String différenceintString [q11]

Compléter la procédure `isEven` de façon à ce que lorsqu'on lui passe un entier sous la forme d'une chaine de caractères `s` (qui est donc de type `String`), elle affiche si cet entier est pair ou non. L'affichage se fera suivant les schémas suivants (où X et Y doivent être remplacés par l'argument) :

```
oui, X est pair
```

```
non, Y est impair
```

▷ `StringMod.java`

Mousse au chocolat (⋆⋆) int String arrondis casparticuliers conditionnellesimple [q12]

Bob veut faire une mousse au chocolat pour `n` personnes pour son anniversaire. Les ingrédients d'une mousse au chocolat pour 4 personnes sont 3 œufs, 100g de chocolat, 30g de sucre.

Compléter la procédure `showRecipe` qui attend un entier `n` et qui affiche la liste d'ingrédients requis pour une mousse au chocolat pour `n` personnes.

Par exemple pour `n=4` personnes, la procédure affichera :

```
3 oeufs
100g de chocolat
30g de sucre
```

et pour `n=2` personnes, la procédure affichera :

```
1 oeuf
50g de chocolat
15g de sucre
```

Remarques :

* Faites très attention au pluriel et au singulier pour "oeuf".

* Vous ne devez pas utiliser de nombre réel dans le code (vous devez uniquement utiliser des entiers).

* Cet exercice demande de bien faire attention à l'ordre des opérations. Par exemple, en Java, `5 / 6 * 2` vaut 0 car Java calcule d'abord `5 / 6` qui vaut 0, et multiplie ensuite par 2.

* On ne peut pas faire de mousse sans oeufs !

* Un nombre négatif ou nul de personnes sera considéré comme aucune personne. Dans ce cas, on affichera une recette avec des quantités nulles pour chaque ingrédient.

▷ `ChocolateMousse.java`

4 Bonus

Trois couleurs (⋆⋆⋆) boucleavecdépendance dessin [q13]

En utilisant les procédures fournies, écrivez un programme qui dessine le drapeau français. On rappelle que ce dernier est constitué de trois bandes de même taille, dont les couleurs sont, de droite à gauche, bleu, blanc et rouge.

Indication : le drapeau dessiné aura une largeur de 600pixels.

▷ `Flag.java`

Ordre lexicographique (⋆⋆⋆) int String boucleavecdépendance conditionnellesimple [q14]

Compléter le code afin que :

- la fonction `minLength`, qui prend en paramêtres deux chaînes de caractères, retourne la longueur de la plus petite des deux

- la fonction `getDifference`, qui prend en paramêtres deux chaînes de caractères `m` et `n`, retourne le plus petit indice `i` tel que :

`m` et `n` différent sur le caractère d'indice `i`

ou `i` est plus grand que la longueur de `m` ou `n`

- la procédure `compareString`, qui prend en paramêtres deux chaînes de caractères `m` et `n`, affiche leur ordre dans le dictionnaire avec le message :

```
X est avant Y dans le dictionnaire.
```

ou

```
X et Y sont identiques.
```

Indications :

- on n'oubliera pas de revenir à la ligne

- on pourra faire appel à la procédure `stringLetterAt` qui étant donné une chaîne de caractères `m` et un entier `i`, renvoie la position dans l'alphabet (1 pour a, 2 pour b, 5 pour e/é/è/ê, etc...) du caractère en position `i`.

▷ `Dictionary.java`

Art minimaliste : PacMan vs. Java (⋆⋆⋆) boucleavecdépendance dessin [q15]

Nous vous donnons l'occasion d'exprimer votre talent artistique dans cet exercice, tout en vous replaçant dans l'univers de l'informatique des années 80.

Il vous suffira pour cela de compléter la procédure `draw`, de façon à ce qu'elle dessine le fameux héros du mythique jeu vidéo PacMan. Pour les marmousets qui auraient l'outrecuidance de ne pas avoir connu les années 80, nous vous rappelons que PacMan est essentiellement une pizza jaune dont on aurait ôté une part. Vous vous servirez intelligemment de la procédure bien nommée `drawArc` afin de dessiner des rayons de cercle.

Indice : vous pourrez essayer dans un premier temps de dessiner un cercle plein, et raffiner votre code ensuite.

▷ `PacMan.java`

Faire ses lignes sans se fatiguer (⋆) String boucleder épétition [q16]

En un temps heureusement révolu, les élèves qui se tenaient mal en classe, ou rataient un devoir, devaient parfois, en guise de punition, recopier quelques centaines de fois une même phrase. Dans cet exercice, nous allons programmer une procédure qui effectuera cette pénible tâche à la place de l'utilisateur.

Complétez la procédure `afficherLignes` qui attend un entier `n` et une chaîne de caractères `ligne` et affiche `n` fois la chaîne `ligne`, en revenant à la ligne à chaque nouvel affichage.

▷ `FaireSesLignes.java`

Compte à rebours de fusée (⋆⋆) boucleavecdépendance [q17]

Compléter la procédure `showCountdown` qui attend un entier `n` et qui affiche un compte à rebours de `n` à `0`. Pour `n=3`, votre code affichera :

```
3
2
1
0
```

Indice : vous remarquerez qu'afficher les entiers de `n` à `0`, revient à afficher les entiers `n-i` pour `i` allant de `0` à `n`.

▷ `Countdown.java`

5 Bac à sable

Test (⋆) [q18]

Faites vos tests ici !

▷ `Test.java`