Java Avancé

ESGI 3DW

2015 - 2016

TP1

Installation, algorithmique
Conversions implicites et explicites (cast)

Installation

- ➔ Installer le Java Development Kit (JDK)
  http://www.oracle.com/technetwork/java/javase/downloads/index.html

- ➔ Installer Eclipse (Eclipse IDE for Java Developers)
  http://www.eclipse.org/downloads/

- ➔ ou : installer intelliJ, un IDE de l'éditeur JetBrains, gratuit pour les élèves du groupe GES https://www.jetbrains.com/idea/download/#section=windows (utilisez votre mail GES pour le compte sur JetBrains)

- ➔ Marquer la page de l'API de Java 7 sur votre navigateur
  http://download.oracle.com/javase/7/docs/api/

Exercices

Exercice 1 compilation

Créez un nouveau Java Project. Créez-y une classe Test ou Main

Créez une méthode main et y placer le code suivant[1]. Compilez et exécutez.

```
public class Test {
            public static void main(String[] args) {
            System.out.println("tapez un mot");
            //Saisie clavier :
            Scanner entree =   new Scanner(System.in);
            String mot = entree.nextLine(); //dans mot est stocké une ligne
            entière
            //Affichage console :
            System.out.println(mot);
                }
            }
```

[1] Eclipse : à la création de la classe, cochez la checkbox *create main method*
IntelliJ : *project from template : line command app*

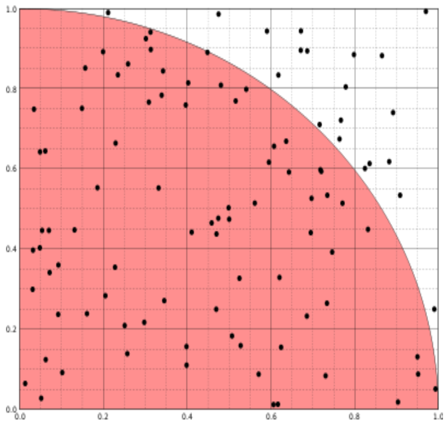
Exercice 2 opérateurs de cast

Ecrire un programme qui transforme un nombre réel x et un entier n (x et n sont définis en dur) et qui affiche le réel x tronqué à n chiffres après la virgule. On pourra utiliser la méthode Mat.pow(a,n) pour l'élévation de a à la puissance n. L'objectif de l'exercice est d'utiliser les opérateurs de cast.

<u>Contraintes</u> : Interdit de transformer x en chaine de caractères, interdit d'utiliser l'opérateur modulo, et la fonction partie entière

Exercice 3 fonction random

Le calcul de π par la méthode de Monte-Carlo consiste à tirer au hasard des nombres x et y dans l'intervalle [0 ; 1]. Le point M (x, y) appartient au disque de centre (0,0) de rayon 1 si et seulement si $x^2 + y^2 < 1$. Or la probabilité que le point M appartienne au disque est π/4 (soit le rapport entre les aires du quart de disque de rayon 1 et du carré de côté 1). En faisant le rapport du nombre de points dans le disque sur le nombre de tirages, on obtient une approximation du nombre π/4 d'autant plus fine que le nombre de tirages est grand.



- ➔ Ecrire une méthode qui prend en paramètre le nombre de points de l'échantillon et qui renvoie l'erreur d'approximation. On pourra utiliser la méthode de classe *Math.random()* qui renvoie un double de l'intervalle [0,1[et la constante de classe *Math.PI*
- ➔ Modifier le programme de manière à ce qu'il fournisse une approximation à 10^{-n} près, n étant un entier fourni par l'utilisateur. Le programme pourra également renvoyer le nombre de points de l'échantillon utilisé. On pourra utiliser *Math.pow(double a, double b)* qui renvoie un *double* valant *a* à la puissance b.

Exercice 4 fonction et tableau

Dans le même projet ou dans un nouveau, écrire une fonction rechDoub qui renvoie un booléen indiquant si un tableau d'entiers en paramètre contient des doublons, ainsi qu'un petit programme de test.

Exercice 5 fonction et switch

Ecrire un programme où l'utilisateur entre un entier entre 0 et 99 et qui affiche ce nombre en toute lettre. On pourra écrire une fonction `String convertir(int n)` et utiliser un switch.

Exercice 6 optimisation

- ➔ Vérifier la conjecture de Syracuse[2] pour les entiers inférieurs à 1000.
- ➔ Afficher les nombres qui connaissent un grand temps de vol, ainsi que ceux qui atteignent une grande altitude maximale. Y en a-t-il qui cumulent les 2 propriétés ?
- ➔ Jusqu'à quel nombre pouvez-vous prouver la conjecture (avant la fin du TP) ?

---

[2] https://fr.wikipedia.org/wiki/Conjecture_de_Syracuse