Projet 3

Rappels

Rendu le 14 avril 2018, 22h, dans la boîte de dépôt `Projet3` de jalon

Format zip ou tar.gz

Nom de l'archive pour une personne seule `Projet3_Nom_Prenom`, pour un groupe de deux `Projet3_Nom_Prenom_Nom2_Prenom2`

Contenu de l'archive L'exécutable du code que vous avez écrit ne doit pas être présent.

- Un fichier `README`, court, contenant le langage et les instructions de compilation et de lancement. Si le langage n'est pas portable (`Java` est portable), il doit fonctionner sur une des plateformes auxquelles j'ai accès (Windows / Fedora, et éventuellement Mac), et vous devez préciser laquelle.
- Le(s) fichier(s) ou le(s) dossier(s) contenant le code demandé, non compilé.
- Si vous utilisez des librairies, les fichiers de ces librairies.

Note 1 point pour le respect des consignes et de la date de rendu ; le reste sera réparti entre le test automatique pour les résultats, et la correction de votre code.

Soyez raisonnables, tant dans le choix du langage, que dans le choix des librairies (vous ne devriez presque pas en avoir besoin) ou de la procédure d'installation/lancement. De plus, le programme doit pouvoir être lancé depuis un script bash (y compris les arguments), et la sortie doit pouvoir être redirigée vers un fichier (classiquement, sortie dans un fichier, ou sortie en console).
Plusieurs archives seront mises à votre disposition sur jalon : un squelette de code (en `Java`, et en `Python3`) pour le programme demandé, un squelette de code pour d'autres langages (sur demande), un fichier de test du format de la sortie (qui ne teste que le format de la sortie).
Vous n'avez aucune obligation d'utiliser le squelette de code proposé.

Mini-projet

Contexte

Afin d'estimer l'aire d'une surface comprise dans un rectangle de dimension connue, la méthode de Monte-Carlo, par simulation, consiste à tirer des points au hasard dans ce rectangle, suivant une loi uniforme, et compter combien parmi ceux-ci sont dans la surface. En divisant ensuite ce nombre par le nombre total de points tirés au hasard, on obtient une estimation du pourcentage du rectangle qui est dans surface, ce qui permet ensuite d'obtenir une approximation de l'aire de la surface.

Dans ce projet, vous allez vous intéresser aux différentes étapes de cette méthode, afin d'obtenir une estimation de la valeur de π (ce qui permettra de vérifier votre algorithme), puis d'estimer l'aire de la surface indiquée sur le graphique 3. Pour cela, il vous faudra d'abord simuler une loi uniforme sur un intervalle quelconque et implémenter la méthode de Monte-Carlo. Vous la testerez en premier pour estimer l'aire du carré bleu, en tirant des nombres dans le rectangle gris (cf graphique 1). Puis, vous estimerez l'aire d'un disque de rayon 1, centré en 0, en tirant des nombres dans le carré gris de côté 2 (cf graphique 2). Puis finalement, vous pourrez réutiliser cette méthode pour estimer l'aire de la surface du dernier graphique (cf graphique 3).

Spécifications

Ecrire un programme qui prend en entrée le nombre n de tirages au hasard pour chacune des deux estimations.

Ce programme devra alors écrire en sortie, pour chaque expérience, le nombre de points tirés qui sont tombés dans la surface, suivie d'une espace, puis de l'estimation de l'aire obtenue grâce à cette simulation, avec 3 chiffres obligatoires après la virgule. En particulier :

Première ligne 1e expérience : estimer l'aire du carré bleu en tirant des nombres dans le rectangle gris (cf graphique 1).

Deuxième ligne 2e expérience : estimer la valeur de π, qui est exactement l'aire du disque rose du graphique 2, en tirant des nombres dans le rectangle gris.

Troisième ligne 3e expérience : estimer la valeur de l'aire de la surface verte du graphique 3, en tirant des nombres dans le rectangle gris.

Les zéros non significatifs doivent être écrits : respectez le nombre obligatoire de chiffres indiqués. Le signe pour la virgule pourra être au choix un point ou une virgule.

Graphiques

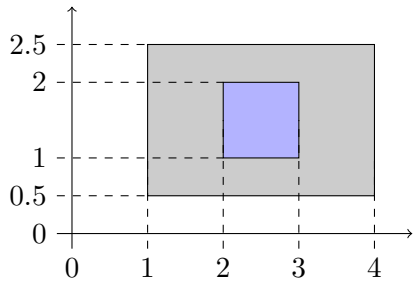

Figure 1: Rectangle gris et carré bleu

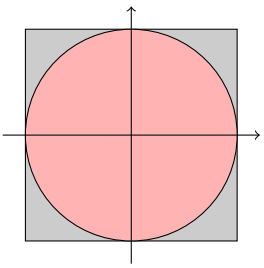

Figure 2: Disque rose de rayon 1

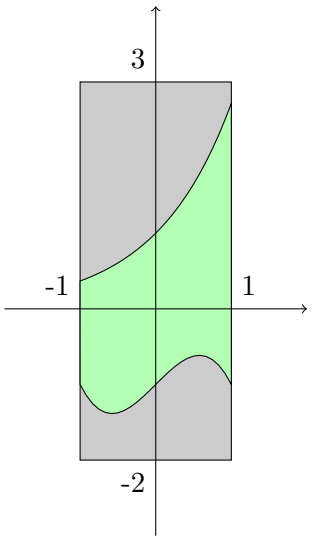

Figure 3: Surface verte
Courbe du haut : $f(x) = e^x$
Courbe du bas : $f(x) = x - x^3 - 1$

Exemple et remarques

Plus précisément, si le programme est appelé avec 1000 simulations, on pourrait obtenir la sortie suivante (par exemple) :

```
170 1.020
782 3.128
419 4.190
```