CollègeAhuntsic

Programmation Web côté serveur III
Devoir 1 | Les Bases Du C#
Ce devoir contient 50 points

. Instructions

- Ce premier devoir doit être remis pour le 11 septembre 2023 à minuit.
- Le travail peut être fait en équipe de 1, 2 ou 3.
- Le travail doit être remis sous la forme d'une solution C#, compréssé en format zip, par courriel au didier.amyot@collegeahuntsic.qc.ca
- Le travail doit contenir un petit README qui indique où sont les réponses aux questions.

1. Moyenne d'une liste

1.a. (1 punkto)

Écrire une fonction qui prend une liste de doubles et qui calcule la moyenne des éléments de cette liste.

1.b. (1 punkto)

Tester la fonction avec des tests unitaires.

2. Ascii art

2.a. (3 punktoj)

Écrire une fonction `CreateStaircase(int height, int width)` qui retourne un `string` de 5 "marches" mesurant `height` de haut et `width` de large.

Par example, ci-bas est le résultat de `CreateStaircase(2, 3)`.

```
          ###
          ###
        ######
        ######
      ########
      ########
    ##########
    ##########
##############
##############
```

2.b. (2 punktoj)

Tester la fonction avec des tests unitaires.

3. Tic-Tac-Toe

Le but de cet exercice est de créer une classe qui représente un jeu de tic-tac-toe qui permettrait à deux joueurs de jouer. Pour la simplicité, le joueur **X** commence toujours.

Dernière mise-à-jour: 2023-10-08T15:01:18Z | Identification: 363748f7

3.a. (3 punktoj)
Lire sur les enums et créer un enum qui représente les joueurs du tic-tac-toe.

3.b. (2 punktoj)
Lire sur les enums et créer un enum qui représente l'état d'un "carré" du jeu de tic-tac-toe.

3.c. (4 punktoj)
Créer une classe `TicTacToe` qui représente l'état du jeu. Vous devez utiliser les enums créés plus haut.

3.d. (3 punktoj)
Ajouter une méthode `XPlay` qui permet au joueur "X" de jouer.

3.e. (3 punktoj)
Ajouter une méthode `OPlay` qui permet au joueur "O" de jouer.

3.f. (2 punktoj)
Ajouter une méthode `NextPlayer` qui retourne le prochain joueur à jouer.

3.g. (3 punktoj)
Ajouter une méthode `TheWinnerIs` qui retourne le joueur gagnant.

3.h. (5 punktoj)
Écrire un test unitaire qui donne un exemple de jeu complet.

4. Approximation de la valeur de π par une méthode Monte-Carlo

Il est possible d'estimer la valeur de π par une méthode qui utilise l'aléatoire. Pour se faire, il faut connaître la formule de l'aire d'un cercle $A = \pi r^2$ où A est l'aire du cercle et r son rayon.

L'idée principale de la méthode est d'estimer l'aire d'un cercle de rayon prédéfini. En effet, avec l'aire et le rayon connus, nous avons que $\pi := \frac{A}{r^2}$. Dans ce cas-ci, prendre un cercle de rayon 1 est très judicieux car la formule ce simplifira par $\pi := A$.

Donc, comment faire pour estimer l'aire d'un cercle? La méthode que nous allons utiliser nécessite d'inscrire le cercle dans un carré comme illustré ci-bas.

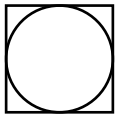

Figure 1: Cercle inscrit dans un carré

Comme il est facile de connaître l'aire du carré A_c, nous allons essayer de trouver la proprotion p du carré qui est couverte par le cercle. Pour se faire, nous allons lancer des points au hazard dans le carré; la proprotion p du carré qui est couverte par le cercle sera alors estimée par le nombre de point ayant tombé dans le cercle divisé par le nombre total de point.

4.a. (1 punkto)
Copier/Utiliser la classe `Point` vue en classe.

4.b. (2 punktoj)
Créer une classe `Cercle` centrée en $(0, 0)$ et de rayon r.

Dernière mise-à-jour: 2023-10-08T15:01:18Z | Identification: 363748f7

4.c. (2 punktoj)
Ajouter une méthode `IsIn` à `Cercle` qui prend en argument un `Point` et retoure `true`/`false` dépendamment si le point est à l'intérieur ou l'extérieur du cercle.

4.d. (2 punktoj)
Créer une classe `Square` centrée en $(0, 0)$, de coté c.

4.e. (1 punkto)
Créer une classe `Area` du carré qui retourne l'aire du carré.

4.f. (2 punktoj)
Créer une méthode `RandomPoint` dans `Square` qui prend en argument un object de classe Random et retourne un point aléatoire (uniformément distribué) situé dans le carré.

4.g. (5 punktoj)
Créer un méthode statique `PiEstimate` qui prend un entier n en paramêtre et retourne une estimation de π faite avec n points.

4.h. (3 punktoj)
Écrire un test unitaire qui démontre que votre valeur de π n'est pas trop loin de la valeur usuelle $3.14159265358979323846264338327950288419 71$.