ITI 1520 - Devoir 2

Disponible: le 28 sept, 2024
Date de remise: mardi, le 8 oct, 2024, 22:00
SVP notez que le devoir n'est pas accepté après cette date.

Vous devez faire ce travail **individuellement**. Vous devez soumettre un répertoire compressé en BrightSpace (le fichier d2_xxxx.zip ou xxxx est votre numéro d'étudiant) contenant le fichier d2.py

Mettez toutes les fonctions requises ci-dessous dans le fichier d2.py
Les noms des fichiers et les noms des fonctions doivent être les noms requis, parce qu'on va utiliser des tests automatiques pour la correction. Si le nom d'une fonction n'est pas le nom requis, la note sera zéro pour la fonction. Ajouter des commentaires pour chaque fonction (avec une description courte et contrat de type). Si votre fichier d2.py donne erreur de syntaxe, la note sera zéro.
Si vous envoyez le devoir plusieurs fois, la dernière version sera note.
SVP noter qu'on va utiliser un outil logiciel pour détecter du plagiat. En cas deux devoirs sont identique ou très similaires, la note sera zéro pour les deux.

Le but de ce devoir est de tester les concepts de Module 3 et Module 4.

Ajouter une déclaration dans un fichier **declaration_VOTRE_NOM.txt**. avec votre soumission en BrightSpace (dans le fichier d2_xxxxx.zip)

Notes pour le fichier declaration_VOTRE_NOM.txt :
Ce fichier doit contenir la référence pour code qui n'est pas écrit par vous même **si c'est le cas**.
Ça inclue code donné par un collègue ou autre personne, ou trouvé sur l'internet, réseaux sociale (comme **Chegg, CourseHero, Stack Overflow, discord,** ou autre), ou généré par **ChatGPT** ou autre assistent virtuel. Ça n'inclue pas le code donne en BrightSpace dans les notes de classe, labo, etc.
Pour chaque question ou vous avez utilisé du code donné ou trouvé, il faut:
1. le numéro de la question
2. copier-coller le code emprunté ou généré. Ça inclue code donné/trouvé et modifier très peu.
3. le nom de la source: personne, site Internet ou autre

Vous pouvez perdre des points pour la question, mais au moins vous évitez une accusation de plagiat. En cas de plagiat, la note est zéro et le cas doit être rapporté au doyen.
Le même est valable si quelqu'un montre son code a un collègue.
Si vous n'utilisez pas cette déclaration, c'est équivalent à déclarer que tous le code a été écrit par vous-même.

Barème : total de **24** points. Devoir 2 est 4% de la note finale.

Question 1. (3 points) La saison de patinage commence sur le canal Rideau lorsqu'une glace de bonne qualité d'au moins 30 cm d'épaisseur sera formée et la température moyenne dans les derniers 10 jours sera au moins -10°C. Ecrivez une fonction Python appelée **patinage** qui prend deux variables (l'épaisseur de la glace en cm et la température moyenne on °C) et retourne True si le canal est ouvert pour patinage et False sinon. Voir des exemples de test ci-desous.
Le contrat de type de la fonction est : (float, float)->bool

Question 2. (3 points) Ecrivez une fonction Python appelée **alphaNote** qui prend une valeur comme entrée (un nombre réel de 0 a 100, inclusive) et retourne une note alphabétique. Pas besoin de vérifier si la valeur est de 0 à 100 pour le moment. Pour cette question, on fait l'hypothèse que ce le cas. On va vérifier ça pour Question 3. Ajoutez en commenatire le contrat de type (devinez-le vous-même). Utilisez la correspondance suivante (de uOttawa):

A+	90-100
A	85-89
A-	80-84
B+	75-79
B	70-74
C+	65-69
C	60-64
D+	55-59
D	50-54
E	40-49
F	0-39

Question 3. (3 points) Ecrivez une fonction Python appelée **alphaNoteVerif** qui utilise la fonction de Question 2, mais avant elle lit le clavier pour prendre la note finale numérique et vérifie que la valeur soit de 0 à 100 (0 et 100 inclues). Si ce ne pas le cas, elle demande l'usager de réintroduire la note, même plusieurs fois, jusqu'au la valeur est dans l'intervalle attendu. On fait l'hypothèse que l'usager donne un nombre entier ou réel, pas besoin de vérifier ça pour le moment. La fonction doit appeler la fonction alphaNote de question 2 est afficher un message avec son résultat. Elle doit aussi afficher le message: "Echoue" si la note a été "E" ou "F", ou le message "Reussi" pour les autres valeurs. Le contrat de type est () ->None.

Question 4. (3 points) Ecrivez une fonction Python appelée **boucles** qui prend un entier **n** et affiche chaque deuxieme valeur, de 1 à **n** (pour les nombres paires, **n** n'est pas inclus) et après ça de n à 1 (pour les nombres paires, **1** n'est pas inclus).
La fonction ne retourne rien. Ajoutez en commentaire le contrat de type (devinez-le vous-même).

Question 5. (3 points) Ecrivez une fonction appelée facteursDeN qui prend un entier n comme paramètre et imprime tous les facteurs de n (à l'exception de 1 et n). Il devrait également imprimer la somme de ces nombres (à l'exception de 1 et n). La fonction doit renvoyer True si la somme des facteurs imprimés est inférieure à n et False dans le cas contraire. Rappelons qu'un facteur est un nombre entre 1 et n qui divise n. Ajoutez en commentaire le contrat de type (devinez-le vous-même).

Question 6. (3 points) Écrivez une fonction Python appelée **carreParfait** qui prend un entier **x** comme paramètre et vérifie si l'entier est un carré parfait. La fonction retourne une valeur booleene,
Si le nombre x est un carré parfait, la fonction doit afficher un message le confirmant, afficher aussi sa racine carrée, et retourner True. Sinon, elle doit afficher que le nombre x n'est pas un carré parfait et retouner False. Écrivez vous-meme le contrat de type en commentaires. Rappel : Un nombre entier x est un carré parfait s'il existe un entier n tel que : $n^2 = x$.

Question 7. (3 points) La formule suivante donne une méthode de calcul de π :

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots$$

Écrivez une fonction `maFun` qui prend en entrée un entier naturel n positif et retourne la valeur :

$$res = \frac{(-1)^n}{2n+1}$$

Le contrat de type de la fonction est : (int)->float

Question 8 . (3 points) Écrivez une fonction `maFun_bis` qui prend en entrée un entier naturel n positif et retourne la valeur :

$$res = 4(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots + \frac{(-1)^n}{2n+1})$$

(approximation de π au rang n)
Votre fonction doit faire appel à la fonction `maFun` de Q7.
Le contrat de type de la fonction est : (int)->float

Testez les fonctions dans l'interpréteur Python ou avec le unit test fourni.
Voici quelques cas de test (testez aussi avec autres valeurs).
Optionnel : Ajouter un programme principal dans votre fichier d2.py pour appeler vos fonctions avec les valeurs ci-dessou.

```
#test Q1
patinage(30, -10)
True
patinage(25.4, -15)
False
patinage(33, -15)
True
patinage(33, -5)
False

#test Q2
alphaNote(100)
'A+'
alphaNote(89)
'A'
alphaNote(56)
'D+'
alphaNote(30)
'F'

#test Q3
alphaNoteVerif()
SVP entrez la note finale (de 0 a 100): 105
SVP entrez la note finale (de 0 a 100): 207.6
SVP entrez la note finale (de 0 a 100): 89
La note alphabetique est: A
Reussi
```

```
alphaNoteVerif()
SVP entrez la note finale (de 0 a 100): -12
SVP entrez la note finale (de 0 a 100): 28
La note alphabetique est: F
Echoue

#test Q4
boucles(13)
1 3 5 7 9 11 13
13 11 9 7 5 3 1
boucles(10)
1 3 5 7 9
10 8 6 4 2

#test Q5
Facteurs de 12  = 2 3 4 6
Somme des Facteurs =  15
False
facteursDeN(9)
Facteurs de 9  = 3
Somme des Facteurs =  3
True
facteursDeN(5)
Facteurs de 5  =
Somme des Facteurs =  0
True

#test Q6
carreParfait(16)
16 est un carre parfait et sa racine carree est 4
True
carreParfait(15)
15 n'est pas un carre parfait
False

#test Q7
maFun(0)
1.0
maFun(1)
-0.3333333333333333
maFun(10)
0.047619047619047616
maFun(2)
0.2

#test Q8
maFun_bis(10)
3.232315809405594
maFun_bis(100)
3.1514934010709914
maFun_bis(1000)
3.1425916543395442
maFun_bis(10000)
3.1416926435905346
```