ITI 1520 - Devoir 3

Disponible: le 5 octobre, 2020
Date de remise: lundi, le 19 octobre, 2020, 8:00
SVP notez que le devoir n'est pas accepter après cette date.

Vous devez faire ce travail **individuellement**. Vous devez soumettre un fichier d3_VOTRE_NUMERO_ETUDIANT.zip contant le fichier d3_VOTRE_NUMERO_ETUDIANT.py avec toutes les fonctions requises et le fichier d3_ VOTRE_NUMERO_ETUDIANT.txt avec vos tests copier-coller de l'interprèteur Python.

Les noms des fichiers et les noms des fonctions doivent être les noms requises, 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, contrat de type et preconditions).
Si un fichier .py donne erreur de syntaxe, la note sera zéro pour la question.
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.

Barème : total de 20 points. Devoir 3 est 7% de la note finale.

Question 1. (2 points) `somme_diviseurs_impaires(n)`
Créez une fonction en Python nommée `somme_diviseurs_impaires` qui prend comme entrée un entier n. Si n est zéro, la fonction ne retourne rien (retourne None). Si n n'est pas zéro, la fonction retourne la somme de tous les diviseurs positive et impaires de n.

Exemples des tests dans l'interpréteur Python :

```
>>> somme_diviseurs_impaires(-9)
13
>>> somme_diviseurs_impaires(0)
>>> somme_diviseurs_impaires(2)
1
>>> somme_diviseurs_impaires(3)
4
>>> somme_diviseurs_impaires(7)
8
>>> somme_diviseurs_impaires(-2001)
2880
```

Question 2. (2 points) somme_de_serie()
Créez une fonction en Python nommée `somme_de_serie` qui demande à l'usager d'entrer au clavier un entier n qui n'est pas négatif. Si l'usager introduit un entier négatif, la fonction retourne None. Sinon, la fonction retourne la somme de la série suivante :
$1000 + 1/1^2 + 1/2^2 + 1/3^2 + 1/4^2 + ... + 1/n^2$.
Par exemple, si n est 0, la fonction retourne 1000; si n est 1, la fonction retourne 1001; si n est 2, la fonction retourne 1001.25, for n = 3, etc.

Exemples des tests dans l'interpréteur Python :

```
>>> somme_de_serie()
SVP entrez un entier pas negatif: -10
>>> somme_de_serie()
SVP entrez un entier pas negatif: 0
1000
>>> somme_de_serie()
SVP entrez un entier pas negatif: 5
1001.463611111111
```

Question 3. (2 points) `somme_liste_div2(l)`
Créez une fonction en Python nommée `somme_liste_div2` qui prend comme entrée une liste des entiers et retourne la somme des éléments de la liste qui de divisent par 2.

Exemples des tests dans l'interpréteur Python :

```
>>> somme_liste_div2([1,4,3,8,5])
12
>>> somme_liste_div2([])
0
>>> somme_liste_div2([4,-10,7])
-6
```

Question 4. (2 points) `countMembers(s)`
Créez une fonction en Python nommée `countMembers` qui prend comme entrée une chaine des caractères s. La fonction retourne le nombre des caractères de s qui sont extraordinaire. Les caractères extraordinaires sont : des lettre minuscules de e à j (inclusive), les lettres majuscule de F à X (inclusive), les chiffre de 2 à 6 (inclusive), le signe d'exclamation (!), la virgule (,), et la barre oblique inverse (backslash) (\). Par exemple, si la chaine s continent deux X, la fonction doit compter ces deux 2 caractères extraordinaire.

Exemples des tests dans l'interpréteur Python :

```
>>> countMembers("\\")
1
>>> countMembers("2\'")
1
>>> countMembers("2aAb3?eE'_13")
4
>>> countMembers("one, Two")
3
```

Question 5. (2 points) `nombre(s)`
Imaginez un client d'une banque qui doit enter un nombre qui représente son argent. Il y a des personnes qui tapent des espaces pour séparer le millier, millions, etc. La banque a besoin de ce nombre sans les espaces.
Créez une fonction en Python nommée `nombre` qui prend comme entrée une chaine des caractères s. La fonction retourne un entier qui représente le nombre. Si s contient autres caractères que des chiffres et des espaces, la fonction retourne None. Si s continent seulement des chiffre et des

espaces, on fait l'hypothèse que les espaces sont bien placés. Par exemple, s peut être ' 119 189 000' mais pas ' 1 1 345 '.

Exemples des tests dans l'interpréteur Python :

```
>>> nombre("251")
251
>>> nombre("1 abc 340")
>>> nombre("1 250")
1250
>>> nombre("-97 500")
-97500
>>> nombre("1 000 001")
1000001
>>> nombre("999 999 100 102")
999999100102
>>> nombre("")
0
```

Question 6. (2 points) `alienNumbers(s)`
Une communication étrange a été intercepte entre deux armées extraterrestre. Les linguistes de NASA ont découvert que les extraterrestres utilisent un système de nombres très étrange. Ils utilisent des symboles pour quelques valeurs numérique et ils additionnent ces valeurs. Les linguistes ont découvert le tableau suivant avec des symboles et leurs valeurs. Il y a beaucoup des messages à décoder, donc il faut une fonction pour faire ça.

Symbole	Valeur
T	1024
y	598
!	121
a	42
N	6
U	1

Par exemple, 'a!ya!U!NaU' représente la valeur of 1095, parce qu'il y a un 'y', 3 '!', 3 'a', 1 'N' et 2 'U' (1*598+3*121 + 3*42+ 1*6 + 2*1).

Créez une fonction en Python nommée `alienNumbers` qui prend comme entrée une chaine des caractères s et retourne la valeur représentée par s dans le tableau ci-dessus. On fait l'hypothèse que s ne contient pas autres caractères que 'T', 'y', '! ', 'a', 'N', 'U'. Utilisez la fonction count de la bibliothèque Python String.
Défi (pas obligatoire): essayez de faire le corps de la fonction seulement sur une ligne de code.

Exemples des tests dans l'interpréteur Python:

```
>>> alienNumbers("a!ya!U!NaU")
1095
>>> alienNumbers("aaaUUU")
129
>>> alienNumbers("")
```

```
0
```

Question 7. (2 points) `alienNumbers2(s)`
Créez une fonction en Python nommée `alienNumbers2` qui prend comme entrée une chaine des caractères s et retourne la valeur représentée par s dans le système de codage extraterrestre de la Question 6. Donc la fonction retourne les mêmes résultats que la fonction `alienNumbers`, mais il ne peut pas utiliser des fonctions de la bibliothèque String. NASA a eu besoin d'implémenter la fonction sur un circuit qui exécute de décodage très rapidement, mais le circuit a un version réduite de Python sans la bibliothèque String. Utilisez une boucle pour vérifier chaque élément de s.

Exemples des tests dans l'interpréteur Python :
```
>>> alienNumbers2("a!ya!U!NaU")
1095
>>> alienNumbers2("aaaUUU")
129
>>> alienNumbers2("")
0
```

Question 8. (2 points) `encrypt(s)`
Créez une fonction en Python nommée `encrypt` qui prend comme entrée une chaine des caractères s et retourne une version chiffrée de s. Un système de cryptage simple écrit le message commencement avec les caractères de la fin, en ordre inverse. Par exemple, 'Hello, world' devient 'dlrow ,olleH'. Pour faire le system plus difficile à décoder, la fonction commence au debout et à la fin, en même temps. Le premier et le dernier caractère deviennent le premier et le deuxième caractère dans la nouvelle chaine, le deuxième et le avant-dernier caractère deviennent le troisième te le quatrième, etc. Par exemple, 'Hello, world' devient 'dHlerlolwo ', (les caractères spécieux, signes de ponctuation, espaces, etc. sont considérés de la même manière) et '0123456789' devient '9081726354'.

Exemples des tests dans l'interpréteur Python :
```
>>> encrypt("Hello, world")
'dHlerlolwo ,'
>>>> encrypt("1234")
'4132'
>>> encrypt("12345")
'51423'
>>> encrypt("1")
'1'
>>> encrypt("123")
'312'
>>> encrypt("12")
'21'
>>> encrypt("Secret Message")
'eSgeacsrseetM '
>>> encrypt(",'4'r")
"r,''4"
```

Question 9. (2 points) `weaveop(s)`
Créez une fonction en Python nommée `weaveop` qui prend comme entrée une chaine des caractères s et retourne une autre chaine de caractères. La fonction vérifie chaque paire des caractères consécutive de s et retourne une chaine avec les lettres o et p insérée dans chaque pair: si le premier caractère dans la paire est une lettre majuscule, la fonction insère O; si le premier caractère est une lettre minuscule, on insère o; si le deuxième caractère est une lettre majuscule, on insère P; si le deuxième caractère est une lettre minuscule, on insère p. Si une des deux caractères n'es pas une lettre in the alphabet, on n'insère rien dans la paire. Si s a 1 ou zéro caractères on retourne une copie de s. Voir la documentation pour les fonctions isalpha et isupper (par exemple avec help(str.isalpha) dans l'interpréteur Python).

Exemples des tests dans l'interpréteur Python :

```
>>> weaveop("aa")
'aopa'
>>> weaveop("aB")
'aoPB'
>>> weaveop("ooo")
'oopoopo'
>>> weaveop("ax1")
'aopx1'
>>> weaveop("abcdef22")
'aopbopcopdopeopf22'
>>> weaveop("abcdef22x")
'aopbopcopdopeopf22x'
>>> weaveop("aBCdef22x")
'aoPBOPCOpdopeopf22x'
>>> weaveop("x")
'x'
>>> weaveop("123456")
'123456'
```

Question 10. (2 points) squarefree `(s)`
Cette question est difficile à résoudre. Utilisez des fragments des chaines (l'operateur :) et testez l'égalité des deux chaines au besoin (s1==s2).
Un mot a la propriété squarefree s'il ne continent pas le même sous-mot deux fois consécutive.
Exemples:
ana est squarefree.
borborygmus n;est pas squarefree, il ya bor deux fois consécutive.
abracadabra est squarefree.
repetitive n'est pas squarefree parce que ti est répété deux fois consécutive.
grammar n'est pas squarefree parce que m est répété deux fois consécutive
gaga n'est pas squarefree parce que ga est répété deux fois consécutive
rambunctious est squarefree.
abcab est squarefree.
abacaba est squarefree.
zrtzghtghtghtq n'est pas squarefree parce que ght est répété deux fois consécutive (trois fois, amis deux fois sont suffisant pour retourner False).

aa n'est pas squarefree parce que a est répété deux fois consécutive
zatabracabrac n'est pas squarefree parce que abrac est répété deux fois consécutive

Créez une fonction en Python nommée `squarefree` qui prend comme entrée une chaine des caractères s et retourne True si s a la propriété squarefree et False sinon.

Exemples des tests dans l'interpréteur Python :

```
>>> squarefree("")
True
>>> squarefree("a")
True
>>> squarefree("zrtzghtghtghtq")
False
>>> squarefree("abcab")
True
>>> squarefree("12341341")
False
>>> squarefree("44")
False
```

Notes pour le fichier declaration_VOTRE_NOM.txt si vous avez besoin:
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 Stack Overflow, discord, ou autre). 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 question
2. copier-coller le code emprunter. Ça inclue code donné/trouvé et modifier très peu.
3. le nom de la source: personne, site Internet ou autre
Vous allez perdre des points 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 ecrit par vous-même.