Informatique (2018-2019)
Devoir surveillé 1

Durée : 2 heures

Exercice 1 (5 pts)

1. On considère le code Python de la fonction `d` suivante :

```
def d(n):
    L=[1]
    for nombre in range(2,n+1):
       if n%nombre==0:
            L.append(nombre)
    return L
```

   Quel est le résultat de l'appel `d(4)` ? Puis de l'appel `d(10)` ?
   Que fait la fonction `d` ?
2. Un *diviseur non-trivial* d'un entier n est un diviseur de n différent de 1 et de n. Ecrire une fonction `DNT`, d'argument n, renvoyant la liste des diviseurs non-triviaux de l'entier n.
3. Ecrire une fonction `sommeCarresDNT`, d'argument n, renvoyant la somme des carrés des diviseurs non-triviaux de l'entier n.
4. Ecrire la suite des instructions permettant d'afficher tous les nombres entiers inférieurs à 1000 et égaux à la somme des carrés de leurs diviseurs non-triviaux.

Exercice 2 (3 pts)

1. Ecrire une fonction `mini` qui prend en argument une liste de mots et renvoie le mot le plus court avec son indice dans la liste. Si des mots ont le même nombre de lettres la fonction renvoie le premier de ces mots rencontrés dans la liste.
   Par exemple, avec la liste `m=['cinq','six','sept','huit','neuf','dix']`, la fonction doit renvoyer le mot `'six'` avec l'indice 1.
2. Comment modifier simplement la fonction pour qu'elle renvoie le dernier des mots rencontrés dans la liste si des mots ont le même nombre de lettres ? Dans ce cas, c'est le mot `'dix'` avec l'indice 5 qui doivent être renvoyés.

Exercice 3 (4 pts)
On considère la fonction g définie sur $\mathbb{R}$ par $g(x) = \mathrm{e}^{\sin x}$.

1. Ecrire le code qui permet de définir la fonction g.
2. Ecrire une fonction `trace(f,a,b,n)` permettant d'obtenir avec la bibliothèque Matplotlib la courbe représentative d'une fonction f sur un intervalle $[a;b]$ en utilisant n points équirépartis sur $[a;b]$, bornes comprises.
   Ecrire alors le code qui permet d'obtenir la représentation graphique de la fonction g sur l'intervalle $[-\pi;\pi]$ en utilisant 100 points.

Exercice 4 (8 pts)

Afin d'utiliser la structure de pile, on suppose données les fonctions `creer_pile(c)` qui permet de créer une pile de capacité `c`, `pile_vide(p)` qui renvoie `True` si la pile `p` est vide et `False` sinon, `empiler(p,x)` qui empile un élément `x` sur la pile `p`, `depiler(p)` qui enlève l'élément sur le dessus de la pile `p` et renvoie sa valeur.

Dans la suite il est demandé de manipuler les piles uniquement au travers de ces fonctions, sans aucune hypothèse sur la représentation effective des piles en mémoire.

1. Ecrire une fonction `parentheses` qui prend en argument une chaîne de caractères `ch` et renvoie la chaîne de caractères composées des parenthèses rencontrées, dans l'ordre, dans `ch`.

   On considère un mot, c'est-à-dire une chaîne de caractères, constitué uniquement de parenthèses, par exemple "( )( )" ou "(( ))( )" ou "( ))" ; les deux premiers mots sont bien parenthésés, le dernier ne l'est pas.
2. Ecrire une fonction `nombres` qui prend en argument un mot `m` comme ci-dessus et renvoie une liste `[n1,n2]` où `n1` est le nombre de parenthèses ouvrantes et `n2` le nombre de parenthèses fermantes.
3. Ecrire une fonction `compare` qui prend en argument un mot `m` comme ci-dessus et renvoie un booléen de valeur `True` si le nombre de parenthèses ouvrantes est égal au nombre de parenthèses fermantes et `False` sinon.
4. Ecrire une fonction `verification` qui prend en argument un mot `m` comme ci-dessus. Le corps de la fonction est décrit ci-dessous :
   - la fonction commence par créer une pile `p` de capacité la longueur du mot et une liste vide nommée `couple` ;
   - ensuite tous les caractères du mot sont parcourus, de gauche à droite ;
     - si le caractère est une parenthèse ouvrante "(", son indice est empilé sur la pile `p` ;
     - sinon, c'est une parenthèse fermante ")" d'indice i et alors si la pile est vide, c'est que le mot n'est pas bien parenthésé et la valeur `False` est renvoyée, sinon, l'indice j de la dernière parenthèse ouvrante rencontrée est dépilé et la liste `[j, i]` est ajouté à la liste `couple` ;
   - enfin la fonction doit vérifier que la pile est bien vide (sinon il resterait des parenthèses ouvrantes) ;
   - la fonction renvoie alors la liste `couple`.
5. Que renvoie l'appel `parentheses("(()(()))")` ?