

INFO-H-100 – Informatique – Prof. Th. Massart
1^{ère} année du grade de Bachelier en Sciences de l'Ingénieur
Examen de juin

Prénom : _____ Nom : _____ Matricule : _____

Remarques préliminaires

- On vous demande de répondre à **chaque question sur une feuille séparée**.
- N'oubliez pas d'inscrire votre nom, prénom et numéro de matricule sur chaque feuille.
- Vous disposez de trois heures et vous ne pouvez pas utiliser de notes.
- Vous pouvez utiliser le verso des feuilles pour répondre.
- Si du code vous est demandé,
 - la réponse à la question doit comprendre le code *Python* structuré et conforme aux règles de bonne pratique et conventions,
 - sauf mention contraire, vous ne pouvez utiliser aucune fonction de librairies (pas d'`import`)
 - veuillez à découper votre réponse en fonctions de manière pertinente
 - veuillez à utiliser des structures de données appropriées.

Question 1 - Fonctionnement de Python (2 points)

Expliquer brièvement comment s'effectue la gestion de la mémoire lors de l'exécution d'un programme Python qui appelle une fonction récursive. Pour vous aider dans vos explications, utilisez un exemple de code fonction Python récursive qui calcule la factorielle d'un naturel n reçu en paramètre. Pour illustrer vos propos utilisez des diagrammes d'état expliquant comment évolue la mémoire lors de l'exécution. Dans vos diagrammes d'états, veuillez à donner complètement les objets existant en mémoire.

Question 2 - Code Python (5 points)

Pour chacun des codes suivants,

- donnez ce qu’il imprime (avec les appels qui sont donnés) et expliquez de façon générale ce que fait la fonction (si elle résout un problème, lequel ...);
- donnez l’état du programme lors de chaque (groupe de) `print` grâce à des diagrammes d’état (comme fait au cours);
- sachant que les types des paramètres sont,
 - n : nombre naturel,
 - x : liste d’entiers que vous pouvez supposer de taille n ,
 - s et t : strings que vous pouvez supposer de taille n contenant uniquement des caractères ASCII (128 caractères différents possibles),

donnez la complexité moyenne et maximale ($\mathcal{O}()$) de la fonction `foo_i` ($i = 1..5$) en temps d’exécution.

Précisez bien les paramètres utilisés pour exprimer la complexité et justifiez bien vos réponses (le résultat seul ne suffit pas pour obtenir des points).

Notez bien : pour expliquer ce que le code fait et donner la complexité, vous pouvez annoter les codes fournis (compléter les docstrings, commenter, ...)

```
def foo_1(x):
    """
    """
    y = x
    z = x[:-1]
    med = x[len(x)//2]
    dico = dict(zip(list(range(len(x))), x))
    print(x)
    print(y)
    print(z)
    print(med)
    print(dico)
    print(x is y)
    print(x is z)
foo_1(list("abcde"))
```

```
def foo_2(x):
    """
    """
    y = x[:]
    x = 1
    y[x] = [4, 5, 6]
    print(y)
    return y
x = [3, 5, 7]
a = foo_2(x)
print(x)
print(a)
print(y)
```

```
def foo_3(n):
    """
    """
    if n == 1:
        res = (0, 1)
    else:
        res = foo_3(n-1)
        res = res + (res[-2] + res[-1], )
    print(res)
    return res
print(foo_3(4))
```

```
- def foo_4(s,t):  
    for e in s:  
        t = t[:t.find(e)]+t[t.find(e)+1:]  
    print()  
    return len(t) == 0  
  
print(foo_4('bonjour', 'nuroobj'))
```

```
- def foo_5(s,t):  
    x = list(s)  
    y = [e for e in t]  
    x.sort()  
    y.sort()  
    print()  
    return x == y  
  
print(foo_5('bonjour', 'nuroobj'))
```

Question 3 - Manipulation de fichiers et de données (5 points)

Considérons le fichier d'une société représentant (de façon extrêmement simplifiée) les achats de clients. Les informations sont mémorisées dans un fichier texte. Chaque ligne du fichier porte sur un achat effectué par le client correspondant. Le nom des clients est toujours formé d'un seul mot et est séparé du montant de l'achat par un espace. Le fichier est trié sur le nom des clients. Par exemple le fichier suivant indique que le Client2 a effectué trois achats différents pour des montants respectifs de 300, 120 et 440 €.

```
Client1 140
Client1 200
Client2 300
Client2 120
Client2 440
Client3 500
Client4 210
Client4 100
Client5 520
Client6 200
```

Ecrire la fonction `pack(fNameIn, fNameOut)` qui écrit dans le fichier `fNameOut` les achats regroupés de façon à ce que les clients n'apparaissent plus qu'une fois (dans le même ordre) avec le montant cumulé (la somme) de leurs achats.

```
Client1 340
Client2 860
Client3 500
Client4 310
Client5 520
Client6 200
```

Vous avez la garantie, sans devoir le vérifier, que le fichier `fNameIn` respecte le format présenté et qu'il contient au moins un achat.

Question 4 - Tri (4 points)

Les secteurs alimentaires du Reblochon, du pain et des filets de maquereaux se disputent la place de l'aliment le plus nutritif. De nombreuses études ont été conduites pour classer les 3 secteurs alimentaires et mènent à des résultats tout à fait différents qui sont rangés et numérotés dans la liste sondages.

Par exemple :

```
sondages = [['1', 'LeReblochon', 'LePain', 'LeMaquereau'],
            ['2', 'LeMaquereau', 'LeReblochon', 'LePain'],
            ['3', 'LeReblochon', 'LeMaquereau', 'LePain'],
            ['4', 'LePain', 'LeMaquereau', 'LeReblochon'],
            ['5', 'LeReblochon', 'LeMaquereau', 'LePain'],
            ['6', 'LePain', 'LeReblochon', 'LeMaquereau'],
            ['7', 'LeMaquereau', 'LeReblochon', 'LePain'],
            ['8', 'LeMaquereau', 'LePain', 'LeReblochon']]
```

(Le nombre de sondages inclus dans sondages peut être très élevé).

Le secteur des filets de maquereaux décide d'analyser tous ces sondages et vous demande d'écrire la fonction `triSondage(sondages, partenaire, concurrent)`. Cette fonction doit classer les sondages dans l'ordre qui met le plus proche de la première place, le partenaire (par exemple 'LeMaquereau'). En cas d'ex-aequo, l'avantage est donné au sondage qui met concurrent (par exemple 'LePain') à la plus mauvaise place possible.

En considérant que les deux variables `partenaire` et `concurrent` sont données comme suit :

```
triSondage(sondages, 'LeMaquereau', 'LePain')
```

La liste `sondages` sera alors modifiée de la manière suivante :

```
sondages = [['2', 'LeMaquereau', 'LeReblochon', 'LePain'],
            ['7', 'LeMaquereau', 'LeReblochon', 'LePain'],
            ['8', 'LeMaquereau', 'LePain', 'LeReblochon'],
            ['3', 'LeReblochon', 'LeMaquereau', 'LePain'],
            ['5', 'LeReblochon', 'LeMaquereau', 'LePain'],
            ['4', 'LePain', 'LeMaquereau', 'LeReblochon'],
            ['1', 'LeReblochon', 'LePain', 'LeMaquereau'],
            ['6', 'LePain', 'LeReblochon', 'LeMaquereau']]
```

Prénom : _____ Nom : _____ Matricule : _____

Question 5 - Récursivité (4 points)

Pour le maintien de l'arbre généalogique d'une famille, chaque personne est représentée par un tuple formé de son prénom et de la liste (éventuellement vide) de ses enfants. Contrairement à des pratiques courantes, les familles qui nous intéressent sont caractérisées par le fait de ne jamais donner deux fois le même prénom à deux personnes différentes. Le prénom suffit donc à identifier une personne.

Voici un exemple d'arbre généalogique :

```
arbre = ("Lucy", [
    ("Bill", [
        ("Mary", []),
        ("Fred", [("Paul", []), ("John", [])]),
        ("Jean", [])
    ]),
    ("Rick", []),
    ("Jack", [
        ("Jane", []),
        ("Bob", [("May", [])])
    ])
])
```

La figure 1 illustre cet arbre généalogique.

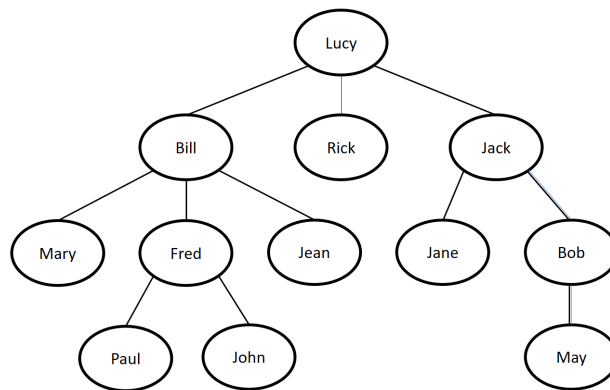


FIGURE 1 – Arbre généalogique

Ecrivez une fonction qui va chercher dans un arbre généalogique, tous les enfants d'un parent donné. Cette fonction a 2 paramètres :

- arbre : la liste qui représente l'arbre généalogique
- parent : le prénom du parent dont il faut trouver les enfants (un string)

La fonction renverra la liste (éventuellement vide) des prénoms des enfants de parent. Si le parent n'apparaît pas dans l'arbre, la fonction renvoie None.

Par exemple, le résultat du bout de code suivant :

```
for parent in ["Lucy", "Fred", "Rick", "Al"]:
    print("Les enfants de", parent, "sont", enfants(arbre, parent))
```

Est :

```

Les enfants de Lucy sont ['Bill', 'Rick', 'Jack']
Les enfants de Fred sont ['Paul', 'John']
Les enfants de Rick sont []
Les enfants de Al sont None
```

Prénom : _____ Nom : _____ Matricule : _____

Prénom : _____ Nom : _____ Matricule : _____