/323

11h14

Nom	
Prénom	
Groupe	E1

Note	15/20

1	9,5
2	2
3	2
4	3,5
5	3
6	4

Algorithmique
INFO-SUP S1
Partiel n° 1 (P1)
3 janvier 2023 - 9h30
Feuilles de réponses

Réponses 1 **(Pile ou file ? –** ***2 points*****)**

	pile	file	aucune
A B C D E F	X		
F E D C B A		X	
C B E F D A			X
C E D F A B			X

Réponses 2 **(Dichotomie : "chemin" de recherche –** ***2 points*****)**

Entourer les bonnes réponses :

① 58 - 33 - 46 - 43 - 39 - 42 — (OUI) – NON

② 40 - 75 - 57 - 53 - 44 - 42 — (OUI) – NON

③ 51 - 43 - 34 - 49 - 41 - 42 — OUI – (NON)

④ 61 - 17 - 38 - 46 - 35 - 42 — OUI – (NON)

***Réponses 3* (Lucas – *3 points*)**

Spécifications :

La fonction `check_lucas(L)` vérifie si chaque élément de la liste d'entiers `L` est la somme des deux précédents, les valeurs des deux premiers éléments pouvant être quelconques.

```
def check_lucas(L):
    if L == []:
        return True
    else:
        first = L[0]
        i = 1
        l = len(L)
        while i<l-1 and (first + L[i]) != L[i+1]:
            first = L[i]
            i += 1
        if i = l-1:
            return False
        else:
            return True
```

Réponses 4 **(Split –** ***4 points*****)**

3/4

Spécifications :

La fonction `my_split(L, sep)`, avec `L` une liste et `sep` une valeur de "séparation" du type des éléments de `L`, retourne une liste contenant les longueurs des sous-listes séparées par des valeurs `sep`.

```
def my_split(L,sep):
    l=len(L)
    nb=0
    new=[]
    for i in range(l):
        if L[i]==sep:
            while i<l-1 and L[i+1]==sep:
                i+=1
            if nb!=0
                new.append(nb)
                nb=0
        else:
            nb+=1
    if nb!=0:
        new.append(nb)
    return new
```

-0,5

Réponses 5 **(Insertion somme –** ***5 points*****)**

Spécifications :

La fonction `insert_sum(L, s, x)` insère (en place) `x` dans la liste d'entiers strictement positifs `L` à la première position, si elle existe, telle que la somme des valeurs précédentes est strictement supérieure à `s` (entier strictement positif).

La fonction place détermine à quel emplacement on doit insérer x. Le cas où ce n'est pas possible est géré dans la fonction principale.

```
def place(L,s):
    total = 0
    l = len(L)
    i = 0
    while i<l and total<=s:
        total += L[i]
        i+=1
    return i

def insert_sum(L,s,x):
    if L!=[]:
        p=place(L,s,x)
        l=len(L)
        if p!=l:
            i=0
            while i<p:
                i+=1
            stock=[]
            while i<l:
                stock.append(L.pop())
            lo=len(stock)-1
*           i+=1
            L.append(x)
            while lo>=0:
                L.append(stock[lo])
                lo=lo-1
```

Pas de null, liste !

Réponses 6 **(Mystery – *4 points*)**

1. Quel est le résultat de l'application de `mystery([6, 24, 30, 22, 29, 20, 29, 8, 40, 7])` ?

   [6,7,8,20,22,24,29,29,30,40]

2. Quel est le résultat de l'application de `mystery([4, 14, 6, 18, 4, 7, 5, 19, 14, 11, 11, 3, 11, 13, 4])` ?

   [3,4,4,4,5,6,7,11,11,11,13,14,14,18,19]

3. Que retourne la fonction `mystery(L)` ?

   Une nouvelle liste contenant les éléments de la liste L triés dans l'ordre croissant

4. Quelles sont les hypothèses que la liste `L` doit respecter pour que `mystery(L)` retourne le résultat décrit à la question précédente ?

   Il faut que la liste L ne soit pas vide (le cas n'est pas géné) et que la liste ne contienne que des valeurs positifs. ou null ?