IPT Devoir 1- Proposition de solutions

Solution 1

```
def g(x,y):
    return x+y,x*y

g=lambda x,y:(x+y,x*y)
```

Solution 2 De for à while

```
def f(n):
    L=[]
    i=n
    while i<2*n:
        L.append(i)
        i=i+1
    return L
```

Le test est `i<2*n` car le dernier rang est $2n-1$ dans l'intervalle `range(n,2*n)`.

Solution 3 Script de la fonction g :

```
def g(x):
    if x<0:
        return np.sin(x)
    elif x<=2:
        return x
    elif x<3:
        return 4-x
    else:
        return np.exp(3-x)
```

Solution 4

```
def e1(n):
    r=1 # terme de rang i=0
    f=1
    for k in range(1,n+1):
        f=f*k  # contient k!
        r=r+1/f
    return r
```

Solution 5 D'autres instructions sont possibles

```
L.append(f)    #L=L+[f]
L=[d]+L
L.append(L.pop(i))
L.insert(p,e)    #L=L[:p]+[e]+L[p:]
L.reverse()    #L=L[::-1]
L.sort()
L.count(a)
L.index(b)
```

Solution 6

Ce script affiche le plus petit entier naturel non nul, n, tel que

$$\frac{1}{n}<x \quad \Leftrightarrow \quad n>\frac{1}{x}$$

Il calcule $\left\lfloor \frac{1}{x} \right\rfloor + 1$.

Solution 7

```
def suite(n):
    a,b=2,-1
    if n==0:
        return 2
    for i in range(2,n+1):
        a,b=b,2*b+3*a
    return b
```

Solution 8 1. Le suivi des variables donne :

len(L)	L
6	[2,-1,3,0,1,2]
5	[2,3,0,1,2]
4	[3,0,1,2]
3	[3,1,2]
2	[3,2]
1	[3]

2. La fonction retourne le maximum de la liste.

Solution 9 **Tri par sélection**

Position d'un minimum

```
def pmax(L):
    '''
    Entree : L une liste de nombres reels
    Sortie : la position d'un minimum
    '''
    position=0
    for i in range(1,len(L)):
        if L[i]>L[position]:
            position=i
    return position
```

Méthode

➜ Le tri par sélection consiste à :
- on cherche le maximum de la liste, on le place dans une liste initialement vide et on le retire de la liste de départ
- puis on recommence, ainsi de suite, jusqu'à ce que la liste de départ soit vidée.

Lycée Saint-Exupery – maffre.saintexupery@gmail.com – MPSI

Tri d'une liste

```
def tri_d(L):
    '''
    Entree : L une liste de nombres reels
    Sortie : T la liste triee
             (methode par selection)
    '''
    LL,T=list(L),[] # copie de L, liste vide
    while len(LL)>0:
        p=pmax(LL)
        T.append(LL.pop(p))
    return T
```

Attention ! L'instruction `LL.pop(p)` renvoie l'élément en position `p` de la liste `LL` et le retire de la liste. Ainsi, il est possible d'emboiter (de composer) les deux instructions : `T.append(LL.pop(p))`.

▶ Description du déroulement du tri appliqué à la liste `[2,-3,-4,5,2,1]` :

boucle	L	pmax(L)	T
X	[2,-3,-4,5,2,1]	X	[]
1	[2,-3,-4,2,1]	5	[5]
2	[-3,-4,2,1]	0	[5,2]
3	[-3,-4,1]	2	[5,2,2]
4	[-3,-4]	2	[5,2,2,1]
5	[-4]	0	[5,2,2,1,-3]
5	[]	0	[5,2,2,1,-3,-4]

La liste ordonnée associée est `[5,2,2,1,-3,-4]`.

Le tri croissant consiste à constituer la liste trier en ajoutant par la gauche et non plus par la droite les maximums successifs.

```
def tri_c(L):
    LL,T=list(L),[] # copie de L, liste vide
    while len(LL)>0:
        p=pmax(LL)
        m=LL.pop(p)
        T=[m]+T
    return T
```