

Question 1

.....

.....

.....

.....

.....

.....

.....

.....

.....

Question 2

Type de tri :

mystere1 :

mystere2 :

mystere3 :

mystere4 :

mystere5 :

Question 3

```
def mystere1(tab,i,j):
    g =i+1
    d=j
    p=tab[i]
    while g<=d :
        while d>=0 and tab[d]>p:
            d=d-1
        while g<=j and tab[g]<=p:
            g=g+1
        if g<d :
            tab[g],tab[d]=tab[d],tab[g]
            d=d-1
            g=g+1
    k=d
    tab[i],tab[d]=tab[d],tab[i]
    return k

def mystere2(tab,i,j):
    if i<j :
        k = mystere3(tab,i,j)
        mystere4(tab,i,k-1)
        mystere5(tab,k+1,j)
```

Question 4

.....

.....

.....

.....

.....

Question 5

DS 2

Question 6

.....

.....

.....

.....

.....

Question 7

Question 8

```
def enveloppe_convexe(tab):
    tri(tab)

    es = creer_pile()
    ei = creer_pile()

    for i in range(len(tab)):
        enveloppe_Inf(tab,ei,i)
        enveloppe_Sup(tab,es,i)

    pop(es)

    while (not(est_vide(es))):
        push(ei,pop(es))

    pop(ei)

    return (ei)
```

Dans le pire des cas, le tri est en $\mathcal{O}(n^2)$. Les algorithmes de recherche d'enveloppe ont une complexité linéaire. Ceux-ci étant eux-mêmes dans une boucle parcourant toute la liste de points, on a une coût de $(n + n)^2$. Enfin, dans le pire des cas, la dernière boucle while a une complexité linéaire. Le coût temporel de l'algorithme est donc $n^2 + (n + n)^2 + n$. La fonction est donc en $\mathcal{O}(n^2)$.