Exercice 5

1a

Si les éléments du tableau sont tous positifs, il suffit d'additionner tous les éléments du tableau pour obtenir la somme maximale (la sous-séquence correspond à l'ensemble du tableau).

1b

Si les éléments du tableau sont tous négatifs, il suffit de prendre l'élément le plus grand du tableau (la sous-séquence est réduite à un seul élément)

2a

```
def somme_sous_sequence(lst, i, j):
      somme = 0
      for ind in range(i,j+1):
            somme = somme + lst[ind]
      return somme
```

2b

Pour un tableau de 10 éléments, nous avons 55 comparaisons (10+9+8+7+6+5+4+3+2+1=55).

2c

```
def pgsp(lst):
    n = len(lst)
    somme_max = lst[0]
    i_max = 0
    j_max = 0
    for i in range(n):
        for j in range(i,n):
            s  = somme_sous_sequence(lst,i,j)
            if s > somme_max:
                somme_max = s
                i_max = i
                j_max = j
    return (somme_max, i_max, j_max)
```

3a

i	0	1	2	3	4	5	6	7
lst[i]	-8	-4	6	8	-6	10	-4	-4
S(i)	-8	-4	6	14	8	18	14	10

3b

```
def pgsp2(lst):
    somme_max = [lst[0]]
    for i in range (1,len(lst)):
        if somme_max[i-1] <= 0:
            somme_max.append(lst[i])
        else :
            somme_max.append(lst[i]+somme_max[i-1])
    return max(somme_max)
```

3c

Cette solution est plus avantageuse, car la complexité en temps de l'algorithme est en O(n) alors que dans le cas précédent il était en $O(n^2)$.