TP 3

DES PILES ET DES FILES

```
## Exercice 1

def consultation(f):
    """ entrée : file f
        précondition : f non vide
        sortie : renvoie la valeur de tête de f sans modifier f """
    g = File()
    a = f.defiler()
    g.enfiler(a)
    while not f.empty():
        g.enfiler(f.defiler())
    while not g.empty():
        f.enfiler(g.defiler())
    return a


def longueurfile(f):
    """ entrée : file f
        sortie : renvoie la longueur de f """
    if f.empty():
        return 0
    g = File()
    l = 0
    while not f.empty():
        g.enfiler(f.defiler())
        l += 1
    while not g.empty():
        f.enfiler(g.defiler())
    return l


def copiefile(f):
    g = File()
    h = File()
    while not f.empty():
        a = f.defile()
        g.enfiler(a)
        h.enfiler(a)
    while not g.empty():
        f.enfiler(g.defiler())
    return h
```

```python
## Exercice 2

def elimination(F, k):
    c = 0
    for _ in range(k-1):
        F.enfiler(F.defiler())
    F.defiler()


def derniers(f, k):
    nb = longueurfile(f)
    while nb > k-1:
        elimination(f, k)
        nb -= 1
    return f


# test = File()
# test.contenu = ["Alice", "Bob", "Charles", "Dorine", "Ernest", "Flavien", "Gérard", "
    Hermione", "Ines", "Jessica"]
# print(derniers(test,2))
# test.contenu = ["Alice", "Bob", "Charles", "Dorine", "Ernest", "Flavien", "Gérard", "
    Hermione", "Ines", "Jessica"]
# print(derniers(test,3))
# test.contenu = ["Alice", "Bob", "Charles", "Dorine", "Ernest", "Flavien", "Gérard", "
    Hermione", "Ines", "Jessica"]
# print(derniers(test,4))


for n in range(2,21):
    f = File()
    for k in range(1,n+1):
        f.enfiler(k)
    g = derniers(f, 2).defiler()
    print(n, g)


# Question 2b
""" S'il y a 2n personnes au début, celui qui est en position k au deuxième tour,
    était en position k+(k-1) = 2k-1 au départ.
    S'il y a 2n+1 personnes au début, alors on commence par éliminer les 2,4,6,...,2n et 1
    puis celui désormais en position k était au départ en position 2k+1.
"""

def J(n):
    if n == 1 or n == 2:
        return 1
    if n%2 == 0:
        return 2*J(n//2)-1
    else:
        return 2*J(n//2)+1



## Exercice 3


# Question 1

def scm(s):
    L, i, n = [], 0, len(s)
    while i < n:
        j = i + 1
        while j < n and s[j] >= s[j-1]:
            j += 1
        L.append((i, j - 1))
        i = j
    return L
```

```
# Question 2

def fusionner(s, r1, r2):
    d1, f1 = r1
    d2, f2 = r2
    L = []
    while d1 <= f1 and d2 <= f2:
        if s[d1] <= s[d2]:
            L.append(s[d1])
            d1 += 1
        else:
            L.append(s[d2])
            d2 += 1
    if d1 > f1:
        for x in s[d2: f2+1]:
            L.append(x)
    else:
        for x in s[d1: f1+1]:
            L.append(x)
    for i in range(r2[1] - r1[0] + 1):
        s[r1[0] + i] = L[i]


# Question 3

def depileFusionneRemplace(s, pile):
    r2 = pile.depiler()
    r1 = pile.depiler()
    fusionner(s, r1, r2)
    pile.empiler((r1[0], r2[1]))


# Question 4

def alphaTri(s):
    def longueur(segment):
        return segment[1] - segment[0] + 1
    liste_scm = scm(s)
    nb_scm = len(liste_scm)
    pile = Pile()
    pile.empiler(liste_scm[0])
    h, i = 1, 1
    # h est la hauteur de la pile, i l'indice de la dernière scm traitée
    while i < nb_scm: # première phase
        if h >= 2:
            top = pile.depiler()
            sous_top = pile.depiler()
            pile.empiler(sous_top)
            pile.empiler(top)
            if longueur(sous_top) < 2*longueur(top):
                depileFusionneRemplace(s, pile)
                h -= 1
            else:
                pile.empiler(liste_scm[i])
                i += 1
                h += 1
        else:
            pile.empiler(liste_scm[i])
            i += 1
            h += 1
    while h > 1: # deuxième phase
        depileFusionneRemplace(s, pile)
        h -= 1
```