TD 1

(4)

1) $\sqrt{n}$ car le plus grand valeur de cpt possible est $\sqrt{n}$ et on l'augment cpt toujours en 1.

2) n car on commence de n et on fait $n-1$ jusqu'à $n=0$.

3) n^3 car si $n \neq 0$ on appelle la fonction elle même au moins 3 fois en diminuant n en 1.

4) $\log_4 n$ car on diminue chaque fois n en divisant par 4 jusqu'à $n=0$

5) $1 \dots \sqrt{n}$ $\quad 1, 2, 4, 8, 16, \sqrt{n}$

$\log_2 \sqrt{n}$

(5) $n \leq N$

Est Dans $(x, L) = l + O(1)$

Est Dans Trié $(x, L) = \log_2 L + O(1)$

Element Commun Dans Triés $(n, N) = n + N + O(1)$

Tri $(n) = n \log_2 n$

Tri $(n) = N \log_2 N$

$\Big\} n \log_2 n + N \log_2 N + n + N$

x in n:
Est Dans T $(x, N) = \log_2 N + O(1)$

$\Big\} n \log_2 N + N \log_2 N = \log_2 N (n+N)$

x in N:
Est Dans T $(x, n) = \log_2 n + O(1)$

$\Big\} N \log_2 n + n \log_2 n = \log_2 n (N+n)$ ← le meilleur.

⑥ Au mieux : 1, au pire : 366

Somme fixe : $\frac{365 \cdot 1 + 366}{366} \approx 2$ euros

2) $n_6 = 3$

000 -> 001 => 1 bits — au mieux

pile

nombre de bits de N : $\log_2(N)+1$

au pire : on change tous les bits, donc : $\log_2(N)$

④

```
if p != 0:
    if b == false:
        if q != 0 :
            return true
        else return false
    else return q == 1
else return b == true
```

```
if p == 0 : return b == true
if b == true : return (q == 1)
return q != 0
```

⑧ 1) vrai ssi $\forall a \in A, Q(a)$

```
pour tout a dans A:
    si non Q(a):
        rendre faux
rendre vrai
```

2) vrai ssi $\exists a \in A, Q(a)$

```
pour tout a dans A:
    si Q(a):
        rendre vrai
rendre faux
```

3) Vrai ssi $\forall a \in A, \exists b \in B, \forall c \in C, P(a,b,c)$

```
pour tout a dans A:
    trouvé_B = faux
    pour tout b dans B
        trouvé_C = vrai
        pour tout c dans C:
            si non P(a,b,c):
                trouvé_C = faux
                break
        si trouvé_C == vrai :
            trouvé_B = true
            Break
    si trouvé_B == faux:
        rendre faux

rendre vrai
```

(9)

1)
```
Q1();
si test:
    code
    Q1();

Q1_iter():
tant que test
    code
```

2)
```
Q2():
    code
    si test
        Q2()

Q2_iter():
    code
    tant que test
        code
```

3)
```
Q3():
    si test1:
        code1
        si test2:
            code2
            Q3()

Q3_iter():
    tant que test1:
        code 1
        si non test2:
            Break
        code 2
```

4) Q4():

```
si test1:
    Code 1
    Si test1_bis:
        Q4()
    sinon:
        tant que test2 et test2 bis:
            Code2
        Si test2 et non test2_bis:
            Code2
            Q4()
```

```
Q4_iter():
    B = vrai
    tant que B:
        Si non test1:
            B = faux
        Si non:
            Code 1
            Si non test1_bis:
                tant que test2 et test2_bis:
                    Code2
                Si non test2:
                    B = faux
                sinon:
                    Code2
```

10