Algo 2 : Etude d'une tâche
Rechercher une valeur dans un tableau trié

Algorithme 4 : `recherche_dichotomique (A, n, x)`
Entrées :
- `A` : tableau **trié**
- `n` : nb d'éléments dans `A`
- `x` : valeur recherchée dans `A`

Sorties :
- soit indice `i` tel que `A[i]=x`
- soit valeur spéciale « `Non Trouvé` »

Procédure :
1. Fixer `p=1` et `r=n`
2. Tant que `p`≤`r`, faire la chose suivante :
   A. Fixer `q` à `partie_entière( (p+r)/2 )`
   B. Si `A[q]=x`, alors retourner `q`
   C. Sinon, si `A[q]>x`, alors fixer `r=q-1`
   D. Sinon, fixer `p=q+1`
3. Retourner la valeur de `Non-Trouvé`

La **partie entière** du nombre réel 2.98 correspond à 2 : on coupe le nombre réel au niveau de la virgule (le point en notation informatique).

Exercice 1

On considère le tableau A = [10, 4, 12, -2, 4] (le même tableau que dans la première séance d'algorithmique).

(1) A la main, ré-écrire le tableau A trié par ordre croissant.

(2) Décrire toutes les étapes de l'algorithme `recherche_dichotomique(A,5,-2)`.

(3) Décrire toutes les étapes de l'algorithme `recherche_dichotomique(A,5,4)`.

(4) Décrire toutes les étapes de l'algorithme `recherche_dichotomique(A,5,12)`.

Exercice 2

On souhaite savoir comment évolue le taux de croissance du temps d'exécution de la recherche dichotomique.

(1) Reprendre la dernière question de l'exercice précédent : à chaque entrée dans la boucle `tant que`, écrire le contenu du sous-tableau `A[p,r]`.

(2) Comment généraliser le résultat de la question précédente à un tableau A de taille n quelconque ?

(3) Quel est le taux de croissance de l'algorithme ? On utilisera la notation $\mathcal{O}()$.

→

Exercice 3 Fonction factorielle récursive

La fonction mathématique $n!$ est appelée « factorielle n ». Elle est définit par les deux propriétés suivantes :

- $0! = 1$
- $n! = n \times (n-1) \times (n-2) \times \cdots \times 3 \times 2 \times 1$

Voici une proposition de réalisation de cette fonction en python :

```
def factorielle (n) :
    if n == 0 :
        return 1
    else :
        resultat = 1
        for i in range(1,n+1) :
            resultat = resultat * i
        return resultat
```

(1) Calculer 0!, 2!, 3! et 5!.

(2) Caractériser le taux de croissance de cet algorithme avec la notation $\mathcal{O}()$.

(3) Proposer une version récursive de cet algorithme.

(4) Caractériser le taux de croissance de ce nouvel algorithme avec la notation $\mathcal{O}()$.