|  |
| --- |
| **Correction activité : Recherche d’une occurrence** |

Une fois les algorithmes faits, on peut compter le nombre d’opérations (affectation, opération arithmétique, teste, itération d’une boucle, affichage, renvoi d’une fonction, etc) pour montrer que la complexité asymptotique des algorithmes utilisés est linéaire : *O(n).*

**Situation 1**

def occurence\_1(L):

n=len(L) *# (1 affectation)*

i=0 #on initialise i à zéro *(1 affectation)*

while ((i<n) and L[i]!=40): *# (n itérations + 2 testes / itération au pire des cas)*

i=i+1 #on incrémente i tant qu’on ne trouve pas le joueur gagnant *(1 affectation + 1 addition)*

if (i<n): #dans ce cas on trouve l’indice du joueur gagnant *(1 teste)*

print(i+1, "est le numéro du joueur gagnant") *# (1 affichage + 1 addition)*

else: #dans ce cas on a parcouru toute la liste L *(1 teste)*

print ("Aucun joueur a fait un score de 40") *# (1 affichage)*

Total :

**Situation 2**

def occurence\_2(L,x):

n=len(L)

i=0

while ((i<n) and L[i]!=x):

i=i+1

if (i<n):

print(i+1,"est le numéro du joueur gagnant")

else:

print ("Aucun joueur a fait un score de",x)

L’algorithme étant pratiquement le même que précédemment, puisqu’on a remplacé 40 par x, alors la complexité asymptotique est la même : *O(n)*.

**Preuve de terminaison de la boucle « while » :**

Dans la bouche « while » de la fonction *occurrence\_1bis()*, on teste si ***i < n*** et si ***L[i] ≠ 40***.

On peut prendre comme variant la variable ***i*** qui est initialisée à zéro.

Si les deux conditions sont vérifiées alors on incrémente ***i*** de ***1*** par l’opération ***i = i + 1***, puis on recommence les testes.

Deux cas possibles se présentent :

* On trouve une valeur pour  ***i*** (compris entre ***0*** et ***n-1***) telle que ***L[i] = 40*** et donc la boucle se termine : c’est le cas où on trouve un gagnant (le joueur ***i + 1***).
* On ne trouve pas de valeur pour  ***i*** (compris entre ***0*** et ***n-1***) telle que ***L[i] = 40***, donc ***i = n*** et par conséquent la boucle se termine aussi : c’est le cas où il n’y a pas de gagnant.

Ainsi, dans tous les cas la boucle se termine bien.

La boucle « while » de la fonction *occurrence\_2bis()* étant construite de la même manière (on remplace juste 40 par x), par conséquent une preuve similaire montre également sa terminaison.