REGNAULT Edgar
MOREAU Noé

CS 211 : TP n°1

I-Bubble Sort

0)Préparation :

Pseudo code Bubble Sort :

n=taille de tab

Pour rang allant de 0 à n-2

Pour i allant de 0 à n-2-rang

Si tab[i]>tab[i+1]

tmp=tab[i]
tab[i]=tab[i+1]
tab[i+1]=tmp

Pseudo code avancé Bubble Sort :

```
fonction swap( adresse entier a, adresse entier b)
{
        soit un entier n
        entier tmp=valeur contenue dans  a
        valeur contenue dans a= valeur contenue dans  b
        valeur contenue dans b=tmp;
swap ne retourne rien
}

fonction trie(tableau à n éléments, taille n du tableau)
{
        pour i allant de 0 à n-2 faire:
        {
                pour j allant de 0 à (n-i-2) faire:
                {
                        si tab[j]>tab[j+1] alors:
                                swap(adresse tab[j],adresse tab[j+1]
                }

        }
}
```

5)
Pour tab1, le programme fait 2303 comparaisons.
Pour tab2, le programme fait 2450 comparaisons.
Pour tab3, le programme fait 1617 comparaisons.
Pour ref, le programme fait 49 comparaisons.

6)
Dans le meilleur des cas (tableau déjà trié), on a une complexité linéaire : O(n).
Dans le pire des cas, on a une complexité quadratique : $O(n^2)$
Cet algorithme est donc adaptatif, car en fonction de si le tableau est déjà trié ou non, il n'effectuera pas le même nombre d'opérations.

7)
La complexité spatiale est en O(1) car on n'utilise qu'une fonction swap utilisant une variable tmp pour y stocker temporairement la valeur d'une des deux valeurs à échanger.
L'algorithme est stable car on effectue les comparaisons avec un inférieur strict : 2 valeurs égales ne seront pas échangées.

II-Insertion Sort

2)
Pour tab1, le programme fait 682 comparaisons.
Pour tab2, le programme fait 1216 comparaisons.
Pour tab3, le programme fait 63 comparaisons.
Pour ref, le programme fait 0 comparaisons.

3)
Dans le meilleur des cas (tableau déjà trié), on a une complexité constante : O(1).
Dans le pire des cas, on a une complexité quadratique : $O(n^2)$
Cet algorithme est donc adaptatif, car en fonction de si le tableau est déjà trié ou non, il n'effectuera pas le même nombre d'opérations.

4)
La complexité spatiale est en O(1) car on n'utilise qu'une fonction swap utilisant une variable tmp pour y stocker temporairement la valeur d'une des deux valeurs à échanger.
L'algorithme est stable car on effectue les comparaisons avec un supérieur strict : 2 valeurs égales ne seront pas échangées.

III-Merge sort

1)
Si on déclare tmp de cette façon: “int* tmp = tab” on crée un pointeur sur un int.
Si on déclare tmp de cette façon: “int tmp[n]”, on crée un tableau de n int permanent de classe statique.

2)
La bonne méthode est:
int* tmp=malloc(n*sizeof(int));
.