

Institut Supérieur du Numérique

Devoir de Structures de données et complexité 2024-2025

Classes : L2 (OWM, RSS, DSI)

Durée : 1h30

Exercice 1 [5 pts] Qu'affiche le programme ci-dessous ?

```
1 #include <iostream>
2 #include <cstring>
3 using namespace std;
4 int main() {
5     struct etudiant{
6         int matricule;
7         string nom;
8     };
9     int x = 5, y = 8;
10    int *ptrx = &x;
11    int *ptry = &y;
12    struct etudiant e = {23500, "Zeinaba
13    Sidi Mesoud"};
14    struct etudiant * eptr = &e;
15    cout<<*ptrx<< " " <<*ptry<<
16    "<<eptr->matricule<<
17    "<<(*eptr).nom<<endl;
18    ptrx = ptry;
19    *ptrx = 12;
20    cout<<x<< " " <<y<< endl;
21    return 0;
22 }
```

Exercice 2 [5 pts] Calcul de nombre d'opération

Soit le code suivant :

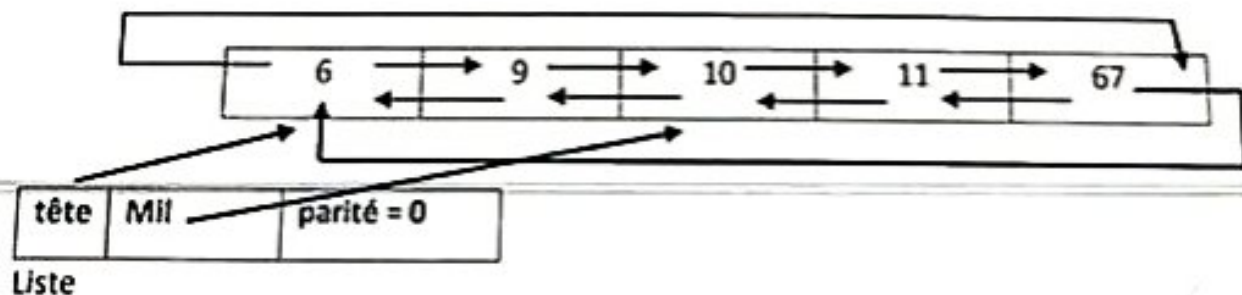
```
for(i = 1 ; i <= n; i++)
    for(j=1; j<=i; j++)
        cout<<"Bonjour la complexité" ;
```

Calculer en fonction de n , le nombre d'exécutions de l'instruction `cout` ?

Exercice 3 [10 pts] Implémentation d'une liste chaînée particulière

On considère une liste chaînée stockant des entiers. Chaque élément doit pointer sur 2 éléments : son suivant et son précédent. Le premier élément est considéré comme le suivant du dernier et le dernier est le précédent du premier.

En plus de garder un pointeur sur la tête de la liste, on doit également garder un pointeur sur l'élément au milieu. Une information permettant à tout moment de savoir si le nombre d'éléments est pair ou non doit être stockée. On doit également maintenir la liste triée selon l'ordre croissant (Voir figure ci-dessous)



1. Écrire la représentation de cette structure en mémoire (Indication : utilise 2 structures)
2. Écrire une fonction C++ qui prend en paramètres une liste L, un entier e et retourne la liste résultat de l'ajout de e à L. L'insertion doit veiller à ce que la liste reste triée, les pointeurs pointent aux bons endroits, et milieu et parité sont actualisés.
3. Écrire une fonction C++ permettant de supprimer un élément à la position i.
4. Écrire une fonction permettant de compter le nombre d'éléments de la liste en parcourant seulement sa moitié.