TP n° 4 - Les pointeurs

Exercice 1 :

Commenter le résultat d'exécution du programme suivant (expliquer ce qui s'affiche) :

```
void main()
{
      int *p;
      int x=17;
      p=&x;
      cout<<&p;
      cout<<p;
      cout<<*p;
      cout<<&x;
      cout<<x;
}
```

Exercice 2 :

Donner le résultat d'exécution des fonctions suivantes et représenter schématiquement l'état de la mémoire centrale.

```
void exemple1() {
       int * p; int x=17;
       p=&x; x++; (*p)++;
       cout<<"x="<<x<<"    *p="<<*p<<endl;
}
void exemple2() {
       char *pc, *pb, car;
       car='t'; pc=&car; pb=pc; car=car-'a'+'A';
       cout<<car<<*pc<<*pb<<endl;
}
void exemple3() {
       char mot[10];
       strcpy(mot,"papillon"); cout<<"*mot="<<*mot<<endl;
       *mot=*(mot+2)='t'; cout<<mot<<endl;
}
```

Exercice 3 : Tableau dynamique

```
#include<iostream>
#include<string>
using namespace std;

#define TAILLETAB 40

typedef int *tabDynamic;

int main(void)
{
//instructions
}
```

Soit **t1**, un tableau de **`TAILLETAB`** entiers et **t2** une variable de type **`tabDynamic`**, écrire les instructions du programme permettant de recopier dans **t1** tous les éléments positifs de **t2**.

Exercice 4 : Gestion d'un ensemble de caractères par une liste doublement chaînée.

L'objectif de ce problème est de programmer la gestion d'un ensemble de caractères par une liste doublement chaînée. Pour cela, vous utiliserez la classe définie ci-dessous et qui vous sera fourni.

```
class Noeud {
public:
      int nombre;
      Noeud * precedant;
      Noeud * suivant;
      Nœud();
      Nœud(int);
…
};
class Liste {
public :
      Nœud tete ;
      Liste() ;
…
};
```

Écrire les méthodes suivantes :

- initialisation de la liste à vide
- test de vacuité
- fonction de test d'appartenance à la liste
- procédure d'affichage des éléments de la liste
- procédure d'ajout d'un nouvel élément dans la liste
- procédure de suppression d'un élément de la liste