Programmation en C/C++

Série 12

Passage du C au C++

 Important !

 Difficile !

Objectifs
Écrire des programmes en langage C++ ; passer du C au C++.

Introduction

C++ est un langage dérivé du langage C. Il a été créé par Bjarne Stroustrup aux USA en 1985. Le langage a été normalisé par l'ISO et l'ANSI en 1998 puis 2003.
C++ est donc très voisin du langage C. Le nom C++ fait un clin d'œil à l'opérateur d'incrémentation "++" du C qui amène à penser que C++ permet de faire plus de choses que C. C++ accepte les librairies standard de C mais dispose également de ses propres librairies. C++ autorise, entre autres, la surcharge des fonctions, la programmation orientée objet.

C++ sous Codeblocks

Vous fichiers sources auront toujours l'extension *.cpp* (**C P**lus **P**lus).

N'oubliez pas de mentionner cette extension lorsque vous enregistrez votre fichier.

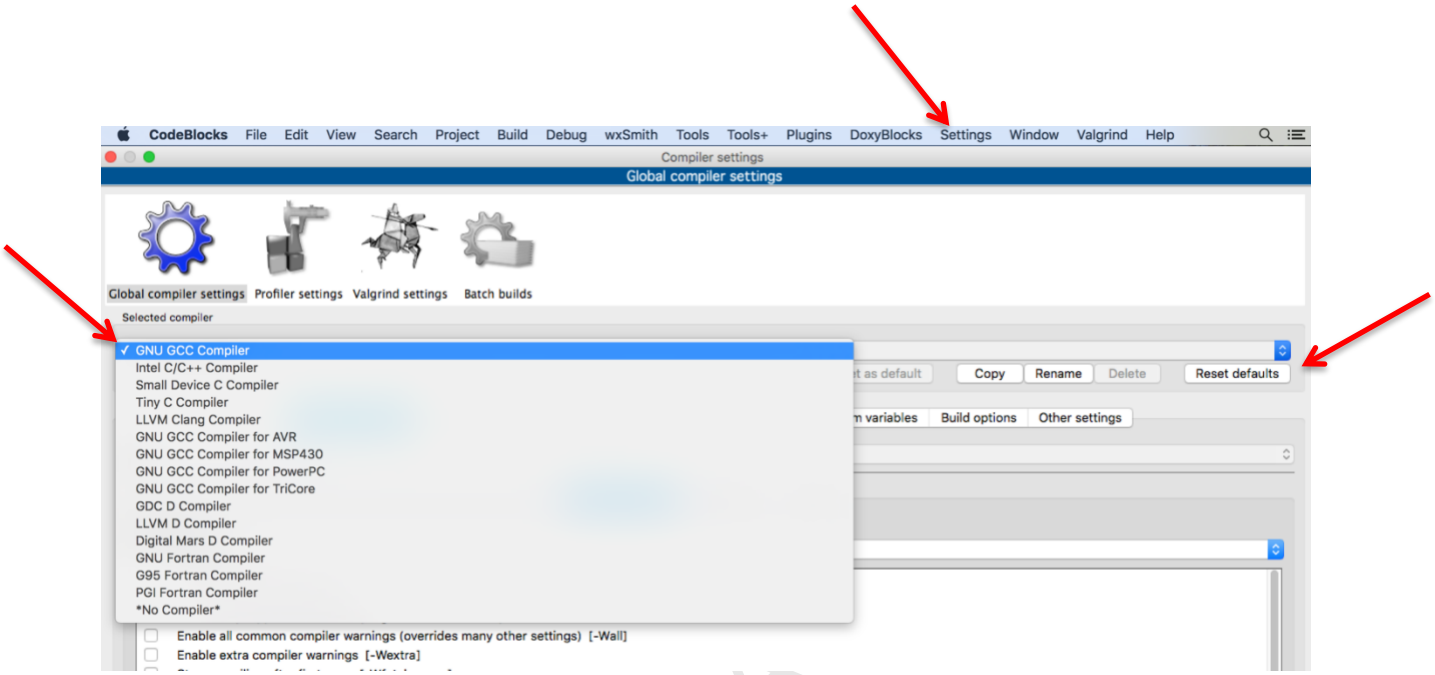
Si vous avez installé Codeblocks à partir d'une version disposant d'un compilateur intégré, un compilateur C ainsi qu'un compilateur C++ ont été normalement installés.

Si ce n'est pas le cas, il faut désinstaller Codeblocks puis télécharger une version avec compilateur intégré puis l'installer.

Lors de l'installation ou de la création du projet ne pas oublier de sélectionner le compilateur C++ (et non pas le compilateur C !).

Lors de la compilation si vous rencontrez des problèmes avec des messages du type "*Nothing to be done*" allez dans le menu :
Settings/Compiler/ puis sélectionner le compilateur *GNU CPP*.

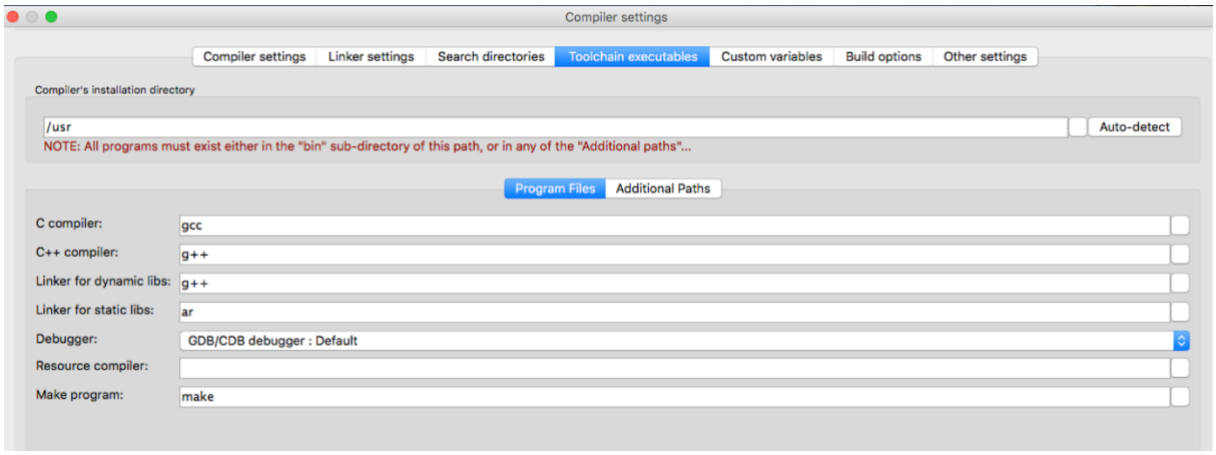
Sur la droite sélectionner "*Reset defaults*" et réinitialiser les paramètres.



Emplacement des programmes C++

Dans Settings/compiler sélectionner l'onglet "*Toolchain executables*" puis cliquer sur *Auto-detect*. Cela vous renseignera où est installé le compilateur : vos fichiers sources et exécutables devront être placés dans ce même répertoire.

Autrement, vous pouvez chercher où est placé le compilateur avec la fonction "Rechercher" du système d'exploitation et glisser ensuite votre fichier.*cpp* dans ce répertoire.



Il faut ensuite que vos fichiers sources soient installés dans le répertoire où se trouve le compilateur C++.

Librairies de base à inclure

Il vous est conseillé d'inclure systématiquement les lignes de code suivantes avant le début de votre programme principal :
#include <cstdio>
#include <cstdlib>
#include <iostream>
using namespace std;

Les trois premières lignes correspondent à des directives de compilation (*headers*) permettant d'inclure des librairies C++ ; la quatrième ligne correspond à "l'espace de noms" *std* qui permettra d'utiliser des variables, opérateurs spéciaux ou fonctions comme *cout/cin* (entrées/sorties).

Attention les *headers* n'ont pas l'extension .h en C++.

Pas de changement pour :

- Le test conditionnel *if*,
- La boucle *for*,
- La boucle *while*,
- Les tableaux,
- L'appel, les prototypes et l'utilisation de fonctions,
- La récursivité,
- Les pointeurs,
- L'écriture et la lecture dans des fichiers texte,

Des entrées/sorties simplifiées

printf est remplacé par *cout <<*

La ligne écrite en C : *printf ("la variable A vaut %d \n" , A);*
deviendra en C++ : *cout << "la variable A vaut : "<< A << endl ;*
ou bien : *cout << "la variable A vaut : "<< A << "\n" ;*
Vous remarquez que les formats *%d, %f, %c, %s* sont supprimés…

Remarque
Le mot clé *endl (*contraction de *End of Line)* permet de réaliser un retour à la ligne lors d'un affichage ou d'une écriture dans un fichier texte *("\n"* est également toléré comme en C*). endl* permet de purger également le flux de données lors d'un affichage ou d'une écriture de données dans un fichier texte.

scanf est remplacé par *cin>>*

La ligne écrite en C : *scanf ("%d " , &A) ;*
deviendra en C++ : *cin >> A ;*
Vous remarquez encore une fois que les formats *%d, %f, %c, %s* sont supprimés…

C++ est donc plus simple à utiliser que C et permet, en plus, la programmation orientée objet.

malloc remplacé par *new*

En langage C, on avait alloué de la mémoire pour un pointeur *P* sur Maillon :
*Maillon *P = malloc (sizeof(*P));*

En C++ l'instruction devient :
*Maillon *P = New (Maillon);*

En C++ : allocation d'un tableau de 100 entiers :
*int *Tab= new int [100];*

free remplacé par *delete*

En langage C, on libérait un emplacement en mémoire avec le mot clé *free* :
free(P);

En C++ l'instruction devient :
Delete P ;

En C++, destruction d'un tableau *Tab* de 100 entiers :
delete [ ] Tab;

Exemples de programmes en C++

Afin de basculer dans le langage C++ très rapidement, voici trois programmes à chercher qui reprennent les notions de variables, entrées/sorties, tests conditionnels, boucles *for* et *while*, fonctions et tableaux.
Ecrivez et saisissez entièrement les codes sources de ces programmes (pas de copier/coller).

Exemple 1 Boucle *do…while* avec des entrées/sorties en C++
Le programme ci-après demande de saisir un entier naturel (contrainte : entier positif ou nul) ; il utilise une boucle *do …while* et des entrées /sorties (instructions *cout* et *cin)*.

```
#include <cstdio>
#include <cstdlib>
#include <iostream>
using namespace std;

int main() {
int nb;

do {
        cout << "Quel entier voulez-vous saisir ? " << endl;
        cin >> nb;
    }   while (nb < 0);

cout << "vous avez saisi l'entier : " << nb << endl;

return 0;
}
```

Exemple 2 Test conditionnel et fonction en C++
Le programme ci-après demande de saisir au clavier deux entiers différents (il vérifie qu'ils sont bien différents) puis affiche le plus grand des deux.

```
#include <cstdio>
#include <cstdlib>
#include <iostream>
using namespace std;

int max(int a,int b) {
if (a>b)
        return a;
else
        return b;
}

int main() {
int x,y;
cout << "Saisir l'entier 1" << endl;
```

```
cin >> x;

do {
cout << "Saisir l'entier 2" << endl;
cin >> y;    } while (y==x);   // on boucle tant que y est égal à x

cout<<"Plus grand : "<<  max(y,x) <<"\n";
}
```

Exemple 3 Tableaux et fonctions en C++

Le programme ci-après demande combien on souhaite posséder d'entiers dans un tableau. Il appelle ensuite la fonction *Saisie()* pour rentrer au clavier les valeurs du tableau. Puis il appelle la fonction *PlusGrand()* pour afficher le plus grand élément du tableau. La directive # *include* <*vector*> est nécessaire pour gérer les tableaux (*vectors*).

```
#include <cstdio>
#include <cstdlib>
#include <iostream>
# include <vector>

using namespace std;
int PlusGrand(vector<int> Tab)
{
int Max=Tab[0];

if (Tab.size()==0) {
        cout << ‘‘Tableau vide’’ << endl;
        return -1;
}


for (int i=0; i<Tab.size();i++)
        if (Tab[i] > Max)
                Max= Tab[i];
        return Max;
}

vector<int> Saisie (int nb)
{
vector<int> T(nb,0);

for (int i=0; i< nb; i++) {
        cout << " saisir un entier ";
        cin >> T[i];
}
return T;
```

```
}

int main() {
int nb;
cout << " Nombre d'entiers dans le tableau ? : ";
cin >> nb;


vector<int> T(nb,0);
T=Saisie(nb);
cout << "Valeur la plus grande : " << PlusGrand(T) << endl;

}
```

EXERCICES (en C++)

Exercice 1 Fonction récursive

Écrire le programme C++ qui demande de saisir un entier puis affiche sa factorielle à l'aide d'une fonction récursive.

Corrigé

```
#include <cstdio>
#include <cstdlib>
#include <iostream>

using namespace std;

int Facto ( int );

int main() {
int x;

cout << " Saisir un entier x " << endl ;
cin >> x;
cout << "Resultat de la factorielle : " << Facto(x) << endl;
}

int Facto (int n) {
if (n > 1)
        return (n * Facto (n - 1));
else
        return 1;
}
```

Exercice 2

Énoncé

Écrire un programme en C++ qui demande de saisir 2 nombres entiers et affiche s'ils sont égaux ou, autrement, le plus grand des deux.

Corrigé

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

int main()
{
int a, b;

cout<< "saisir l'entier a: " << endl;
cin>>a;
```

```
cout<<"\nsaisir l'entier b: "<< endl;
cin>>b;
if (a==b)
        cout<<"\nLes deux nombres sont egaux\n" ;
else    if (a>b)
                {
                        cout <<"Le nombre "<<a<< " est plus grand que le nombre " <<b
                << endl;
                }
        else
                {
                cout<<"Le nombre "<< b << " est plus grand que le nombre " << a<<
                endl;
                }
return 0;
}
```

Exercice 3

Énoncé

Écrire un programme C++ qui demande de saisir une note et qui repose la question indéfiniment si la note tapée n'est pas comprise entre 0 et 20.

Corrigé

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

int main()
{
int Note;

cout<< "Saisir votre note :  " << endl;
cin>>Note;

while (Note <0 || Note >20)
{
cout<<"\nSaisissez une note comprise entre 0 et 20\n";
cin>>Note;
}

cout<<" Bravo ! " << endl;
return 0;
}
```

Exercice 4

Énoncé

Écrire le programme C++ qui demande de saisir un nombre entier. Si le nombre saisi est par exemple 5 il s'affichera :

```
*
 *
  *
   *
    *
```

Chaque étoile est décalée d'une espace (nom féminin en typographie) supplémentaire à chaque ligne. (0 espace avant l'étoile sur la première ligne, 1 espace avant l'étoile sur la ligne 2…)

Corrigé

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

int main()
{
   int nb, i, j;
   cout << "Entrer un nombre entier: "<< endl;
   cin >>nb;

   cout <<"*" << endl;
   for (i = 1; i < nb; i++) {
      for (j = 1; j <= i; j++) {
         cout << " " ;
      }
      cout << "*" << endl;
   }
   return 0;
}
```

Exercice 5

Énoncé

Écrire le programme C++ qui demande de saisir un nombre entier. Si le nombre saisi est par exemple 4 il s'affichera :

```
Saisir un entier:4
11
12
13
14
21
22
23
24
31
32
33
34
41
42
43
44
MacBook-Pro-de-G:~ G$
```

Si on a saisi 3 il s'affichera :

```
Saisir un entier:3
11
12
13
21
22
23
31
32
33
MacBook-Pro-de-G:~ G$
```

Corrigé

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

int main(){
int i,j;
int a;

cout <<"Saisir un entier:"<< endl;
cin>>a;

for(i=1;i<=a;i++)
{
   for(j=1;j<=a;j++)
   {
      cout << i << " " << j << endl ;
   }
}
```

return 0;

}

Exercice 6

Énoncé

Écrire un programme C++ qui écrit la lettre T à l'aide d'étoiles. On demandera en premier de saisir un nombre entier *n* (avec *n* supérieur ou égal à 3). La branche verticale du "T" aura toujours *n* étoiles ; La branche horizontale du "T" aura *n* étoiles si *n* est impair et (*n-1*) étoiles si *n* est pair. Il faudra gérer les étoiles et les espaces.

Exemples :

```
Saisissez un nombre entier superieur ou egal a 3 :
5
*****
  *
  *
  *
  *
```

Exemple avec la valeur *n* =5

```
Saisissez un nombre entier superieur ou egal a 3 :
6
*****
  *
  *
  *
  *
  *
```

Exemple avec la valeur *n* =6

Corrigé

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

int main(){
   int nb, i, j;
   cout << "Saisissez un nombre entier superieur ou egal a 3 :\n" << endl;
   cin>>nb;

//Cas des nombres impairs
   if (nb%2 != 0) // on teste si le reste de la division par 2 vaut 0
      {   //branche horizontale
          for (i=1; i<=nb; i++)
              cout <<"*" ;

          cout << "\n";
           // branche verticale
          for (i=1;i<=nb-1;i++)
```

```
            {
                for(j=1;j<=((nb+1)/2)-1;j++)
                    cout << " " ;

                cout << "*\n";
            }
    }

  //Cas des nombres pairs
  else
    {   //branche horizontale
        for (i=1;i<nb;i++)
            cout << "*";

        cout << "\n";

         //branche verticale
        for (i=1;i<=nb-1;i++)
            {
                for(j=1;j<=(nb/2)-1;j++)
                    cout << " ";

                cout << "*\n";
            }
    }
  return 0;
}
```


Exercice 7 Tableau

Énoncé

Écrire le programme C++ qui demande de saisir 5 réels au clavier, puis les affiche ainsi que le plus grand élément de ce tableau.

Corrigé

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

int main() {
int a;
float T [5];
float max;

for ( a=0; a<5; a++)   {
   cout<<"Saisir un reel:\n";
   cin>>T[a];
}
```

```
//affichage
for(a=0;a<5;a++)
   cout<<T[a]<<endl;

max =T[0];

for(a=0; a<5; a++)  {
   if (T[a]>max)
   max = T[a];
}
cout<<"Le plus grand element est " << max<<endl;

return 0;
}
```

Exercice 8 Tableau 2D

Énoncé

Soit le tableau à deux dimensions suivant :

Age	2	6	16
Poids	6	15	45

Écrire le programme C++ qui crée « en dur » un tableau avec ces valeurs puis affiche la moyenne des poids de ces enfants.

Corrigé

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

int main() {
float somme = 0;
int j;
int T[2][3] = {
{2, 6, 16},
{6,15 , 45}
};
for ( j=0; j<3; j++)
somme +=T[1][j];

somme/=3.0;
cout<<"Poids moyen : "  <<somme <<endl;

return 0;
}
```

Exercice 9 Fonctions

Énoncé

Écrire le programme principal C++ qui appelle une fonction qui affecte les valeurs entières 100, 99, 98, 97, 96 aux 5 éléments d'un tableau d'entiers puis appelle une autre fonction qui affichera le contenu de ce tableau.

Corrigé

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

void Initialisation (int Tab[], int n) {
int i;

for(i=0; i<n; i++)
    Tab[i] = 100-i;
}

void Affichage (int Tab[], int taille) {
int i;

for(i=0; i<taille; i++)
cout<< Tab[i]<<"\t";

cout<<"\n";
}

int main() {
int T[5];
Initialisation(T, 5);
Affichage(T, 5);
return 0;
}
```

Exercice 10 Tableau dynamique et libération de mémoire

Énoncé

Écrire un programme principal C++ qui demande le nombre d'éléments dans un tableau de réels. Puis alloue de la mémoire dynamiquement ; demande la saisie des réels, les affiche, désalloue la mémoire et affiche à nouveau les valeurs. Que constate-t-on ?

Corrigé

```
#include <cstdio>
#include <cstdlib>
#include <iostream>


using namespace std;


int main() {
int taille, i;


cout<<"\nSaisissez le nombre de cases du tableau : \t ";
cin>>taille;

//allocation dynamique de mémoire
float *Tab= new float [taille];

//Saisie
for (i=0; i<taille; i++) {
 cout<<"Saisissez la valeur  "<<(i+1)<<"\t ";
cin>> Tab[i];
}

//affichage
cout<<"\n";
for (i=0; i<taille; i++)
cout<<"Valeur : " <<(i+1) <<" : "<< Tab[i]<<endl;

//libération de la mémoire
delete [] Tab;

// nouvel affichage
cout<<"\n";
for (i=0; i<taille; i++)
cout<<"Valeur : " <<(i+1) <<" : "<< Tab[i]<<endl;

return 0;
}
```


Remarque
Après le *delete* de libération de la mémoire les données s'affichent toujours : les données étant toujours présentes au même emplacement mémoire ; par contre, si on modifie le code pour demander de saisir à nouveau les données le programme plante et il s'affiche bien pointeur non alloué.

```
Saisissez le nombre de cases du tableau :        3
Saisissez la valeur  1   1
Saisissez la valeur  2   2
Saisissez la valeur  3   3

Valeur : 1 : 1
Valeur : 2 : 2
Valeur : 3 : 3
Sources(983,0x115e0f5c0) malloc: *** error for object 0x7ff24e400630: pointer be
ing freed was not allocated
Sources(983,0x115e0f5c0) malloc: *** set a breakpoint in malloc_error_break to d
ebug
```

Exemple si on demande de re-saisir les données après le delete

Exercice 11 Pointeurs

Énoncé

Écrire un programme principal C++ qui demande de saisir 2 réels dans des variables et appelle ensuite une fonction qui va permuter les contenus de ces 2 variables. On n'utilisera pas de variable globale mais des pointeurs.

Corrigé

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

void Permut (double* VAR1, double* VAR2)
{
double tampon;
tampon=*VAR1;
*VAR1=*VAR2;
*VAR2=tampon;
}

int main()
{
double A,B;
cout<<"Valeur de a ?"<<endl ;
cin>>A;
cout<<"Valeur de b ?"<<endl ;
cin>>B;
Permut(&A,&B);
cout<<"A vaut : "<< A<<endl ;
cout<<"B vaut : "<< B<<endl ;
return 0;
}
```

Exercice 12 Écriture dans un fichier texte

Énoncé

A l'aide d'un programme C++, écrire à la volée 5 nombres entiers saisis au clavier, à la suite, sur une même ligne, séparés par une espace, dans un fichier texte nommé Test.txt.

Corrigé

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

int main()
{
int Nombre,i ;
FILE* fic = NULL;
fic = fopen("Test2.txt","w");

if (fic != NULL)
        {
        for (i=1 ; i<= 5 ; i++)
        {
                cout<<"Saisir un entier"<<endl;
                cin>>Nombre;
                fprintf(fic, "%d ", Nombre); // attention espace après le %d
        }

        fclose (fic);
        cout<<"Nombres enregistres dans le fichier";
        }
else
        cout<<"Ouverture impossible";

return 0;
}
```

Exercice 13 Lecture d'un fichier texte

Énoncé

Écrire un texte dans un fichier texte nommé *Test.txt* en utilisant uniquement des points comme symbole de ponctuation ; enregistrez le fichier.

Écrire ensuite le programme C++qui appelle une fonction qui :

- Affiche ce texte,
- Renvoie le nombre de mots présents dans ce texte.

Corrigé

```
#include <iostream>
#include <cstdio>
#include <cstdlib>

using namespace std;

int NbMot(FILE *fic);

int main() {
```

```
FILE *fic;
fic = fopen("Test.txt","r");
if (fic != NULL){
   cout<<"le nombre de mot(s) est : "<< NbMot (fic)<<endl;
   fclose(fic);
   }
else{
   cout<<"ouverture impossible\n";
   }
return 0;
}

int NbMot(FILE *fic){
int NbEspaces =0;
char caract;

while (!feof(fic)){
   caract = fgetc(fic);
   if (caract!=EOF)
        cout<<caract;
   if (caract == 32 )
        NbEspaces ++;
}
cout<<"\n";
return ++NbEspaces;
}
```


Exercice 14 Liste chaînée

Énoncé

Créer un maillon "10" puis la liste chaînée 10 - 9 – 8 – 7 - 6…. 0 - NULL où les maillons sont ajoutés les uns après les autres, par le bas.

Pour rappel le mot clé *malloc* est remplacé par l'instruction *new*, en C++.

Corrigé

```
#include <iostream>
#include <cstdio>
#include <cstdlib>

using namespace std;

typedef struct Maillon Maillon;
struct Maillon{
int valeur;
Maillon *suivant; };
```

```
int main(){
int i=0;
Maillon *maListe ;
Maillon *maListebis ;
Maillon *element = new (Maillon);

//création du premier maillon
element->valeur = 10;
element->suivant=NULL;

maListe = element;

// duplication de la liste
maListebis=maListe;

for (i=9; i>=0;i--) {
        //Création d'un nouveau maillon
        Maillon *element = new (Maillon);
        element->valeur = i;
        element->suivant = NULL;

        //parcours de la liste
        while (maListebis->suivant != NULL)
                {
                maListebis = maListebis->suivant;
                }

        //branchement du maillon en fin de liste
        maListebis->suivant = element;

        //Remontée de maListebis en tête de liste
        maListebis=maListe;
        }

//affichage de la liste
while (maListebis != NULL)    {
        cout << maListebis->valeur;
        maListebis = maListebis->suivant;
```

```
        }
cout <<" NULL";

}
```