



ECOLE NORMALE SUPÉRIEURE DE L'ENSEIGNEMENT
TECHNIQUE MOHAMMEDIA
DÉPARTEMENT MATHÉMATIQUES ET INFORMATIQUE

25/03/2022

Compte rendu

Des travaux pratique de la
chapitre 1

**C++ COMME UN LANGAGE C
AVANCE-PRINCIPAUX APPORTS**

PRÉPARÉE POUR
TAFFAH ACHRAF

ENCADRÉ PAR
PR. KHALIFA MANSOURI



TABLE DES MATIERES

Introduction	3
Cadre général	Erreur ! Signet non défini.
Partie pratique	4
1. Exemples de cour	4
REMARQUE :	4
EXEMPLE 1 :	4
EXEMPLE 2 :	5
EXEMPLE 3 :	5
EXEMPLE 4 :	6
EXEMPLE 5 :	7
EXEMPLE 6 :	7
EXEMPLE 7 :	8
EXEMPLE 8 :	9
EXEMPLE 9 :	10
EXEMPLE 10 :	10
2. Exercices des travaux pratique	11
Exercice 1 :	11
Exercice 2 :	11
Exercice 3 :	12
Exercice 4 :	13
Exercice 5 :	14
Exercice 6 :	15
Exercice 7 :	16
Exercice 8 :	17
Exercice 9 :	19
Conclusion	20

INTRODUCTION

IL EST IMPORTANT D'APPRENDRE LA PROGRAMMATION PROCEDURELE AVANT D'APPRENDRE LA PROGRAMMATION ORIENTEE OBJET PARCE QUE LA PROGRAMMATION MODULAIRE QUI EST GENERALEMENT MELANGEE ET CONFONDUE AVEC LA PROGRAMMATION PROCEDURELE, PEUT ETRE APPLIQUEE A LA POO. PAR CONSEQUENT, EN PLUS LA PLUPART DES LANGAGES DE PROGRAMMATION SONT MULTIPARADIGME, A UN CERTAIN NIVEAU, MEME SI LEURS CONCEPTEURS OU DEVELOPPEURS COMMUNS DISENT LE CONTRAIRE, ET AUSSI PARCE QUE LA POO EST PLUS COMPLEXE QUE LA PROGRAMMATION PROCEDURELE, IL EST DONC PREFERABLE D'APPRENDRE LA PROGRAMMATION PROCEDURELE EN PREMIER.

PARTIE PRATIQUE

1. Exemples de cour

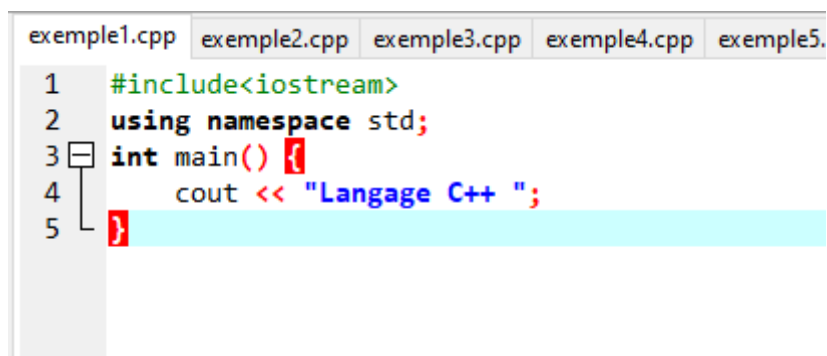
REMARQUE :

Dans les exemples suivants J'ai trouvé les erreurs suivantes et j'ai les corrigés :

1. Il ne faut pas Inclure une bibliothèque qui n'existe pas « iostream.h"
2. Dépréciatif conversion from string
3. La fonction main de type **void** ça ne marche pas bien.

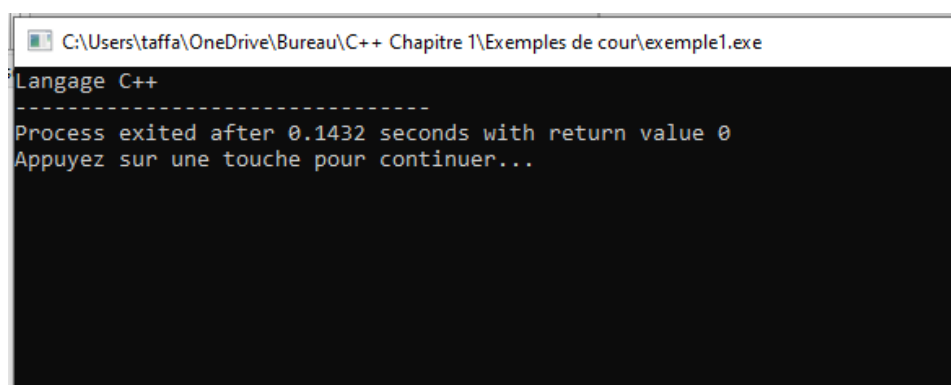
EXEMPLE 1 :

Ce programme permet d'afficher « Langage c++ ».



```
exemple1.cpp  exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.
1  #include<iostream>
2  using namespace std;
3  int main() {
4      cout << "Langage C++ ";
5  }
```

Voilà l'affichage après l'exécution de ce programme.



```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple1.exe
Langage C++
-----
Process exited after 0.1432 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXEMPLE 2 :

Ce programme permet d'afficher deux valeurs de type différentes.

```
exemple1.cpp | exemple2.cpp | exemple3.cpp | exemple4.cpp | exemple5.cpp | exemple6.cpp | exemple7.cpp
1  #include<iostream>
2  using namespace std;
3  int main(){
4      float Pi = 3.14;
5      cout << " La valeur de Pi est : "; //Cas 1
6      cout << Pi;
7
8      cout << "\n La valeur de Pi est : " << Pi; //Cas 2
9  }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple2.exe
La valeur de Pi est : 3.14
La valeur de Pi est : 3.14
-----
Process exited after 0.2959 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXEMPLE 3 :

Ce programme permet d'afficher la puissance deuxième d'un entier donné.

```
exemple1.cpp | exemple2.cpp | exemple3.cpp | exemple4.cpp | exemple5.cpp | exemple6.cpp | exemple7.cpp
1  #include<iostream>
2  using namespace std;
3  int main(){
4      int N;
5      cout << " Entrer un nombre entier : ";
6      cin >> N;
7      cout << "Le carré du nombre entié est : " << N*N;
8      return 0;
9  }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple3.exe
Entrez un nombre entier : 15
Le carré du nombre entier est : 225
-----
Process exited after 2.474 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXEMPLE 4 :

Ce programme permet de transformer le type d'une variable à un autre type.

```
exemple1.cpp  exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.cpp  exemple6.cpp  exemp
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  int main() {
5      char c='m' , d=25,e;
6      int i=42,j;
7      float r=678.9,s;
8      j = c;
9      cout << j << "\n"; // j vaut 109
10     j = r;
11     cout << j << "\n"; // j vaut 678
12     s = d;
13     cout << s<<"\n"; // s vaut 25.0
14     cout << s<<"\n"; // s vaut 25
15     e = i;
16     cout << e<<"\n"; // e vaut *
17     getch();
18 }
19
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple4.exe
109
678
25
25
*
```

EXEMPLE 5 :

Dans ce programme l'opérateur de résolution « : » permet d'accéder aux variables globales plutôt qu'aux variables locales.

```
exemple1.cpp  exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.cpp  exemple6.cpp  exemple7.cpp
1  #include<iostream>
2  using namespace std;
3  int i=11;
4  int main (){
5      int i = 34;
6      {
7          int i = 23;
8          ::i = ::i+1;
9          cout << ::i<< " " << i << endl;
10     }
11     cout << ::i << " " << i << endl;
12 }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple5.exe
12 23
12 34
-----
Process exited after 0.1604 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXEMPLE 6 :

Dans ce programme nous avons réalisé trois fonctions de même nom « somme », mais avec des signatures différentes.

```
exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.cpp  exemple6.cpp  exemple7.cpp  e
1  #include<iostream>
2  using namespace std;
3
4  int somme (int n1, int n2){
5      return n1 + n2;
6  }
7  int somme (int n1, int n2, int n3){
8      return n1 + n2 + n3;
9  }
10 int somme (double n1, double n2){
11     return n1 + n2;
12 }
13 int main(){
14     cout << "1 + 2 = " << somme(1,2) << endl;
15     cout << "1 + 2 + 3 = " << somme(1,2,3) << endl;
16     cout << "1.2 + 2.3 = " << somme(1.2, 2.3) << endl;
17 }
```

Voilà l’affichage après l’exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple6.exe
1 + 2 = 3
1 + 2 + 3 = 6
1.2 + 2.3 = 3
-----
Process exited after 0.4168 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXEMPLE 7 :

Dans ce programme nous avons réalisé trois fonctions de même nom « affiche », mais avec des signatures différentes.


```

exemple1.cpp  exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.cpp  exemple6.cpp  [*] exemple7.cpp  exemple8.cpp  exemple9.cpp
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  struct complexe{
5      double reel,im;
6  };
7  void affiche(int);
8  void affiche(double);
9  void affiche(complexe);
10 int main(){
11     int a=5;
12     double b=0.0; //error nome variable d
13     complexe c={1.0,-1.0};
14     affiche(a); //Appel la fonction (1)
15     affiche(b); //Appel la fonction (2)
16     affiche(c); // Appel la fonction (3)
17 }
18 void affiche(int i){
19     cout << "Type de variable (int) :" << endl;
20     cout << "Valeur : " << i << endl;
21 }
22 void affiche(double d){
23     cout << "Type de variable (double) :" << endl;
24     cout << "Valeur : " << d << endl;
25 }
26 void affiche(complexe c){
27     cout << "Type de variable (complexe) :" << endl;
28     cout << "Valeur : " << c.reel << endl;
29     cout << "Valeur : " << c.im << endl;
30 }

```

Voilà l'affichage après l'exécution de ce programme.

```

C:\Users\taffa\OneDrive\Bureau\C++ Chapitre 1\Exemples de cour\exemple7.exe
Type de variable (int) :
Valeur : 5
Type de variable (double) :
Valeur : 0
Type de variable (complexe) :
Valeur : 1
Valeur : -1

-----
Process exited after 0.207 seconds with return value 0
Appuyez sur une touche pour continuer...

```

EXEMPLE 8 :

L'opérateur new permet d'allouer la mémoire dans ce programme.

exemple1.cpp	exemple2.cpp	exemple3.cpp	exemple4.cpp	exemple5.cpp	exemple6.cpp	exemple7.cpp	exemple8.cpp	exemple9.cpp
--------------	--------------	--------------	--------------	--------------	--------------	--------------	--------------	--------------

```

1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4
5  int main(){
6      int *ptr1,*ptr2,*ptr3;
7      // Allocation dynamique d'un entier
8      ptr1 = new int;
9
10     // Allocation d'un tableau de 10 entiers
11     ptr2 = new int [10];
12
13     // Allocation d'un entier avec initialisation
14     ptr3 = new int(10);
15
16     struct date {
17         int jour,mois,an;
18     };
19     date *ptr4, *ptr5, *ptr6, d = {25, 4, 1952};
20
21     // Allocation dynamique d'une structure
22     ptr4 = new date;
23
24     // Allocation dynamique d'un tableau d'un tableau de structure
25     ptr5 = new date[10];
26
27     // Allocation dynamique d'une structure avec initialisation
28     ptr6 = new date(d);
29     return 0;
30 }

```

EXEMPLE 9 :

L'opérateur **new** permet d'allouer la mémoire et l'opérateur **delete** permet de libérer ladite dans ce programme.

exemple1.cpp	exemple2.cpp	exemple3.cpp	exemple4.cpp	exemple5.cpp	exemple6.cpp	exemple7.cpp	exemple8.cpp	exemple9.cpp
--------------	--------------	--------------	--------------	--------------	--------------	--------------	--------------	--------------

```

1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  int main(){
5      int *pi = new int; // il faut allouer avant de desallouer
6      delete pi; // desalloue la zone adressée par pi
7      // pi existe encore mais pas pi*
8      char *pc = new char[100];
9
10     delete pc; // desalloue la zone de 100 caractères
11     // delete [100]pc; // instruction équivalente
12     return 0;
13 }

```

EXEMPLE 10 :

Dans ce programme nous avons alloué et désalloué un tableau d'objets.

```

exemple1.cpp  exemple2.cpp  exemple3.cpp  exemple4.cpp  exemple5.cpp  exemple6.cpp  exemple7.cpp  exemple8.cpp  exemple9.cpp  exemple10.cpp
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4
5  struct complexe{
6      double reel,im;
7  };
8
9  int main(){
10     complexe *z;
11     z = new complexe[50];
12     delete z; // ne libère que le premier élément
13     // delete [50]z;
14     // ou
15     delete []z;
16     return 0;
17 }

```

2. Exercices des travaux pratique

EXERCICE 1 :

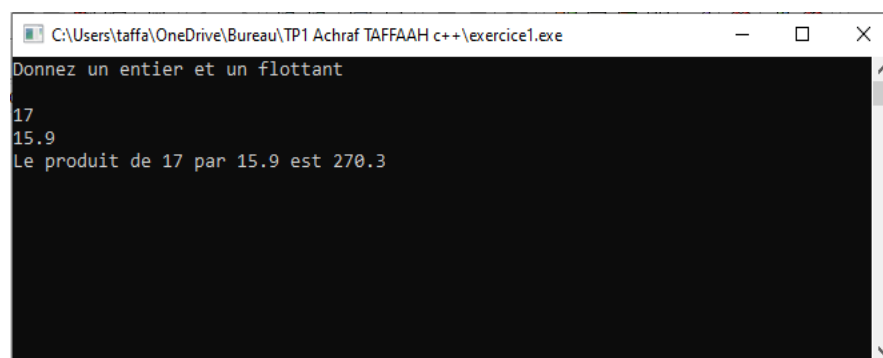
Ce programme permet de scanner un entier et un réel, puis d'afficher la valeur des variables.

```

exercice1.cpp
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  int main(){
5      int n;
6      float x;
7
8      cout << "Donnez un entier et un flottant\n" << endl;
9      cin >> n >> x;
10     cout << "Le produit de " << n << " par " << x << " est " << n*x << "\n" << endl;
11     getch();
12     return 0;
13 }

```

Voilà l'affichage après l'exécution de ce programme.



```

C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice1.exe
Donnez un entier et un flottant
17
15.9
Le produit de 17 par 15.9 est 270.3

```

EXERCICE 2 :

J'ai testé le programme original et J'ai trouvé les erreurs suivantes :

4. Il ne faut pas Inclure une bibliothèque qui n'existe pas « iostream.h »
5. Dépréciatif conversion from string
6. La fonction main de type **void** ça ne marche pas bien.

```

1 #include <iostream> //il faut utilisé iostream au lieux de iostream.h
2 #include <conio.h>
3 using namespace std;
4 int main(){
5     int i,n=25, *p;
6     char *CH="On est à l'IGA !"; //Deprecated conversion from string to 'char*'
7     float x=25.359;
8
9     cout<<"Bonjour\n";
10    cout<<CH<<"\n";
11    cout<<"BONJOUR\n"<<CH<<"\n";
12    cout<<"n= "<<n<<" x= "<<x<<" p= "<<p<<"\n";
13    getch();
14 }

```

Voilà l'affichage après l'exécution de ce programme.

```

C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice2.exe
Bonjour
On est à l'IGA !
BONJOUR
On est à l'IGA !
n= 25 x= 25.359 p= 0

```

EXERCICE 3 :

J'ai testé le programme original et J'ai trouvé les erreurs suivantes :

1. Il ne faut pas inclure une bibliothèque qui n'existe pas « iostream.h"
2. La fonction « main » de type **void** ça ne marche pas bien.
3. Il faut utiliser un **namespace** dans le programme.

```

1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 int main(){
5     int n;
6     char tc[30],c;
7     float x;
8     cout<<"Saisir un entier:";
9     cin>>n;
10    cout<<"Saisir un réel:";
11    cin>>x;
12    cout<<"Saisir une phrase:";
13    cin>>tc;
14    cout<<"Saisir une lettre:";
15    cin>>c;
16    cout<<"Affichage : "<<n<<" "<<x<<" "<<tc<<" "<<c<<"\n";
17    return 0;
18    getch();
19 }

```

Voilà l'affichage après l'exécution de ce programme.

C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice3.exe

```
Saisir un entier:19
Saisir un réel:15.5
Saisir une phrase:achraf
Saisir une lettre:a
Affichage : 19 15.5 achraf a
```

```
-----
Process exited after 11.82 seconds with return value 0
Appuyez sur une touche pour continuer...
```

Je peux conclure qu'il faut toujours donner à la fonction main le type **Int** au lieu de **void**, et aussi il faut inclure des bibliothèques existentes.

EXERCICE 4 :

Voilà le code du programme qui calcule la puissance nième de x.

```
1  #include <iostream>
2  #include <conio.h>
3  using namespace std;
4  float puissance(float x, int n=4){
5      if ( n == 0 )
6          return 1;
7      if (n>0){
8          float pow_x_n=1;
9          for(int i=1;i<=n;i++){
10             pow_x_n=pow_x_n*x;
11          }
12          return pow_x_n;
13      }
14      if (n<0){
15          float pow_x_n_negative=1/puissance(x,-n);
16          return pow_x_n_negative;
17      }
18  }
19  void display(float value){
20      cout << value << "\n" << endl;
21  }
22  int main(){
23      float r1,r2,r3,r4,r5,r6,r7,r8;
24      // calcul
25      r1=puissance(2,2); r2=puissance(47,10); r3=puissance(9,66); r4=puissance(45,2.5);
26      r5=puissance(7,10.2); r6=puissance(18,0); r7=puissance(2,-2); r8=puissance(-10,-1);
27      // Affichage
28      display(r1); display(r2); display(r3); display(r4); display(r5); display(r6); display(r7); display(r8);
29      return 0;
30      getch();
31  }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice4.exe
4
5.25991e+016
inf
2025
2.82475e+008
1
0.25
-0.1
```

EXERCICE 5 :

J'ai testé le programme original et J'ai trouvé les erreurs suivantes :

1. Il ne faut pas inclure une bibliothèque qui n'existe pas « `iostream.h` »
2. La fonction « `main` » de type `void` ça ne marche pas bien.
3. Il faut utiliser un **namespace** dans le programme.

```
1 #include <iostream>
2 #include <conio.h>
3 using namespace std;
4 void test(int n=0, float x=2.5){
5     cout << "Fonction N1 : ";
6     cout << "n= " << n << " x= " << x << "\n";
7 }
8 void test(float x=4.1, int n=2){
9     cout << "Fonction N2 : ";
10    cout << "n= " << n << " x= " << x << "\n";
11 }
12 int main(){
13     int i=5; float r=3.2;
14     test(i,r);
15     test(r,i);
16     test(i);
17     test(r);
18     // Les appels suivants, ambigus, sont rejetés par le compilateur
19     // test();
20     // test(i,i)
21     // test(r,r)
22     // Les initialisations par défaut de x à la valeur 4.1
23     // et de n à 0 sont inutilisables
24     return 0;
25     getch();
26 }
```

Voilà l'affichage après l'exécution de ce programme.

C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice5.exe

```
Fonction N1 : n= 5 x=3.2
Fonction N2 : n= 5 x=3.2
Fonction N1 : n= 5 x=2.5
Fonction N2 : n= 2 x=3.2

-----
Process exited after 0.2301 seconds with return value 0
Appuyez sur une touche pour continuer...
```

EXERCICE 6 :

J'ai testé le programme original et J'ai trouvé les erreurs suivantes :

1. Il ne faut pas inclure une bibliothèque qui n'existe pas « iostream.h »
2. La fonction « main » de type **void** ça ne marche pas bien.
3. Il faut utiliser un **namespace** dans le programme.

```
1 #include<iostream>
2 #include<conio.h>
3 using namespace std;
4
5 void essai(float x,char c,int n=0){
6     cout <<"Fonction N1: x = " << x <<" c = " << c <<" n = " << n <<"\n";
7 }
8
9 void essai(float x,int n){
10     cout <<"Fonction N2 : x = " << x <<" n = " << n <<"\n";
11 }
12
13 int main(){
14     char l='z';
15     int u=4;
16     float y = 2.0;
17     essai(y,l,u); /* fonction N1 */
18     essai(y,l); /* fonction N1 */
19     essai(y,u); /* fonction N2 */
20     essai(u,u); /* fonction N2 */
21     essai(u,l); /* fonction N1 */
22     // essai(y,y); /* rejet par le compilateur */
23     essai(y,y,u); /* fonction N1 */
24     getch();
25 }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice6.exe
Fonction N1: x = 2 c = z n = 4
Fonction N1: x = 2 c = z n = 0
Fonction N2 : x = 2 n = 4
Fonction N2 : x = 4 n = 4
Fonction N1: x = 4 c = z n = 0
Fonction N1: x = 2 c = 0 n = 4
```

EXERCICE 7 :

Voilà le code du programme qui calcule et affiche la puissance nième de x, qui ont le même nom mais avec une signature différente.

```
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  void affiche(float,int);
5  void affiche(int,float);
6
7  int main(){
8      affiche((float)7.8,2); affiche(7,(float)2.8);
9      affiche((float)1.9,2); affiche(10,(float)1.8);
10     affiche((float)6.7,7); affiche(2,(float)4.9);
11     getch();
12 }
13
14 void affiche(float x,int n=0){
15     if( (n == 0) || (x == 0 && n == 0))
16         cout << "puissance("<x<<","<n<<") est : 1 " << endl;
17     if(n>0){
18         float pow_x_n=1;
19         for(int i=1;i<=n;i++){
20             pow_x_n=pow_x_n*x;
21         }
22         cout << "puissance("<x<<","<n<<") est : " << pow_x_n << endl;
23     }
24 }
25
26 void affiche(int n,float x=0){
27     affiche(x,n);
28 }
```

Voilà l’affichage après l’exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice7.exe
puissance(7.8,2) est : 60.84
puissance(2.8,7) est : 1349.29
puissance(1.9,2) est : 3.61
puissance(1.8,10) est : 357.047
puissance(6.7,7) est : 606071
puissance(4.9,2) est : 24.01
```


EXERCICE 8 :

J'ai testé le programme original et J'ai trouvé les erreurs suivantes :

1. Il ne faut pas inclure une bibliothèque qui n'existe pas « `iostream.h` »
2. La fonction « `main` » de type `void` ça ne marche pas bien.
3. Il faut utiliser un `namespace` dans le programme.

- ✓ Dans le premier cas je vois qu'il y a un passage par valeur, alors la valeur des variables ne change pas dans la mémoire.

```
1  #include <iostream>
2  #include <conio.h>
3  using namespace std;
4  void echange(int a,int b){
5      int tampon;
6      tampon = b; b=a;a=tampon;
7      cout<<"Pendant l'échange: a = "<<a<<" b = "<<b<<"\n";
8  }
9  int main(){
10     int u=5,v=3;
11     cout<<"Avant échange: u = "<<u<<" v = "<<v<<"\n";
12     echange(u,v);
13     cout<<"Après échange: u = "<<u<<" v = "<<v<<"\n";
14     getch();
15     return 0;
16 }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice8_v1.exe
Avant Échange: u = 5 v = 3
Pendant l'Échange: a = 3 b = 5
Après Échange: u = 5 v = 3
```

- ✓ Dans le deuxième cas je vois qu'il y a un passage par référence, alors la valeur des variables va changer dans la mémoire.

```

1  #include <iostream>
2  #include <conio.h>
3  using namespace std;
4  void echange(int *a,int *b){
5      int tampon;
6      tampon = *b; *b=*a;*a=tampon;
7      cout<<"Pendant l'échange: a = "<<*a<<" b = "<<*b<<"\n";
8  }
9  int main(){
10     int u=5,v=3;
11     cout<<"Avant échange: u = "<<u<<" v = "<<v<<"\n";
12     echange(&u,&v);
13     cout<<"Après échange: u = "<<u<<" v = "<<v<<"\n";
14     getch();
15     return 0;
16 }

```

Voilà l'affichage après l'exécution de ce programme.

```

C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice8_v2.exe
Avant Échange: u = 5 v = 3
Pendant l'Échange: a = 3 b = 5
AprPs Échange: u = 3 v = 5

```

- ✓ Dans le troisième cas je vois qu'il y a un passage par adresse, alors la valeur des variables va changer dans la mémoire.

```

1  #include <iostream>
2  #include <conio.h>
3  using namespace std;
4  void echange(int &a,int &b){
5      int tampon;
6      tampon = b; b=a;a=tampon;
7      cout<<"Pendant l'échange: a = "<<a<<" b = "<<b<<"\n";
8  }
9  int main(){
10     int u=5,v=3;
11     cout<<"Avant échange: u = "<<u<<" v = "<<v<<"\n";
12     echange(u,v);
13     cout<<"Après échange: u = "<<u<<" v = "<<v<<"\n";
14     getch();
15     return 0;
16 }

```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice8_v3.exe
Avant l'change: u = 5 v = 3
Pendant l'change: a = 3 b = 5
Après l'change: u = 3 v = 5
```

EXERCICE 9 :

J'écris un programme d'essai de la fonction **Remise_a_zero** qui permet de remettre à zéro les 2 champs d'une structure de ce type, transmise en argument soit par adresse ou bien par référence, ce programme va afficher les valeurs d'une structure de ce type, après l'appel de cette fonction.

```
2  #include<conio.h>
3  using namespace std;
4
5  struct essai{
6      int n;
7      float x;
8  };
9
10 void Remise_a_zero(struct essai* e){
11     e->n=0;
12     e->x=0;
13 }
14
15 void Remise_a_zero(struct essai &e){
16     e.n=0;
17     e.x=0;
18 }
19
20 void display(struct essai* e){
21     cout << "la valeur de n est : " << e->n << "\nla valeur de x est : " << e->x << endl;
22 }
23
24 int main(){
25     // ::::::::::::::::::::Remise_a_zero par référence::::::::::::::::::
26     cout << "Cas 1 :\nAvant la remise a zero" << endl;
27     essai *test = new essai;
28     display(test);
29     cout << "Après la remise a zero" << endl;
30     Remise_a_zero(test);
31     display(test);
32     // ::::::::::::::::::::Remise_a_zero par adresse::::::::::::::::::
33     cout << "\nCas 2 :\nAvant la remise a zero" << endl;
34     essai *test2 = new essai;
35     display(test2);
36     cout << "Après la remise a zero" << endl;
37     Remise_a_zero(*test);
38     display(test);
39     return 0;
40 }
```

Voilà l'affichage après l'exécution de ce programme.

```
C:\Users\taffa\OneDrive\Bureau\TP1 Achraf TAFFAAH c++\exercice9.exe
Cas 1 :
Avant la remise a zero
la valeur de n est : 12540064
la valeur de x est : 0
Après la remise a zero
la valeur de n est : 0
la valeur de x est : 0

Cas 2 :
Avant la remise a zero
la valeur de n est : 12540064
la valeur de x est : 0
Après la remise a zero
la valeur de n est : 0
la valeur de x est : 0

-----
Process exited after 0.1042 seconds with return value 0
Appuyez sur une touche pour continuer...
```

CONCLUSION

DANS CES TRAVAUX PRATIQUE J'APPRENDRAI LES PRINCIPES DE LA PROGRAMMATION PROCEDURAL EN C++, MAIS CONTRAIREMENT A LA PROGRAMMATION PROCEDURALE QUI REPOSE SUR DES SERIES D'INSTRUCTIONS ET DE LOGIQUE, LA PROGRAMMATION ORIENTEE OBJET EST UN STYLE DE PROGRAMMATION QUI S'ARTICULE AUTOUR D'OBJETS RENFERMANT DES DONNEES ET DES MECANISMES.