ECOLE NORMALE SUPÉRIEURE
DE L'ENSEIGNEMENT TECHNIQUE DE
MOHAMMEDIA

UNIVERSITÉ HASSAN II DE
CASABLANCA

ENSET

المدرسة العليا لأساتذة التعليم التقني

المحمدية
جامعة الحسن الثاني بالدار البيضاء

COMPTE RENDU

chapitre 1 les exemples de cour + TP

Abderrahmane Ettounani

II-BDCC

I- TP

Exercice 1 :

```cpp
#include <iostream> // cin et cout sont definis dans iostream
using namespace std;
int main()
{
    int n;
    float x;

    cout << "donnez un entier et un flottant \n"; // equivalent de printf
    cin >> n;                                     // equivalent de scanf
    cin >> x;
    cout << "le produit de " << n << " par " << x << "\n est " << n * x << endl;
    return 0;
}
```

```
donnez un entier et un flottant
1
3.2
le produit de 1 par 3.2
 est 3.2
```

Exercice 2 :

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
int main()
{
    int i,n=25,*p;
    char *CH="On est à l'IGA !";
    float x=25.359;

    cout <<"BONJOUR\n";
    cout <<CH<<"\n";
    cout <<"BONJOUR \n"<<CH<<"\n";
    cout <<"n= "<<n<<" x= "<<x<<" p= "<<p<<"\n";

    //getch();

    return 0;
}
```

```
BONJOUR
On est à l'IGA !
BONJOUR
On est à l'IGA !
n= 25 x= 25.359 p= 0
```

Exercice 3 :

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
int main()
{
    int n;
    char tc[30], c;
    float x;

    cout << "Saisir un entier : ";
    cin >> n;

    cout << "Saisir un réel : ";      You, 44 seconds ago • Uncommitted changes
    cin >> x;

    cout << "Saisir un phrase : ";
    cin >> tc;

    cout << "Saisir un lettre : ";
    cin >> c;

    cout << "Affichage :  " << n << " " << x << " " << tc << " " << c << "\n";
    cin >> tc;

    // getch();

    return 0;
}
```

||

```
Saisir un entier : 2
Saisir un réel : 2.4
Saisir un phrase : salam
Saisir un lettre : enset
Affichage :  2 2.4 salam e
(base) ettounani@tounani:/media/ettounani/D/GitHub/tp-mansouri/tp-mansouri/ex3$ ./ex3
Saisir un entier : k
Saisir un réel : Saisir un phrase : Saisir un lettre : Affichage :  0 3.08398e-41 ▯8▯▯
(base) ettounani@tounani:/media/ettounani/D/GitHub/tp-mansouri/tp-mansouri/ex3$ ./ex3
Saisir un entier : 2.4
Saisir un réel : Saisir un phrase : 5
Saisir un lettre : bonjour
Affichage :  2 0.4 5 b
(base) ettounani@tounani:/media/ettounani/D/GitHub/tp-mansouri/tp-mansouri/ex3$ ./ex3
Saisir un entier : 22
Saisir un réel : 4.5
Saisir un phrase : enset media
Saisir un lettre : Affichage :  22 4.5 enset m
```

1 2 3 4

on conclura que cin stop quand il lit un espace (4)
il faut respecter les types des données (2)

Exercice 4 :

```
You, 1 second ago | 1 author (You)
#include <iostream>
//#include <conio.h>
using namespace std;

float puissance(float x, int n = 4)
{
    float resultat = 1, i = 0;
    while (i < n)
    {
        resultat = resultat * x;
        i++;
    }
    return resultat;
}

int main()
{

    cout << puissance(2, 3.4);

    // getch();

    return 0;
}
```

⇒ ⇒ ⇒

```
la puissance est : 8
```

on conclure que la fonction prendre seulement la partie decimal de 3.4

donc il fait un casting

Exercice 5:

ex5 > ex5.cpp > main()

```cpp
#include <iostream>
using namespace std;
void test(int n=0,float x=2.5){
    cout <<"Fontion N°1 : ";
    cout << "n= "<<n<<" x= "<<x<<"\n";
}
void test(float n=4.1,int x=2){
    cout <<"Fontion N°2 : ";
    cout << "n= "<<n<<" x= "<<x<<"\n";
}
int main()
{
    int i=5;float r=3.2;
    test(i,r);//Fontion N°1
    test(r,i);//Fontion N°2
    test(i);//Fontion N°1
    test(r);//Fontion N°2      You, now
    return 0;
}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS

```
[Running] cd "/media/ettounani/D/GitHub/tp-ma
Fontion N°1 : n= 5 x= 3.2
Fontion N°2 : n= 3.2 x= 5
Fontion N°1 : n= 5 x= 2.5
Fontion N°2 : n= 3.2 x= 2
```

1

on conclure que le compilateur choisissait la fonction qui va être exécuté selon les types des arguments

Exercice 6:

```cpp
#include <iostream>
#include <stdio.h>
using namespace std;
void essai(float x, char c, int n = 0)
{
    cout << "Fontion N°1 : x= " << x << "c = " << c << "n = " << n << "\n";
}

void essai(float x, int n)
{
    cout << "Fontion N:°2 : x = " << x << "n = " << n << "\n";
}
int main()
{       You, 2 hours ago • Update ex6.cpp …
    char l = 'z';
    int u = 4;
    float y = 2.0;
    essai(y, l, u); // fonction N°1
    essai(y, l);    // fonction N°1
    essai(y, u);    // fonction N°2
    essai(u, u);    // fonction N°2
    essai(u, l);    // fonction N°1
    // essai(y,y); rejet par le compilateur
    essai(y, y, u); // fonction N°1
    // getch();
    return 0;
}
```

```
Fontion N°1 : x= 2c = zn = 4
Fontion N°1 : x= 2c = zn = 0
Fontion N:°2 : x = 2n = 4
Fontion N:°2 : x = 4n = 4
Fontion N°1 : x= 4c = zn = 0
Fontion N°1 : x= 2c = n = 4
```

on conclure que le compilateur choisissait la fonction qui va être exécuté selon les types des arguments .

Mais si il ne trouve pas une fonction avec les types des agruments correspondant à les types qui dans la déffinition

Il fait un castion (le cas de float et int).

Exercice 7:

```cpp
#include <iostream>
#include <stdio.h>
using namespace std;
void affiche(float x, int n = 0)
{

    if (n == 0)
    {
        cout << "result = 1";
        return;      You, 2 hours ago
    }
    int i = 0;
    float result = 1;
    while (i < n)
    {
        result = result * x;
        i++;
    }
    cout << "result = " << result;
}
```

```cpp
void affiche(int n, float x = 0)
{

    if (x == 0)
    {
        cout << "result = 0";
        return;
    }
    int i = 0;
    float result = 1;
    while (i < n)
    {
        result = result * x;
        i++;
    }
    cout << "result = " << result;
}
```

```cpp
int main()
{
    int n = 0;
    float x = 4.0;
    affiche(x, n);
    affiche(n, x);
    n = 2;
    x = 4.3;
    affiche(x, n);
    affiche(n, x);
    // getch();
    return 0;
}
```

```
result = 1
result = 1
result = 18.49
result = 18.49
```

Exercice 8:

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
void echange(int a, int b)
{      You, 2 hours ago • add section 1 in ex8 …
    int tampon;
    tampon = b;
    b = a;
    a = tampon;
    cout << "Pendant l'échange : a= " << a << "b= " << b << "\n";
}

int main()
{

    int u = 5, v = 3;
    cout << "Avant échange : u= " << u << "v = " << v << "\n";
    echange(u, v);
    cout << "Aprés échange : u= " << u << "v = " << v << "\n";
    // getch();
    return 0;
}
```

⇒ ⇒

```
Avant échange : u= 5v = 3
Pendant l'échange : a= 3b= 5
Aprés échange : u= 5v = 3
```

on constate que les valeurs de a et b ne sent pas échanger

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
void echange(int *a, int *b)
{
    int tampon;
    tampon = *b;
    *b = *a;
    *a = tampon;
    cout << "Pendant l'échange : a= " << *a << "b= " << *b << "\n";
}

int main()
{

    int u = 5, v = 3;
    cout << "Avant échange : u= " << u << "v = " << v << "\n";
    echange(&u, &v);
    cout << "Aprés échange : u= " << u << "v = " << v << "\n";
    // getch();
    return 0;
}      You, 2 hours ago • add section 2 in ex8 …
```

⇒ ⇒

```
Avant échange : u= 5v = 3
Pendant l'échange : a= 3b= 5
Aprés échange : u= 3v = 5
```

on constate que les valeurs de a et b sent échanger

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
void echange(int &a, int &b)
{
    int tampon;
    tampon = b;
    b = a;
    a = tampon;
    cout << "Pendant l'échange : a= " << a << "b= " << b << "\n";
}

int main()
{

    int u = 5, v = 3;
    cout << "Avant échange : u= " << u << "v = " << v << "\n";
    echange(u, v);
    cout << "Aprés échange : u= " << u << "v = " << v << "\n";
    // getch();      You, 2 hours ago • add section 3 in ex8 …
    return 0;
}
```

⇒ ⇒

```
Avant échange : u= 5v = 3
Pendant l'échange : a= 3b= 5
Aprés échange : u= 3v = 5
```

on constate que les valeurs de a et b sent échanger

on conclure qu'il y a deux méthodes pour échanger les valeurs des vareables soient par passage par adresse ou par référence

Exercice 9:

```cpp
#include <iostream>
//#include <conio.h>
using namespace std;
You, 12 seconds ago | 1 author (You)
struct essai{
    int n;
    float x;
};
void remize_a_zero(struct essai &test){
    test.n = 0;
    test.x = 0;
}
void remize_a_zero(struct essai *test){
    test->n = 0;
    test->x = 0;
}
void afficher(struct essai test){
    cout << "n= " << test.n << " x = " << test.x;
}
int main(){
    struct essai test;
    remize_a_zero(test);
    afficher(test);
    remize_a_zero(&test);
    afficher(test);
    return 0;
}       You, 3 hours ago • Update ex9.cpp …
```

II- les exemples de cour

exemple 1 :

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

int main()
{
    cout << "Langage C++ ";
    return 0;
}
```

exemple 2 :

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

int main()
{
    float PI = 3.14;
    cout << "La valeur de PI est : ";
    cout << PI;
    return 0;
}
```

exemple 3 :

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

int main()
{
    int N;

    cout << "Entrer un nombre entier : ";
    cin >> N;
    cout << "le carre du nombre entier est :" << N * N;
    return 0;
}
```

exemple 4 :

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

int main()
{
    char c = 'm', d = 25, e;
    int i = 42, j;
    float r = 678.9, s;
    j = c;
    cout << j << "\n"; // j vaut 109
    j = r;
    cout << j << "\n"; // j vaut 687
    s = d;
    cout << s << "\n"; // s vaut 25.0
    e = i;
    cout << e << "\n"; // e vaut *
    return 0;
}
```

exemple 5 :

```cpp
#include <iostream>
int i = 11;
using namespace std;

void main()
{
    int i = 34;
    {
        int i = 23;
        ::i = ::i + 1;
        cout << ::i << " " << i << endl;
    }
    cout << ::i << " " << i << endl;
}
```

exemple 6 :

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

void f1(int n = 3)
{
    cout << n;
}
void f2(int n, float x = 2.35)
{
    cout << n << "   " << x;
}
void f3(char c, int n = 3, float x = 2.35)
{
    cout << n << "   " << x << " " << c;
}

void main()
{
    char a = 0;
    int i = 2;
    float r = 5.6;
    f1(i);
    f1();
    f2(i, r);
    f2(i);
    f3(a, i, r);
    f3(a, i);
    f3(a);
}
```

exemple 7 :

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

int somme(int n1, int n2)
{
    return n1 + n2;
}
int somme(int n1, int n2, int n3)
{
    return n1 + n2 + n3;
}
double somme(double n1, double n2)
{
    return n1 + n2;
}

void main()
{
    cout << "1+2 = " << somme(1, 2) << endl;
    cout << "1+2+3 = " << somme(1, 2, 3) << endl;
    cout << "1.2+2.3 = " << somme(1.2, 2.3) << endl;
}
```

exemple 8 :

```cpp
#include <iostream>
using namespace std;
struct complexe{
    double reel, im;
};
void affiche(int);
void affiche(double);
void affiche(complexe);
int main(void){
    int a = 5;
    double d = 0.0;
    complexe c = {1.0, -1.0};
    affiche(a);
    affiche(d);
    affiche(c);
}
void affiche(int i){
    cout << "type de variable (int) " << endl;
    cout << "valeur : " << i << endl;
}
void affiche(double d){
    cout << "type de variable (double ) " << endl;
    cout << "valeur : " << d << endl;
}
void affiche(complexe c){
    cout << "type de variable (complexe ) " << endl;
    cout << "valeur : " << c.reel << endl;
    cout << "valeur : " << c.im << endl;
}
```