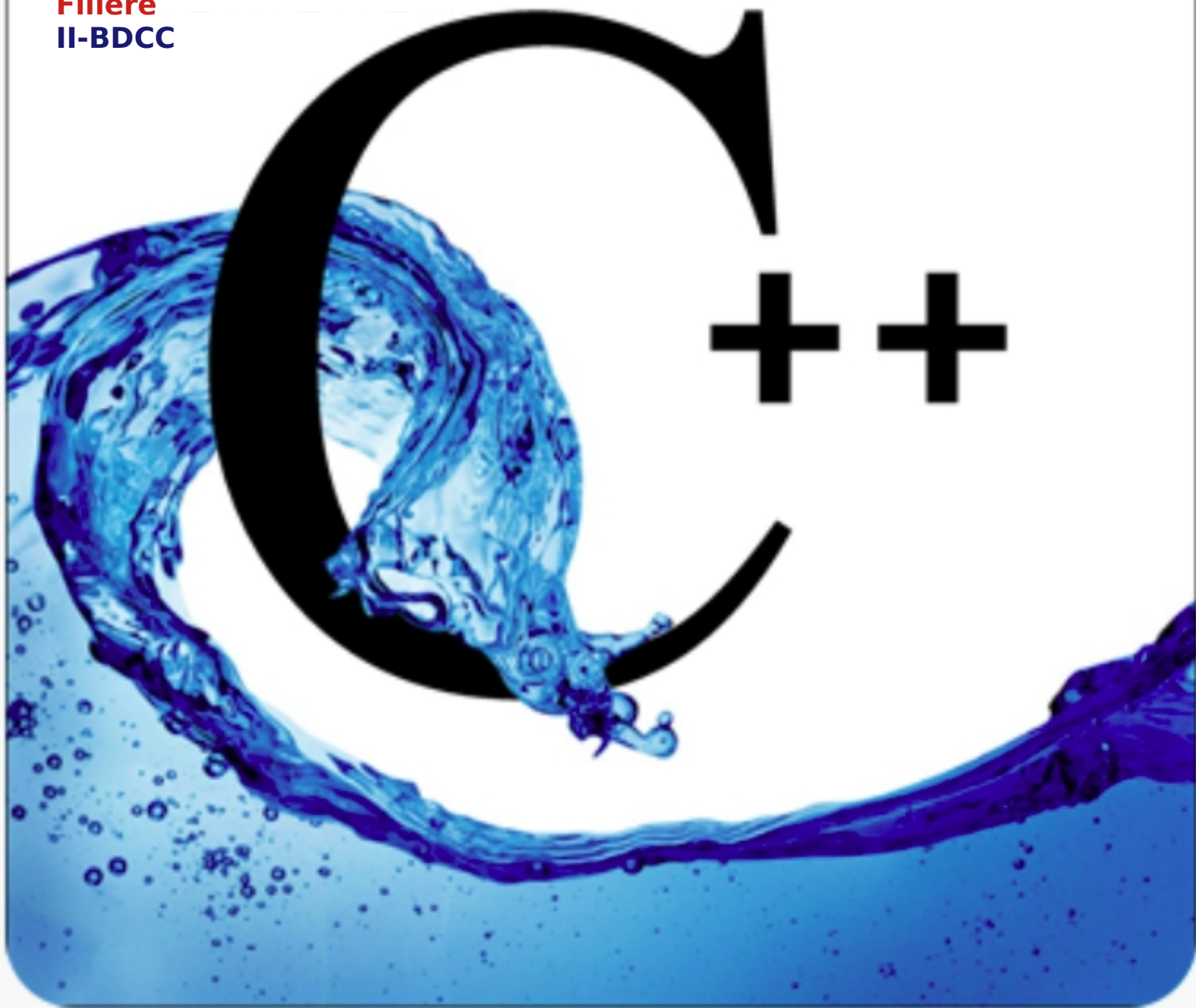


COMPTE RENDU DE CHAPITRE 4 + TPS

Réalisé par :
Abderrahmane ETTOUNANI
Filière
II-BDCC

Encadré par :
Monsieur K.MANSOURI



les exemples de cours & tps

```
Exemple 01.cpp > ...
1  #include <iostream>
2  using namespace std;
3  class point
4  {
5      int x,y;
6  public:
7      point(int,int);
8      ~point();
9  };
10 point :: point(int abs,int ord)
11 {
12     x=abs;y=ord;
13     cout <<"Construction du point " <<x<<" " <<y<<"\n";
14 }
15 point::~~point()
16 {
17     cout<<"destruction du point " <<x<<" " <<y<<"\n";
18 }
19 void test()
20 {
21     cout <<"Debut de test() \n";
22     point u(3,7);
23     cout<<"Fin de test()\n";
24 }
25 int main()
26 {
27     cout<<"Debut de test()\n";
28     point a(1,4);
29     test();
30     point b(5,10);
31     for(int i=0;i<3;i++)
32     {
33         cout <<"Boucle tour numero " <<i<<"\n";
34         point(7+i,12+i);
35     }
36     cout<<"Fin de Main() \n";
37     return 0;
38 }
```

```
Debut de test()
Construction du point 1  4
Debut de test()
Construction du point 3  7
Fin de test()
destruction du point 3  7
Construction du point 5  10
Boucle tour numero 0
Construction du point 7  12
destruction du point 7  12
Boucle tour numero 1
Construction du point 8  13
destruction du point 8  13
Boucle tour numero 2
Construction du point 9  14
destruction du point 9  14
Fin de Main()
destruction du point 5  10
destruction du point 1  4
```

```

1  #include <iostream>
2  using namespace std;
3  class point
4  {
5      int x,y;
6  public:
7      point(int,int);
8      ~point();
9  };
10 point :: point(int abs,int ord)
11 {
12     x=abs;y=ord;
13     cout <<"Construction du point " <<x<<" " <<y<<"\n";
14 }
15 point::~~point()
16 {
17     cout<<"destruction du point " <<x<<" " <<y<<"\n";
18 }
19 point a(1,4);
20 int main()
21 {
22     cout<<"Debut de main()\n";
23     point b(5,10);
24     cout<<"Fin de Main() \n";
25     return 0;
26 }
27
28

```

```

Construction du point 1 4
Debut de main()
Construction du point 5 10
Fin de Main()
destruction du point 5 10
destruction du point 1 4

```

```

1  #include <iostream>
2  using namespace std;
3  class point
4  {
5      int x,y;
6  public:
7      point(int,int);
8      ~point();
9  };
10 point :: point(int abs,int ord)
11 {
12     x=abs;y=ord;
13     cout <<"Construction du point " <<x<<" " <<y<<"a l'adresse " <<this<<"\n";
14 }
15 point::~~point()
16 {
17     cout<<"destruction du point " <<x<<" " <<y<<"a l'adresse : " <<this <<"\n";
18 }
19 int main()
20 {
21     cout<<"Debut de main()\n";
22     point a(0,0);
23     a=point(1,2);
24     a=point(3,5);
25     cout<<"Fin de Main() \n";
26     return 0;
27 }
28
29

```

```

Debut de main()
Construction du point 0 0a l'adresse 0x7ffedf0ee140
Construction du point 1 2a l'adresse 0x7ffedf0ee148
destruction du point 1 2a l'adresse : 0x7ffedf0ee148
Construction du point 3 5a l'adresse 0x7ffedf0ee150
destruction du point 3 5a l'adresse : 0x7ffedf0ee150
Fin de Main()
destruction du point 3 5a l'adresse : 0x7ffedf0ee140

```

```

#include <iostream>
using namespace std;
class point
{
public:
    int x,y;
    point(int,int);
    ~point();
};
point :: point(int abs,int ord)
{
    x=abs;y=ord;
    cout <<"Construction du point " <<x<<" " <<y<<"\n";
}
point::~~point()
{
    cout<<"destruction du point " <<x<<" " <<y<<"\n";
}
int main()
{
    void fct(point *);
    point *adr;
    cout<<"Debut de main()\n";
    adr=new point(3,7);
    fct(adr);
    cout<<adr->x<<" . " <<adr->y;
    delete adr;
    cout<<"Fin de Main() \n";
    return 0;
}
void fct(point *adp)
{
    cout <<"Debut de la fonction \n";
    adp->x=4;
    adp->y=6;
    cout<<adp->x<<adp->y<<"Fin de la fonction \n";
}

```

```

Debut de main()
Construction du point 3 7
Debut de la fonction
46Fin de la fonction
4 . 6destruction du point 4 6
Fin de Main()

```

```

1  #include <iostream>
2  using namespace std;
3  class point
4  {
5
6  public:
7      int x,y;
8      point(int,int);
9      ~point();
10 };
11 point :: point(int abs,int ord)
12 {
13     x=abs;y=ord;
14     cout <<"Construction du point " <<x<<" " <<y<<"\n";
15     cout<<"son adresse est:"<<this<<"\n";
16 }
17 point::~~point()
18 {
19     cout<<"destruction du point " <<x<<" " <<y<<"\n";
20     cout<<"son adresse est:"<<this<<"\n";
21 }
22 int main()
23 {
24     cout<<"Debut de main()\n";
25     point a(3,7);
26     point b=a;
27     cout<<"Fin de Main() \n";
28     return 0;
29 }
30

```

```

Debut de main()
Construction du point 3 7
son adresse est:0x7ffe421aeab8
Fin de Main()
destruction du point 3 7
son adresse est:0x7ffe421aeac0
destruction du point 3 7
son adresse est:0x7ffe421aeab8

```

```
#include <iostream>
```

```
using namespace std;
```

```
class liste
```

```
{
```

```
int taille;
```

```
float *adr;
```

```
public:
```

```
    liste(int);
```

```
    ~liste();
```

```
};
```

```
liste :: liste(int t)
```

```
{
```

```
    taille=t;
```

```
    adr=new float[taille];
```

```
    cout <<"Construction " << "adresse de l'objet: " <<this<<"\n";
```

```
    cout <<"Adresse de la liste : " <<adr<<"\n";}
```

```
liste::~~liste()
```

```
{
```

```
    cout<<"destruction de l'objet avec l'adresse \n"<<this<<"\n";
```

```
    cout<<"l'adresse de la liste est:"<<adr<<"\n";
```

```
    delete adr;
```

```
}
```

```
int main()
```

```
{
```

```
    cout<<"Debut de main()\n";
```

```
    liste a(3);
```

```
    liste b=a;
```

```
    cout<<"Fin de Main() \n";
```

```
    return 0;
```

```
}
```

```
Debut de main()
```

```
Construction adresse de l'objet: 0x7fff04760e60
```

```
Adresse de la liste : 0x555a3fae8ec0
```

```
Fin de Main()
```

```
destruction de l'objet avec l'adresse
```

```
0x7fff04760e70
```

```
l'adresse de la liste est:0x555a3fae8ec0
```

```
destruction de l'objet avec l'adresse
```

```
0x7fff04760e60
```

```
l'adresse de la liste est:0x555a3fae8ec0
```



```

1  #include <iostream>
2  using namespace std;
3  class liste
4  {
5  int taille;
6  float *adr;
7  public:
8      liste(int);
9      liste(liste &);
10     ~liste();
11 };
12 liste::liste(liste &v)
13 {
14     taille=v.taille;
15     adr=new float[taille];
16     for(int i=0;i<taille;i++)
17     {
18         adr[i]=v.adr[i];
19     }
20     cout<<" \n Constructeur par recopie";
21     cout<<" Adresse de l'objet : "<<this;
22     cout<<" Adresse de la liste : "<<adr<<"\n";
23 }
24 liste :: liste(int t)
25 {
26     taille=t;
27     adr=new float[taille];
28     cout <<"Construction " << "adresse de l'objet: "<<this<<"\n";
29     cout <<"Adresse de la liste : "<<adr<<"\n";}
30 liste::~~liste()
31 {
32     cout<<"destruction de l'objet avec l'adresse \n"<<this<<"\n";
33     cout<<"l'adresse de la liste est:"<<adr<<"\n";
34     delete adr;
35 }
36 int main()
37 {
38     cout<<"Debut de main()\n";
39     liste a(3);
40     liste b=a;
41     cout<<"Fin de Main() \n";
42     return 0;
43 }
44

```

```

Debut de main()
Construction adresse de l'objet: 0x7ffec356cac0
Adresse de la liste : 0x560fa6051ec0

Constructeur par recopie Adresse de l'objet : 0x7ffec356cad0 Adresse de la liste : 0x560fa6051ee0
Fin de Main()
destruction de l'objet avec l'adresse
0x7ffec356cad0
l'adresse de la liste est:0x560fa6051ee0
destruction de l'objet avec l'adresse
0x7ffec356cac0
l'adresse de la liste est:0x560fa6051ec0

```



```

3  class point
4  {
5  int x,y;
6  public:
7      point(int,int);
8      point(point &);
9      ~point();
10     point symetrique();
11     void affiche(){cout<<"x="<<x<<" y="<<y<<"\n";};
12 };
13 point::point(point &pt)
14 {
15     x=pt.x;y=pt.y;
16     cout<<" \n Constructeur par recopie";
17     cout<<" construction du point : "<<x<<" "<<y;
18     cout<<" Adresse de l'objet : "<<this<<"\n";
19 }
20 point :: point(int abs=0,int ord=0)
21 {
22     x=abs;y=ord;
23     cout<<" construction du point : "<<x<<" "<<y;
24     cout<<" Adresse de l'objet : "<<this<<"\n";
25 }
26 point ::~point()
27 {
28     cout<<"destruction du point "<<x<<" "<<y;
29     cout<<"son adresse est "<<this<<"\n";
30 }
31 point point::symetrique()
32 {
33     point res;
34     cout <<"*****\n";
35     res.x=-x;
36     res.y=-y;
37     cout<<"#####\n";
38     return res;
39 }
40 int main()
41 {
42     cout<<"Debut de main()\n";
43     point a(1,4),b;
44     cout<<"Avant Appel a Symetrique\n";
45     b=a.symetrique();
46     b.affiche();
47     cout<<"Apres appel a symetrique et Fin de Main() \n";
48     return 0;
49 }

```

```

Debut de main()
| construction du point : 1 4 Adresse de l'objet : 0x7ffd41fa60e0
| construction du point : 0 0 Adresse de l'objet : 0x7ffd41fa60e8
Avant Appel a Symetrique
| construction du point : 0 0 Adresse de l'objet : 0x7ffd41fa60f0
*****
#####
destruction du point -1 -4son adresse est 0x7ffd41fa60f0
x=-1 y=-4
Apres appel a symetrique et Fin de Main()
destruction du point -1 -4son adresse est 0x7ffd41fa60e8
destruction du point 1 4son adresse est 0x7ffd41fa60e0

```

```

#include <iostream>
using namespace std;
class liste
{
int taille;
float *adr;
public:
    liste(int);
    liste(liste &);
    void saisie();
    void affiche();
    liste oppose();
    ~liste();
};

liste::liste(liste &v)
{
    taille=v.taille;
    adr=new float[taille];
    for(int i=0;i<taille;i++)
    {
        adr[i]=v.adr[i];
    }
    cout<<" \n Constructeur par recopie";
    cout<<" Adresse de l'objet : "<<this<<"\n";
    cout<<" Adresse de la liste : "<<adr<<"\n";
}

liste :: liste(int t)
{
    taille=t;
    adr=new float[taille];
    cout <<"Construction " << "adresse de l'objet: "<<this<<"\n";
    cout <<"Adresse de la liste : "<<adr<<"\n";}

liste::~~liste()
{
    cout<<"destruction de l'objet avec l'adresse \n"<<this<<"\n";
    cout<<"l'adresse de la liste est:"<<adr<<"\n";
    delete adr;
}

void liste::saisie()
{
    int i;
    for(i=0;i<taille;i++)
    {
        cout<<"Entrer un Nombre : ";
        cin>>*(adr+i);
    }
}

void liste::affiche()
{
    int i;
    for(i=0;i<taille;i++)
    {
        cout<<*(adr+i)<<" ";
    }
    cout<<"\n Adresse de l'objet : "<<this<<" Adresse de liste : "<<adr<<"\n";
}

liste liste::oppose()
{
    liste res(taille);
    for(int i=0;i<taille;i++)
    {
        res.adr[i]=-adr[i];
    }
    for(int i=0;i<taille;i++)
    {
        cout<<res.adr[i]<<" ";
    }
    cout <<"\n";
    return res;
}

int main()
{
    cout<<"Debut de main()\n";
    liste a(3),b(3);
    a.saisie();
    a.affiche();
    b=a.oppose();
    b.affiche();
    cout<<"Fin de Main() \n";
    return 0;
}

```



```

Debut de main()
Construction adresse de l'objet: 0x7ffd17d16290
Adresse de la liste : 0x555f09e9b2c0
Construction adresse de l'objet: 0x7ffd17d162a0
Adresse de la liste : 0x555f09e9b2e0
Entrer un Nombre : 2
Entrer un Nombre : 4
Entrer un Nombre : 5
2 4 5
  Adresse de l'objet : 0x7ffd17d16290 Adresse de liste : 0x555f09e9b2c0
Construction adresse de l'objet: 0x7ffd17d162b0
Adresse de la liste : 0x555f09e9b760
-2 -4 -5
destruction de l'objet avec l'adresse
0x7ffd17d162b0
l'adresse de la liste est:0x555f09e9b760
0 0 5.59874e-33
  Adresse de l'objet : 0x7ffd17d162a0 Adresse de liste : 0x555f09e9b760
Fin de Main()
destruction de l'objet avec l'adresse
0x7ffd17d162a0
l'adresse de la liste est:0x555f09e9b760

```

```

#include <iostream>
using namespace std;

class point
{
    int x,y;
public:
    point(int abs=0,int ord=0)
    {
        x=abs;
        y=ord;
        cout<<"Constructeur point "<<x<<" "<<y<<"\n";
    }
};

class poincol
{
    point p;
    int couleur;
public:
    poincol(int,int,int);
};

poincol :: poincol(int abs,int ord,int coul) :p(abs,ord)
{
    couleur =coul;
    cout <<"Constructeur poincol "<<couleur<<"\n";
}

int main()
{
    poincol a(1,3,9);
    return 0;
}

```

```

Constructeur point 1 3
Constructeur poincol 9

```

-----partie 1 :-----

```
3  class note
4  {
5      float value;
6  public :
7      void input();
8      void print();
9      void set (float);
10     float get();
11     char* apprecier();
12 };
13 void note :: input ()
14 {
15     cout << "saisissez la note : ";
16     cin >> value;
17     if (value > 20 || value <0)
18     {
19         cout << "la note ne doit pas etre >20 ou <0: ";
20         note :: input ();
21     }
22 }
23 void note :: print ()
24 {
25     cout << "la note est : " << value;
26 }
27 void note :: set(float v)
28 {
29     value = v;
30 }
31 float note :: get ()
32 {
33     return value;
34 }
35 char* note :: apprecier ()
36 {
37     if (value >= 16)
38         return "tres bien";
39     else if (value <= 15 && value >=12)
40         return "bien";
41     else if (value <12 && value >=10)
42         return "passable";
43     else return "mauvaise note";
44 }
45 main()
46 {
47     note a;
48     a.input();
49     char* chaine = a.apprecier();
```

```
(base) ettounani@tounani
saisissez la note : 10
```

```
passable
(base) ettounani@tounani
saisissez la note : 8
```

```
mauvaise note
(base) ettounani@tounani
saisissez la note : 18
```

```
tres bien
(base) ettounani@tounani
```

-----partie 2-----

rtie 02.cpp > harmonise(note &)

```
#include <iostream>
using namespace std;
class note
{
    float value;
public :
    note (float v)
    {
        value = v;
    }
    void input();
    void print();
    void set (float);
    float get();
    char* apprecier();
    void harmonise(note&);
    void moyenne(note *, int);
    char* apprecier(note*, int);
};

void note :: input (){
    cout << "saisissez la note : ";
    cin >> value;
    if (value > 20 || value<0)
    {
        cout << "la note ne doit pas etre >20 ou <0 : ";
        note :: input ();
    }
}

void note :: print (){
    cout << "la note est : " << value;
}

void note :: set(float v){
    value = v;
}

float note :: get (){
    return value;
}

char* note :: apprecier (){
    if (value >= 16)
        return "tres bien";
    else if (value <= 15 && value >=12)
        return "bien";
    else if (value <12 && value >=10)
        return "passable";
    else return "mauvaise note";
}
```

```

void harmonise(note &a){
    if (a.get() < 8)
        a.set(0);
    else a.set(8);
}

float moyenne(note *t, int n)
{
    float somme=0;
    for (int i = 0; i < n; i++)
        somme += t[i].get();
    return (somme/n);
}

void apprecier(note *t, int n)
{
    for (int i = 0; i < n; i++)
    {
        cout <<"\n" << t[i].get() <<" .. " << t[i].apprecier() <<"\n";
    }
}

main()
{
    note T[4] = {13, 10, 4, 12, 19};
    cout <<"\n" << moyenne(T, 4)<<"\n";
    apprecier(T, 4);
}

```

9.75

13 .. bien

10 .. passable

4 .. mauvaise note

12 .. bien

-----partie 3-----

```
1  #include <iostream>
2  using namespace std;
3  class note
4  {
5      float value;
6
7  public :
8
9      note (float v)
10     {
11         value = v;
12     }
13     void input();
14     void print();
15     void set (float);
16     float get();
17     char* apprecier();
18     void harmonise(note&);
19     void moyenne(note *, int);
20     char* apprecier(note*, int);
21
22 };
23
24
25 void note :: input ()
26 {
27     cout << "saisissez la note : ";
28     cin >> value;
29     if (value > 20 || value<0)
30     {
31         cout << "la note ne doit pas etre >20 ou <0 : ";
32         note :: input ();
33     }
34 }
35
36 void note :: print ()
37 {
38     cout << "la note est : " << value;
39 }
40
41 void note :: set(float v)
42 {
43     value = v;
44 }
45
```

```

float note :: get ()
{
    return value;
}

char* note :: apprecier ()
{
    if (value >= 16)
        return "tres bien";
    else if (value <= 15 && value >=12)
        return "bien";
    else if (value <12 && value >=10)
        return "passable";
    else return "mauvaise note";
}

void harmonise(note &a)
{
    if (a.get() < 8)
        a.set(0);
    else a.set(8);
}

float moyenne(note *t, int n)
{
    float somme=0;
    for (int i = 0; i < n; i++)
        somme += t[i].get();
    return (somme/n);
}

void apprecier(note *t, int n)
{
    for (int i = 0; i < n; i++)
    {
        cout <<"\n" << t[i].get() <<" .. " << t[i].apprecier() <<"\n";
    }
}

```



```

int main()
{
    int n;
    float v;
    note *T;
    cout << " donnez le nbr d'eleves : ";
    cin >> n;
    T = new note[n];

    for(int i = 0; i < n; i++)
    {
        cout << "\n Donner note : ";
        cin >> v;
        T[i].set(v);
    }
    apprecier(T, n);
    cout << "\n moyenne des notes avant Harmonise " << moyenne(T, n) << "\n";

    for (int i = 0; i < n; i++)
    {
        if (T[i].get() < 15)
            harmonise(T[i]);
    }
    cout << "\n moyenne des notes apres Harmonise " << moyenne(T, n) << "\n";
}

```

```

    donnez le nbr d'eleves : 4

Donner note : 13
Donner note : 5
Donner note : 15
Donner note : 20

13 .. bien
5 .. mauvaise note
15 .. bien
20 .. tres bien

moyenne des notes avant Harmonise 13.25
moyenne des notes apres Harmonise 10.75

```