Lab-10

Title: Type conversion

Objective:

• To be familiar with type conversion

Theory:

Introduction to type conversion

- **Basic to class type**
- **Class to basic type**
- **Class to class type With their example and syntax**

1. Write a program to read height of person in meter and convert into feet and inches by using suitable type conversion method.

```
#include<iostream>
using namespace std;
class Height{
        private:
                int feet;
                float inches;
        public:
                Height()
                {
                }
                Height(float m)
                {
           float f=3.28083*m;
           feet=int(f);
           inches=12*(f-feet);
          }
          void display()
          {
                cout<<feet<<"Feet and "<<inches<<"inches"<<endl;
                }
};
int main(){
        float meter;
        Height h1;
        cout<<"Enter the height in meter"<<endl;
        cin>>meter;
        h1=meter;
        cout<<"Height in feet and inches"<<endl;
        h1.display();
        return 0;
}
```

2. Write a program to read height of person in feet and inches and convert into meter by using suitable type conversion method.

```
#include<iostream>
using namespace std;
class Height{
        private:
                int feet;
                float inches;
        public:
                Height(int f,float i){
                        feet=f;
                        inches=i;
                }
                void display(){
                        cout<<feet<<"feets and "<<inches<<" inches"<<endl;
                }
                operator float(){
                        float f=inches/12;
                        f=f+feet;
                        return(f/3.28083);
                }
};
int main(){
        Height h1(5,0.11);
        float m;
        m=h1;
        h1.display();
        cout<<"height of person in meter:"<<m<<endl;
        return 0;
}
```

3. Make a class called memory with member data to represent bytes, kilobytes and megabytes .Read the value of memory in bytes from the user as basic types and display the result in user defined memory type. Like for m (basic type) = 108766, your program should display as: 1 megabyte 38 kilobytes 177 bytes. [Hint: Use basic to user defined (basic-to-class) conversion method].

```
#include<iostream>
using namespace std;
class Memory{
        private:
                int mb;
                int kb;
                int byte;
        public:
                Memory()
                {
                }
                Memory(int m)
                {
                 int rem;
           mb=m/(1024*1024);
           rem=m%(1024*1024);
           kb=rem/1024;
           byte=rem%1024;
          }
          void display()
          {
                cout<<mb<<" Megabyte"<<endl;
                cout<<kb<<" Kilobyte"<<endl;
                cout<<byte<<" Bytes"<<endl;
                }
};
int main(){
        int m;
        Memory m1;
        cout<<"Enter the memory in bytes"<<endl;
        cin>>m;
        m1=m;
        m1.display();
        return 0;
}
```

4. Write a program to convert total memory in bytes into mb, kb and bytes using type conversion. (1kb=1024bytes,1mb=1024kb)
(Remaining)

5. Write a complete program to convert the polar coordinates into rectangular coordinates.(hint: polar-coordinates(radius, angle) and rectangular co-ordinates (x,y) where x=r*cos(angle) and y=r*sin(angle))

```
#include<iostream>
#include<math.h>
using namespace std;
class Rectangle{
        private:
                int x;
                int y;
        public:
                Rectangle(){
                        x=0.0;
                        y=0.0;
                }
                Rectangle(int a,int b){
                        x=a;
                        y=b;
                }
                void display(){
                        cout<<"("<<x<<","<<y<<")"<<endl;
                }
};
class Polar{
        private:
                float radius;
                float angle;
        public:
                Polar(){
                        radius=0.0;
                        angle=0.0;
                }
                Polar(float r,float a){
                        radius=r;
                        angle=a;
                }
                void display(){
                        cout<<"("<<radius<<","<<angle<<")"<<endl;
                }
                operator Rectangle(){
                float x=radius*sin(angle);
                float y=radius*cos(angle);
                return Rectangle(x,y);
                }
};
int main(){
        Polar p1(11.25,7.54);
        Rectangle r1;
        r1=p1;
        cout<<"Polar coordinate:";
        p1.display();
        cout<<"Rectangular coordinate:";
        r1.display();
        return 0;
}
```

6. Write a program to convert Rectangle coordinate to Polar coordinate by using:

- conversion routine in source class

```
#include<iostream>
#include<math.h>
using namespace std;
class Polar{
        private:
                float radius;
                float angle;
        public:
                Polar(){
                        radius=0.0;
                        angle=0.0;
                }
                Polar(float r,float a){
                        radius=r;
                        angle=a;
                }
                void display(){
                        cout<<"("<<radius<<","<<angle<<")"<<endl;
                }
};
class Rectangle{
        private:
                int x;
                int y;
        public:
                Rectangle(){
                        x=0.0;
                        y=0.0;
                }
                Rectangle(int a,int b){
                        x=a;
                        y=b;
                }
                void display(){
                        cout<<"("<<x<<","<<y<<")"<<endl;
                }
                operator Polar(){
                float r=sqrt(x*x+y*y);
                float a=atan(y/x);
                return Polar(r,a);
                }
};
int main(){
        Rectangle r1(4,6);
        Polar p1;
        p1=r1;
        cout<<"Rectangular coordinate:";
        r1.display();
        cout<<"Polar coordinate:";
        p1.display();
        return 0;
}
```

- conversion routine in destination class

```
#include<iostream>
#include<math.h>
using namespace std;
class Rectangle{
        private:
                int x;
                int y;
        public:
                Rectangle(){
                        x=0.0;
                        y=0.0;
                }
                Rectangle(int a,int b){
                        x=a;
                        y=b;
                }
                void display(){
                        cout<<"("<<x<<","<<y<<")"<<endl;
                }
                float getx(){
                        return x;
                }
                float gety(){
                        return y;
                }
};
class Polar{
        private:
                float radius;
                float angle;
        public:
                Polar(){
                        radius=0.0;
                        angle=0.0;
                }
                void display(){
                        cout<<"("<<radius<<","<<angle<<")"<<endl;
                }
                Polar(Rectangle r1){
                        float x=r1.getx();
                        float y=r1.gety();
                        radius=sqrt(x*x+y*y);
                        angle=atan(y/x);
                }
};
int main(){
        Rectangle r1(4,6);
        Polar p1;
        p1=r1;
        cout<<"Rectangular coordinate:";
        r1.display();
        cout<<"Polar coordinate:";
        p1.display();
        return 0;
}
```

7. Write a program to convert class Polar to Rectangle and vice versa by using type conversion.

```
#include<iostream>
#include<math.h>
using namespace std;
class Polar;
class Rectangle{
        private:
                int x;
                int y;
        public:
                Rectangle(){
                        x=0.0;
                        y=0.0;
                }
                Rectangle(int a,int b){
                        x=a;
                        y=b;
                }
                void display(){
                        cout<<"("<<x<<","<<y<<")"<<endl;
                }
                operator Polar();
};
```

```
class Polar{
        private:
                float radius;
                float angle;
        public:
                Polar(){
                        radius=0.0;
                        angle=0.0;
                }
                Polar(float r,float a){
                        radius=r;
                        angle=a;
                }
                void display(){
                        cout<<"("<<radius<<","<<angle<<")"<<endl;
                }
                operator Rectangle(){
                float x=radius*cos(angle);
                float y=radius*sin(angle);
                return Rectangle(x,y);
                }

};
Rectangle::operator Polar(){
                        float a=atan(y/x);
                        float r=sqrt(x*x+y*y);
                        return Polar(r,a);
                }

int main(){
        Polar p1(11.25,7.54);
        Rectangle r1;
        r1=p1;
        Rectangle r2(3,10);
        Polar p2;
        p2=r2;
        cout<<"for polar to rectangle:"<<endl;
        cout<<"Polar coordinate:";
        p1.display();
        cout<<"Rectangular coordinate:";
        r1.display();
        cout<<"for rectangle to polar:"<<endl;
        cout<<"Rectangular coordinate:";
        r2.display();
        cout<<"Polar coordinate:";
        p2.display();
        return 0;
}
```

8. Write a program to convert degree Celsius to degree Fahrenheit by using class to class type conversion.

```
#include<iostream>
using namespace std;
class Fahrenheit{
        private:
                float fahr;
        public:
                Fahrenheit(){

                }
                Fahrenheit(float f){
                        fahr=f;
                }
                void display(){
                   cout<<fahr<<" fahrenheit"<<endl;
                }
};
class Degree{
        private:
                float degree;
        public:
                Degree(){

                }
                Degree(float d){
                        degree=d;
                }
                void display(){
                        cout<<degree<<" degree celsius "<<endl;
                }
                operator Fahrenheit(){
                        float f=1.8*degree+32;
                        return Fahrenheit(f);
                }
};
int main(){
        Fahrenheit f1;
        Degree d1(50);
        f1=d1;
        cout<<"Temperature in degree:"<<endl;
        d1.display();
        cout<<"Temperature in fahrenheit:"<<endl;
        f1.display();
        return 0;
}
```