Assignment – 29 A Job Ready Bootcamp in C++, DSA and IOT MySirG

Type Casting and Conversion

1.Write a C++ program to convert Primitive type to Complex type.
Example -

```
int main()
{
Complex c1;
Int x=5;
c1=x;
return 0;
}
```

Program -

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

class Complex
{
    int x;

    public:
        void operator=(int x)
        {
            this->x = x;
        }
        void display()
        {
            cout<<"Complex class x = "<<x<<endl;
        }
};

int main ()
{
    Complex c1;
    int x = 5;
    c1 = x;
    c1.display();
    return 0;
}
```

Output -
Complex class x = 5

---

2. Write a C++ program to convert Complex type to Primitive type.
Example -

```
int main()
{
```

Complex c1;
c1.setData(3,4);
int x;
x=c1;
return 0;
}

Program -

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

class Complex
{
    int real, imag;
    public:
        void setData(int real, int imag)
        {
            this->real = real;
            this->imag = imag;
        }

        operator int()
        {
            return real+imag;
        }
};

int main ()
{
    Complex c1;

    c1.setData(3,4);

    int x;
    x = c1;
    cout<<"x = "<<x<<endl;
    return 0;
}
```

Output -
x = 7

---

3. Create a Product class and convert Product type to Item type using constructor
int main()
{
Item i1;
Product p1;
p1.setData(3,4);
i1=p1;
return 0;

```
}
```

Program -

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

class Product
{
    int p1, p2;

    public:
        void setData(int x, int y)
        {
            p1 = x;
            p2 = y;
        }
        void display()
        {
            cout<<"Product1 = "<<p1<<", "<<"Product2 = "<<p2<<endl;
        }
        int getProd1()
        {
            return p1;
        }
        int getProd2()
        {
            return p2;
        }
};

class Item
{
    int i1, i2;

    public:
        Item() {}
        Item(Product P)
        {
            i1 = P.getProd1();
            i2 = P.getProd2();
        }
        void display()
        {
            cout<<"Item1 = "<<i1<<", "<<"Item2 = "<<i2<<endl;
        }
};

int main ()
{
    Item i1;
    Product p1;
```

```
    p1.setData(3,4);
    i1 = p1;
    i1.display();
    return 0;
}
```

Output -
Item1 = 3, Item2 = 4

---

4. Create Product class and convert Product type to Item type using casting operator

```
int main()
{
Item i1;
Product p1;

p1.setData(3,4);
i1=p1;
return 0;
}
```

Program -

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

class Item
{
    int i1, i2;

    public:
        void setData(int x, int y)
        {
            i1 = x;
            i2 = y;
        }
        void display()
        {
            cout<<"Item1: "<<i1<<", "<<"Item2: "<<i2<<endl;
        }
        int &getItem1()
        {
            return i1;
        }
        int &getItem2()
        {
            return i2;
        }
};
```

```
class Product
{
    int p1, p2;

    public:
        void setData(int x, int y)
        {
            p1 = x;
            p2 = y;
        }
        void display()
        {
            cout<<"Product1: "<<p1<<", "<<"Product2: "<<p2<<endl;
        }
        operator Item()
        {
            Item obj;
            int &val1 = obj.getItem1();
            int &val2 = obj.getItem2();
            val1 = p1;
            val2 = p2;
            return obj;
        }
};

int main ()
{
    Item i1;
    Product p1;
    p1.setData(3,4);
    i1 = p1;
    i1.display();
    return 0;
}
```

Output -
Item1: 3, Item2: 4

---

5. Create two classes Invent1 and Invent2 and also add necessary constructors in it. Now add functions to support Invent1 to float and Invent1 to Invent2 type.
Example -

```
int main()
{
Invent1 s1=4;
Invent2 d1;
float tv;
tv=s1;
d1=s1;
return 0;
}
```

Program -

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

class Invent1
{
    float x;

    public:
        Invent1(){}
        Invent1(float a):x(a)
        {

        }
        operator float()
        {
            return x;
        }
        float getValue()
        {
            return x;
        }
};

class Invent2
{
    int y;
    public:
        Invent2(){}

        Invent2(Invent1 obj)
        {
            y = obj.getValue();
        }
        void display()
        {
            cout<<"Invent2: "<<y<<endl;
        }
};

int main ()
{
    Invent1 s1 = 4;
    Invent2 d1;
    float tv;
    tv = s1;
    d1 = s1;
    cout<<"tv: "<<tv<<endl;
    d1.display();
    return 0;
}
```

Output -
tv: 4
Invent2: 4

---

6. Create a Time class and take Duration in seconds. Now you need to convert seconds(i.e in int) to Time class.

Example-

```
int main()
{
int duration;
cout<<”Enter time duration in minutes”;
cin>>duration;
Time t1 = duration;
t1.display();
return 0;
}
```

Program -

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

class Time
{
    int T;

    public:
        Time(){}

        Time(int t):T(t)
        {

        }

        void display()
        {
            cout<<"Duration: "<<T<<endl;
        }
};

int main ()
{
    int duration;
    cout<<"Enter time duration in minutes: ";
    cin>>duration;
    Time t1 = duration;
    t1.display();
    return 0;
```

```
}
```

Output -
Enter time duration in minutes: 56
Duration: 56

---

7. Create two class Time and Minute and add required getter and setter including constructors. Now you need to type cast Time object into Minute to fetch the minute from Time and display it.
Example -

```
int main()
{
Time t1(2,30);
t1.display();
Minute m1;
m1.display();
m1=t1 // Fetch minute from time
t1.display();
m1.display();
return 0;
}
```

Program -

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

class Minute
{
    int mins;

    public:
        Minute()
        {
            mins = 0;
        }
        Minute(int m)
        {
            mins = m;
        }
        void display()
        {
            cout<<"Minutes: "<<mins<<endl;
        }
};

class Time
{
    int hours, minutes;
```

```
    public:
        Time(){}

        Time(int h, int m): hours(h), minutes(m) {}

        void display()
        {
            cout<<"Time: "<<hours<<" hours, "<<minutes<<"
minutes"<<endl;
        }

        operator Minute()
        {
          return (hours * 60 + minutes);
        }
};

int main()
{
    Time t1(2,30);
    t1.display();
    Minute m1;
    m1 = t1; // Fetch minute from time
    m1.display();
    return 0;
}
```

Output -
Time: 2 hours, 30 minutes
Minutes: 150

---

8. Create a Rupee class and convert it into int. And Display it.

Example-

```
int main()
{
Rupee r = 10;
int x = r;
cout<<x;
return 0;
}
```

Program -

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

class Rupee
{
    int r;
```

```
    public:
        Rupee() {}

        Rupee(int x)
        {
            r = x;
        }

        operator int()
        {
            return r;
        }
        void display()
        {
            cout<<"Rupees = "<<r<<endl;
        }
};

int main()
{
    Rupee r = 10;
    int x = r;
    cout<<x;
    return 0;
}
```

Output -
10

---

9. Create a Dollar class and add necessary functions to support int to Dollar type conversion.

Example-
int main()
{
int x = 50;
Dollar d;
d = x;
d.display();
return 0;
}

Program -

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

class Dollar
{
    int d;
```

```
    public:
        Dollar() {}

        Dollar(int x)
        {
            d = x;
        }

        void display()
        {
            cout<<"Dollar = "<<d<<endl;
        }
};

int main()
{
    int x = 50;

    Dollar d;
    d = x;
    d.display();

    return 0;
}
```

Output -
Dollar = 50

---

10. Create two classes Rupee and Dollar and add necessary functions to support Rupee to Dollar and Dollar to Rupee conversion.

Example-

```
int main()
{
    Rupee r = 23;
    Dollar d = r; // Rupee to Dollar conversion
    d.display();
    r.display();
    r = d; // Dollar to Rupee Conversion
    d.display();
    r.display();
    return 0;
}
```

Program -

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

```
class Dollar
{
    float d;

    public:
        Dollar(){}

        Dollar(float x)
        {
            d = x;
        }

        void display()
        {
            cout<<"Dollar = "<<d<<endl;
        }

        operator float()
        {
            return d;
        }

        float getDollar()
        {
            return d;
        }
};

class Rupee
{
    float r;

    public:
        Rupee(){}

        Rupee(Dollar d)
        {
            r = d.getDollar()*80.55;
        }

        Rupee(float x)
        {
            r = x;
        }

        void display()
        {
            cout<<"Rupee = "<<r<<endl;
        }
```

```
        operator float()
        {
            return r;
        }

        operator Dollar()
        {
            return r/80.55;
        }
};

int main()
{
    Rupee r = 23;
    Dollar d = r; // Rupee to Dollar conversion
    d.display();
    r.display();
    r = d; // Dollar to Rupee Conversion
    d.display();
    r.display();
    return 0;
}
```

Output -
Dollar = 0.285537
Rupee = 23
Dollar = 0.285537
Rupee = 23
