New chat

```
class Triangle: public Shape {
    public:
        Triangle(double w = 0, double h = 0): Shape(w, h) { }

        double area() {
            cout << "Triangle class area :" << endl;
            return (width * height / 2);
        }
};

int main() {
    Shape* shape;
    Rectangle rec(10, 7);
    Triangle tri(10, 5);

    shape = &rec;
    shape->area();

    shape = &tri;
    shape->area();

    return 0;
}
```

In the above code, we have defined a base class "Shape" and two derived classes "Rectangle" and "Triangle". The base class has a virtual function "area", which is overridden in the derived classes. The main function creates objects of the derived classes and assigns their addresses to a pointer of the base class. The virtual function is called through the pointer, which results in the appropriate derived class function being called based on the object pointed to. This is an example of dynamic polymorphism because the decision of which function to call is made at runtime based on the type of the object pointed to.

I hope these examples help you understand polymorphism in C++.