C++ Classes and Objects II

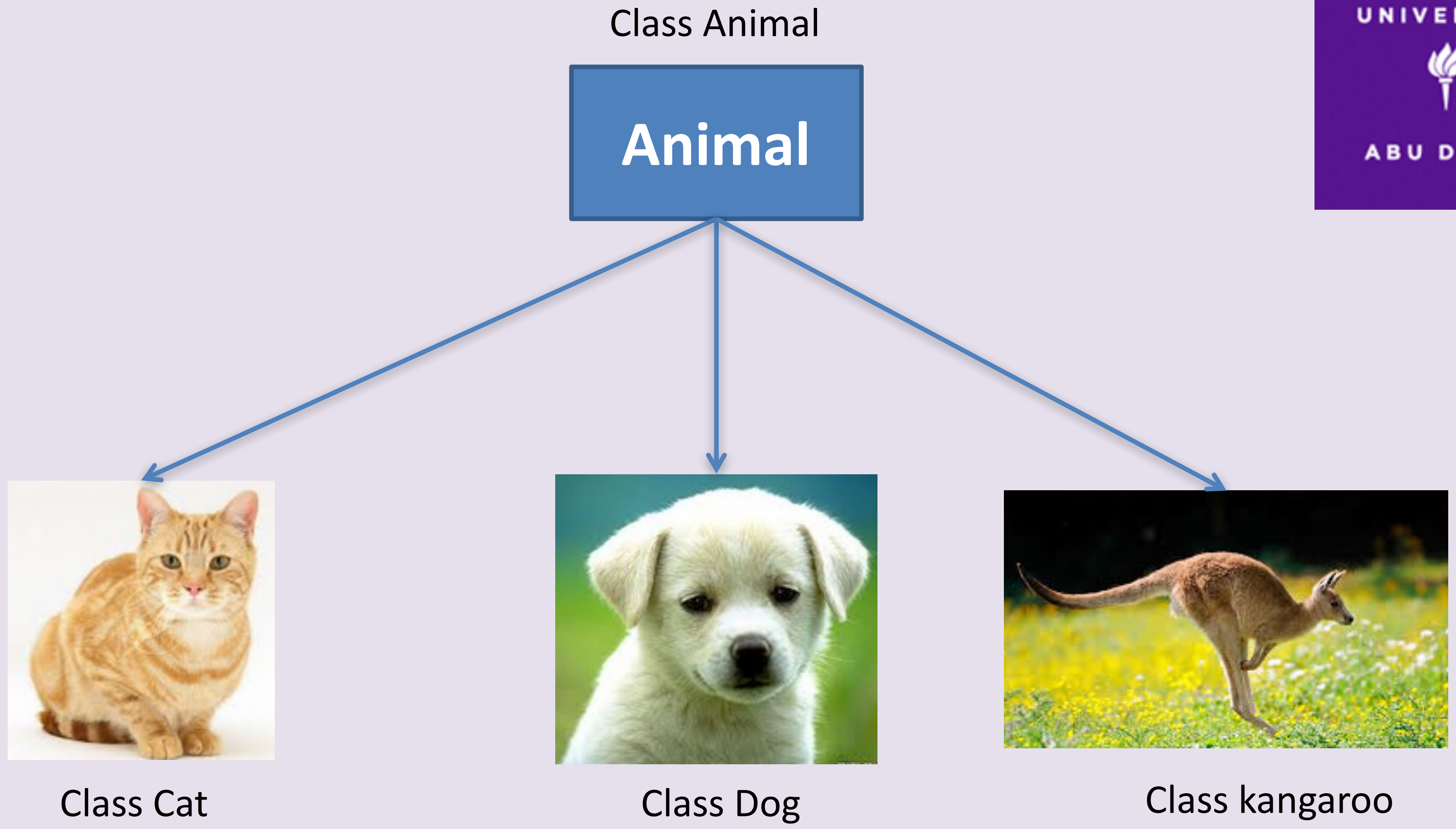
NEW YORK UNIVERSITY ABU DHABI

- C++ Inheritance
- C++ Polymorphism

- C++ Inheritance
- C++ Polymorphism

C++ Inheritance

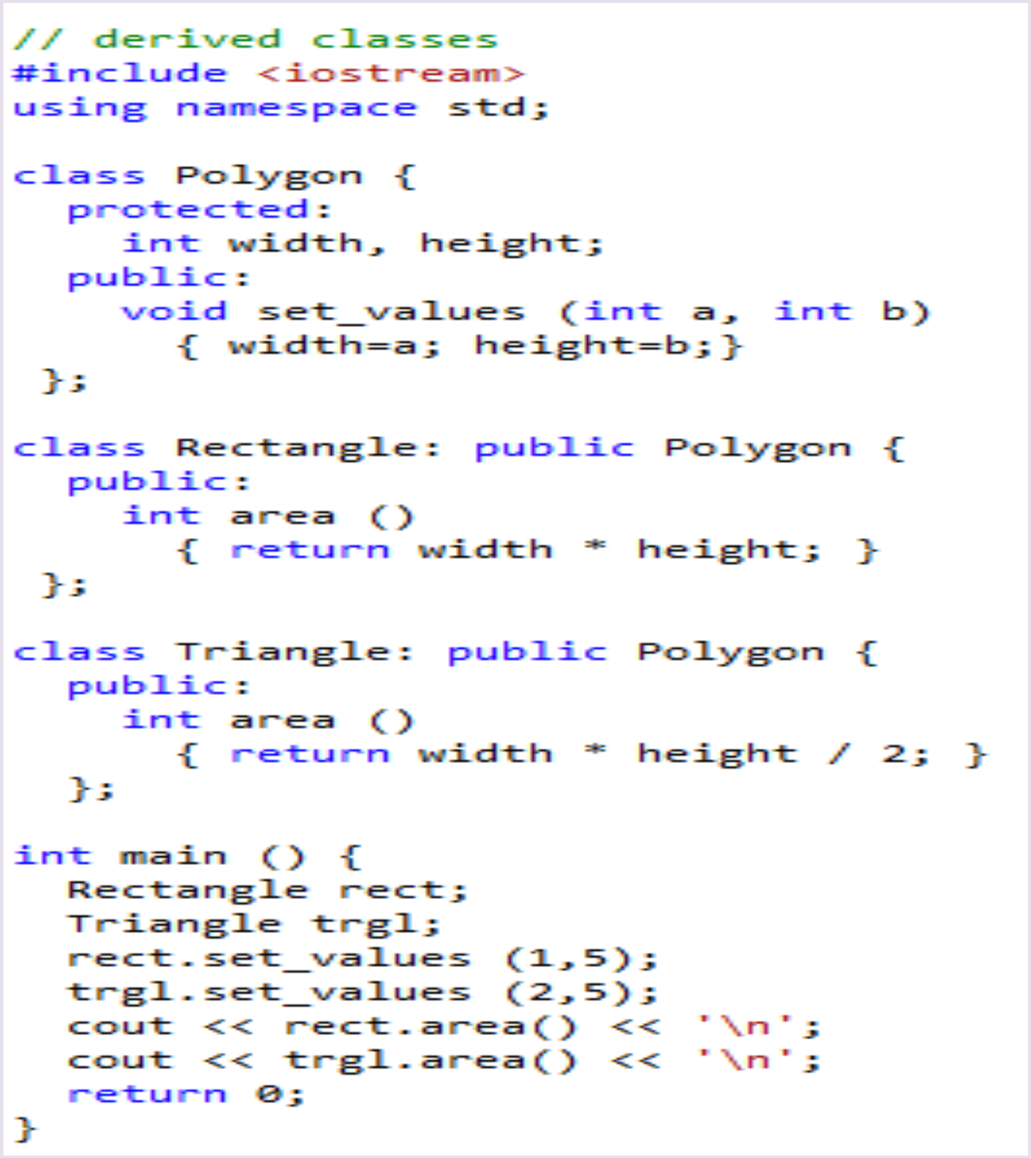
Classes in C++ can be extended, creating new classes which retain characteristics of the base class. This process, known as inheritance, involves a base class and a derived class: The derived class inherits the members of the base class, on top of which it can adds its own members.


Class Animal
Animal
Class Cat
Class Dog
Class kangaroo

C++ Inheritance

```
class derived_class_name: public base_class_name
{

 /*…*/



 };

class Cat: public Animal
{
/*…*/
}
```

Example



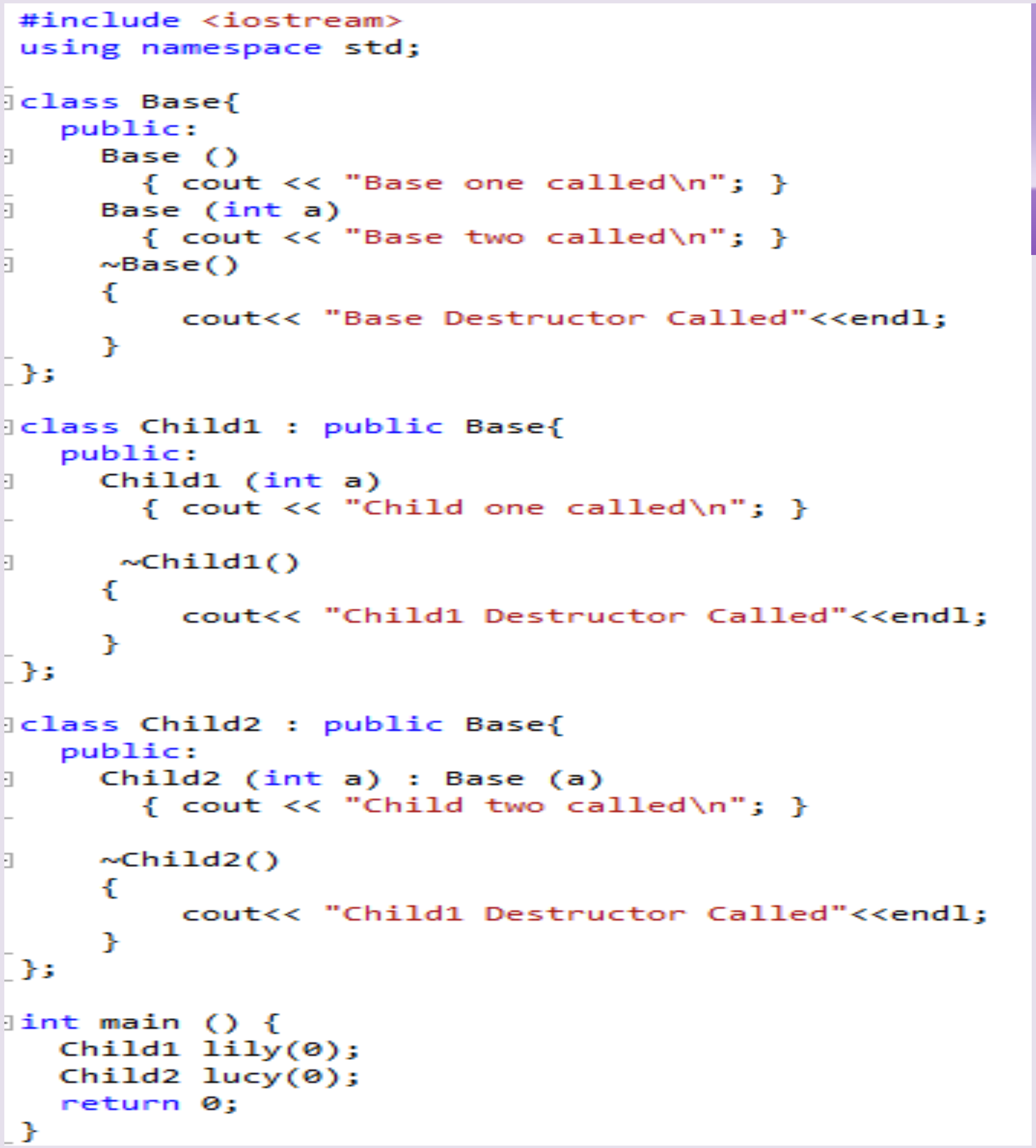
```
// derived classes
#include <iostream>
using namespace std;

class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b;}
 };

class Rectangle: public Polygon {
  public:
    int area ()
      { return width * height; }
 };

class Triangle: public Polygon {
  public:
    int area ()
      { return width * height / 2; }
  };

int main () {
  Rectangle rect;
  Triangle trgl;
  rect.set_values (1,5);
  trgl.set_values (2,5);
  cout << rect.area() << '\n';
  cout << trgl.area() << '\n';
  return 0;
}
```

Acknowledge: http://www.cplusplus.com/doc/tutorial/polymorphism/

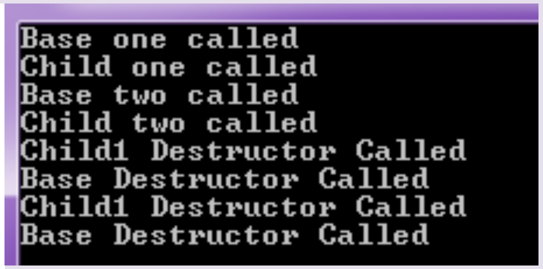
What is inherited from the base class?

- Its constructors and its destructor
- Its assignment operator members (operator=)

What is inherited from the base class?



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

class Base{
   public:
      Base ()
        { cout << "Base one called\n"; }
      Base (int a)
        { cout << "Base two called\n"; }
      ~Base()
      {
          cout<< "Base Destructor Called"<<endl;
      }
};

class Child1 : public Base{
   public:
      Child1 (int a)
        { cout << "Child one called\n"; }

       ~Child1()
      {
          cout<< "Child1 Destructor Called"<<endl;
      }
};

class Child2 : public Base{
   public:
      Child2 (int a) : Base (a)
        { cout << "Child two called\n"; }

      ~Child2()
      {
          cout<< "Child1 Destructor Called"<<endl;
      }
};

int main () {
   Child1 lily(0);
   Child2 lucy(0);
   return 0;
}
```

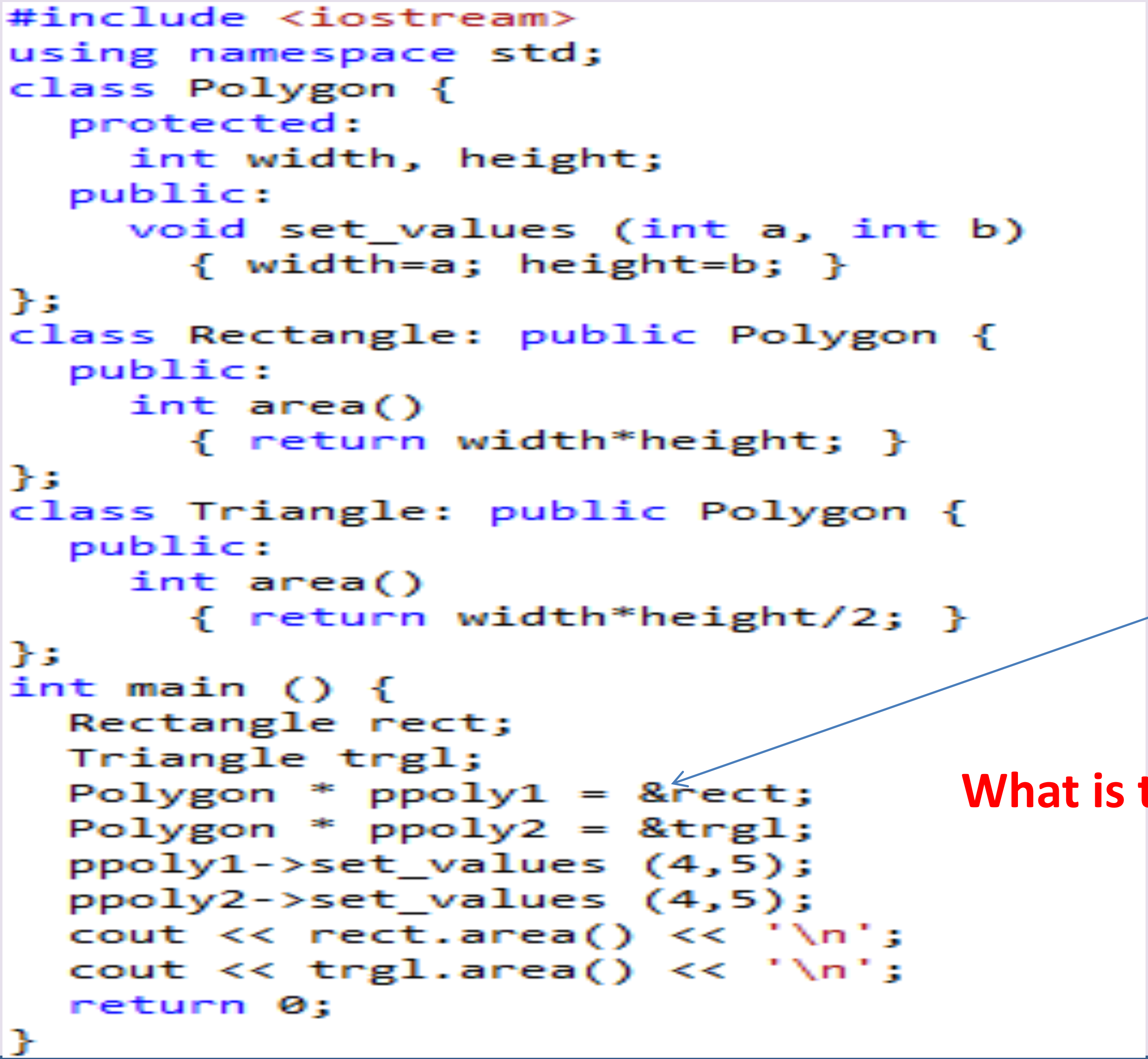


```
Base one called
Child one called
Base two called
Child two called
Child1 Destructor Called
Base Destructor Called
Child1 Destructor Called
Base Destructor Called
```

Engineering

Pointers to base class

A pointer to a derived class is type-compatible with a pointer to its base class.



```
#include <iostream>
using namespace std;
class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b; }
};
class Rectangle: public Polygon {
  public:
    int area()
      { return width*height; }
};
class Triangle: public Polygon {
  public:
    int area()
      { return width*height/2; }
};
int main () {
  Rectangle rect;
  Triangle trgl;
  Polygon * ppoly1 = &rect;
  Polygon * ppoly2 = &trgl;
  ppoly1->set_values (4,5);
  ppoly2->set_values (4,5);
  cout << rect.area() << '\n';
  cout << trgl.area() << '\n';
  return 0;
}
```

Pointer to Derived Class

What is the problem here?

C++ Polymorphism

- C++ Inheritance
- C++ Polymorphism

C++ Polymorphism

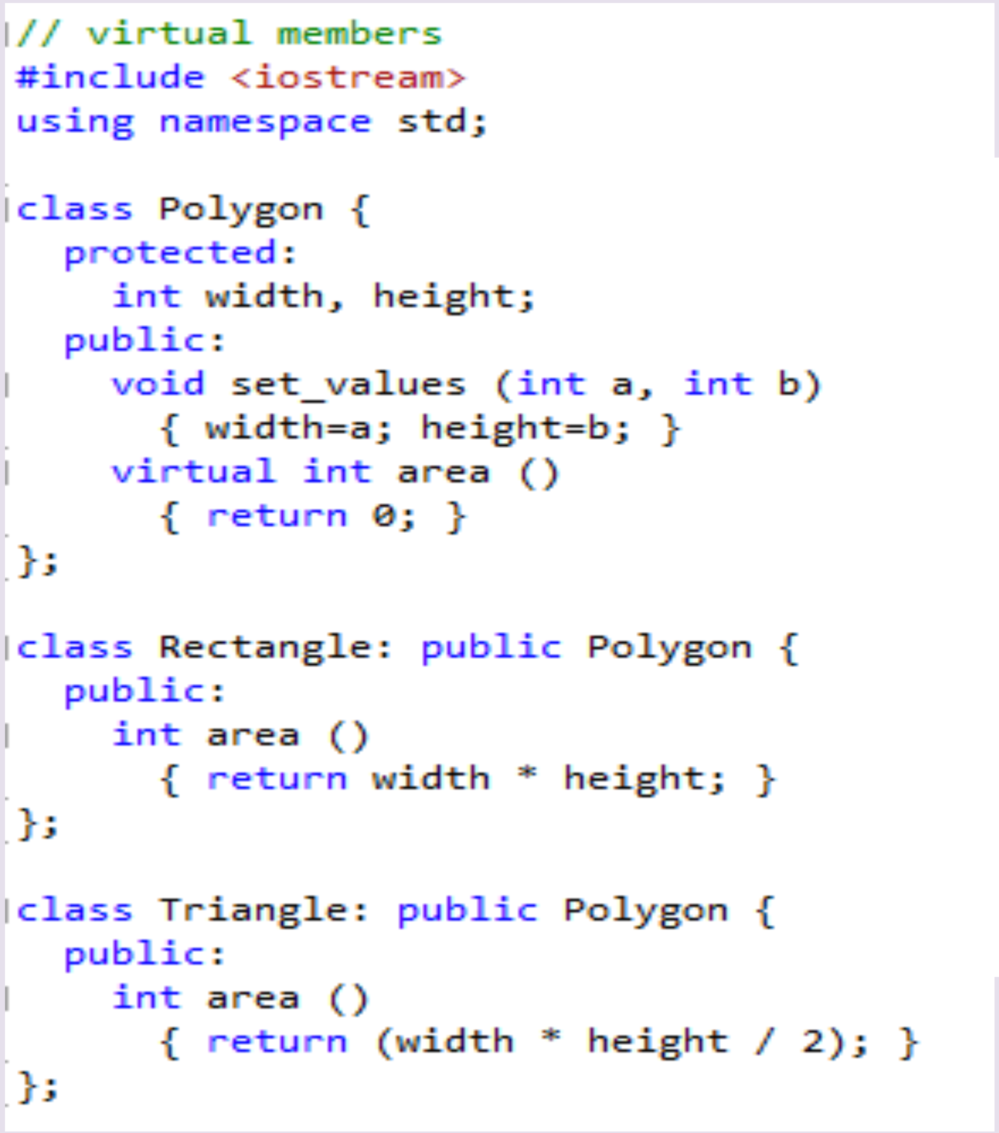
Polymorphism means that some code or operations or objects behave differently in different contexts.

Normally, when the term polymorphism in C++, refers to using virtual methods

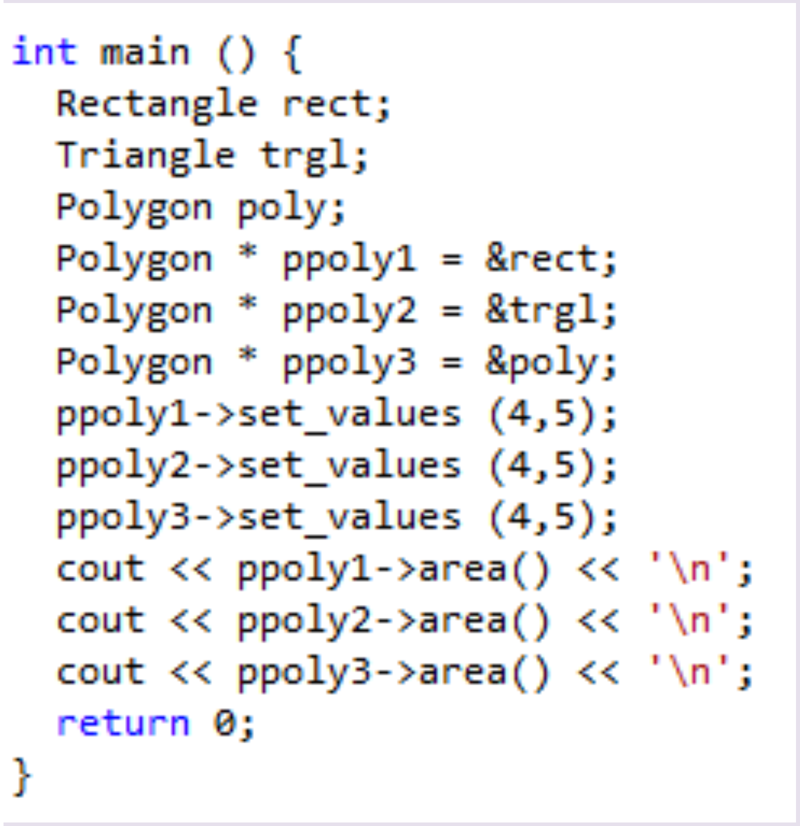
Virtual member

A virtual member is a member function that can be redefined in a derived class, while preserving its calling properties through references.

Virtual member



```cpp
// virtual members
#include <iostream>
using namespace std;

class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b; }
    virtual int area ()
      { return 0; }
};

class Rectangle: public Polygon {
  public:
    int area ()
      { return width * height; }
};

class Triangle: public Polygon {
  public:
    int area ()
      { return (width * height / 2); }
};
```



```cpp
int main () {
  Rectangle rect;
  Triangle trgl;
  Polygon poly;
  Polygon * ppoly1 = &rect;
  Polygon * ppoly2 = &trgl;
  Polygon * ppoly3 = &poly;
  ppoly1->set_values (4,5);
  ppoly2->set_values (4,5);
  ppoly3->set_values (4,5);
  cout << ppoly1->area() << '\n';
  cout << ppoly2->area() << '\n';
  cout << ppoly3->area() << '\n';
  return 0;
}
```

Acknowledge: http://www.cplusplus.com/doc/tutorial/polymorphism/