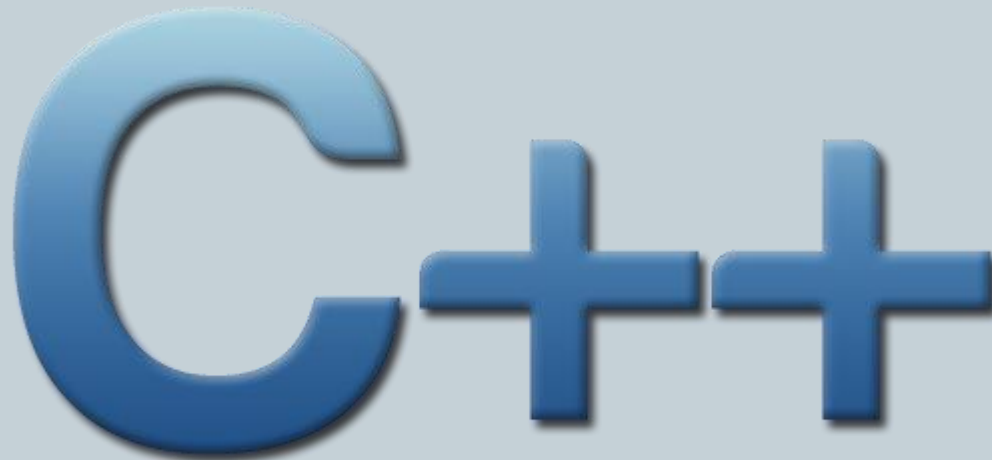


And Even More and More C++

A large, blue, 3D-rendered logo of the C++ programming language. The letter 'C' is on the left, followed by two plus signs. The characters have a slight shadow and a gradient, giving them a three-dimensional appearance. The logo is centered horizontally in the middle of the slide.

Outline

- **C++ Classes**
 - Friendship
 - Inheritance
 - Multiple Inheritance
 - **Polymorphism**
 - **Virtual Members**
 - **Abstract Base Classes**
- **File Input/Output**

Polymorphism

- **Key concept:**
 - A pointer to a derived class is type-compatible with a pointer to its base class
 - This means we can use base class operations on different types of derived classes
 - And that means... polymorphism!

Polymorphism Example

```
// pointers to 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;
}
```

```
20
10
```

Virtual Members

- A member function that can be redefined in a derived class
 - Like an abstract method in Python

```
virtual int area ()  
{ return 0; }
```

Virtual Member Example

```
// 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); }
};

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;
}
```

```
20
10
0
```

Abstract Base Classes

- Very similar concept to Python abstract classes (but we don't need to import the abc library)
 - A class that is not intended to be instantiated
 - Can contain “pure virtual” functions
 - ✦ Functions with no definition
 - A class with at least one pure virtual function is an abstract class

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

Abstract Base Class Example

```
// abstract 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; }
    virtual int area (void) =0;
};

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

class Triangle: public Polygon {
public:
    int area (void)
        { 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 << ppoly1->area() << '\n';
    cout << ppoly2->area() << '\n';
    return 0;
}
```

```
20
10
```


Dynamic Allocation and Polymorphism

```
// dynamic allocation and polymorphism
#include <iostream>
using namespace std;

class Polygon {
protected:
    int width, height;
public:
    Polygon (int a, int b) : width(a), height(b) {}
    virtual int area (void) =0;
    void printarea()
        { cout << this->area() << '\n'; }
};

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

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

int main () {
    Polygon * ppoly1 = new Rectangle (4,5);
    Polygon * ppoly2 = new Triangle (4,5);
    ppoly1->printarea();
    ppoly2->printarea();
    delete ppoly1;
    delete ppoly2;
    return 0;
}
```

20
10

Summary

- **C++ Classes**
 - Friendship
 - Inheritance
 - Multiple Inheritance
 - **Polymorphism**
 - **Virtual Members**
 - **Abstract Base Classes**
- **File Input/Output**

