CS F211: DATA STRUCTURES & ALGORITHMS (2ND SEMESTER 2024-25) INTRODUCTION TO C++

Chittaranjan Hota, PhD
Sr. Professor of Computer Sc.
BITS-Pilani Hyderabad Campus
hota[AT]hyderabad.bits-pilani.ac.in

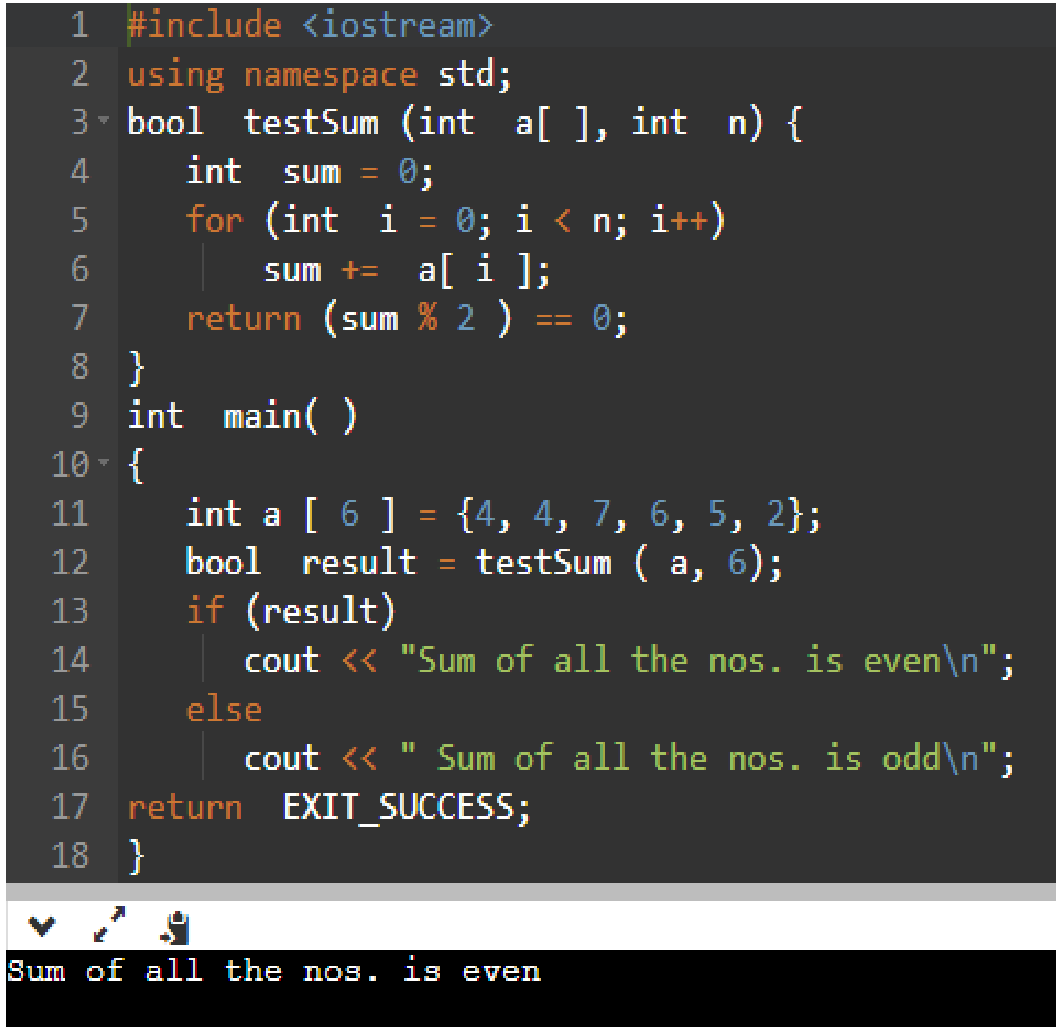
WHY C++ FOR CS F211?

- ✓Developed (in 1979) by Bjarne Stroustrup: Why is it called C++?
- ✓Mid-level: Used for both application level and system level programming tasks.
- ✓Has Object-oriented features improving the quality and reusability of the program.
- ✓Rich library (iostream, iomanip, cmath, cstdlib, iterator, algorithm etc.), Efficiency and speed (competitive coding) …
- ✓Adobe (Photoshop, Illustrator etc are developed using C++, Microsoft used C++ for all of its versions of OS starting from Windows 95, Microsoft Office too is developed using C++, Apple uses C++ to code its OS, MySQL also is written using C++, Mozilla uses a subset of C++, Amazon AWS SDK for C++. Meta, Capgemini, IBM, …

C++ EXAMPLES

```
#include <iostream>
using namespace std;
int main() {
    double num1, num2;
    cout << "Enter the first number: ";
    cin >> num1;
    cout << "Enter the second number: ";
    cin >> num2;
    cout << "Sum: " << ??? << endl;
    cout << "Difference: " << ???   << endl;
    cout << "Product: " << ??? << endl;
    if (num2 != 0) {
        cout << "Quotient: " << ??? << endl;
    }
    else {
        cout << "Division by zero is not allowed." << endl;  }
    return 0;
}
```

Functions and Arrays



```
#include <iostream>
using namespace std;
bool  testSum (int  a[ ], int  n) {
   int  sum = 0;
   for (int  i = 0; i < n; i++)
       sum +=  a[ i ];
   return (sum % 2 ) == 0;
}
int  main( )
{
   int a [ 6 ] = {4, 4, 7, 6, 5, 2};
   bool  result = testSum ( a, 6);
   if (result)
      cout << "Sum of all the nos. is even\n";
   else
      cout << " Sum of all the nos. is odd\n";
return  EXIT_SUCCESS;
}
```

```
Sum of all the nos. is even
```

OBJECT-ORIENTED DESIGN: GOALS AND PRINCIPLES

What is Object-Oriented Design?

- Style of writing computer programs using objects, and their interactions. (Minor degree admissions at BITS, Hyderabad: How many objects and what are their interactions)

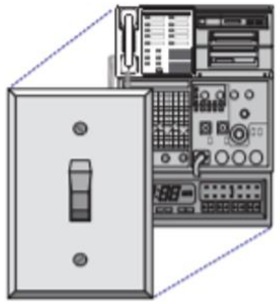
What are the Design Goals?

- Robustness
- Adaptability
- Reusability

(Design Principles)

Abstraction (ADTs are realized by classes in C++)

Encapsulation (Access to data is provided through member functions)



Modularity (different components): supported through hierarchy.

CLASSES IN C++

- Class: A user-defined type or data structure that has data and functions as its members whose access is governed by the access specifiers.
- Object: A variable declared to be of some class, hence includes both data and functions for that object.
- Usage: A variable is an instance of a type. Similarly, an object is an instance of a class.
- Ex:

```
class  Passerger {
      private:
          string           name;
          MealType         mealPref;
          bool             isFreqflyer;
          string           freqFlyerNo;
public:
   Passenger( );
   bool  isFrequentFlyer( ) const { return  isFreqFlyer; }
   void makeFrequentFlyer(const  string&  newFreqFlyerNo) {
         isFreqFlyer = true;
         freqFlyerNo = newFreqFlyerNo;
         }
} ;
```

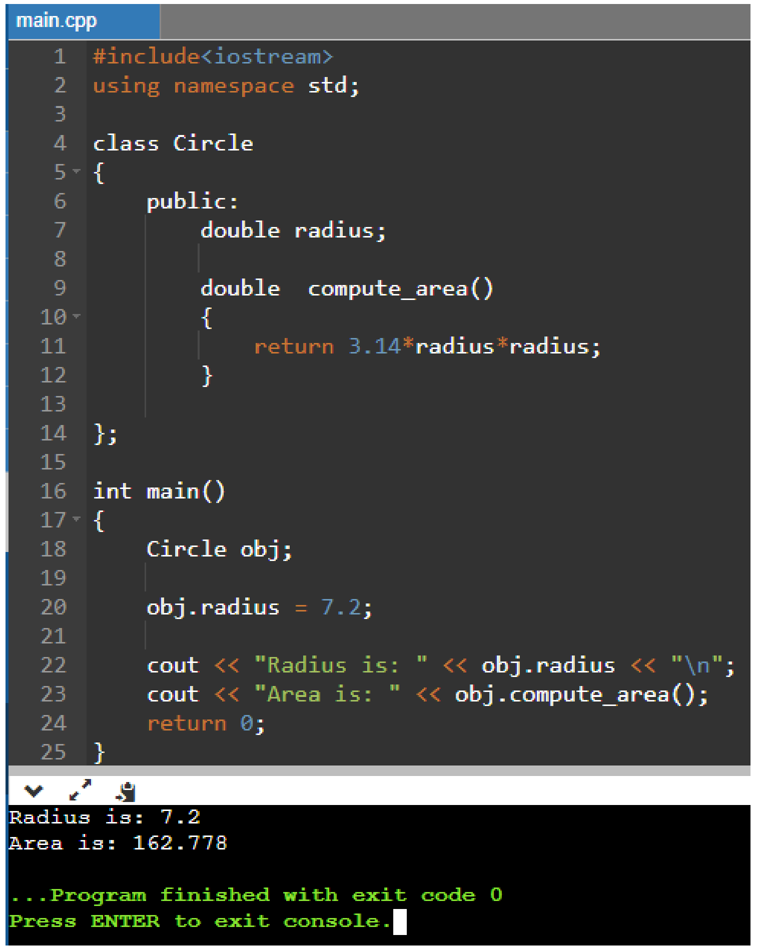
Member variables

Member functions

```
Passenger pass;
if (!pass.isFrequentFlyer()) { pass.makeFrequentFlyer ("12345"); }
```

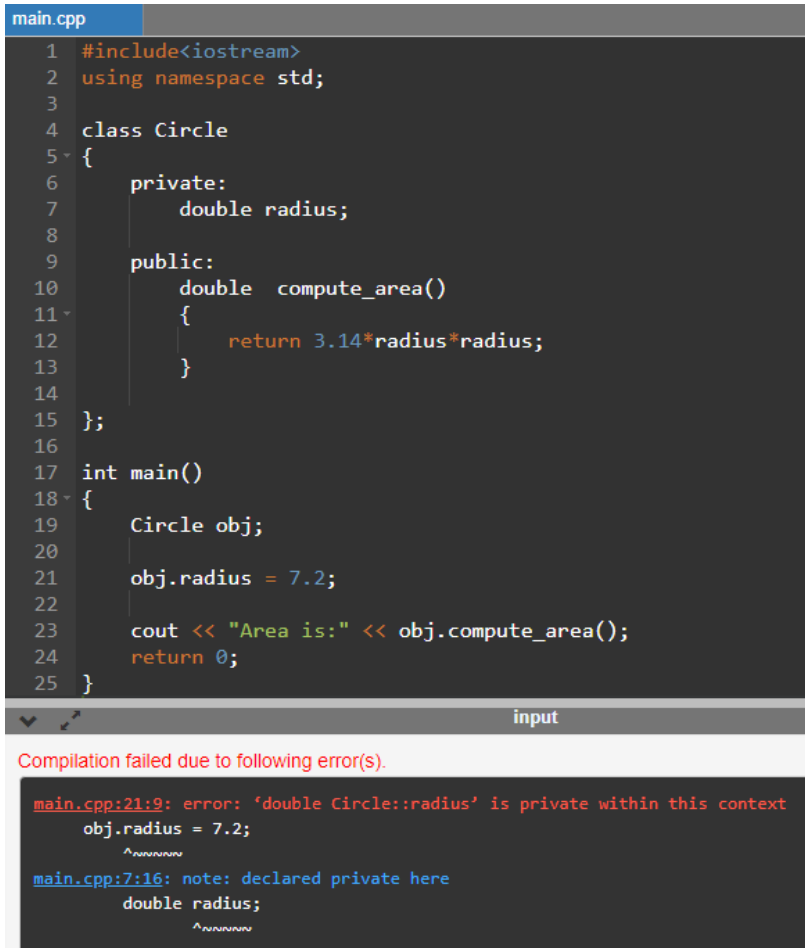
ILLEGAL: pass.name = "Amit";

ACCESS MODIFIERS: PUBLIC, PRIVATE



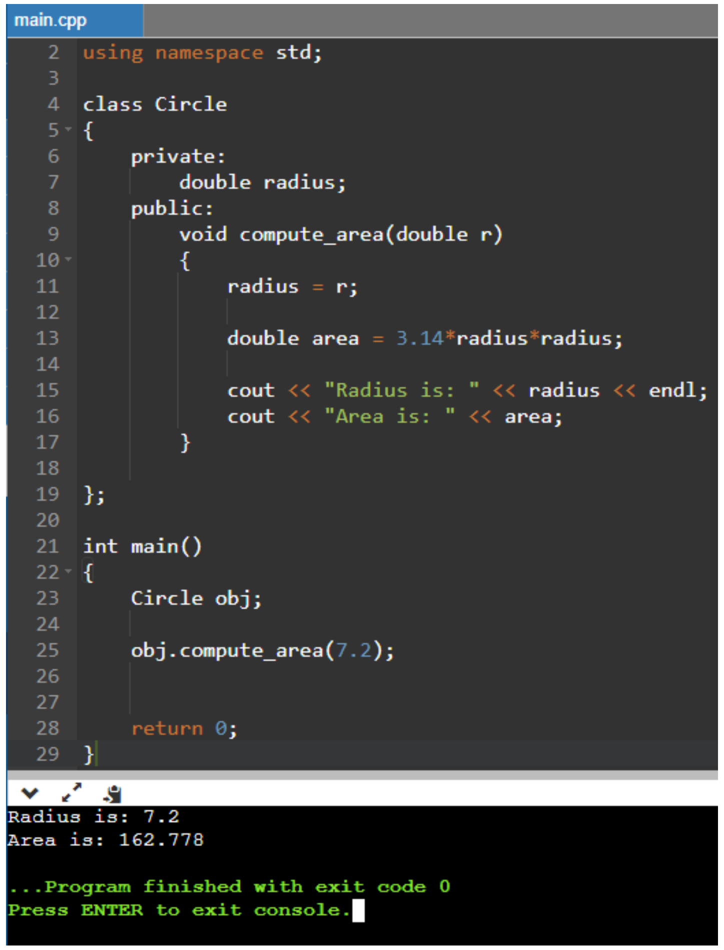
```cpp
#include<iostream>
using namespace std;

class Circle
{
    public:
        double radius;

        double  compute_area()
        {
            return 3.14*radius*radius;
        }

};

int main()
{
    Circle obj;

    obj.radius = 7.2;

    cout << "Radius is: " << obj.radius << "\n";
    cout << "Area is: " << obj.compute_area();
    return 0;
}
```

```
Radius is: 7.2
Area is: 162.778

...Program finished with exit code 0
Press ENTER to exit console.
```



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

class Circle
{
    private:
        double radius;

    public:
        double  compute_area()
        {
            return 3.14*radius*radius;
        }

};

int main()
{
    Circle obj;

    obj.radius = 7.2;

    cout << "Area is:" << obj.compute_area();
    return 0;
}
```

```
Compilation failed due to following error(s).

main.cpp:21:9: error: ‘double Circle::radius’ is private within this context
     obj.radius = 7.2;
         ^~~~~~
main.cpp:7:16: note: declared private here
         double radius;
                ^~~~~~
```



```cpp
using namespace std;

class Circle
{
    private:
        double radius;
    public:
        void compute_area(double r)
        {
            radius = r;

            double area = 3.14*radius*radius;

            cout << "Radius is: " << radius << endl;
            cout << "Area is: " << area;
        }

};

int main()
{
    Circle obj;

    obj.compute_area(7.2);


    return 0;
}
```

```
Radius is: 7.2
Area is: 162.778

...Program finished with exit code 0
Press ENTER to exit console.
```

PROTECTED ACCESS MODIFIER

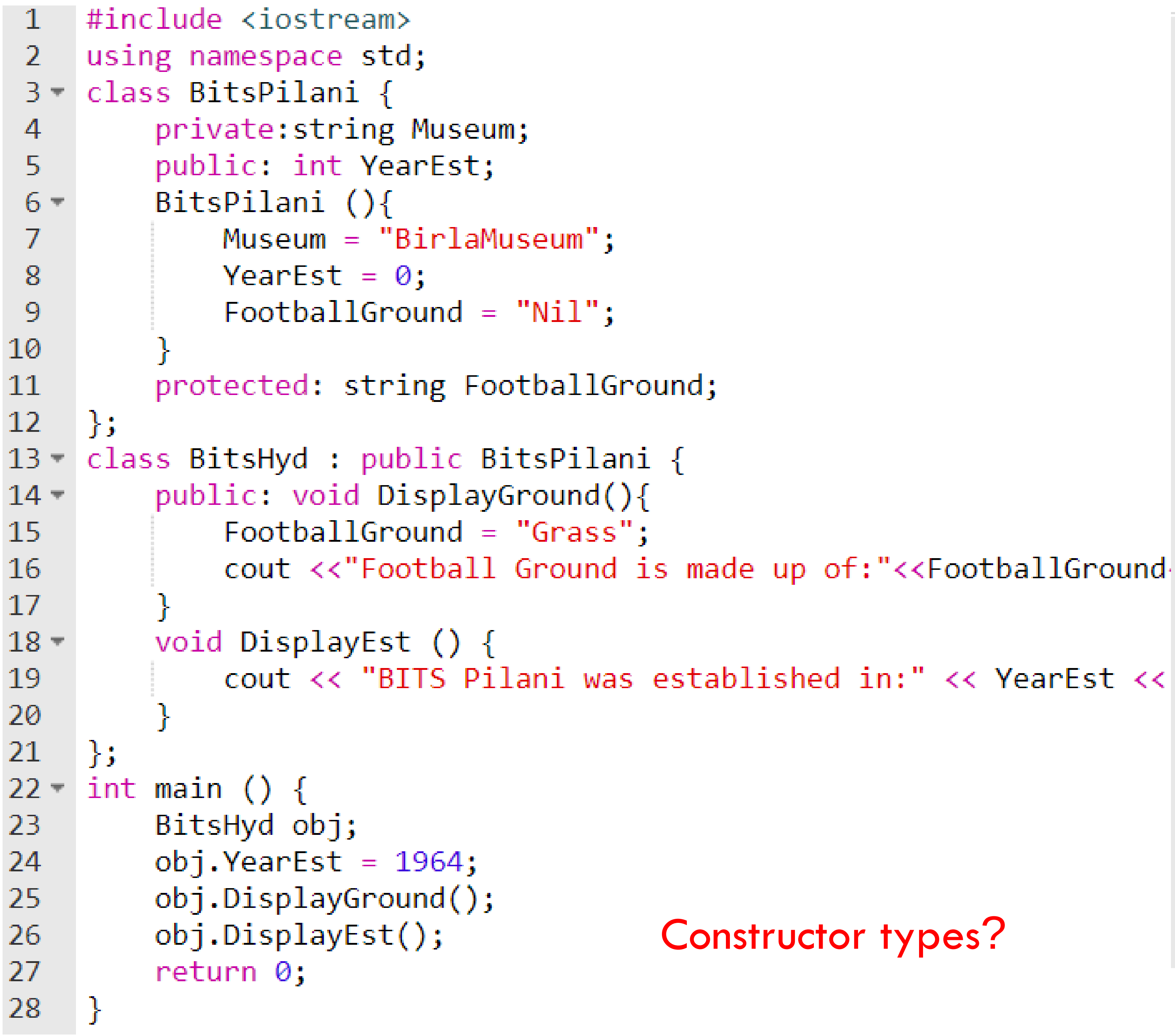
```
class Student{
protected:
    string name;
    int rollNumber;
public:
    Student(string n, int roll) {
        name = n;
        rollNumber = roll;
    }
    void displayBasicDetails() {
      cout << "Name: " << name << endl;
      cout << "Roll Number: " <<
                   rollNumber << endl;
    }
};
```

What type of Constructor is used here?

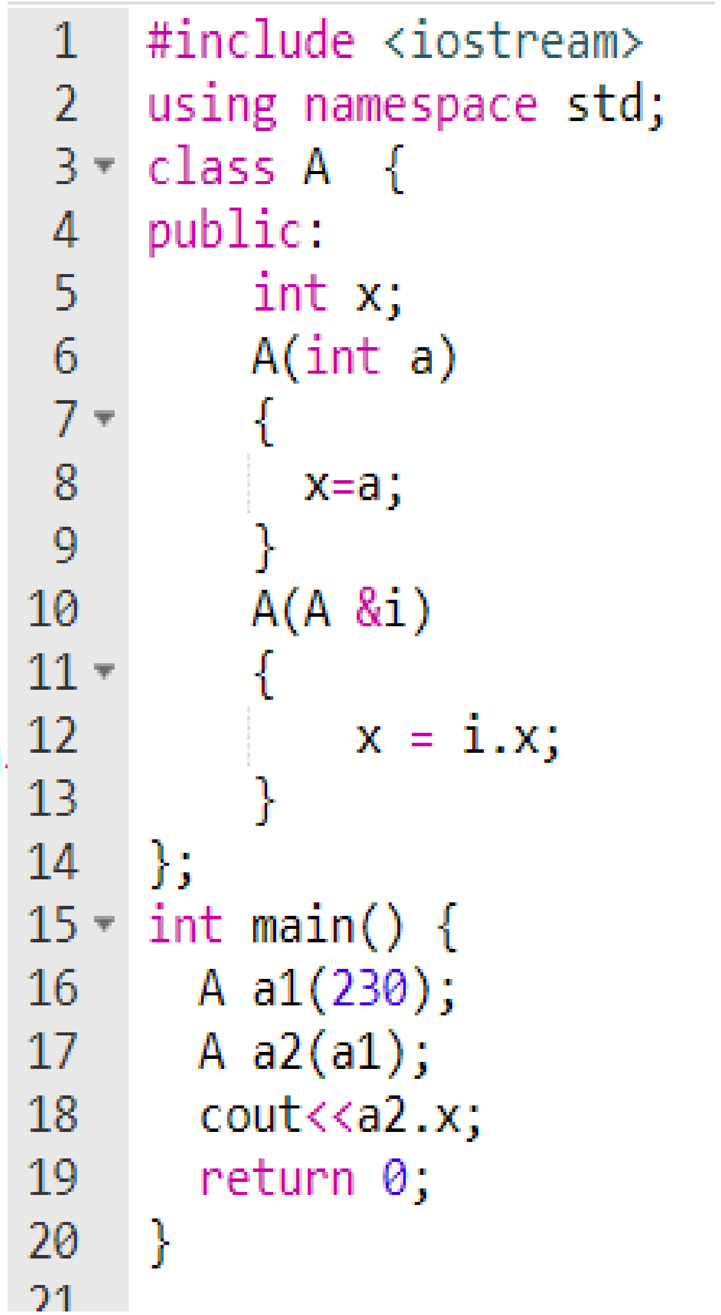
```
class Result : public Student {
private:
    float marks;

public:
  Result(string n, int roll, float m):
                  Student(n, roll) {
        marks = m;
    }
  void displayCompleteDetails() {
     cout << "Name: " << name << endl;
     cout << "Roll Number: " <<
                rollNumber << endl;
     cout << "Marks: " << marks << endl;
    }
};
```

Are there any return types for constructors?

CONTINUED…



```cpp
#include <iostream>
using namespace std;
class BitsPilani {
    private:string Museum;
    public: int YearEst;
    BitsPilani (){
        Museum = "BirlaMuseum";
        YearEst = 0;
        FootballGround = "Nil";
    }
    protected: string FootballGround;
};
class BitsHyd : public BitsPilani {
    public: void DisplayGround(){
        FootballGround = "Grass";
        cout <<"Football Ground is made up of:"<<FootballGround
    }
    void DisplayEst () {
        cout << "BITS Pilani was established in:" << YearEst <<
    }
};
int main () {
    BitsHyd obj;
    obj.YearEst = 1964;
    obj.DisplayGround();
    obj.DisplayEst();
    return 0;
}
```



```cpp
#include <iostream>
using namespace std;
class A  {
public:
    int x;
    A(int a)
    {
      x=a;
    }
    A(A &i)
    {
        x = i.x;
    }
};
int main() {
  A a1(230);
  A a2(a1);
  cout<<a2.x;
  return 0;
}
```

Constructor types?

CLASS INHERITANCE IN C++

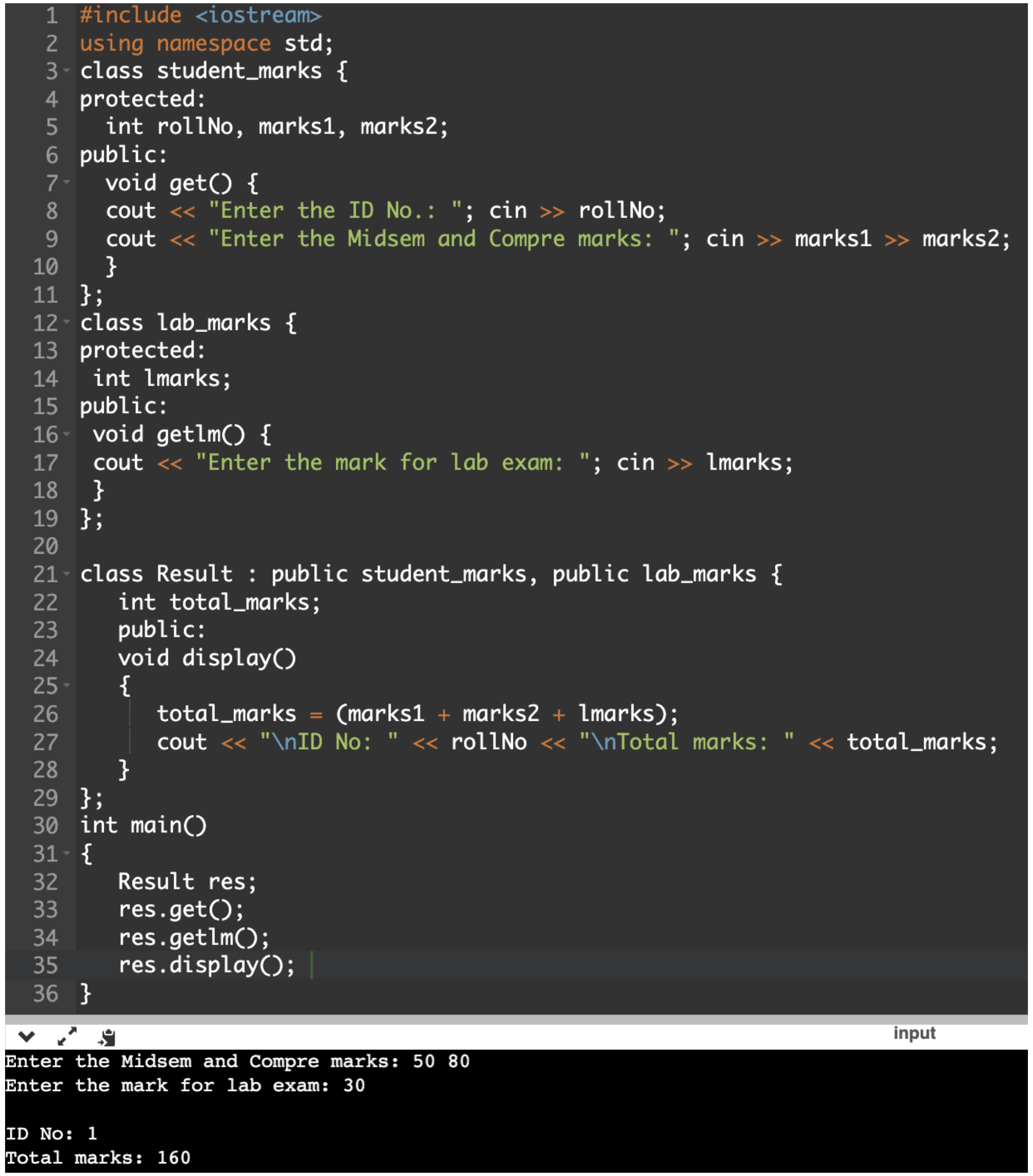
Why is inheritance used in C++?

```
class Person {
  private:
    string name;
    int Aadhaar;
  public:
    void print();
    string getName();
};
```

```
class Student : public Person {
  private:
    string branch;
    int gradYear;
    double cgpa;
    string idNo;
  public:
    void print();
};
```

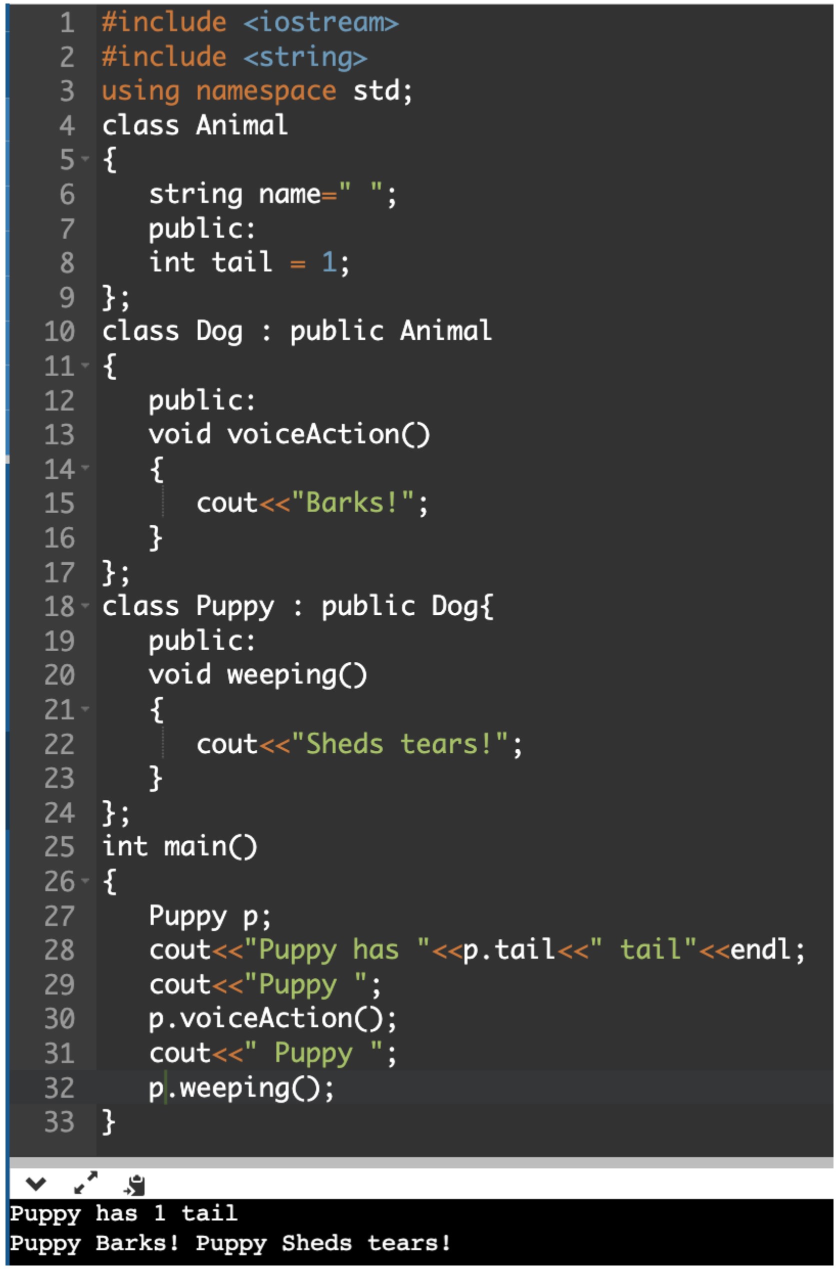
How will you draw the class inheritance diagram?

CONTINUED…



```cpp
#include <iostream>
using namespace std;
class student_marks {
protected:
  int rollNo, marks1, marks2;
public:
  void get() {
  cout << "Enter the ID No.: "; cin >> rollNo;
  cout << "Enter the Midsem and Compre marks: "; cin >> marks1 >> marks2;
  }
};
class lab_marks {
protected:
 int lmarks;
public:
 void getlm() {
 cout << "Enter the mark for lab exam: "; cin >> lmarks;
 }
};

class Result : public student_marks, public lab_marks {
   int total_marks;
   public:
   void display()
   {
      total_marks = (marks1 + marks2 + lmarks);
      cout << "\nID No: " << rollNo << "\nTotal marks: " << total_marks;
   }
};
int main()
{
   Result res;
   res.get();
   res.getlm();
   res.display();
}
```

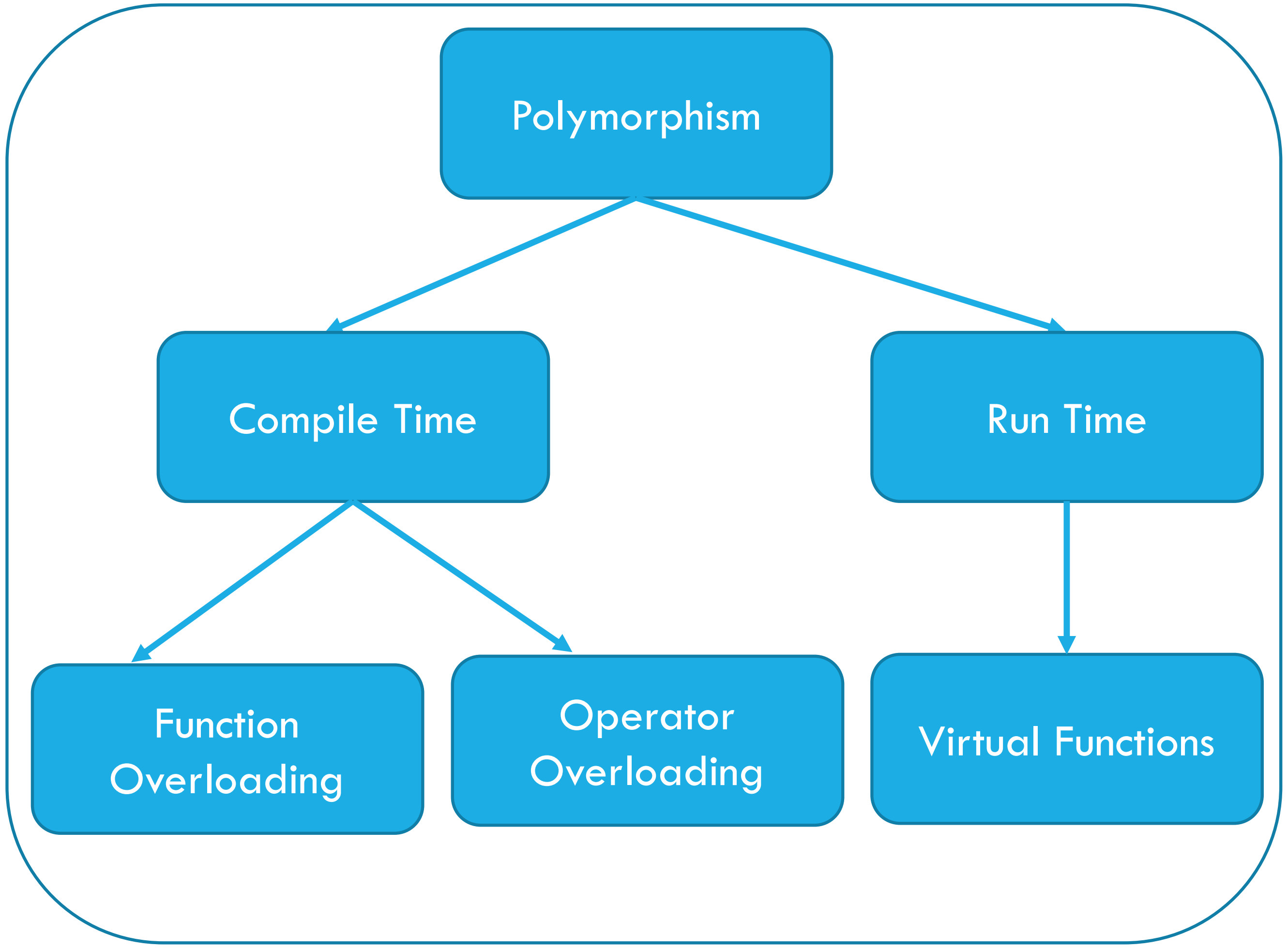
```
Enter the Midsem and Compre marks: 50 80
Enter the mark for lab exam: 30

ID No: 1
Total marks: 160
```

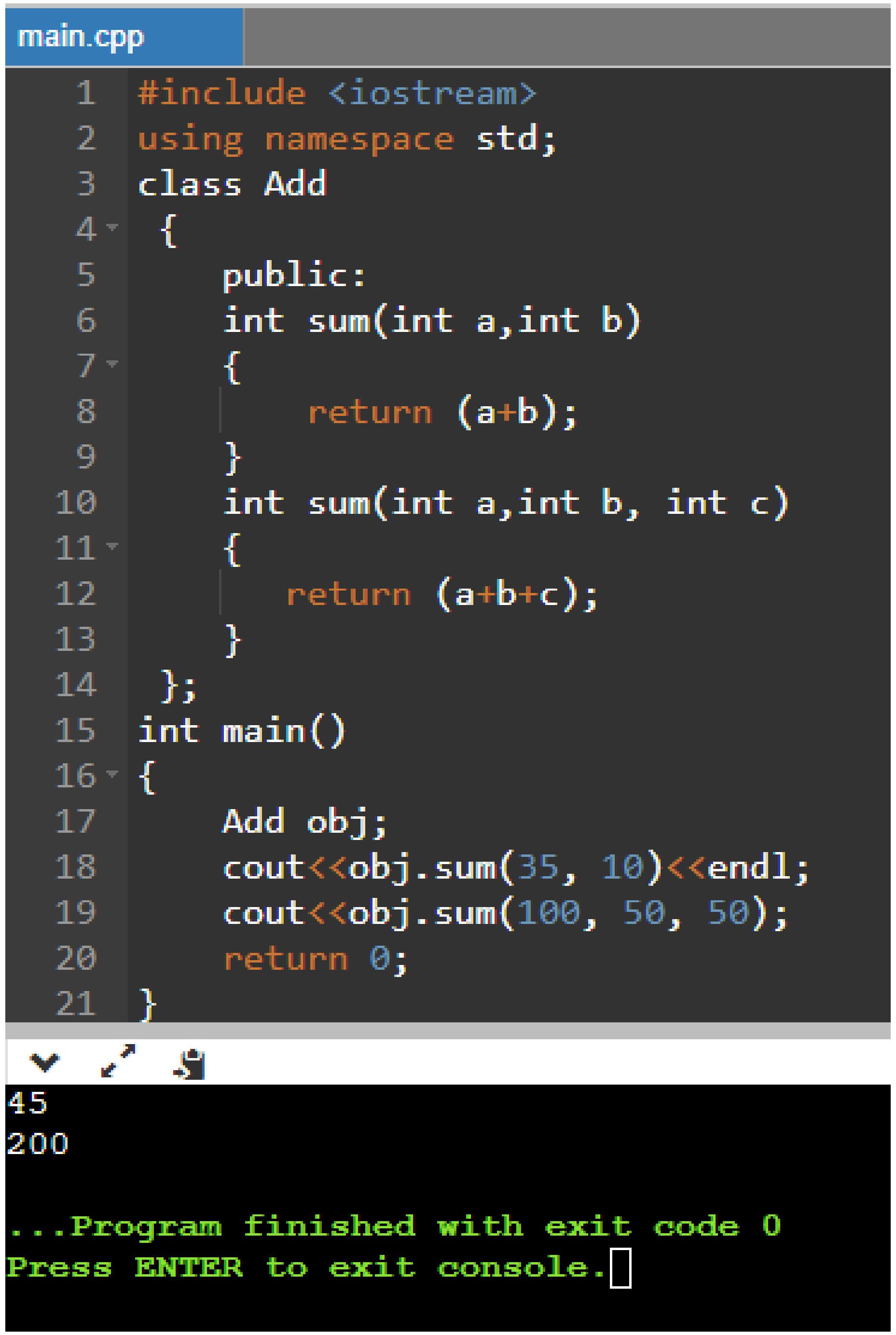


```cpp
#include <iostream>
#include <string>
using namespace std;
class Animal
{
   string name=" ";
   public:
   int tail = 1;
};
class Dog : public Animal
{
   public:
   void voiceAction()
   {
      cout<<"Barks!";
   }
};
class Puppy : public Dog{
   public:
   void weeping()
   {
      cout<<"Sheds tears!";
   }
};
int main()
{
   Puppy p;
   cout<<"Puppy has "<<p.tail<<" tail"<<endl;
   cout<<"Puppy ";
   p.voiceAction();
   cout<<" Puppy ";
   p.weeping();
}
```

```
Puppy has 1 tail
Puppy Barks! Puppy Sheds tears!
```

POLYMORPHISM IN C++

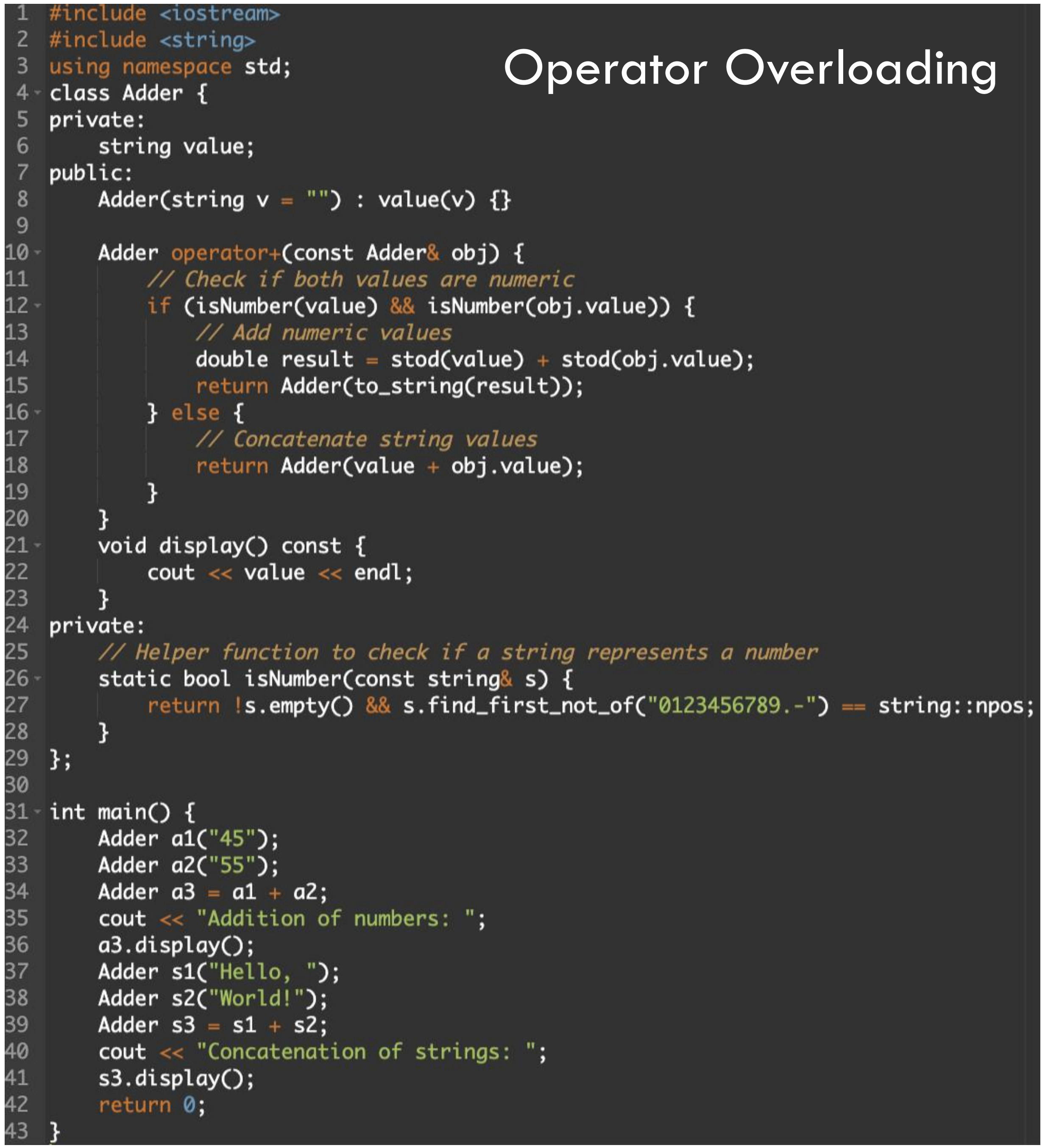



main.cpp

```
#include <iostream>
using namespace std;
class Add
 {
     public:
     int sum(int a,int b)
     {
         return (a+b);
     }
     int sum(int a,int b, int c)
     {
        return (a+b+c);
     }
 };
int main()
{
     Add obj;
     cout<<obj.sum(35, 10)<<endl;
     cout<<obj.sum(100, 50, 50);
     return 0;
}
```

```
45
200

...Program finished with exit code 0
Press ENTER to exit console.
```

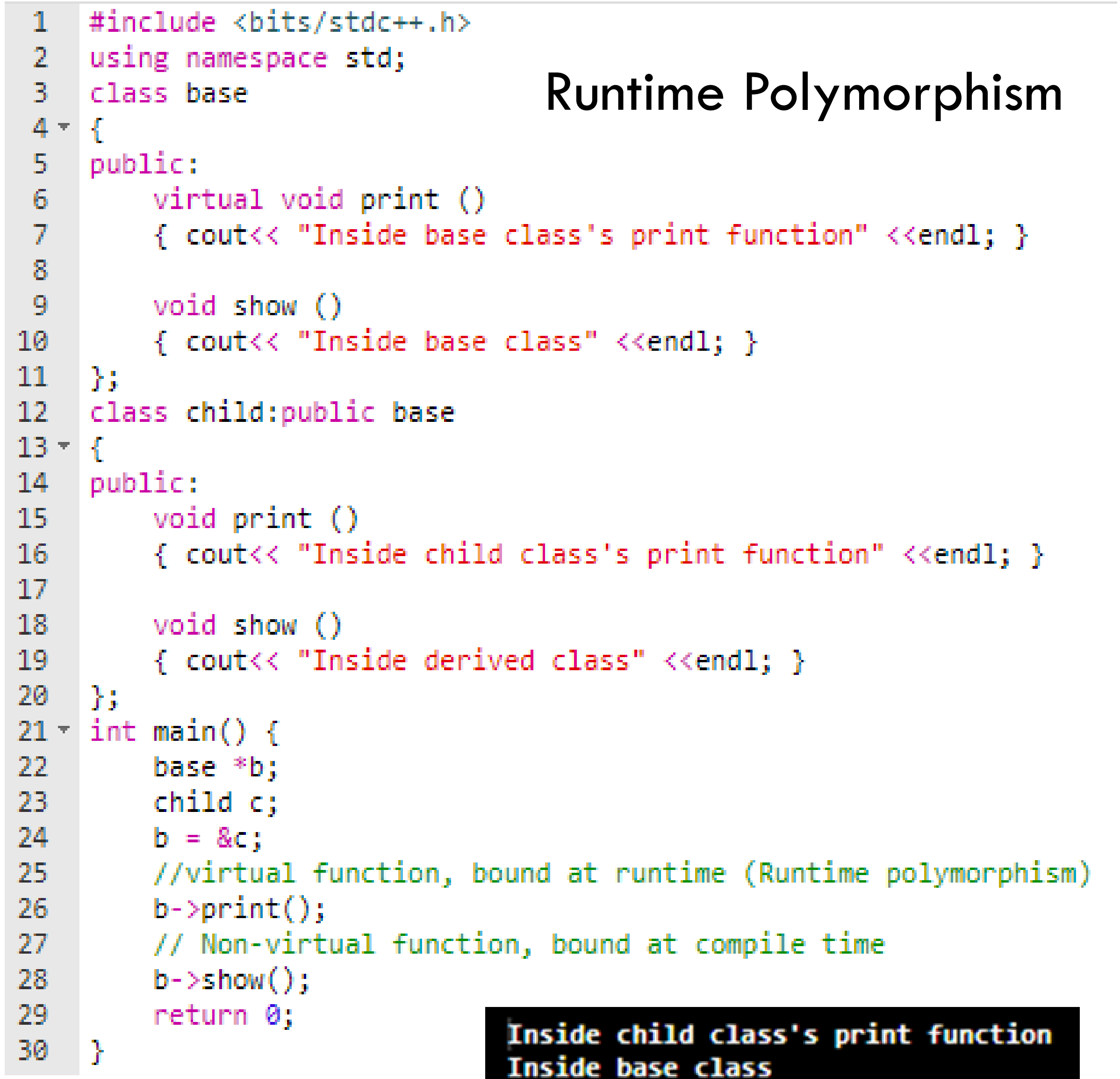
Operator Overloading



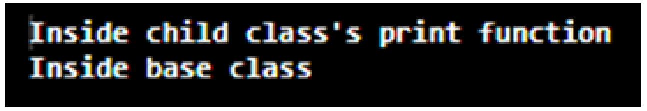
```cpp
#include <iostream>
#include <string>
using namespace std;
class Adder {
private:
    string value;
public:
    Adder(string v = "") : value(v) {}

    Adder operator+(const Adder& obj) {
        // Check if both values are numeric
        if (isNumber(value) && isNumber(obj.value)) {
            // Add numeric values
            double result = stod(value) + stod(obj.value);
            return Adder(to_string(result));
        } else {
            // Concatenate string values
            return Adder(value + obj.value);
        }
    }
    void display() const {
        cout << value << endl;
    }
private:
    // Helper function to check if a string represents a number
    static bool isNumber(const string& s) {
        return !s.empty() && s.find_first_not_of("0123456789.-") == string::npos;
    }
};

int main() {
    Adder a1("45");
    Adder a2("55");
    Adder a3 = a1 + a2;
    cout << "Addition of numbers: ";
    a3.display();
    Adder s1("Hello, ");
    Adder s2("World!");
    Adder s3 = s1 + s2;
    cout << "Concatenation of strings: ";
    s3.display();
    return 0;
}
```

- A C++ virtual function is a member function in the base class that you redefine in a derived class.

Runtime Polymorphism



```cpp
#include <bits/stdc++.h>
using namespace std;
class base
{
public:
    virtual void print ()
    { cout<< "Inside base class's print function" <<endl; }

    void show ()
    { cout<< "Inside base class" <<endl; }
};
class child:public base
{
public:
    void print ()
    { cout<< "Inside child class's print function" <<endl; }

    void show ()
    { cout<< "Inside derived class" <<endl; }
};
int main() {
    base *b;
    child c;
    b = &c;
    //virtual function, bound at runtime (Runtime polymorphism)
    b->print();
    // Non-virtual function, bound at compile time
    b->show();
    return 0;
}
```

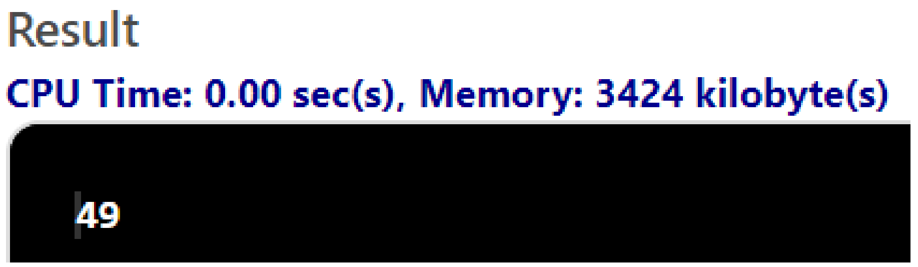


```
Inside child class's print function
Inside base class
```

Function Overriding

FRIEND CLASS IN C++

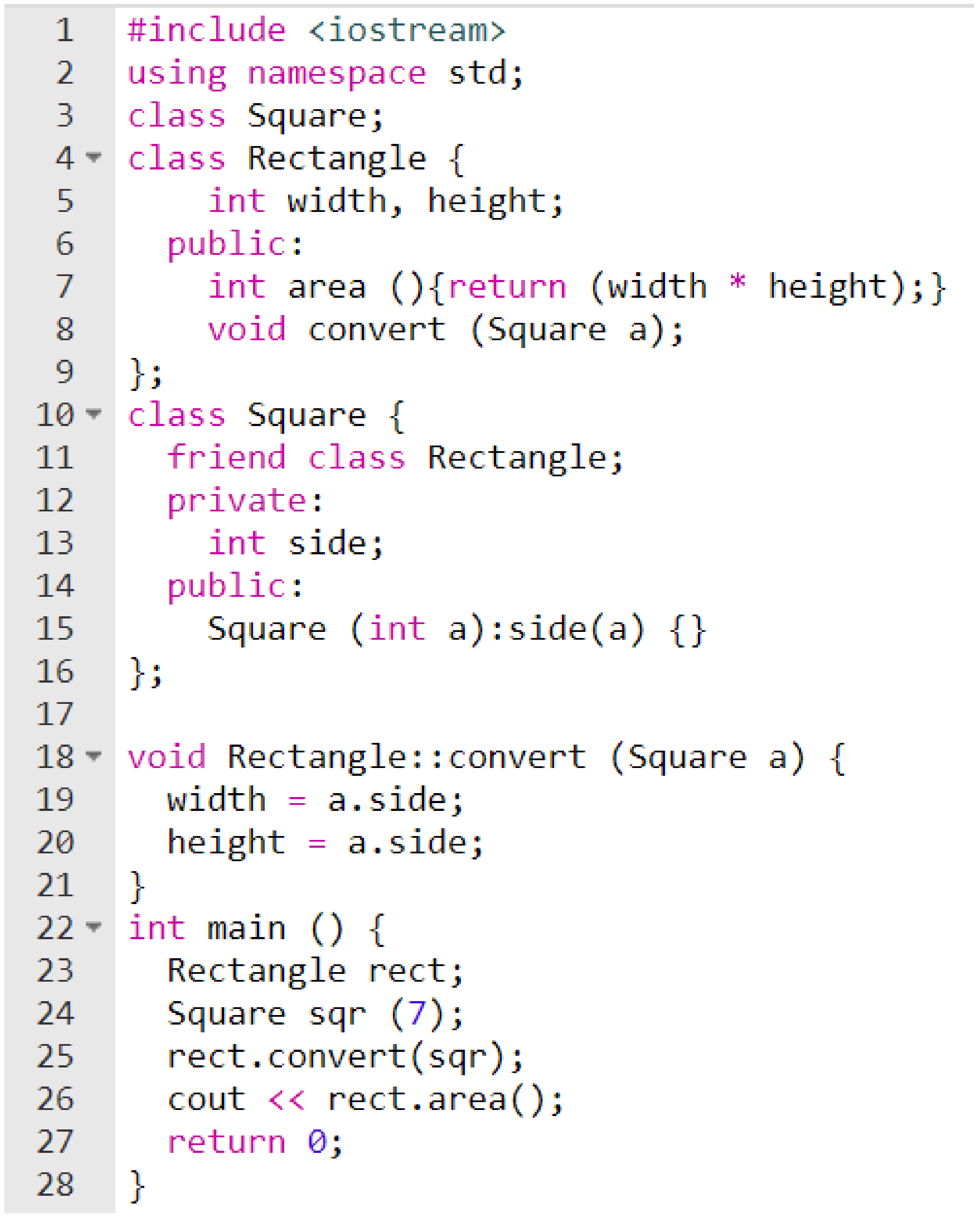
- ✓ A friend class is a class whose members have access to the private members of another class.
- ✓ Rectangle is a friend of Square allowing Rectangle's member functions to access what members of Square?
- ✓ Is friendship transitive?
- ✓ Can a friend not access protected members?



Result

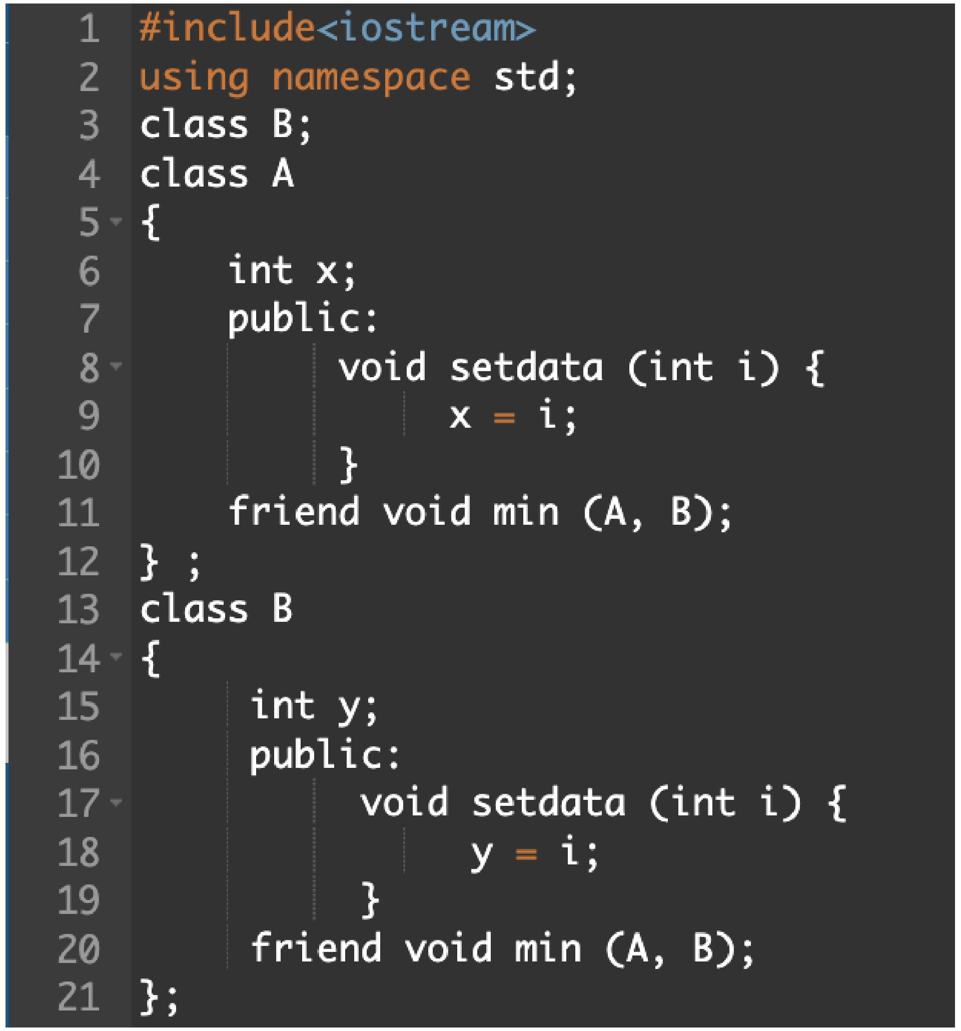
CPU Time: 0.00 sec(s), Memory: 3424 kilobyte(s)

```
49
```

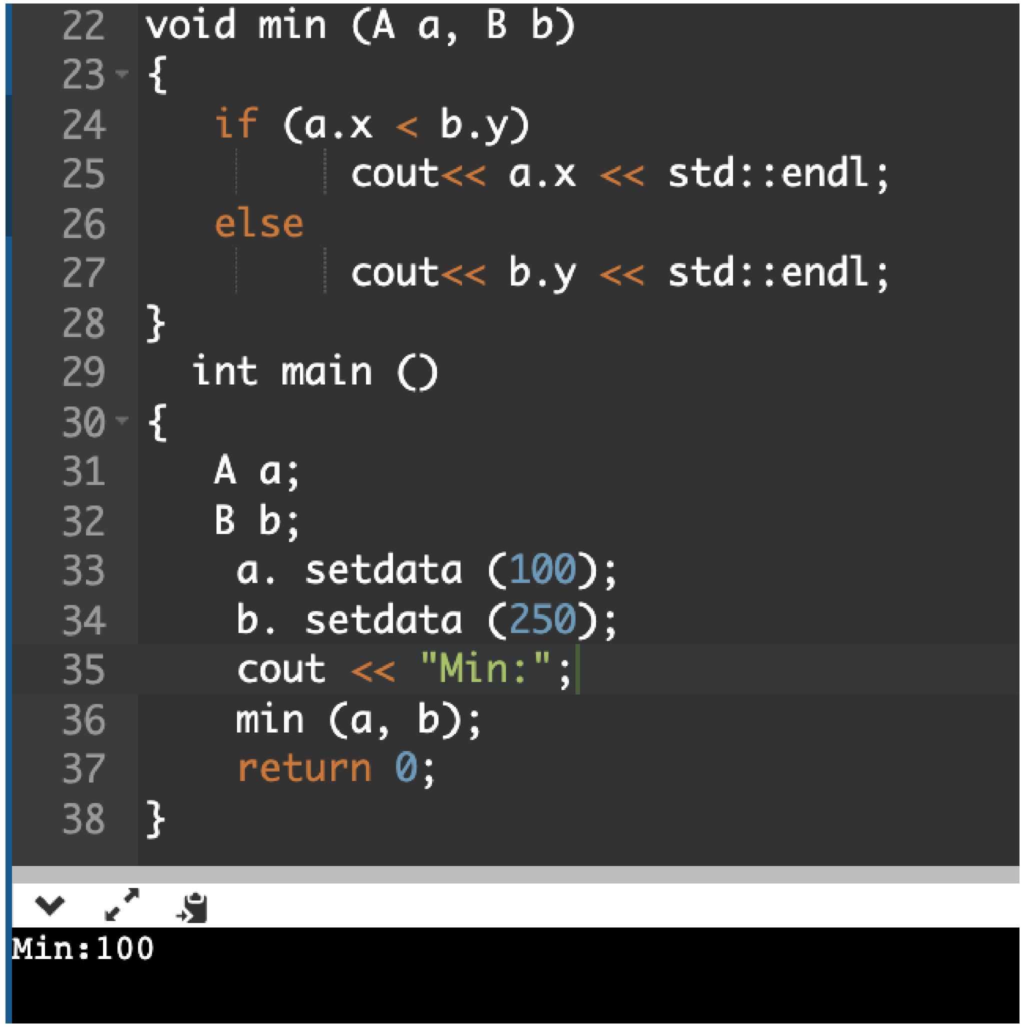


```cpp
#include <iostream>
using namespace std;
class Square;
class Rectangle {
    int width, height;
  public:
    int area (){return (width * height);}
    void convert (Square a);
};
class Square {
  friend class Rectangle;
  private:
    int side;
  public:
    Square (int a):side(a) {}
};

void Rectangle::convert (Square a) {
  width = a.side;
  height = a.side;
}
int main () {
  Rectangle rect;
  Square sqr (7);
  rect.convert(sqr);
  cout << rect.area();
  return 0;
}
```

FRIEND FUNCTION IN C++



```cpp
#include<iostream>
using namespace std;
class B;
class A
{
    int x;
    public:
        void setdata (int i) {
            x = i;
        }
    friend void min (A, B);
} ;
class B
{
     int y;
     public:
         void setdata (int i) {
             y = i;
         }
     friend void min (A, B);
};
```

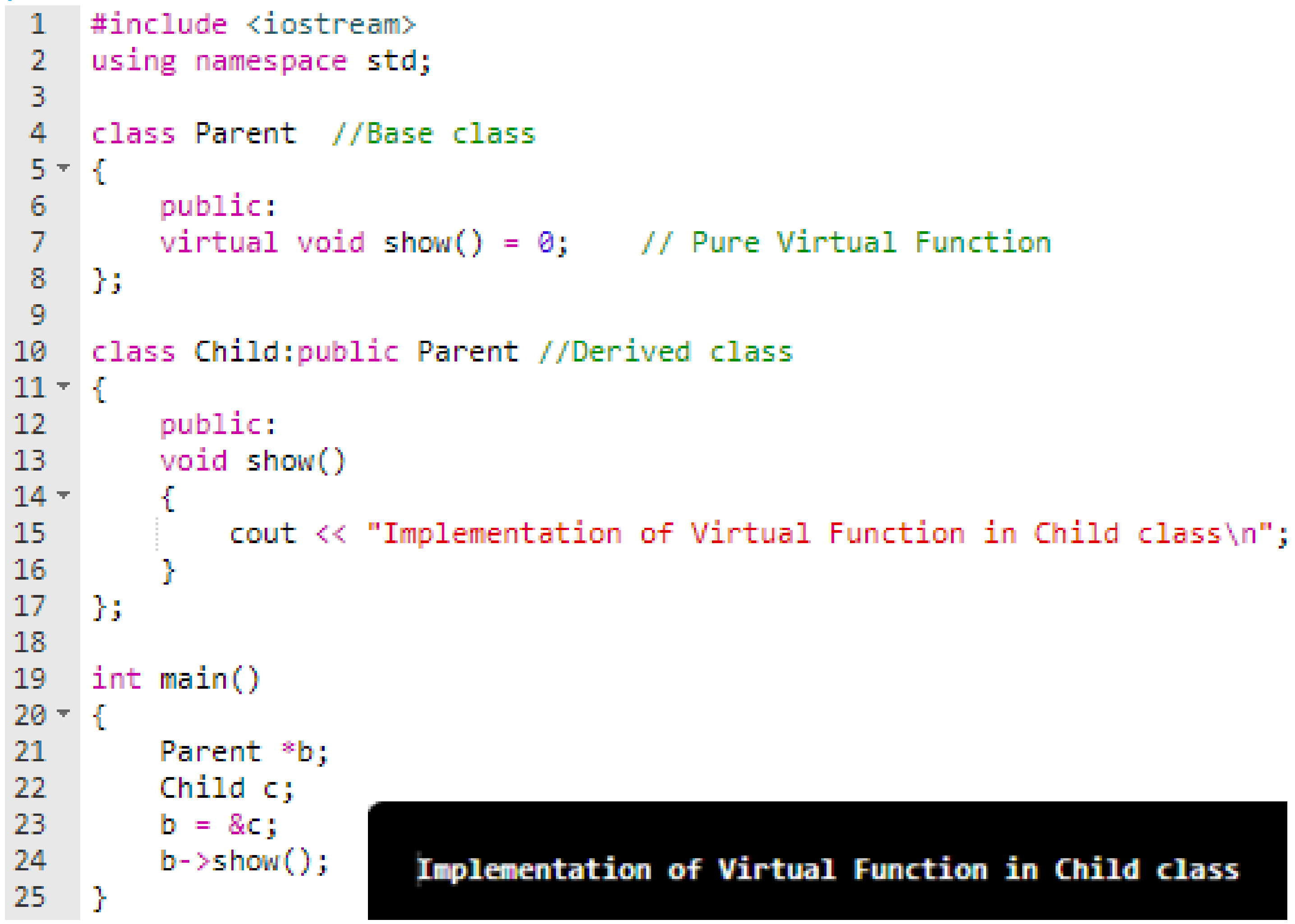


```cpp
void min (A a, B b)
{
   if (a.x < b.y)
         cout<< a.x << std::endl;
   else
         cout<< b.y << std::endl;
}
  int main ()
{
   A a;
   B b;
    a. setdata (100);
    b. setdata (250);
    cout << "Min:";
    min (a, b);
    return 0;
}
```

```
Min:100
```

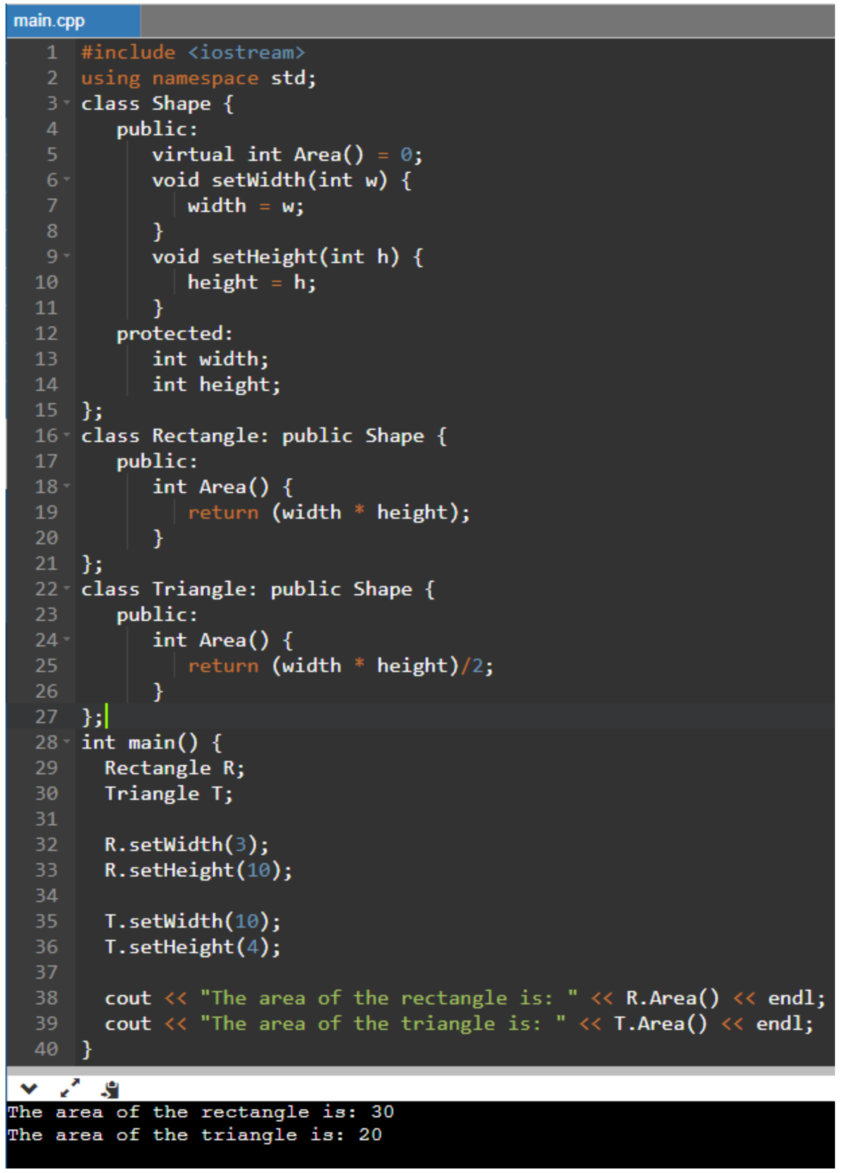
What all accesses min() has from A and B ?

ABSTRACT CLASSES IN C++



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

class Parent  //Base class
{
    public:
    virtual void show() = 0;    // Pure Virtual Function
};

class Child:public Parent //Derived class
{
    public:
    void show()
    {
        cout << "Implementation of Virtual Function in Child class\n";
    }
};

int main()
{
    Parent *b;
    Child c;
    b = &c;
    b->show();
}
```

```
Implementation of Virtual Function in Child class
```



main.cpp

```cpp
#include <iostream>
using namespace std;
class Shape {
   public:
      virtual int Area() = 0;
      void setWidth(int w) {
         width = w;
      }
      void setHeight(int h) {
         height = h;
      }
   protected:
      int width;
      int height;
};
class Rectangle: public Shape {
   public:
      int Area() {
         return (width * height);
      }
};
class Triangle: public Shape {
   public:
      int Area() {
         return (width * height)/2;
      }
};
int main() {
  Rectangle R;
  Triangle T;

  R.setWidth(3);
  R.setHeight(10);

  T.setWidth(10);
  T.setHeight(4);

  cout << "The area of the rectangle is: " << R.Area() << endl;
  cout << "The area of the triangle is: " << T.Area() << endl;
}
```

```
The area of the rectangle is: 30
The area of the triangle is: 20
```

DESIGN PATTERNS: TEMPLATES IN C++

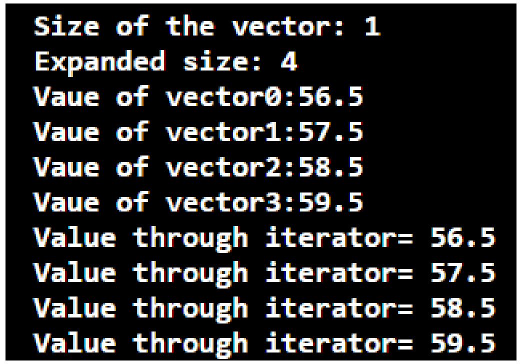
```
template <typename T>
T myMax(T x, T y)
{
   return (x > y)? x: y;
}

int main()
{
   cout << myMax<int>(4, 8) << endl;
   cout << myMax<char>('b', 'm') << endl;
   cout << myMax<double>(7.2, 5.0) << endl;
   return 0;
}
```

Compiler will internally generate what code?

STANDARD TEMPLATE LIBRARY (STL) IN C++

- A library of container classes, algorithms, and iterators.

Can you name some?



```
Size of the vector: 1
Expanded size: 4
Vaue of vector0:56.5
Vaue of vector1:57.5
Vaue of vector2:58.5
Vaue of vector3:59.5
Value through iterator= 56.5
Value through iterator= 57.5
Value through iterator= 58.5
Value through iterator= 59.5
```

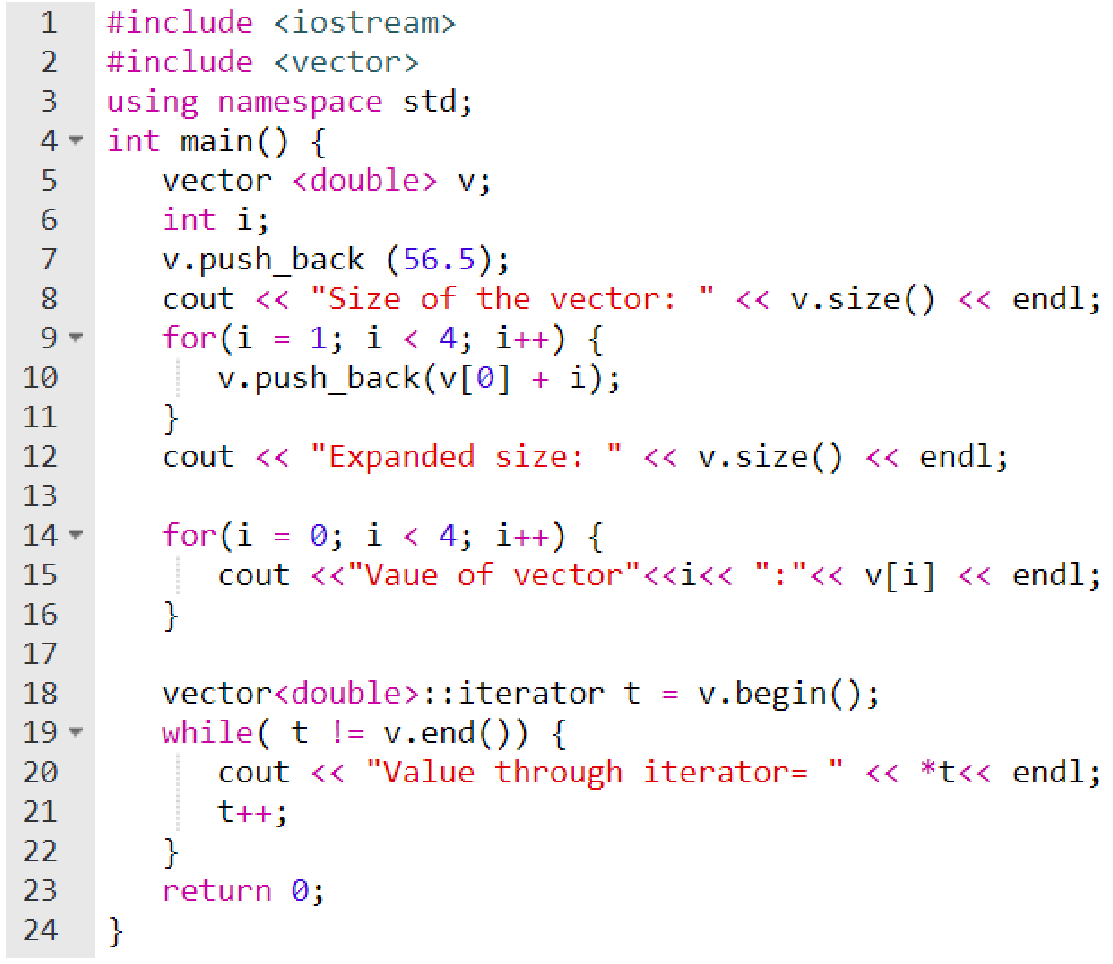
$$\text{InitialValue} + \sum_{i=0}^{n-1} a[i]$$

How to do this using STLs?

Can you name some of the STL functions in this code?

STL on strings:

insert, append, swap, size, resize, reverse etc.



```cpp
#include <iostream>
#include <vector>
using namespace std;
int main() {
    vector <double> v;
    int i;
    v.push_back (56.5);
    cout << "Size of the vector: " << v.size() << endl;
    for(i = 1; i < 4; i++) {
        v.push_back(v[0] + i);
    }
    cout << "Expanded size: " << v.size() << endl;

    for(i = 0; i < 4; i++) {
        cout <<"Vaue of vector"<<i<< ":"<< v[i] << endl;
    }

    vector<double>::iterator t = v.begin();
    while( t != v.end()) {
        cout << "Value through iterator= " << *t<< endl;
        t++;
    }
    return 0;
}
```

THANK YOU!

Next Class: Elementary data structures (Arrays and Linked lists)…