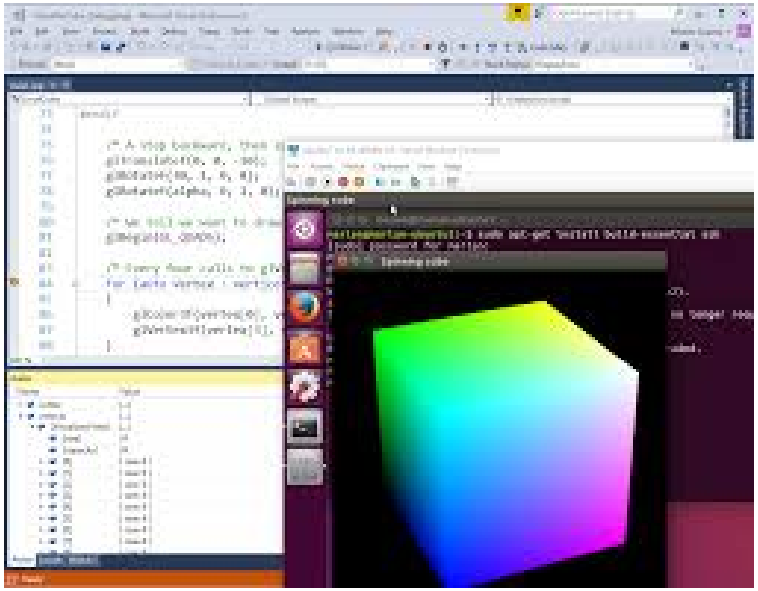
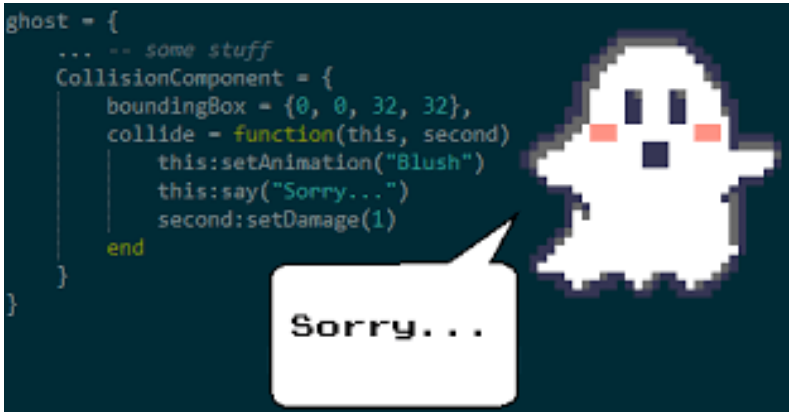
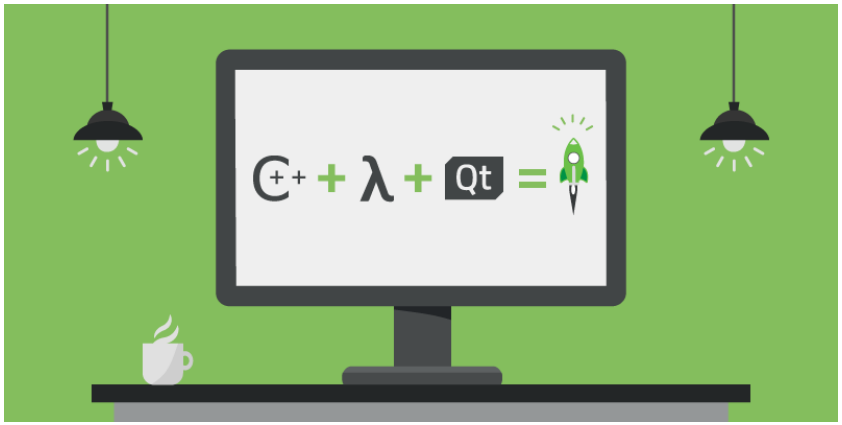
C/C++ Programming: Intro to C++ (1/3)







Course **EE3093** – Lecturer: Dr F. Verdicchio

Historical Timeline

Origins:

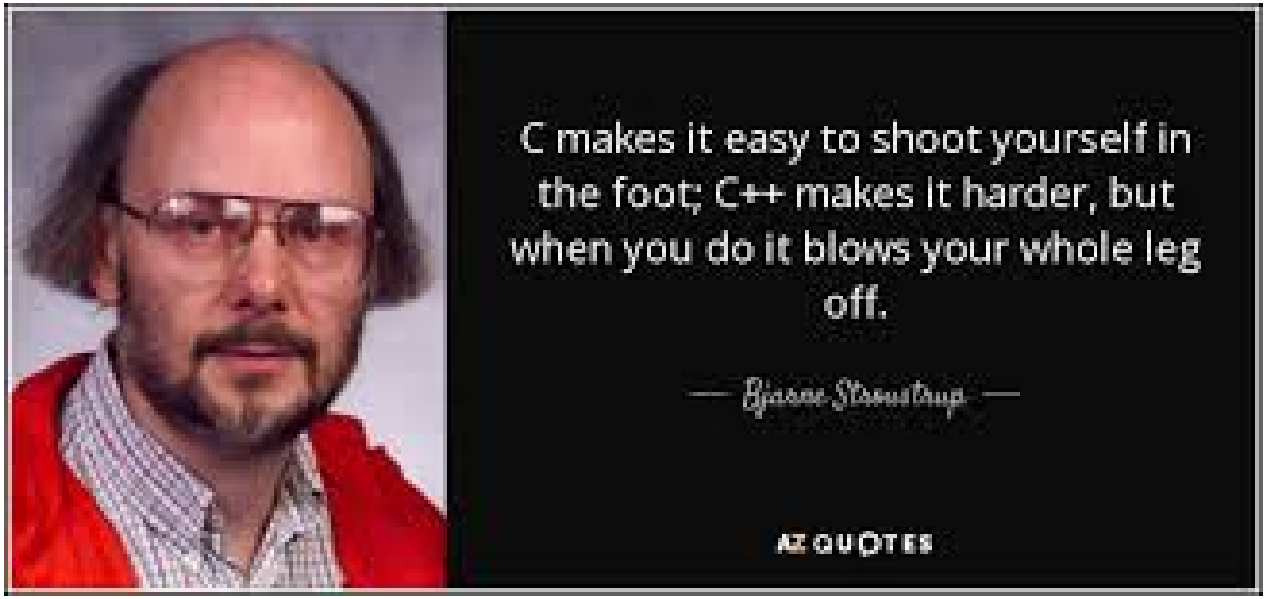
Danish computer scientist Bjarne Stroustrup began work on “C with Classes” in 1979. Working in AT&T Bell Labs, Stroustrup set out to enhance the C language, including **classes**, **derived classes**, strong typing, inlining and default arguments.

In 1983, “C with Classes” was renamed to “C++”, adding new features including **virtual functions**, function/operator **overloading**, type-safe free-store memory allocation (new/delete). Furthermore, it included the development of a standalone compiler for C++, Cfront.

In 1985, the first edition of The C++ Programming Language was released, which became the definitive reference for the language, as there was not yet an official standard. The first commercial implementation of C++ was released in October of the same year.

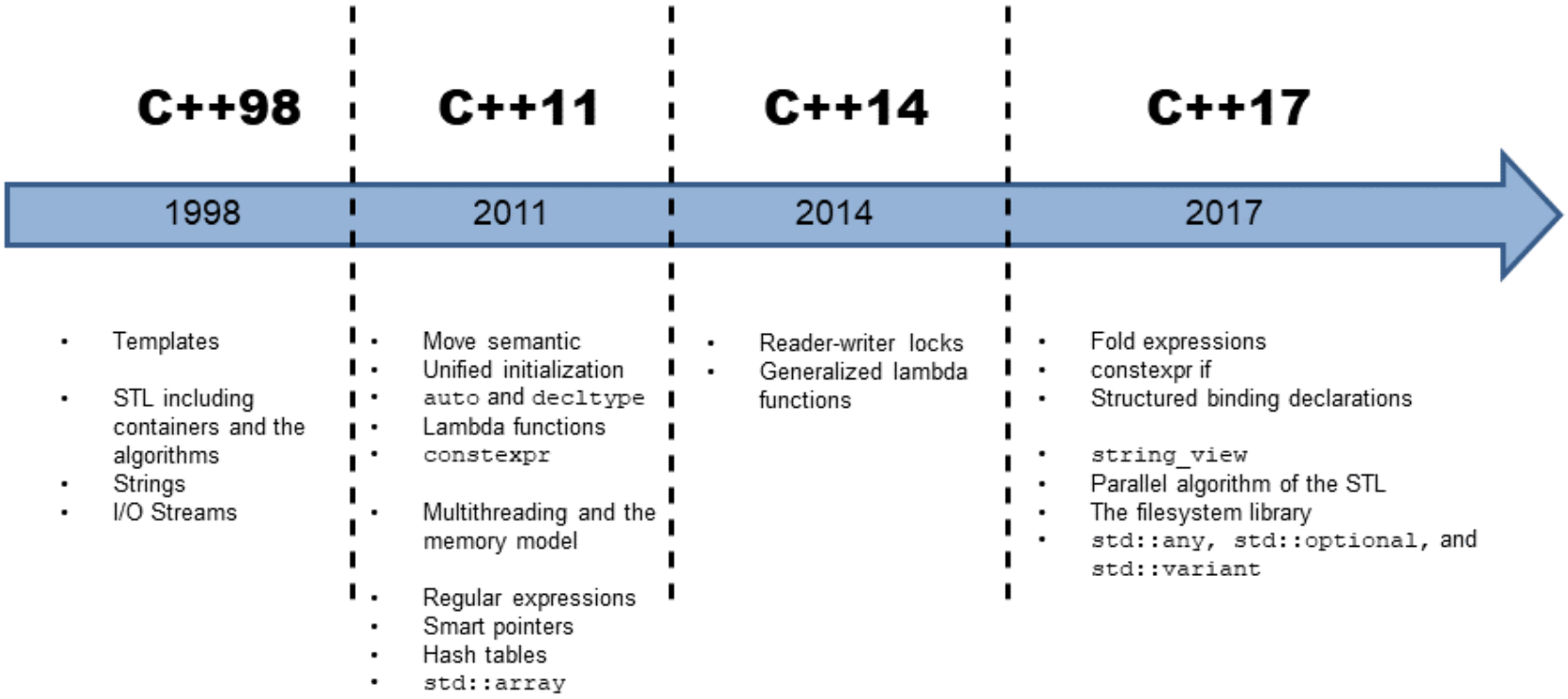
Historical Timeline

Bjarne Stroustrup:



Historical Timeline

Modern developments:



Where do we start?

C++ is an **object-oriented** programming language; C++ support for **classes** enables creation of "objects".

A **class** is **the blueprint** to create a **software object**; it includes:

- Variables (data members);
- Functions (member function).

An **Object** is **an instance** of a **Class**, i.e. an actual variable that belongs to that class.

When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated for that object.

Program with some Class

A **C++ class** has the features of a **C struct**, a collection of different variables (fields), and many additional ones, including:

- Member functions
- Public/private/protected members
- Constructor/destructor
- More advanced features: Inheritance, Virtualization, Friend classes

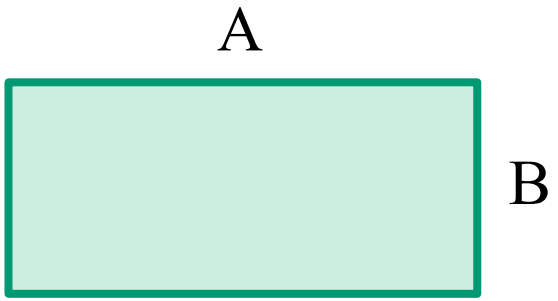
Example: “Rectangle”

Geometric object: “rectangle” is specified by defining:

- Length of two sides (A & B).

This (automatically) gives:

- Area (area = A*B);
- Perimeter (perimeter = 2*(A+B).



SW object: “rectangle” is specified by defining:

- Length of two sides (type *double*).

The object should then be able to (autonomously) compute:

- Area;
- Perimeter.

Example: class “Rectangle”

Solution Explorer

Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

100 %

So... Cl... Pr... Te...

Server Explorer | Toolbox

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Keyword class.
Definition inside braces;
terminate with semicolon.

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Protected members.

100 %

So... Cl... Pr... Te...

Ready Ln 6 Col 1 Ch 1 INS

Server Explorer Toolbox

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Protected members: these are the *inner parts* of the object.

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return a
    double getSide(int sidenum

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Protected members:

100 %

So... Cl... Pr... Te...

Ready | Ln 6 | Col 1 | Ch 1 | INS

Server Explorer | Toolbox

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Protected (private) member variables.

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Protected (private) member function.

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};

#endif
```

Public members

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};

#endif
```

Public member functions.

100 %

Ready | Ln 6 | Col 1 | Ch 1 | INS

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};

#endif
```

Public members: openly accessible to the user (interface).

100 %

So... Cl... Pr... Te...

Ready Ln 6 Col 1 Ch 1 INS

Server Explorer Toolbox

Example: class “Rectangle”

Solution Explorer

- Solution 'EE3093_examples' (1 prc
- EE3093_examples
  - External Dependencies
  - Header Files
    - RectangleExample.h
  - Resource Files
  - Source Files
    - tester.cpp

tester.cpp | RectangleExample.h*

(Global Scope)

```
#ifndef RectangleExample_h
#define RectangleExample_h

#include <iostream>
using namespace std;

class rectangle{
protected:
    double sideA, sideB;
    double area;
    double perimeter;

    void computeArea()
    {
        area=sideA*sideB;
    }

public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};

#endif
```

Public members: openly accessible to the user (interface).

100 %

So... Cl... Pr... Te...

Ready Ln 6 Col 1 Ch 1 INS

Server Explorer Toolbox

Example: class “Rectangle”

ester.cpp × RectangleExample.h

(Global Scope) main()

Now: Let’s **use** the object!

```
#include "RectangleExample.h"

void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;

    cout << endl << endl;

    // set values using the other member function
    testrec2.inputSidesFromKeyboard();
    cout << "rectangle side A is: " << testrec2.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec2.getSide(2) << endl;
    cout << "rectangle area is: " << testrec2.getArea() << endl;

}
```

Example: class “Rectangle”

ester.cpp × RectangleExample.h

(Global Scope) main()

```
#include "RectangleExample.h"

void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;

    cout << endl << endl;

    // set values using the other member function
    testrec2.inputSidesFromKeyboard();
    cout << "rectangle side A is: " << testrec2.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec2.getSide(2) << endl;
    cout << "rectangle area is: " << testrec2.getArea() << endl;

}
```

Include the header file with the class definition.

Example: class “Rectangle”

ester.cpp × RectangleExample.h

(Global Scope) main()

```
#include "RectangleExample.h"

void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;

    cout << endl << endl;

    // set values using the other member function
    testrec2.inputSidesFromKeyboard();
    cout << "rectangle side A is: " << testrec2.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec2.getSide(2) << endl;
    cout << "rectangle area is: " << testrec2.getArea() << endl;

}
```

Language-defined variable type **double**

User-defined variable type **rectangle**

Example: class "Rectangle"

ester.cpp × RectangleExample.h

(Global Scope) main()

```
#include "RectangleExample.h"

void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;
```

"input" functions allow the user *supervised input access* to member variables

```
    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;

    cout << endl << endl;

    // set values using the other member function
    testrec2.inputSidesFromKeyboard();
    cout << "rectangle side A is: " << testrec2.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec2.getSide(2) << endl;
    cout << "rectangle area is: " << testrec2.getArea() << endl;

}
```

Example: class “Rectangle”

ester.cpp × RectangleExample.h

(Global Scope) main()

```
#include "RectangleExample.h"

void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;

    cout << endl << endl;

    // set values using the other member function
    testrec2.inputSidesFromKeyboard();
    cout << "rectangle side A is: " << testrec2.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec2.getSide(2) << endl;
    cout << "rectangle area is: " << testrec2.getArea() << endl;

}
```

“get” functions allow the user *read access* to member variables (not allowed to change)

Example: class “Rectangle”

```
void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // this is not allowed: variable member "sideA" is NOT public; the user cannot access it directly
    testrec1.sideA=10;

    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle 1: " << endl;
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;
    //cout << "rectangle perimeter is: " << testrec1.getPerimeter() << endl;
    cout << " Count of rectangle objects is: " << testrec1.getObjectCount() << endl;

    cout << endl << endl;
```

Variables that are **NOT public** cannot be accessed directly (e.g. in a line of code inside main): these are **only accessible** from **within** a **member function**

Example: class “Rectangle”

```
void main()
{
    // these are two variables of the "standard" type "double"
    double test_sideA, test_sideB;
    test_sideA=10;
    test_sideB=15.5;

    // these are two instantiations (objects) of the class rectangle
    // i.e. two variables of the user-defined type "rectangle"
    rectangle testrec1, testrec2;

    // this is not allowed: function member is NOT public; the user cannot access it directly
    testrec1.computeArea();
```

Similarly for member functions that are **NOT public**

```
    // set values using one member function
    testrec1.inputSides(test_sideA, test_sideB);
    cout << "rectangle 1: " << endl;
    cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
    cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
    cout << "rectangle area is: " << testrec1.getArea() << endl;
    //cout << "rectangle perimeter is: " << testrec1.getPerimeter() << endl;
    cout << " Count of rectangle objects is: " << testrec1.getObjectCount() << endl;

    cout << endl << endl;
```

Any question?

Example: class “Rectangle”

tester.cpp | RectangleExample.h

rectangle | getSide(int sidenum)

```
public:
    double getArea() {return area;}
    double getSide(int sidenum)
    {
        double out;
        switch(sidenum){
            case 1:
                out=sideA;
                break;
            case 2:
                out=sideB;
                break;
            default:
                cout << "Error in getSide(): Incorrect sidenum values" << endl;
                out=-1;
        }

        return out;
    }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Closer look at the implementation

Example: class “Rectangle”

tester.cpp | RectangleExample.h

rectangle | getSide(int sidenum)

```
public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB)
    {
        if(in_sideA>0 && in_sideB>0)
        {
            sideA=in_sideA;
            sideB=in_side
            computeArea();
            computePerimeter();
        }
        else
            cout << "Error in inputSides(): Incorrect input values" << endl;

    }

    void inputSidesFromKeyboard() { ... }
};


#endif
```

Closer look at the implementation

Example: class “Rectangle”

tester.cpp | RectangleExample.h

rectangle | getSide(int sidenum)

```
public:
    double getArea() {return area;}
    double getSide(int sidenum) { ... }

    void inputSides(double in_sideA, double in_sideB) { ... }

    void inputSidesFromKeyboard()
    {
        double in_sideA, in_sideB;

        cout << "Please enter dimension of side A of the Rectangle (then hit ENTER)" << endl;
        cin >> in_sideA;

        cout << "Please enter dimension of side B of the Rectangle (then hit ENTER)" << endl;
        cin >> in_sideB;

        inputSides(in_sideA, in_sideB);

    }
};
```

Closer look at the implementation

Any question?

Relevant topics to ask questions:

- C++ Objects; member variables and functions
- C structs (difference with Object)
- C functions (difference with Object member functions)

Example: class “Rectangle”

Improve the implementation:

- Compute perimeter
- Return perimeter to the user

Notes on using the **iostream** library objects in the following appendix

Appendix: iostream

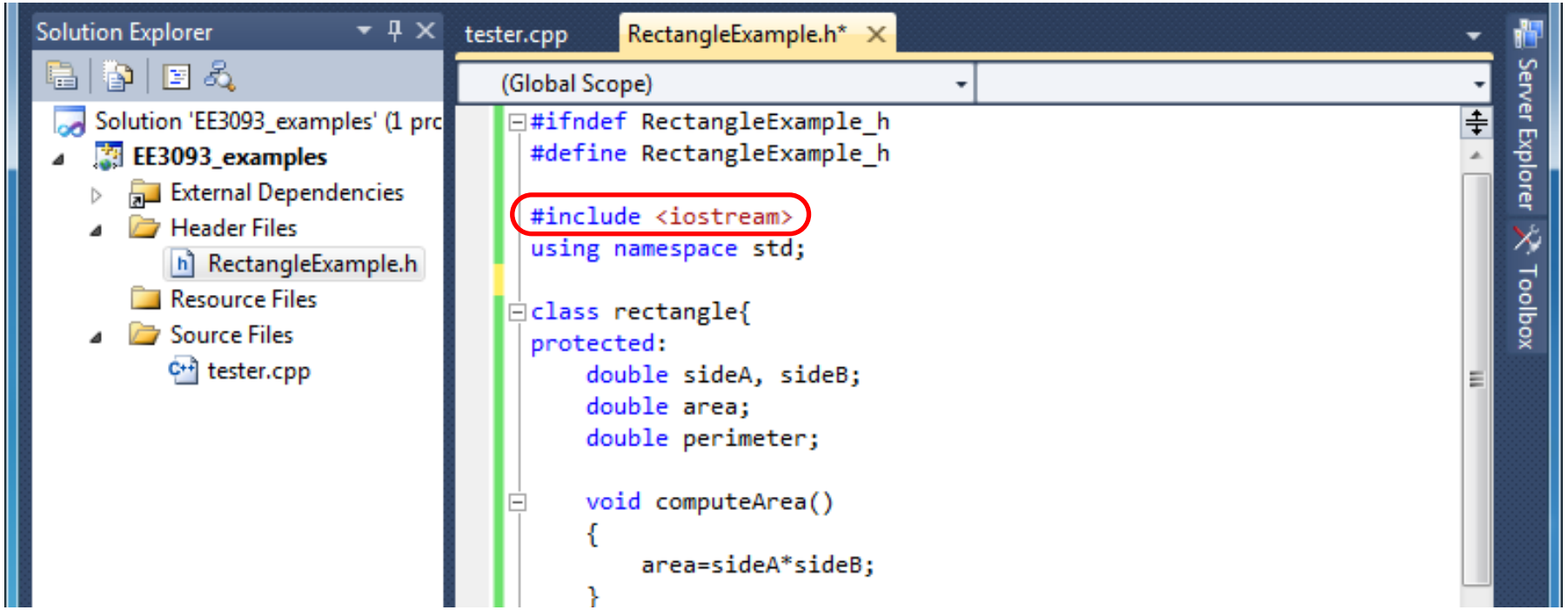
Equivalent to the input/output functionality provided in C by stdio.h, **iostream** provides basic input/output services for C++ programs.

The main **objects** defined in iostream are **cin** and **cout** that allow users to send/receive data to/from standard streams input/output.

Appendix: iostream

How to use it:

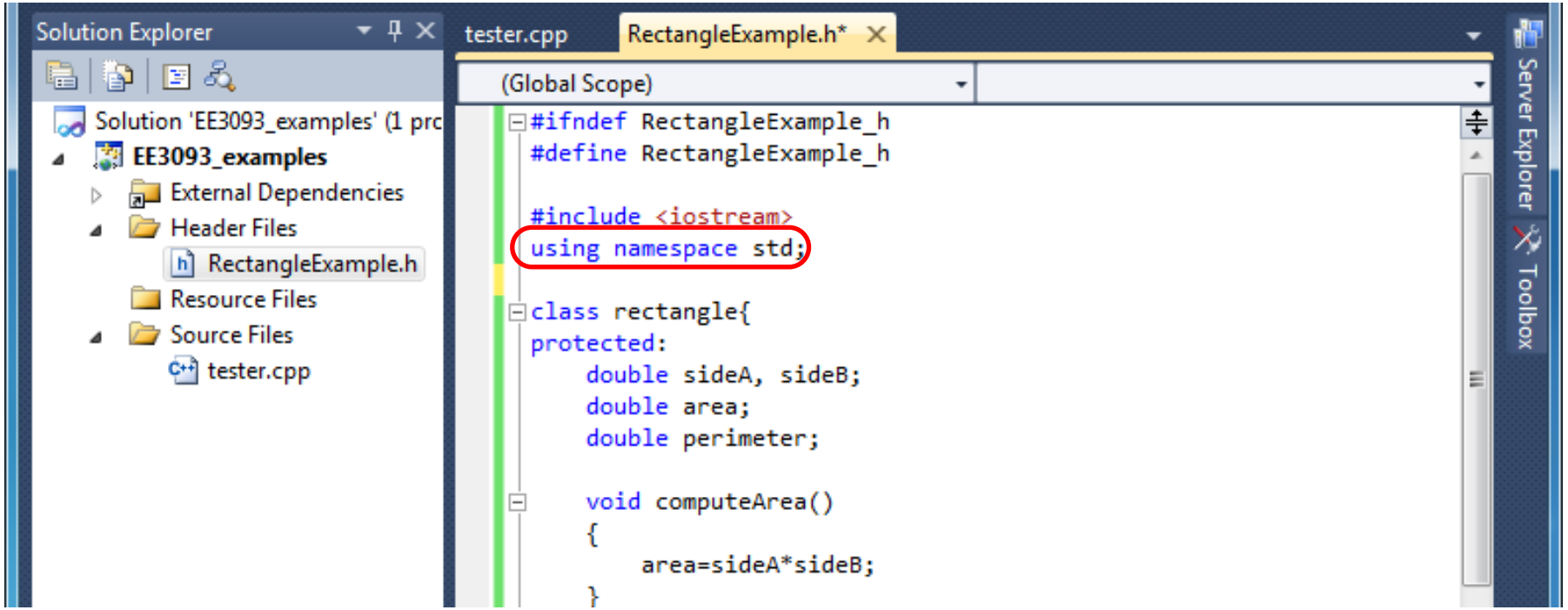
- Include the library (header file)
  - **#include** to tell the compiler to search and include the file.
  - angle brackets < > around the filename tell the compiler to search the file in pre-designated directories.
  - Note: C++ Standard Library Files do not have the **.h** name extension that is now only used by user-defined header files.



Appendix: iostream

How to use it:

- Include the library (header file)
- Optionally, use the namespace **std** defined in this Standard Library
  - This means you can refer to the object name **cin,** which is defined within **std,** simply as **cin**, as opposed to **std::cin**. The same applies to other class names, e.g. **cout**, **endl**, defined in **std**.



Appendix: iostream

How to use it:

- Include the library (header file)
- Optionally, use the namespace **std** defined in this Standard Library
- Output: print to screen using **cout**
  - make **cout** receive a string **"like this sequence of words within inverted commas"** using the operator << ; **cout** will then print that string to screen;
  - make **cout** receive a numerical value, be it a variable or the output of a function; **cout** will then print that value to screen (with proper formatting);
  - make **cout** receive **endl** to move to a new line when printing.

```
// set values using one member function
testrec1.inputSides(test_sideA, test_sideB);
cout << "rectangle side A is: " << testrec1.getSide(1) << endl;
cout << "rectangle side B is: " << testrec1.getSide(2) << endl;
cout << "rectangle area is: " << testrec1.getArea() << endl;
```

Appendix: iostream

How to use it:

- Include the library (header file)
- Optionally, use the namespace **std** defined in this Standard Library
- Output: print to screen using **cout**
- Input: read to keyboard **cin**
  - **cin** prompts the user to input data (into the command window, via keyboard);
  - make cin send the data (it received from keyboard) to a variable the operator >> ;
  - **cout** passes the value (properly formatted) to that variable.

```
void inputSidesFromKeyboard()
{
    double in_sideA, in_sideB;

    cout << "Please enter dimension of side A of the Rectangle (then hit ENTER)" << endl;
    cin >> in_sideA;

    cout << "Please enter dimension of side B of the Rectangle (then hit ENTER)" << endl;
    cin >> in_sideB;

    inputSides(in_sideA, in_sideB);

}
```