

The C++ Language

- ❖ Minimalist language with few *Keywords*
- ❖ Extensive use of *Library Functions*
- ❖ C++ is *Case Sensitive*
- ❖ All C++ keywords *Lower Case*
- ❖ C++ program is sequence of statements
- ❖ Comments syntax:
 - `/* ... */` for C
 - `// ...` for C++ comment to end of line
- ❖ All statements terminate with semicolon;
- ❖ Ignores *White Space* = space, tab, new line

Copyright © 2005 R.M. Laurie 1

The C++ Source Code File

Blank.cpp Source File - This program does nothing

```

1. int main()           //Program start
2. {                   //Program body start
3.     //STATEMENTS GO HERE
4.     return 0;       //Program ends
5. }                   //Program body end
    
```

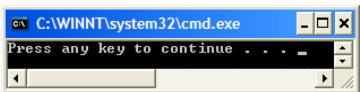
Source File
Blank.cpp

→

Compile C++

Target CPU & Target OS

Executable
Object File
Blank.exe



Copyright © 2005 R.M. Laurie 2

Program Output

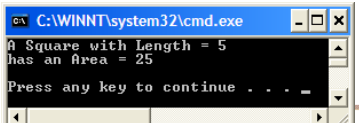
- ❖ C++ does not have output command
- ❖ However, *Library Functions* available
 - `#include <iostream>`
 - `using namespace std;`
 - ◆ `#include` must be before `int main()`
 - ◆ `using namespace` after `#includes`
- ❖ `cout` represents the console output device
 - ◆ The *Output Operator* `<<` is placed between `cout` and the output ASCII string or variable
 - ◆ `endl` represents sending a *new line* with `cout`

Copyright © 2005 R.M. Laurie 3

SquareOutput.cpp

```

1.  /*****
2.  * PROGRAM: SquareOutput.cpp
3.  *****/
4.  #include <iostream>           // Library functions
5.  using namespace std;        // Required Standard C++
6.  int main()                   // Start of program
7.  {
8.     cout << "A Square with Length = ";
9.     cout << 5 << endl;
10.    cout << "has an Area = 25" << endl << endl;
11.    return 0;                 // End of program
12. }
    
```



Copyright © 2005 R.M. Laurie 4

Arithmetic Operators

❖ Operators used to perform arithmetic operations on variables and values

❖ Order-of-operation defined by *precedence*

()	Parenthesis	[highest]
* /	Multiplication and Division	
+ -	Addition and Subtraction	
=	Assignment	[lowest]

Table 7-2 page 280 has all C++ Operator Precedence

❖ Insert parenthesis if order-of-operation is not apparent in source code

❖ Assignment operator =

- ◆ Evaluate expression on right and assign result to left
- ◆ Has lowest precedence

Copyright © 2005 R.M. Laurie 5

Arithmetic Operator Examples

1. nLength = 7;
2. nSquarePerimeter = nLength * 4;
3. nSquareArea = nLength * nLength;
4. nRectangleArea = nLength * nWidth;
5. nRectanglePerimeter = nLength*2+nWidth*2;
6. cout << nSquareArea;
7. cout << "Area = " << nRectangleArea;
8. cout << "Area = " << nLength * nWidth;
9. cout<<"Perimeter = "<<nLength*2+nWidth*2;
10. nScore = 93;
11. nScoreCount = nScoreCount + 1;
12. nTotalScore = nTotalScore + nScore;
13. fAvgScore = nTotalScore / nScoreCount;
14. cout << "Average Score = " << fAvgScore;

Copyright © 2005 R.M. Laurie 6

SquareVariable.cpp

```

1.  /*****
2.  * PROGRAM: SquareVariable.cpp
3.  *****/
4.  #include <iostream>
5.  using namespace std;
6.  int main()
7.  {
8.  int nArea;           // Variable declaration
9.  int nLength;        // Variable declaration
10. nLength = 6;        // Variable initialization
11. nArea = nLength * nLength;
12. cout << "A Square with Length = ";
13. cout << nLength << endl;
14. cout << "has an Area = " << nArea << endl;
15. return 0;
16. }
    
```

A Square with Length = 6
has an Area = 36
Press any key to continue . . .

Copyright © 2005 R.M. Laurie 7

Declaration Statements

❖ Variable is a container for data

nLength 6

❖ *Declaration Statements* allocate memory to hold a *Variables*

- ◆ Specifies *Data Type* `int nLength;`
- ◆ Followed by *Identifier*
- ◆ Terminate C++ statement with semicolon ;
- ◆ Optionally, may declare several variables of the same data type (comma separated)
`int nLength, nArea;`
- ◆ Optionally, may initialize variables in declaration statement
`int nLength = 5;`

Copyright © 2005 R.M. Laurie 8

Data Type Declaration

- ❖ **int** Reserves one *word* of RAM memory to represent a signed whole number:
 - ◆ 16 bits for Turbo C++ =
 - ◆ Signed integer 32,767 to -32,768 (Default)
 - ◆ 32 bits for Borland C++
 - ◆ Signed integer $\pm 2,147,483,648$ (Default)
 - ◆ Example:
 - ◆ `int nLength;`
 - ◆ `int nHeight = 100, nWidth = 50;`
- ❖ **float** Reserves 32 bits of RAM memory
 - ◆ Represents floating point values in range: $\pm 3.402823466 \times 10^{\pm 38}$
 - ◆ Example:
 - ◆ `float fWeight = 2.35;`

Copyright © 2005 R.M. Laurie 9

C++ Identifier Requirements

- ❖ Name of *variable* or *function*
 - ◆ Max Length ≥ 32 Characters
- ❖ C++ Identifier Naming Restrictions:
 - ◆ A to Z a to z 0 to 9 _
 - ◆ No spaces allowed
 - ◆ Must begin with letter or underscore
 - ◆ Case sensitive
 - ◆ Not a reserved word (See pp. 39)
- ❖ Use an Identifier naming convention

Copyright © 2005 R.M. Laurie 10

Identifier Naming Conventions

- ❖ Use descriptive names (self documenting)
- ❖ Variables begins with lower case letter(s) to indicate data type and remaining name in title case
 - ◆ `int nLength = 5, nSquareArea;`
 - ◆ `float fLength=2.5, fWidth= 4, fRectangleArea;`
- ❖ Constants use ALL CAPITAL letters.
- ❖ Library functions use lower case letters
- ❖ User functions use title case no spaces

Copyright © 2005 R.M. Laurie 11

SquareVariableShort.cpp

```

1. /*****
2. * PROGRAM: SquareVariableShort.cpp
3. * More compact version of previous program
4. *****/
5. #include <iostream>
6. using namespace std;
7. int main()
8. {
9.     int nLength = 6; // Declare and Intialize
10.    cout << "A Square with Length = " << nLength
11.        << " inches" << endl;
12.    cout << "has an Area = " << nLength * nLength
13.        << " square inches" << endl;
14.    return 0;
15.}
    
```

```

A Square with Length = 6
has an Area = 36
Press any key to continue . . .
    
```

Copyright © 2005 R.M. Laurie 12

Program Input

- ❖ The preceding programs would be far more useful if they had the flexibility to prompt and receive input values from the user
- ❖ The `<iostream>` library functions also provides user input capability
- ❖ `cin` represents the console input device
 - ◆ The *Input Operator* `>>` is placed between `cin` and the variable to where the input data will be stored
 - ◆ Entered text is converted to the variable data type
- ❖ Examples:


```
cin >> nLength;
cin >> fLength;
cin >> nWeight;
```

Copyright © 2005 R.M. Laurie 13

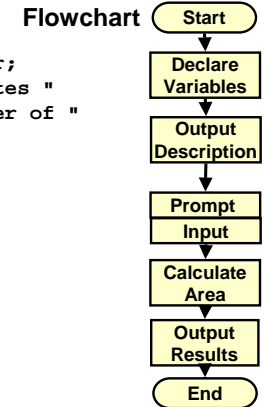
Input Example

This program calculates the area and perimeter of a square
 Enter Length: 20
 Area = 400 Perimeter = 80
 Press any key to continue . . .

```
#include <iostream>
using namespace std;
int main()
{
    int nLength, nArea, nPerimeter;
    cout << "This program calculates "
         << "the area and perimeter of "
         << "a square" << endl;
    cout << "Enter Length: ";
    cin >> nLength;

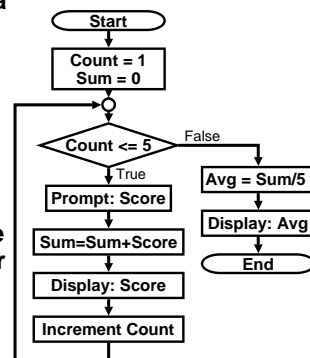
    nArea = nLength * nLength;
    nPerimeter = nLength * 4;

    cout << "Area = " << nArea
         << " Perimeter = "
         << nPerimeter << endl;
    return 0;
}
```



Flowcharting

- ❖ A program is little more than a sequence of events
- ❖ A Flowchart is a graphical model used to represent a sequence of events
- ❖ Flowcharting is a useful tool for designing a computer program
- ❖ A Flowchart can represent the sequence of events that occur in a program before code is written
- ❖ Flowchart can be verified using known test data
- ❖ PowerPoint can be used to create a flowchart



Copyright © 2005 R.M. Laurie 15

Assignment 3: Programming

- ❖ Create a C++ program that prompts the user for the base and height of a right triangle and calculates the area and perimeter of a triangle
- ❖ Determine known test data that will be used to verify your algorithms and program results
- ❖ Create a flowchart to model the sequence of events that will occur in the program
- ❖ Determine the mathematical equations
- ❖ Verify using known test data
- ❖ Write the C++ code and debug syntax errors
- ❖ Verify your C++ program works by entering known test data and debug logic errors

Copyright © 2005 R.M. Laurie 16