Lecture_1
Programming Essentials in c++

Dr. Sara Mohamed

Introduction

A computer **program** is…

- A set of instructions for a computer to follow

- Computer **software** is …

The collection of programs used by a computer

- Includes:
  - Editors
  - Translators
  - System Managers

Types of programming languages

- **Low Level Languages**
- **High level Languages**
- **Fourth Generation Languages**

Low-level Languages

- A low-level language, often known as a computer's native language, is a sort of programming language. It is very close to writing actual machine instructions, and it deals with a computer's hardware components like :
- **Machine Language:**The lowest-level programming language consisting of binary digits (0 and 1), directly executed by the CPU.
-

High-level Languages

Common programming languages include …

C C++ Java Pascal Visual Basic FORTRAN

COBOL Lisp Scheme Ada

These high – level languages

- ❑Resemble human languages
- ❑Are designed to be *easy* to read and write
- ❑Use more complicated instructions than the CPU can follow
- ❑Must be translated to zeros and ones for the CPU to execute a program

Compilers

- Translate high-level language to machine language
  - **Source code**
    - the original program in a high-level language
  - **Object code**
    - the translated version in machine language

Software Life Cycle

1. Analysis and specification of the task (problem definition)
2. Design of the software (algorithm design)
3. Implementation (coding)
4. Maintenance and evolution of the system

Introduction to C++

- **Where did C++ come from?**
  - Derived from the C language
  - C was derived from the B language
  - B was derived from the BCPL language

C++ History

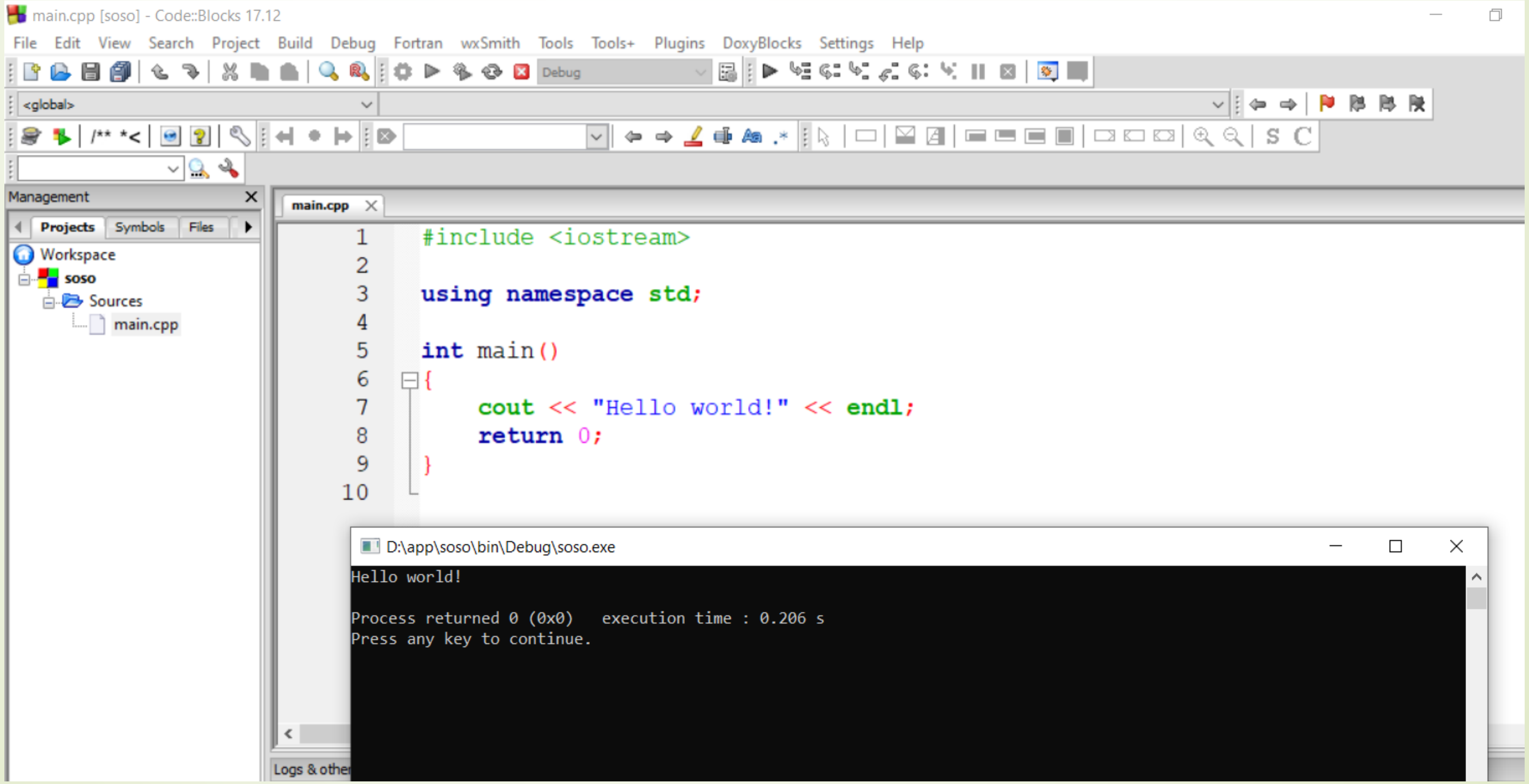
- **C developed by Dennis Ritchie at AT&T Bell Labs in the 1970s.**
  - Used to maintain UNIX systems
  - Many commercial applications written in c
- **C++ developed by Bjarne at AT&T Bell Labs in the 1980s.**
  - Overcame several shortcomings of C
  - Incorporated object-oriented programming
  - C remains a subset of C++

Some applications were created with c++

- Google Chrome Mozilla Firefox ‘
- Microsoft Office
- Adobe Photoshop‘
- Adobe Illustrator
- Microsoft windows Xp
- Vista..

A Sample C++ Program

- A simple C++ program begins this way.
- Code::Blocks is a **free**, **open-source Integrated Development Environment (IDE)** mainly used for programming in **C**, **C++**, **and Fortran**.
-

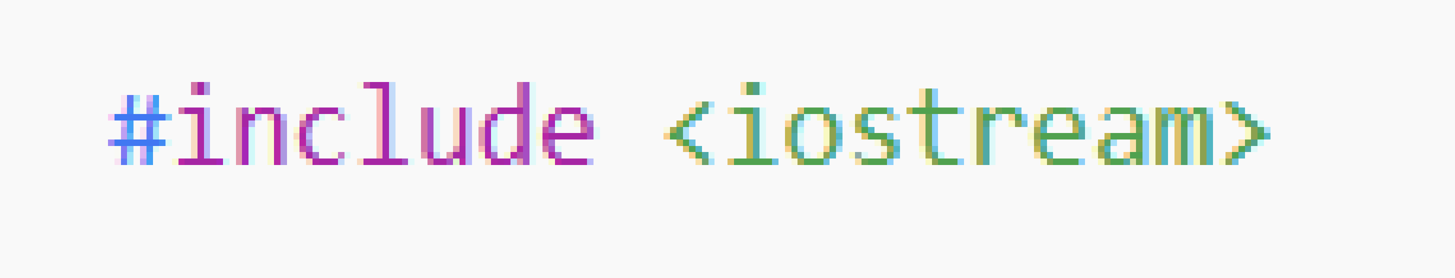


main.cpp [soso] - Code::Blocks 17.12
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
Management
Projects Symbols Files
Workspace
soso
Sources
main.cpp
main.cpp
D:\app\soso\bin\Debug\soso.exe
Logs & other

```cpp
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    return 0;
}

```

```
Hello world!

Process returned 0 (0x0)   execution time : 0.206 s
Press any key to continue.
```

Explanation of code



```
#include <iostream>
```

#Includes <iostream>:#include is known as a preprocessor directive, which is used to load files.< > indicate the start and end of file name to be loaded.. you can use " " quotes too instead of <>.

In this case, iostream is a file containing code for input/output operations.

The **iostream** library provides functionality for:

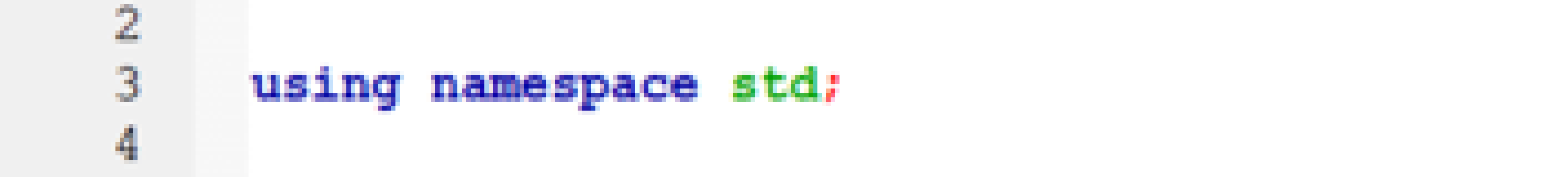
cin → standard input (keyboard)

cout → standard output (screen)

cerr → standard error (error messages)

clog → logging messages

Without including <iostream>, you cannot use these standard input/output objects.



```
using namespace std;
```

- Tells the compiler to use names in iostream in a “standard” way
- std::cout and std::cin
- **Std:: scope resolution**
- **Namespace:** A namespace is a declarative region that provides a scope to the identifiers (the names of types, functions, variables, etc) inside it.
- **Namespaces** are used to organize code into logical groups and to prevent name collisions that can occur especially when your code base includes multiple libraries.

Program Layout

- To begin the main function of the program

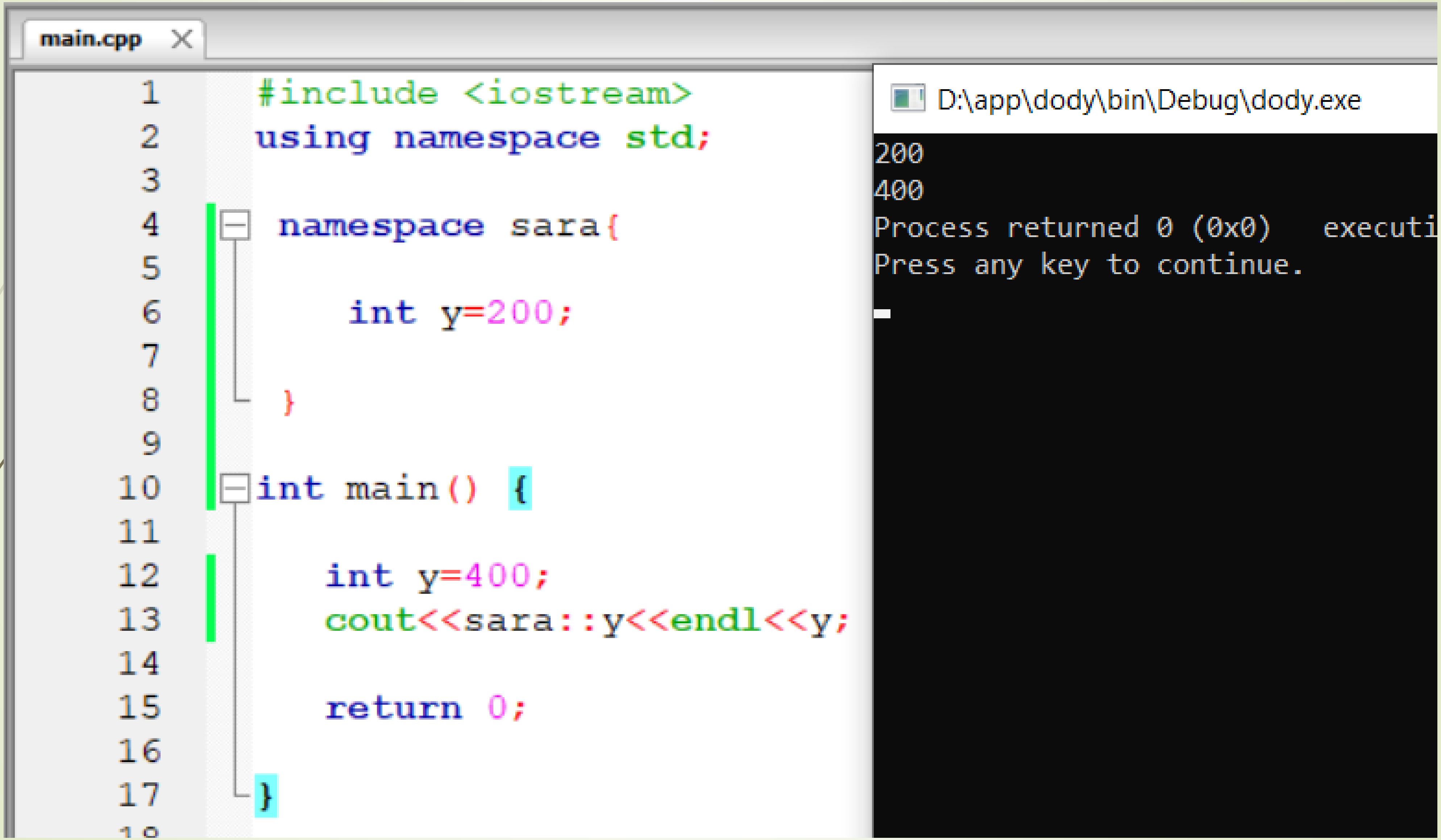
```
int main()
     {
```

- To end the main function

```
         return 0;
       }
```

- Main function ends with a return statement.

Namespace Example



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

 namespace sara{

     int y=200;

 }

int main() {

    int y=400;
    cout<<sara::y<<endl<<y;

    return 0;

}
```

```
200
400
Process returned 0 (0x0)   executi
Press any key to continue.
```

Input and Output

Cout(console output)

Program statement

cout << “Press return after entering a number.\n”;

- cout (see-out) used for output to the monitor
- Think of cout as a name for the monitor
  - “<<“ points to where the data is to end up
- ‘\n’ causes a new line to be started on the monitor

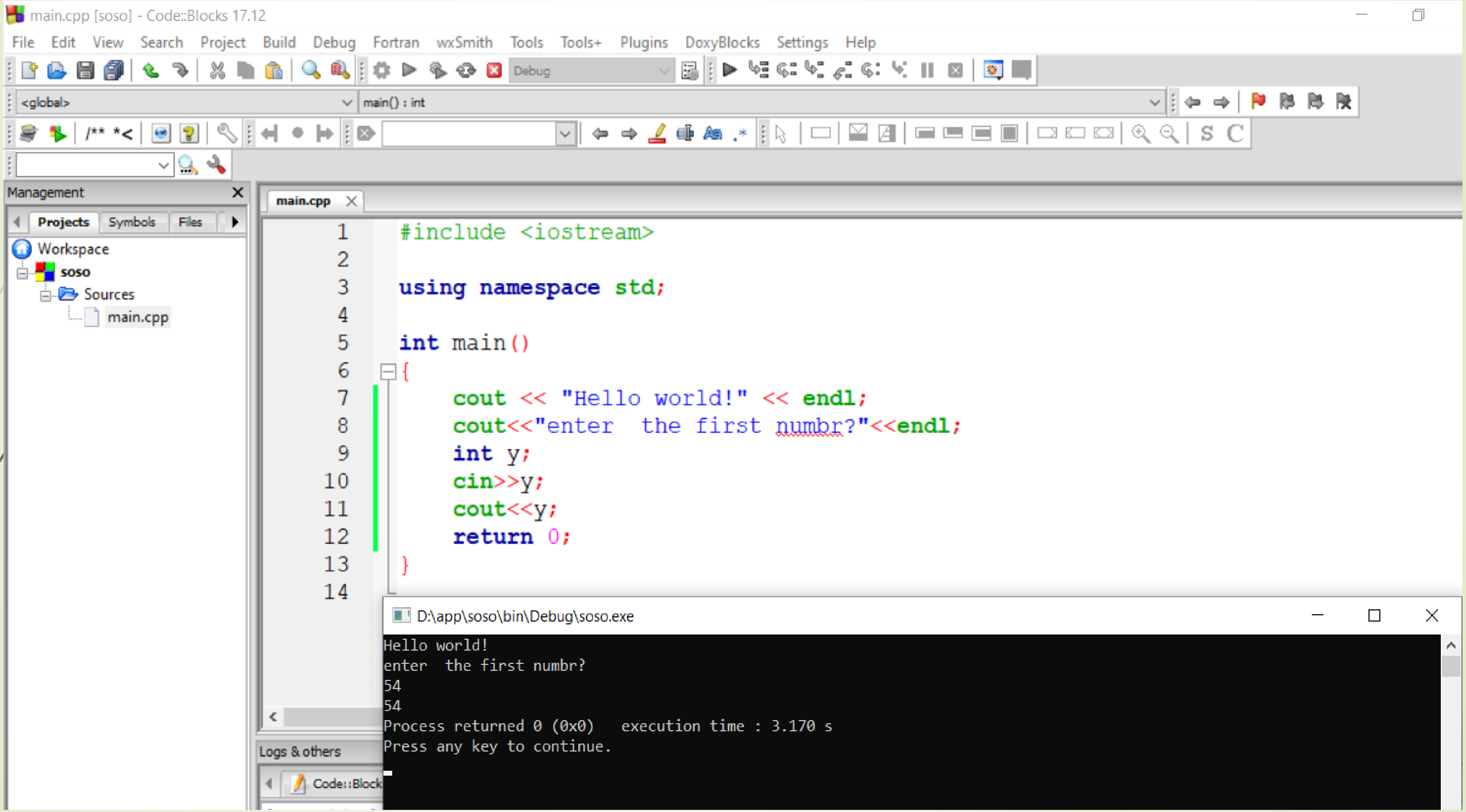
Cin(console input)

- Program statement:
  cin >> var;

✓ cin is a predefined variable that reads data from the keyboard

✓ with the extraction operator (>>).

Example



```cpp
#include <iostream>

using namespace std;

int main()
{
    cout << "Hello world!" << endl;
    cout<<"enter  the first numbr?"<<endl;
    int y;
    cin>>y;
    cout<<y;
    return 0;
}
```

```
Hello world!
enter  the first numbr?
54
54
Process returned 0 (0x0)   execution time : 3.170 s
Press any key to continue.
```

Data and variables types

- **Variables** are used to store information to be referenced and manipulated in a computer program.
- A **variable** in C++ is a **named storage location** in memory that holds a value which can change during program execution.

Think of it like a container with a label.

Variables and Assignments

- Variables are like small blackboards
  - ❑We can write a number on them
  - ❑We can change the number
  - ❑We can erase the number
- C++ variables are names for memory locations
  - ❑We can write a value in them
  - ❑We can change the value stored there
  - ❑We cannot erase the memory location
  - ❑Some value is always there

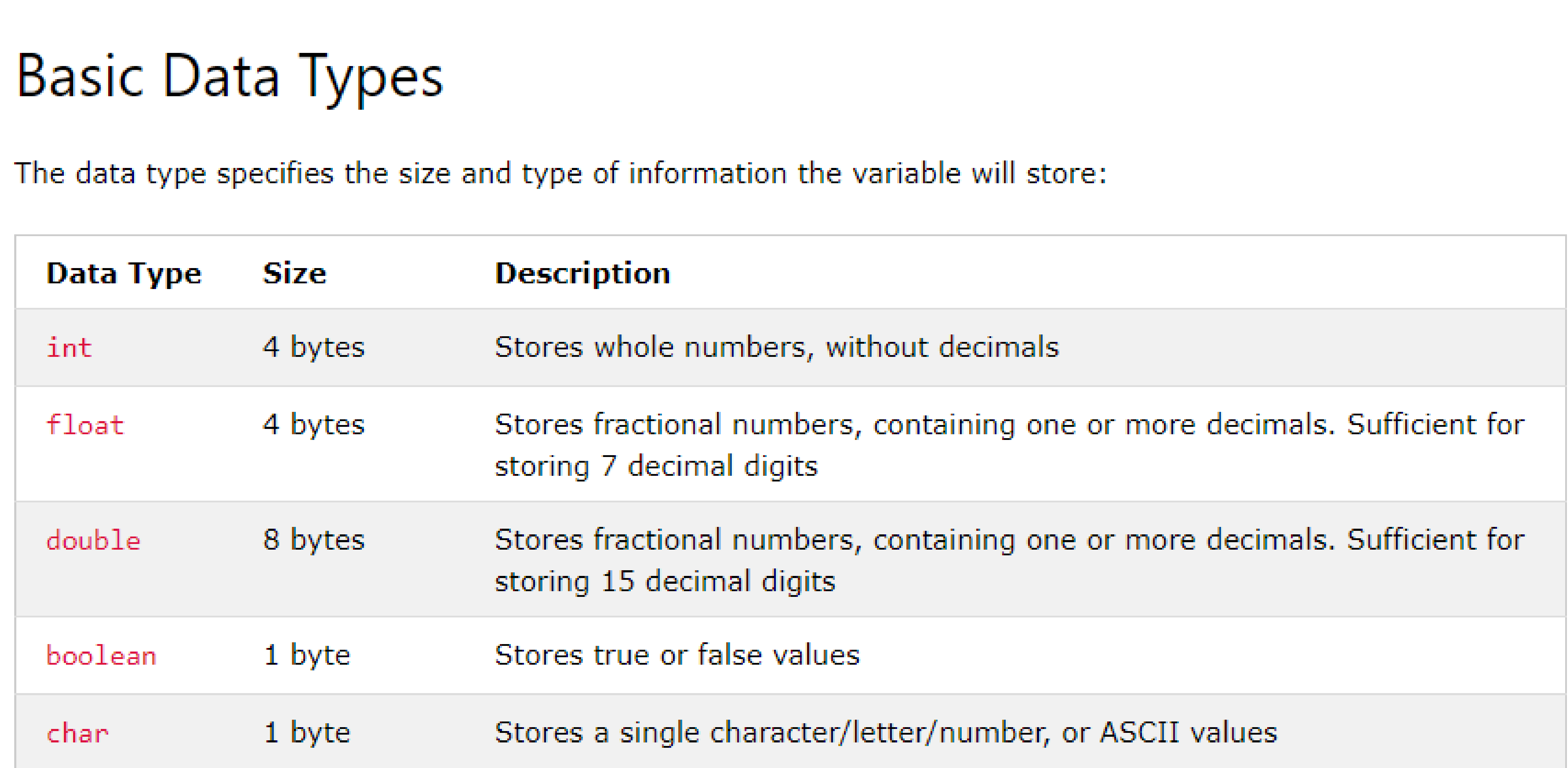
Identifiers

- Variables names are called identifiers
- Choosing variable names
  - Use meaningful names that represent data to be stored
  - First character must be
    - a letter
    - the underscore character
  - Remaining characters must be
    - letters
    - numbers
    - underscore character

Keywords

- Keywords (also called reserved words)
  - Are used by the C++ language
  - Must be used as they are defined in the programming language
  - Cannot be used as identifiers

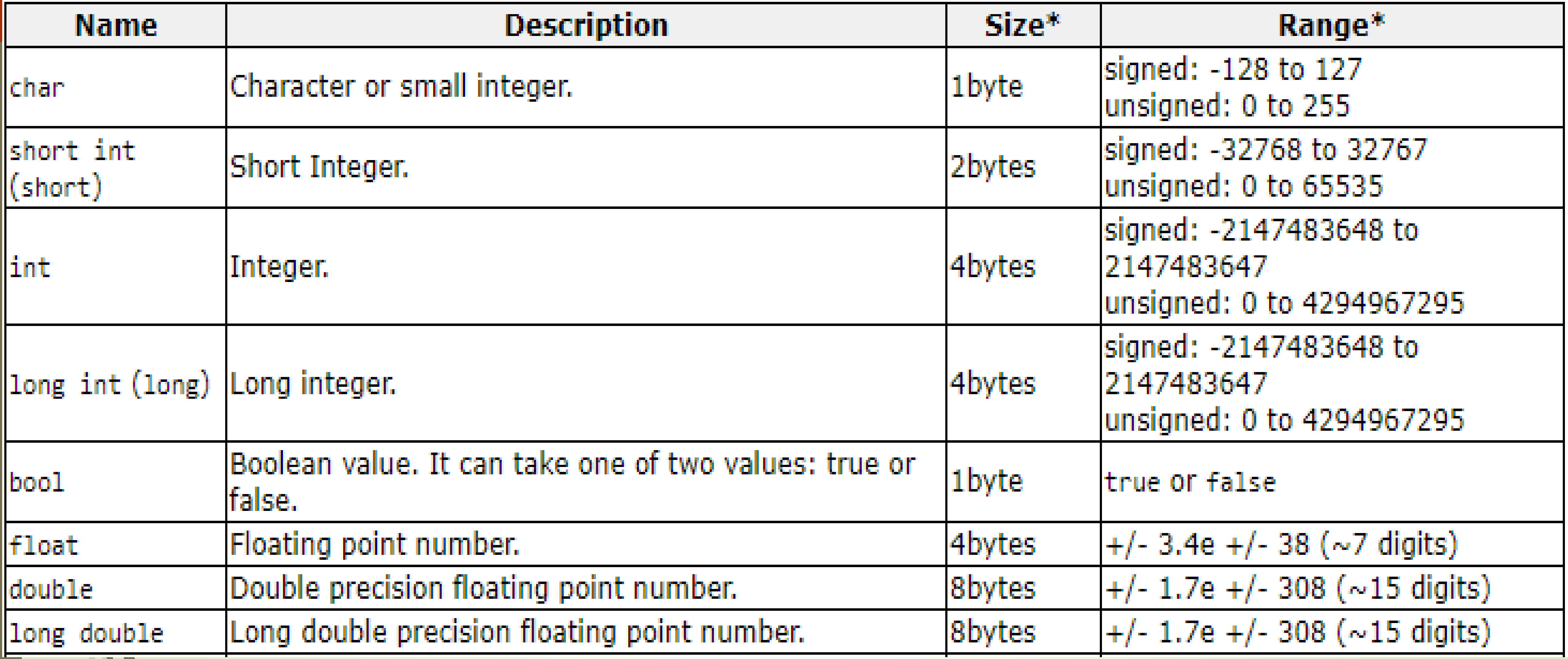
Data types



Basic Data Types

The data type specifies the size and type of information the variable will store:

Data Type	Size	Description
`int`	4 bytes	Stores whole numbers, without decimals
`float`	4 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 7 decimal digits
`double`	8 bytes	Stores fractional numbers, containing one or more decimals. Sufficient for storing 15 decimal digits
`boolean`	1 byte	Stores true or false values
`char`	1 byte	Stores a single character/letter/number, or ASCII values



Name	Description	Size*	Range*
char	Character or small integer.	1byte	signed: -128 to 127 unsigned: 0 to 255
short int (short)	Short Integer.	2bytes	signed: -32768 to 32767 unsigned: 0 to 65535
int	Integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
long int (long)	Long integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
bool	Boolean value. It can take one of two values: true or false.	1byte	true or false
float	Floating point number.	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	Double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	Long double precision floating point number.	8bytes	+/- 1.7e +/- 308 (~15 digits)

Data Types and Expressions

- 2 and 2.0 are not the same number
  - A whole number such as 2 is of type int
  - A real number such as 2.0 is of type double

- Numbers of type int are stored as exact values

- Numbers of type double may be stored as approximate values due to limitations on number of significant digits that can be represented

Integer types

- long or long int (often 4 bytes)
  - Equivalent forms to declare very large integers

```
long  big_total;
long int big_total;
```

- short or short int (often 2 bytes)
  - Equivalent forms to declare smaller integers

```
short small_total;
short int small_total;
```

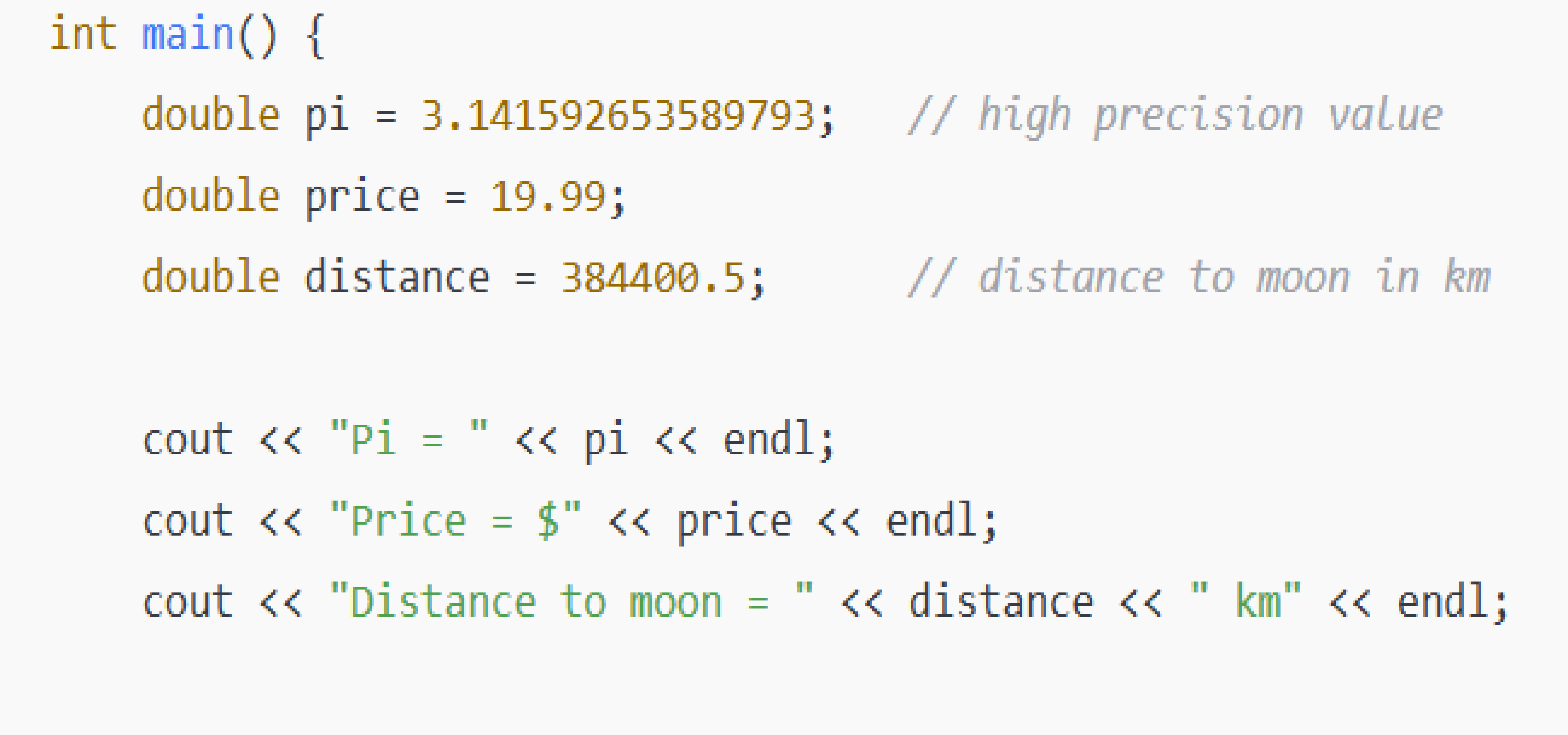
Floating point types

- float (often 4 bytes)
  - Declares floating point numbers with up to 7 significant digits

```
float not_so_big_number;
```

- double
  - ❑ Size: Typically, 8 bytes (64 bits) in most compilers.
  - ❑ Precision: About 15–16 decimal digits of accuracy.
  - ❑ float → ~7 digits

    double → ~15 digits



```
int main() {
    double pi = 3.141592653589793;   // high precision value
    double price = 19.99;
    double distance = 384400.5;      // distance to moon in km

    cout << "Pi = " << pi << endl;
    cout << "Price = $" << price << endl;
    cout << "Distance to moon = " << distance << " km" << endl;
```

Type char

- Computers process character data too char
    - Short for character
    - Can be any single character from the keyboard

- To declare a variable of type char:

```
char letter;
```

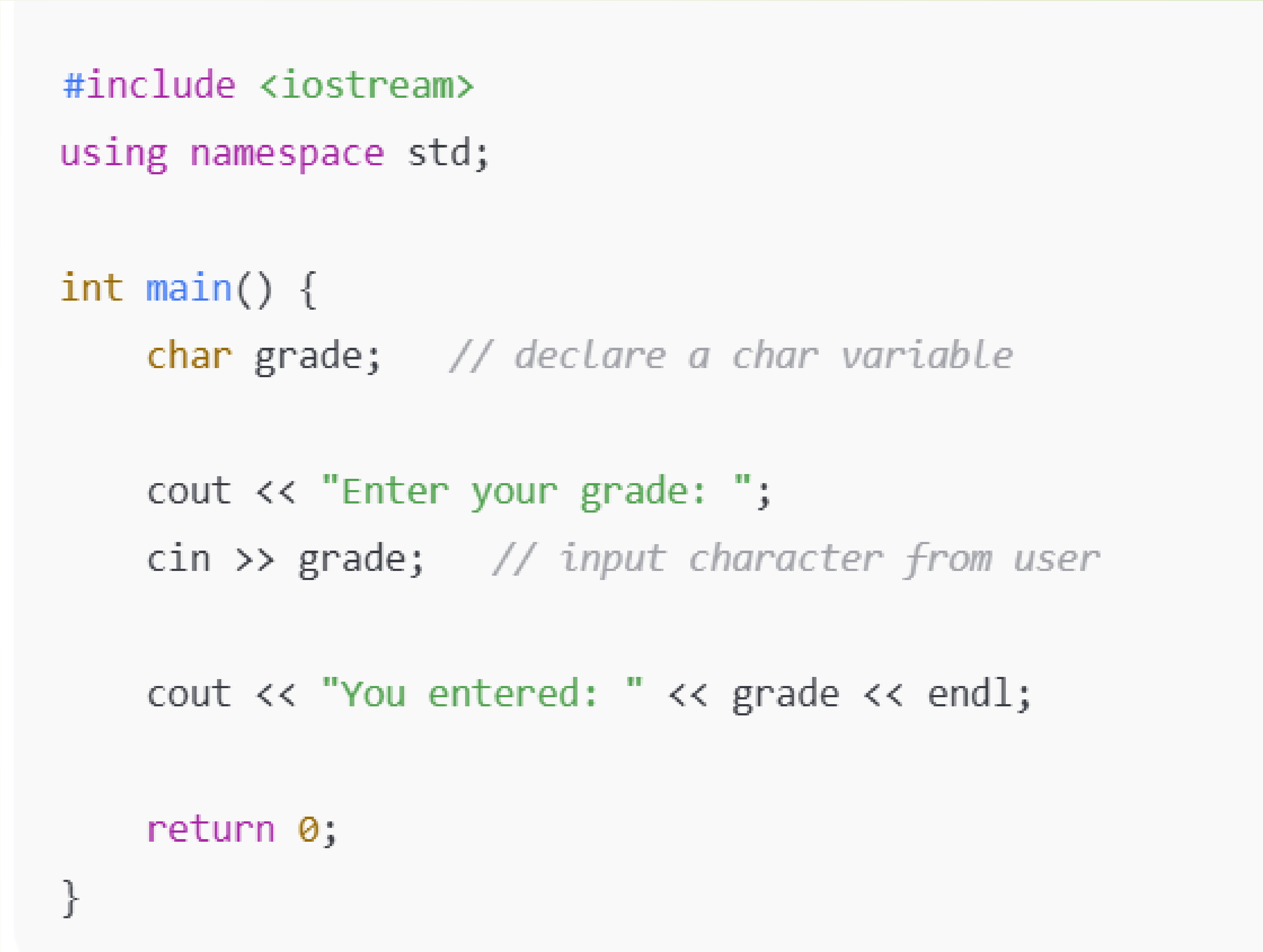
Char data type

- Character constants are enclosed in single quotes

```
char letter = 'a';
```

- Strings of characters, even if only one character is enclosed in double quotes
  - "a" is a string of characters containing one character
  - 'a' is a value of type character

Reading Character Data



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

int main() {
    char grade;   // declare a char variable

    cout << "Enter your grade: ";
    cin >> grade;   // input character from user

    cout << "You entered: " << grade << endl;

    return 0;
}
```

Type bool

- bool is a new addition to C++
  - Short for boolean
  - Boolean values are either true or false
- To declare a variable of type bool:

```
bool old_enough;
```

Type Compatibilities

- In general store values in variables of the same type
  - This is a type mismatch:

```
int int_variable;
int_variable = 2.99;
```

  - If your compiler allows this, int_variable will most likely contain the value 2, not 2.99

int ←→ double (part 1)

- Variables of type double should not be assigned to variables of type int

```
int int_variable;
double double_variable;
double_variable = 2.00;
 int_variable = double_variable;
```

  - If allowed, int_variable contains 2, not 2.00.

int ←→ double (part 2)

- Integer values can normally be stored in variables of type double

```
double double_variable;
double_variable = 2;
```

  - double_variable will contain 2.0

char ← → int

- The following actions are possible but generally not recommended!
- It is possible to store char values in integer variables

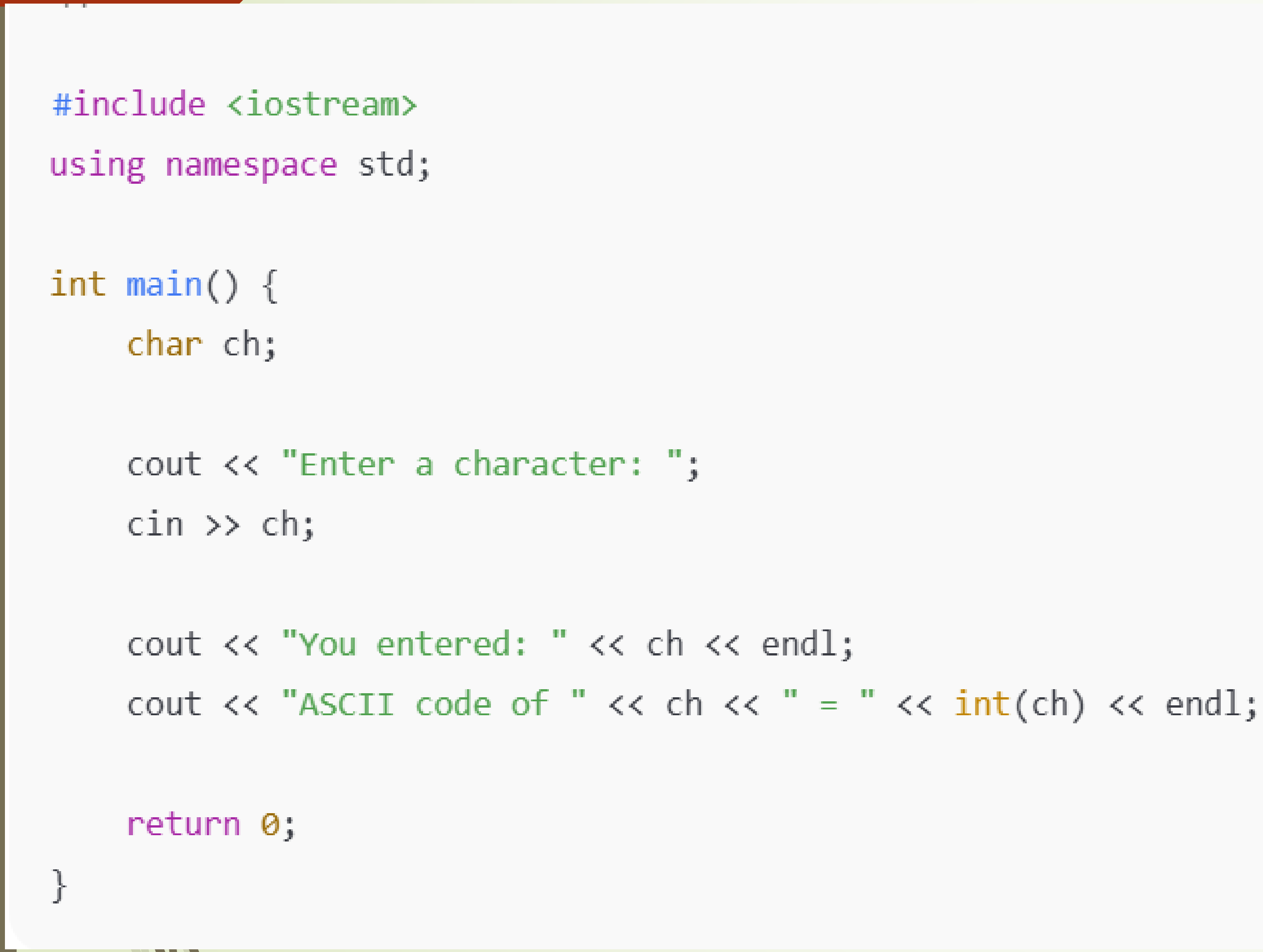
```
        int value = 'A';
```

value will contain an integer representing 'A'

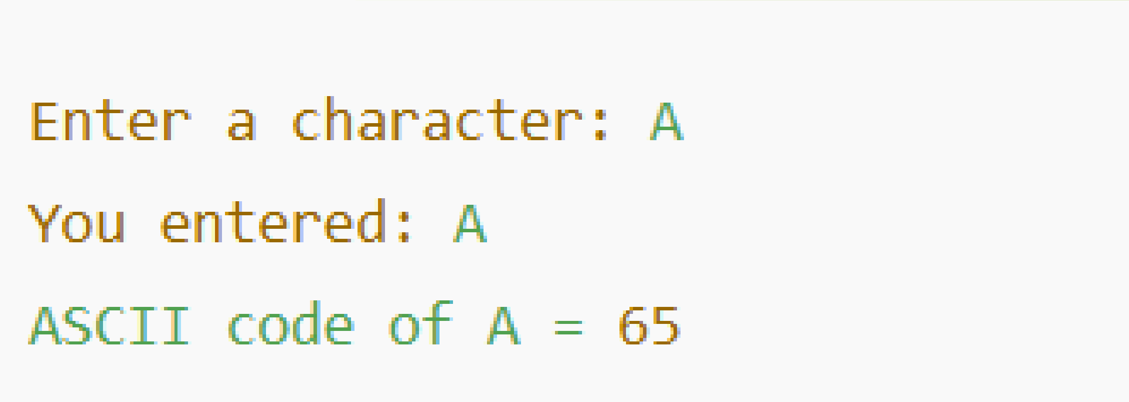
- It is possible to store int values in char variables

```
        char letter = 65;
```

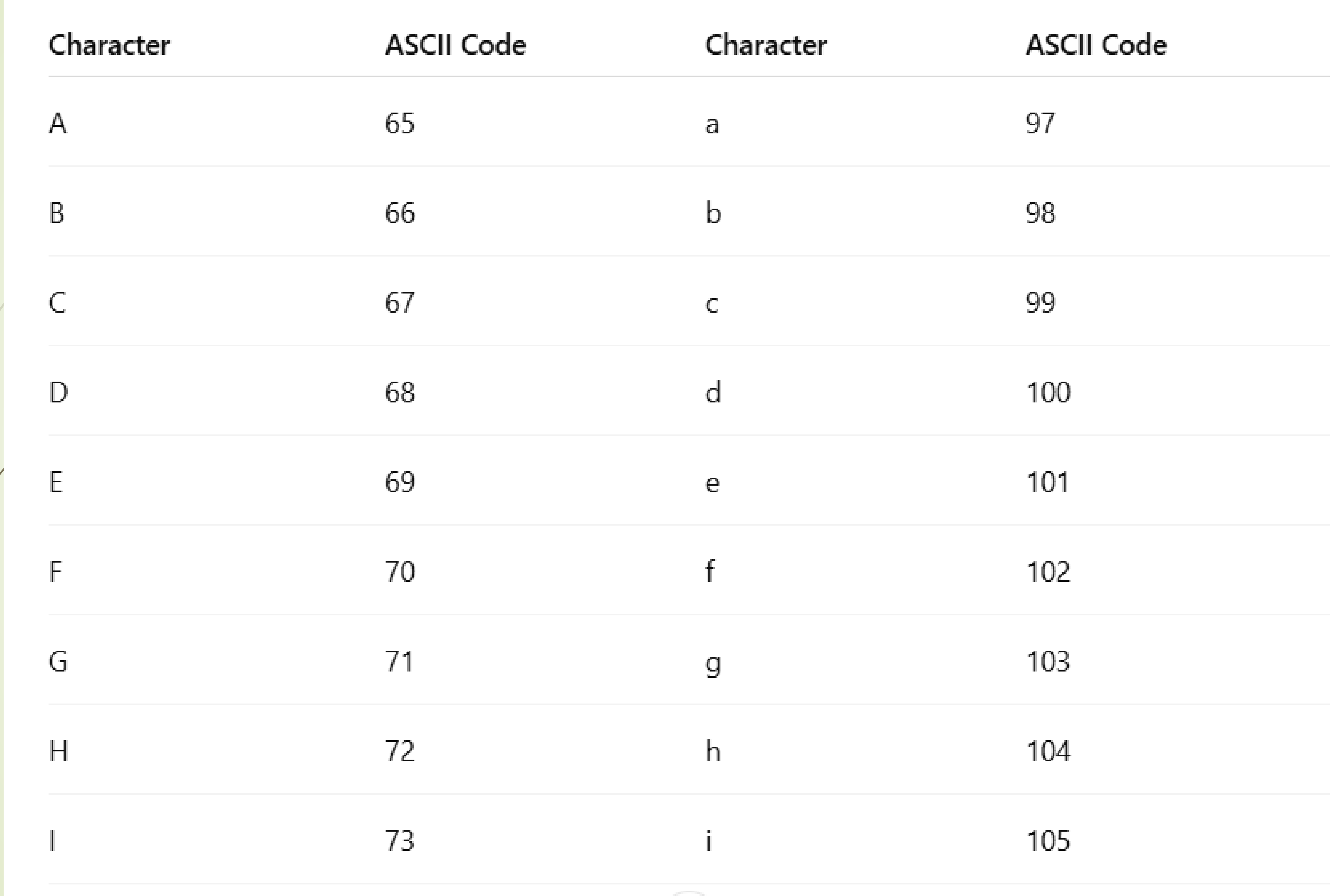
Note :**ASCII (American Standard Code for Information Interchange)** assigns numbers to characters.



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

int main() {
    char ch;

    cout << "Enter a character: ";
    cin >> ch;

    cout << "You entered: " << ch << endl;
    cout << "ASCII code of " << ch << " = " << int(ch) << endl;

    return 0;
}
```



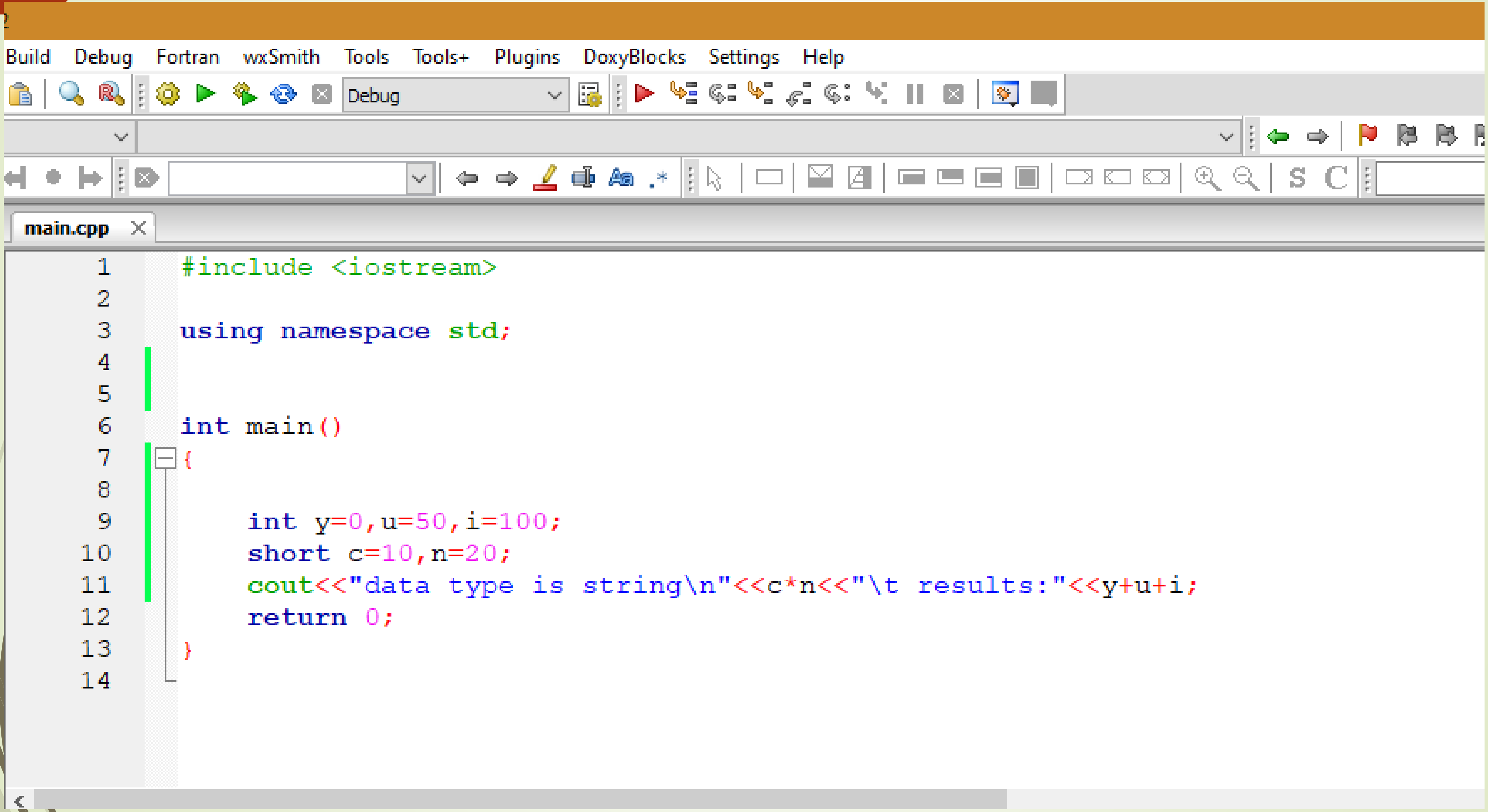
```
Enter a character: A
You entered: A
ASCII code of A = 65
```



Character	ASCII Code	Character	ASCII Code
A	65	a	97
B	66	b	98
C	67	c	99
D	68	d	100
E	69	e	101
F	70	f	102
G	71	g	103
H	72	h	104
I	73	i	105

Variable declaration/initialization

- Int x; declaration
- X=10; initialization
- Char c; declaration
- c=‘v’; initialization
- Double m; declaration
- m=2.58; initialization
- String c; declaration
- C=“mona”; initialization



```cpp
#include <iostream>

using namespace std;


int main()
{

    int y=0,u=50,i=100;
    short c=10,n=20;
    cout<<"data type is string\n"<<c*n<<"\t results:"<<y+u+i;
    return 0;
}

```

Escape Sequences

- **Escape sequences tell the compiler to treat characters in a special way**
- **'\' is the escape character**
  - **To create a newline in output use**

    \n – cout << "\n";

    or the newer alternative

    cout << endl;

  - **Other escape sequences:**

    \t -- a tab

    \\ -- a backslash character

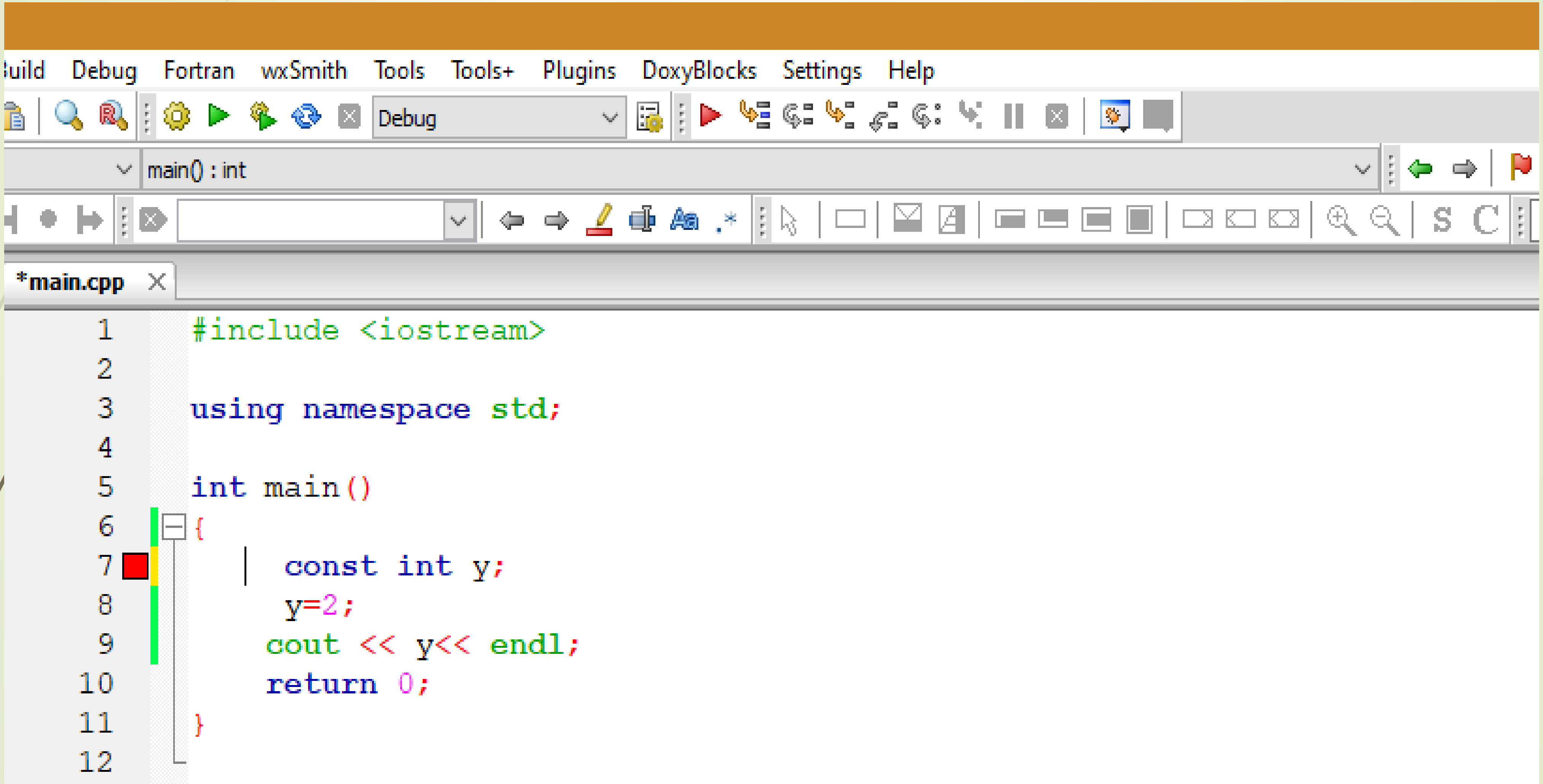
    \" -- a quote character

Constants

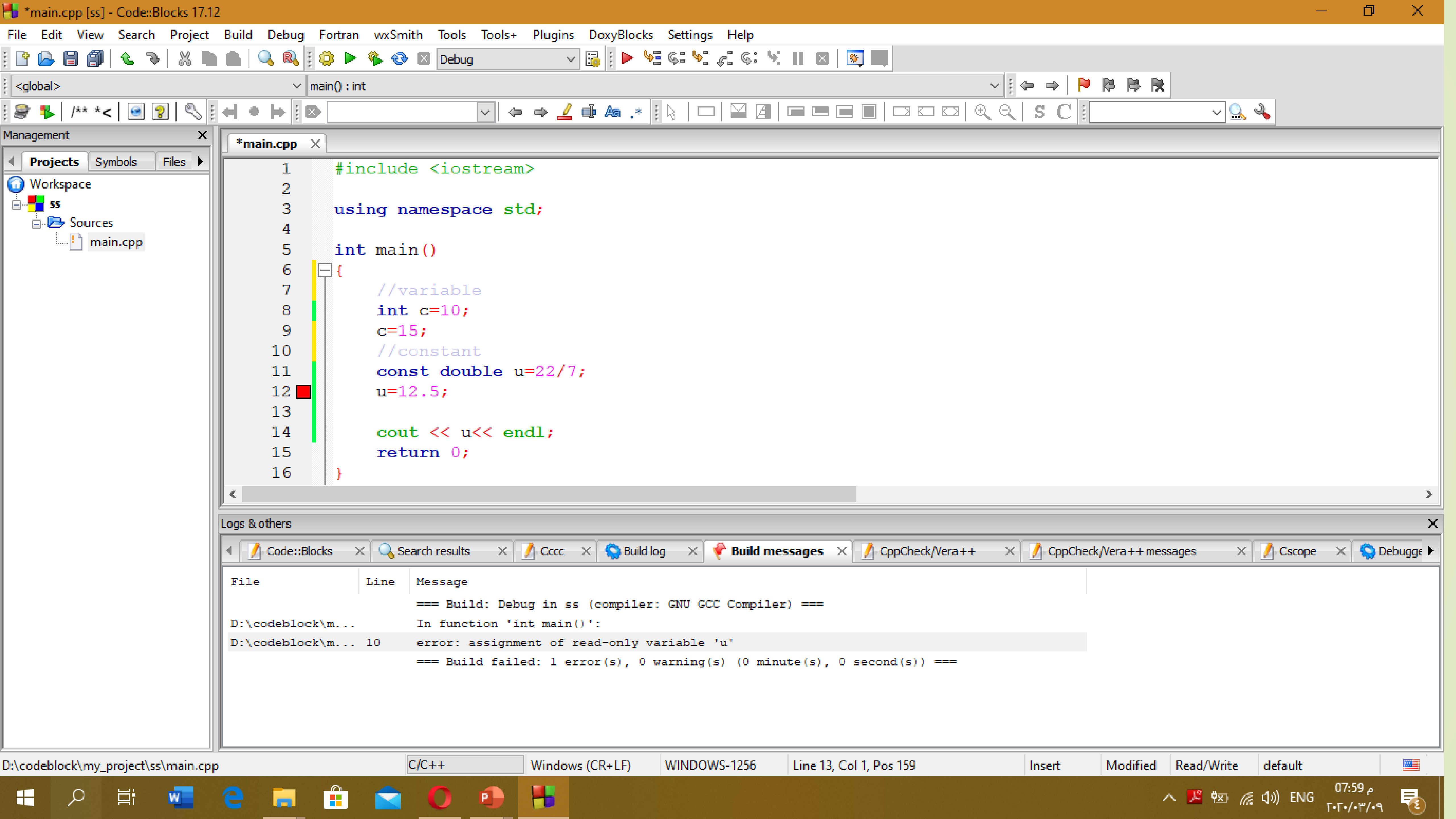
- Used to define a variable whose value cannot be changed.
- Constant declaration syntax: const type constant name;
- Must declare and initialize constant in same time.
- Constant can’t reinitialize.

```
const double PI = 3.14;
 PI = 2.9; //Error
```

Example


Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
main() : int
*main.cpp

```cpp
#include <iostream>

using namespace std;

int main()
{
    const int y;
    y=2;
   cout << y<< endl;
   return 0;
}

```

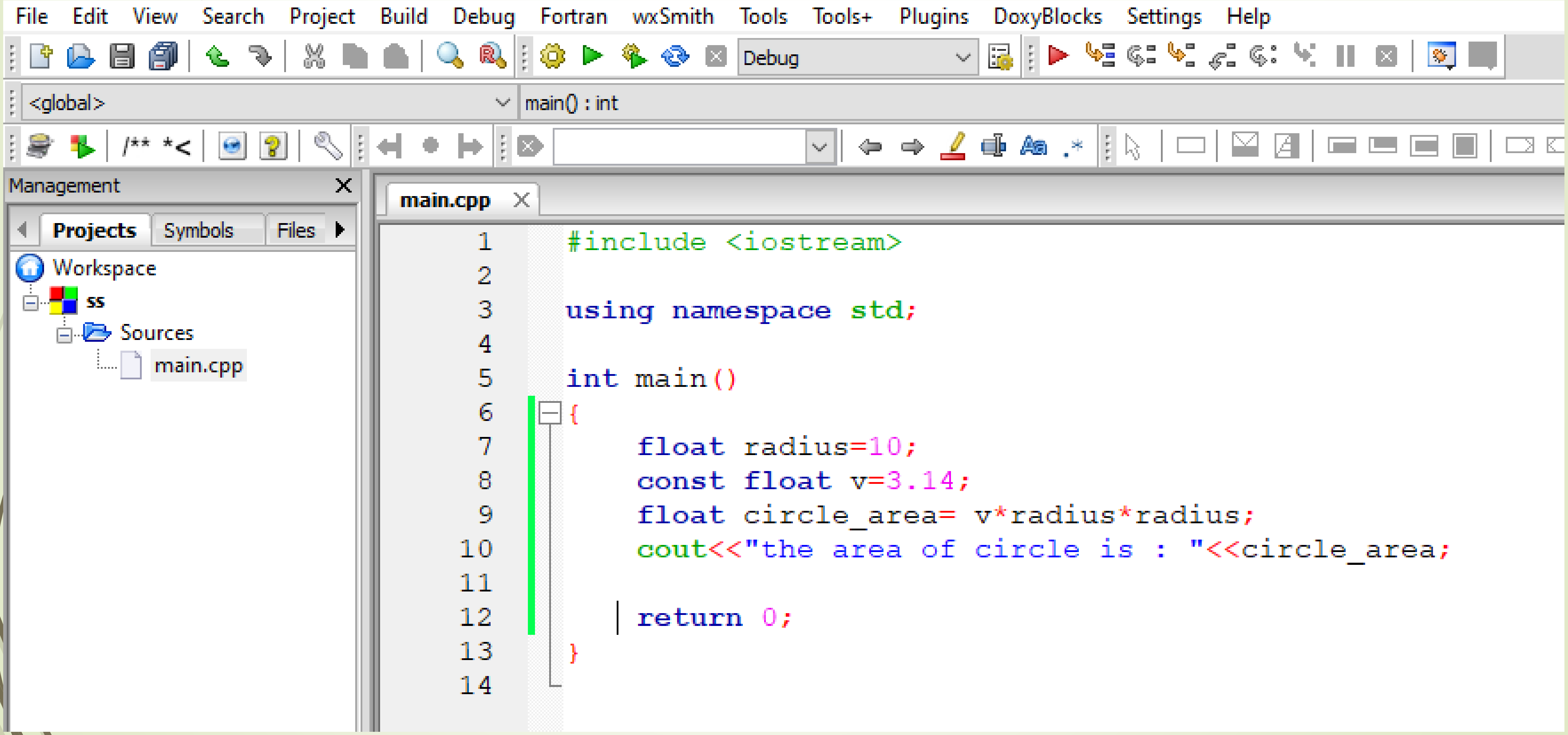


```
#include <iostream>

using namespace std;

int main()
{
    //variable
    int c=10;
    c=15;
    //constant
    const double u=22/7;
    u=12.5;

    cout << u<< endl;
    return 0;
}
```

```
File                Line   Message
                           === Build: Debug in ss (compiler: GNU GCC Compiler) ===
D:\codeblock\m...          In function 'int main()':
D:\codeblock\m...   10     error: assignment of read-only variable 'u'
                           === Build failed: 1 error(s), 0 warning(s) (0 minute(s), 0 second(s)) ===
```

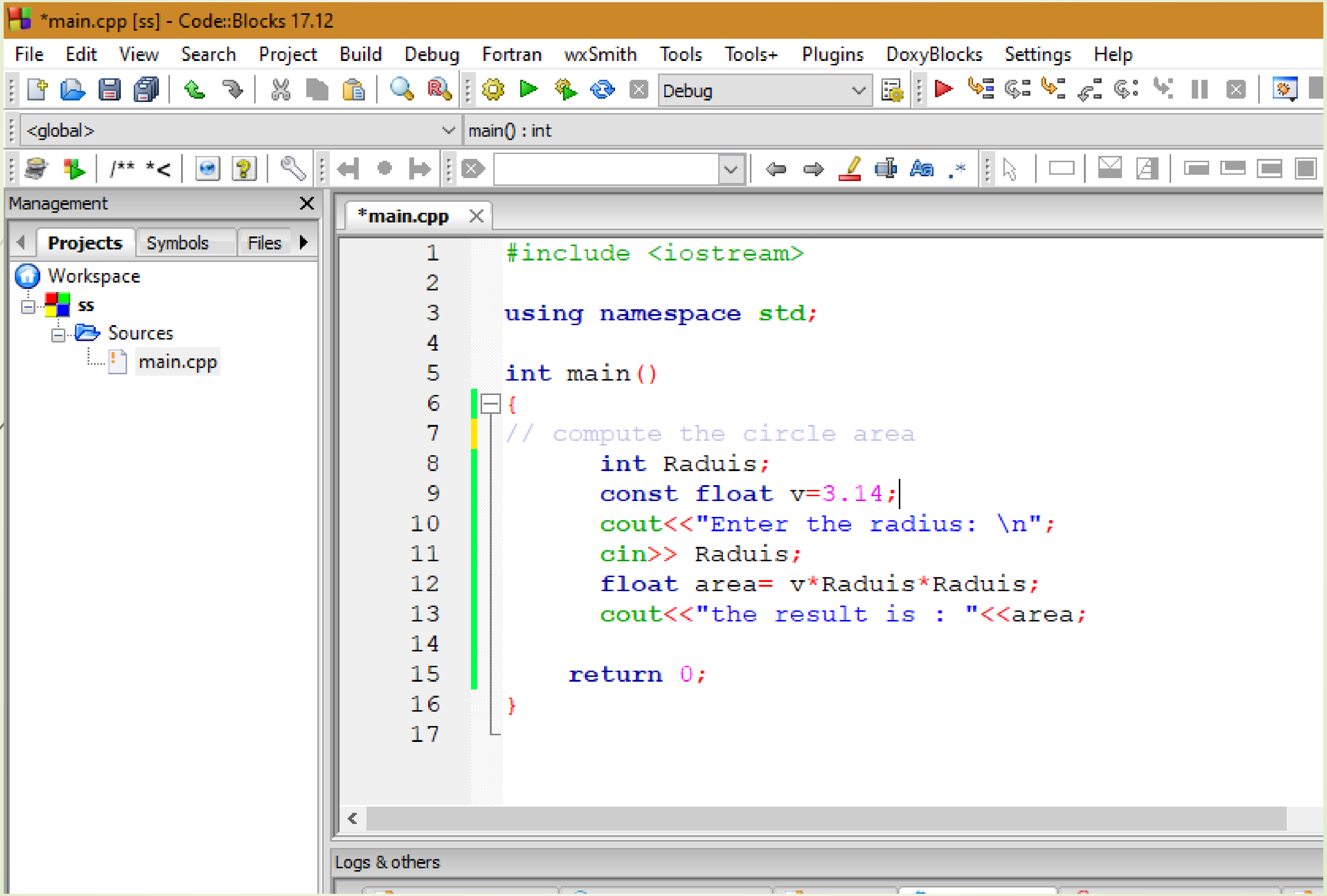
D:\codeblock\my_project\ss\main.cpp | C/C++ | Windows (CR+LF) | WINDOWS-1256 | Line 13, Col 1, Pos 159 | Insert | Modified | Read/Write | default

Compute circle area


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
main() : int
Management
Projects Symbols Files
Workspace
ss
Sources
main.cpp
main.cpp

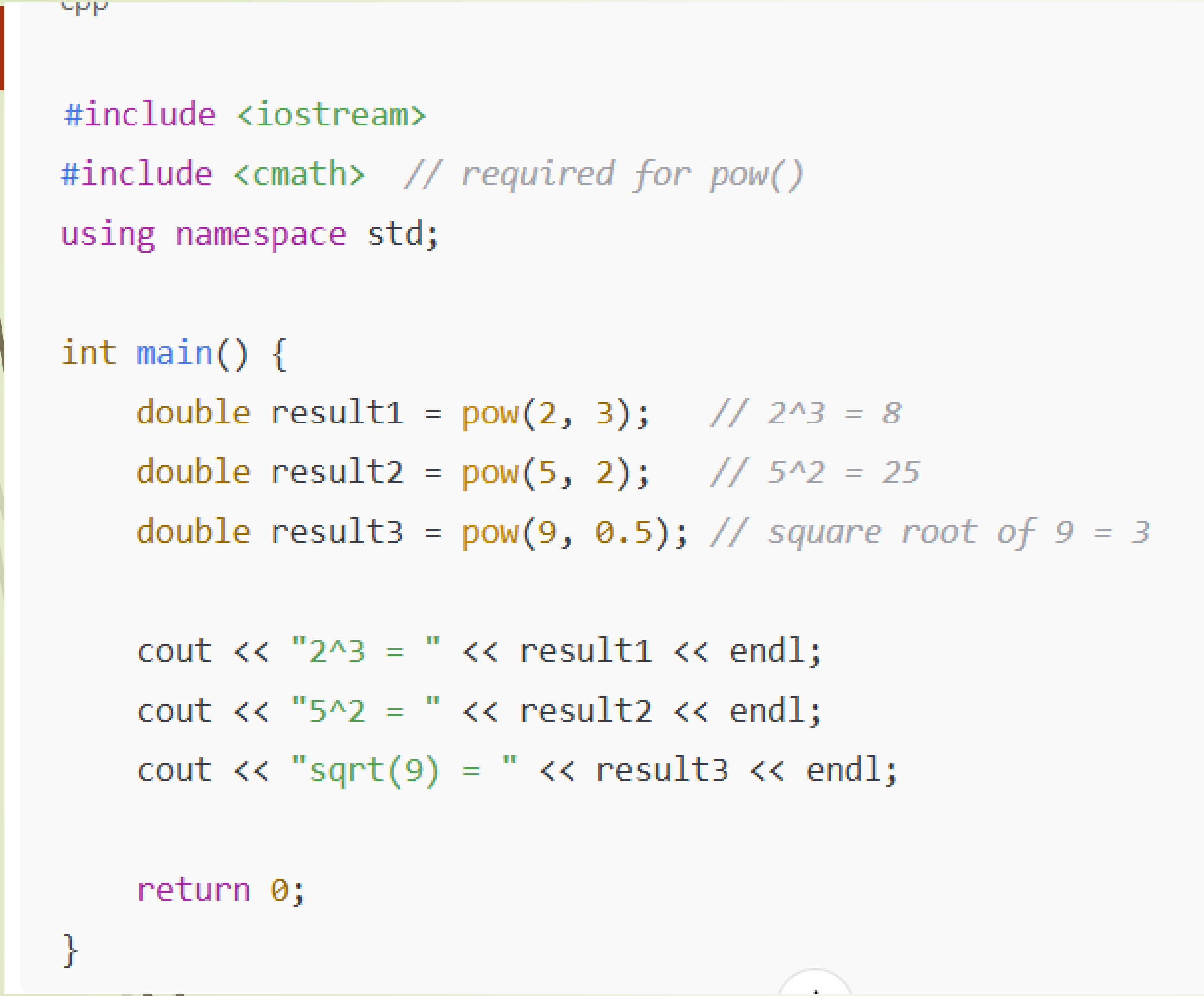
```cpp
#include <iostream>

using namespace std;

int main()
{
    float radius=10;
    const float v=3.14;
    float circle_area= v*radius*radius;
    cout<<"the area of circle is : "<<circle_area;

    return 0;
}

```

Example2



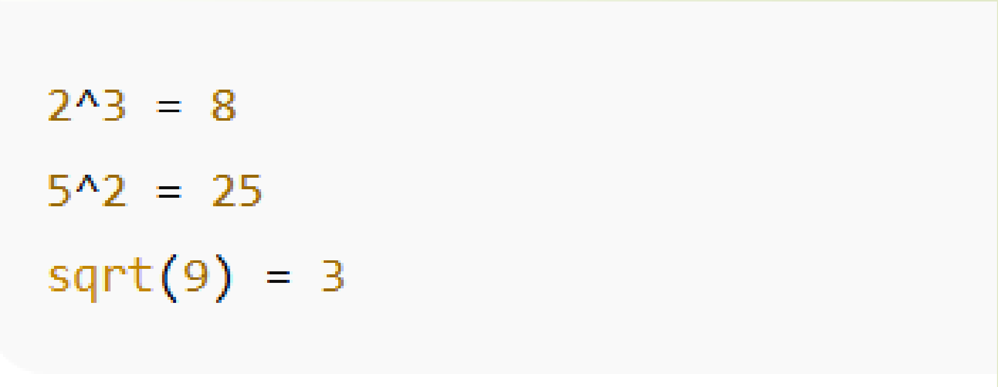
*main.cpp [ss] - Code::Blocks 17.12
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Debug
<global>
main() : int
Management
Projects Symbols Files
Workspace
ss
Sources
main.cpp
*main.cpp
Logs & others

```
#include <iostream>

using namespace std;

int main()
{
// compute the circle area
        int Raduis;
        const float v=3.14;
        cout<<"Enter the radius: \n";
        cin>> Raduis;
        float area= v*Raduis*Raduis;
        cout<<"the result is : "<<area;

      return 0;
}

```



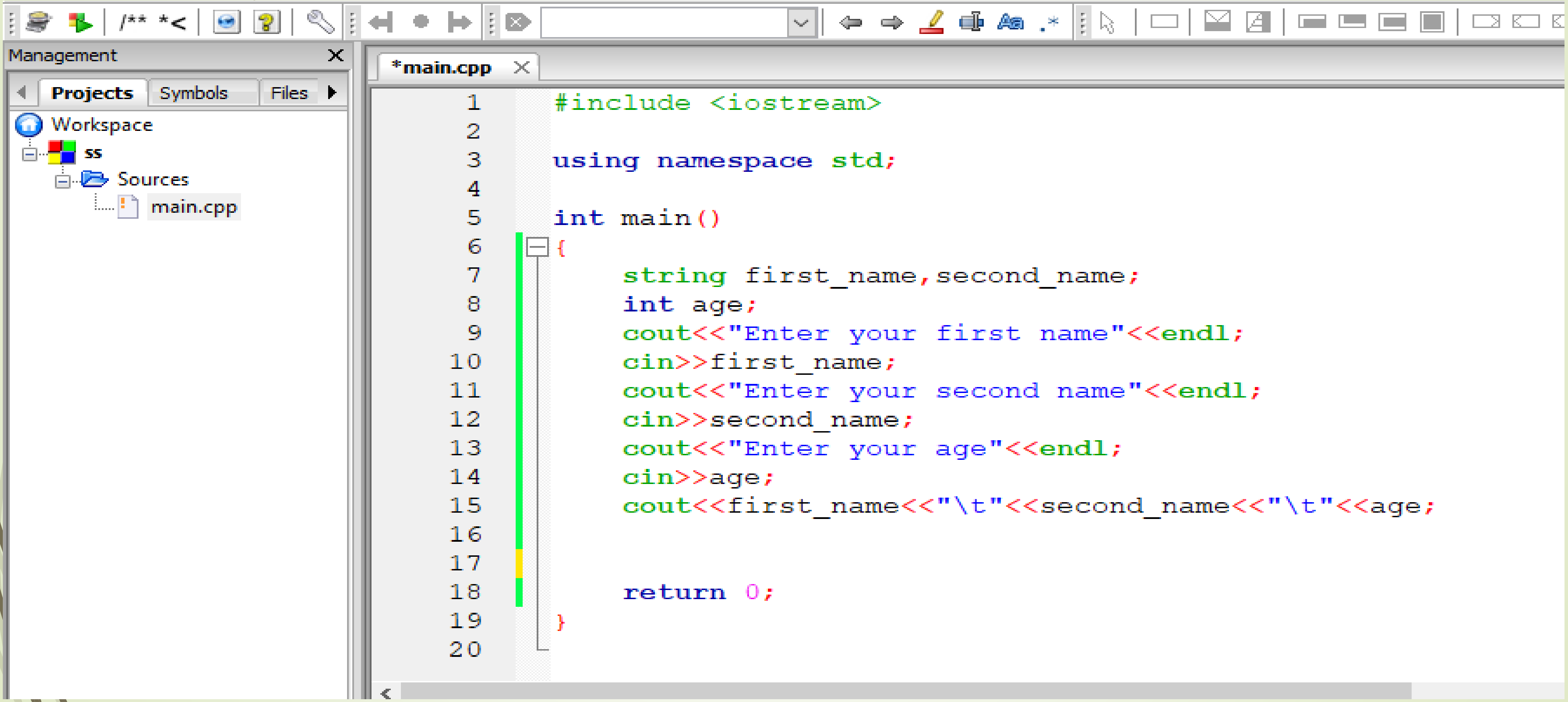
cpp

```cpp
#include <iostream>
#include <cmath>  // required for pow()
using namespace std;

int main() {
    double result1 = pow(2, 3);   // 2^3 = 8
    double result2 = pow(5, 2);   // 5^2 = 25
    double result3 = pow(9, 0.5); // square root of 9 = 3

    cout << "2^3 = " << result1 << endl;
    cout << "5^2 = " << result2 << endl;
    cout << "sqrt(9) = " << result3 << endl;

    return 0;
}
```



```
2^3 = 8
5^2 = 25
sqrt(9) = 3
```

Write c++ program that ask the user to enter his/her name and age then print them in same line.



```cpp
#include <iostream>

using namespace std;

int main()
{
    string first_name,second_name;
    int age;
    cout<<"Enter your first name"<<endl;
    cin>>first_name;
    cout<<"Enter your second name"<<endl;
    cin>>second_name;
    cout<<"Enter your age"<<endl;
    cin>>age;
    cout<<first_name<<"\t"<<second_name<<"\t"<<age;


    return 0;
}

```

Thank you