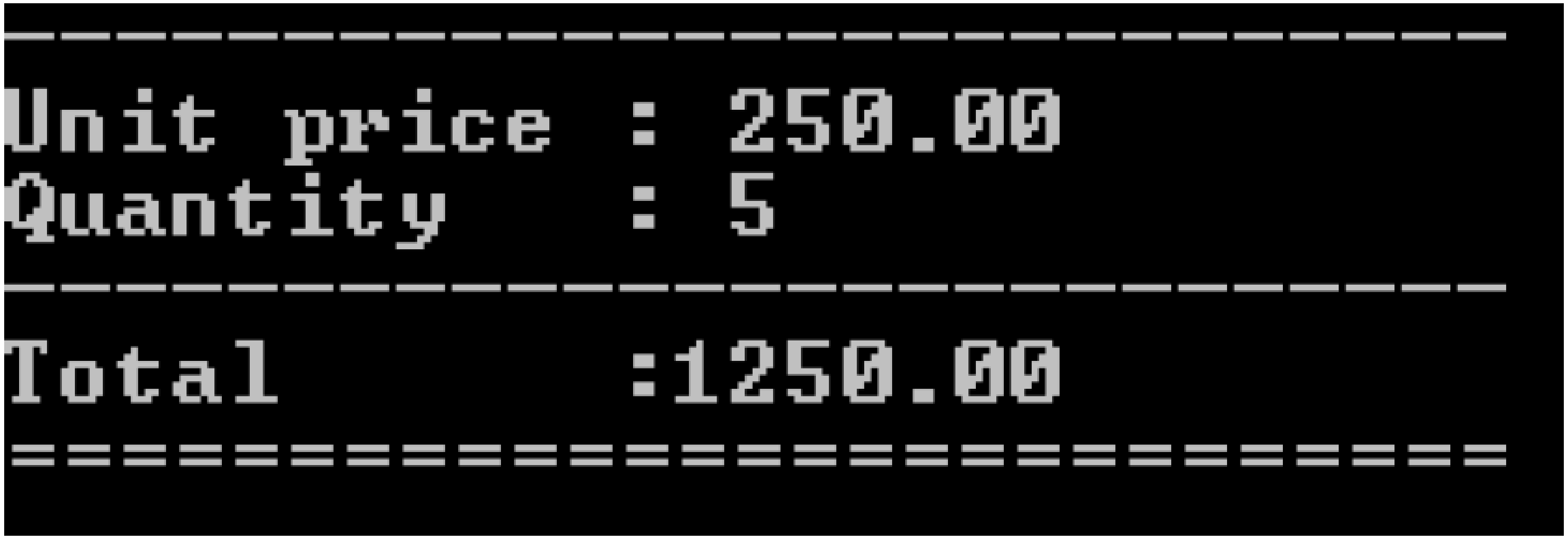
Basic Computer Programming and Networking
Data types & variables

RHNS Jayathissa
Department of Computer Engineering
Faculty Of Computing

Exercise 3.1

- Write a C++ program to display the following output.



```
---------------------------
Unit price : 250.00
Quantity   : 5
---------------------------
Total      :1250.00
===========================
```

Exercise 3.2

- Write a C++ program to calculate and display total amount of the given unit price and quantity of an item.

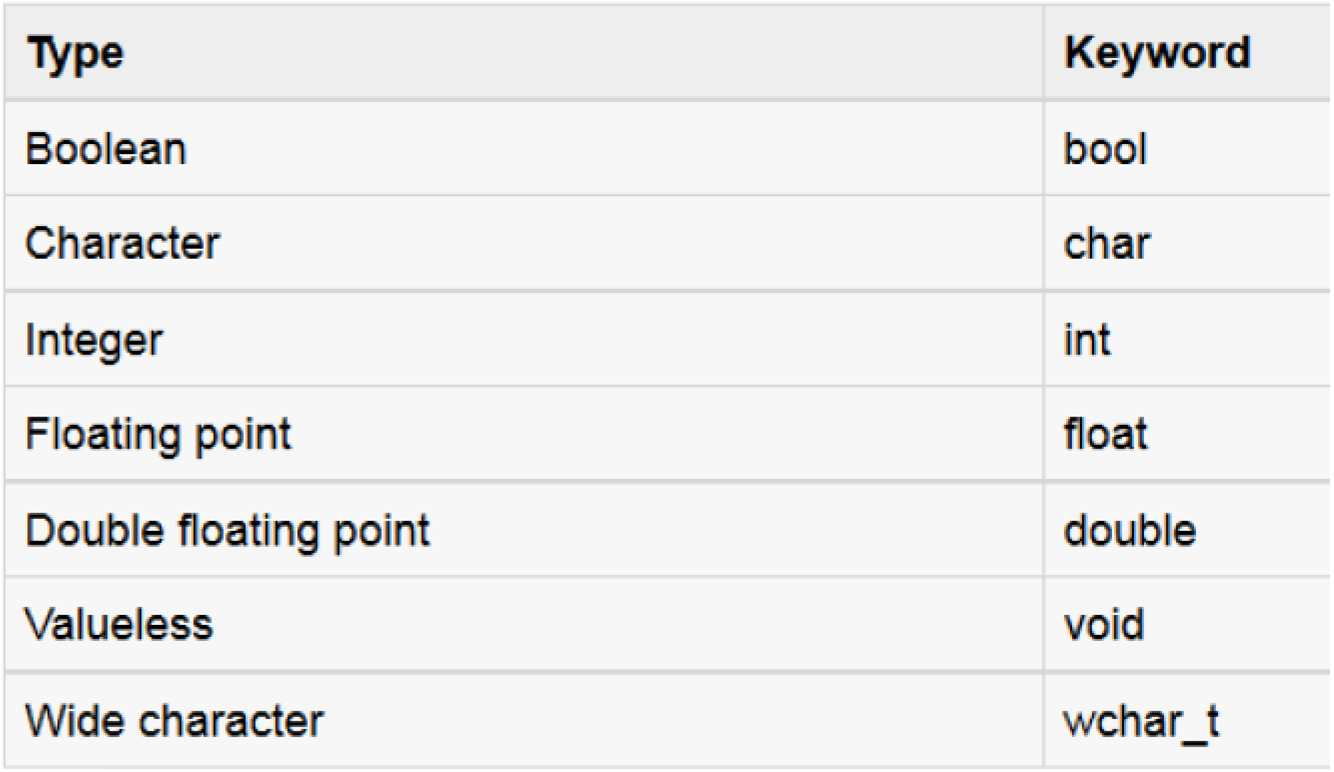
Data types

Data Types

- Use various variables to store various information
- Variables are reserved memory locations to store values
- When you create a variable you reserve some space in memory
- Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory

C++ Data Types

- Seven basic C++ data types



Type	Keyword
Boolean	bool
Character	char
Integer	int
Floating point	float
Double floating point	double
Valueless	void
Wide character	wchar_t

C++ Data Types

In C++, data types are declarations for variables. This determines the type and size of data associated with variables. For example,

int age = 13;

Here, age is a variable of type int. Meaning, the variable can only store integers of either 2 or 4 bytes.

C++ int

The int keyword is used to indicate integers.

Its size is usually 4 bytes. Meaning, it can store values from -2147483648 to 2147483647. {(2^32)/2= from -2147483648 to 2147483647}

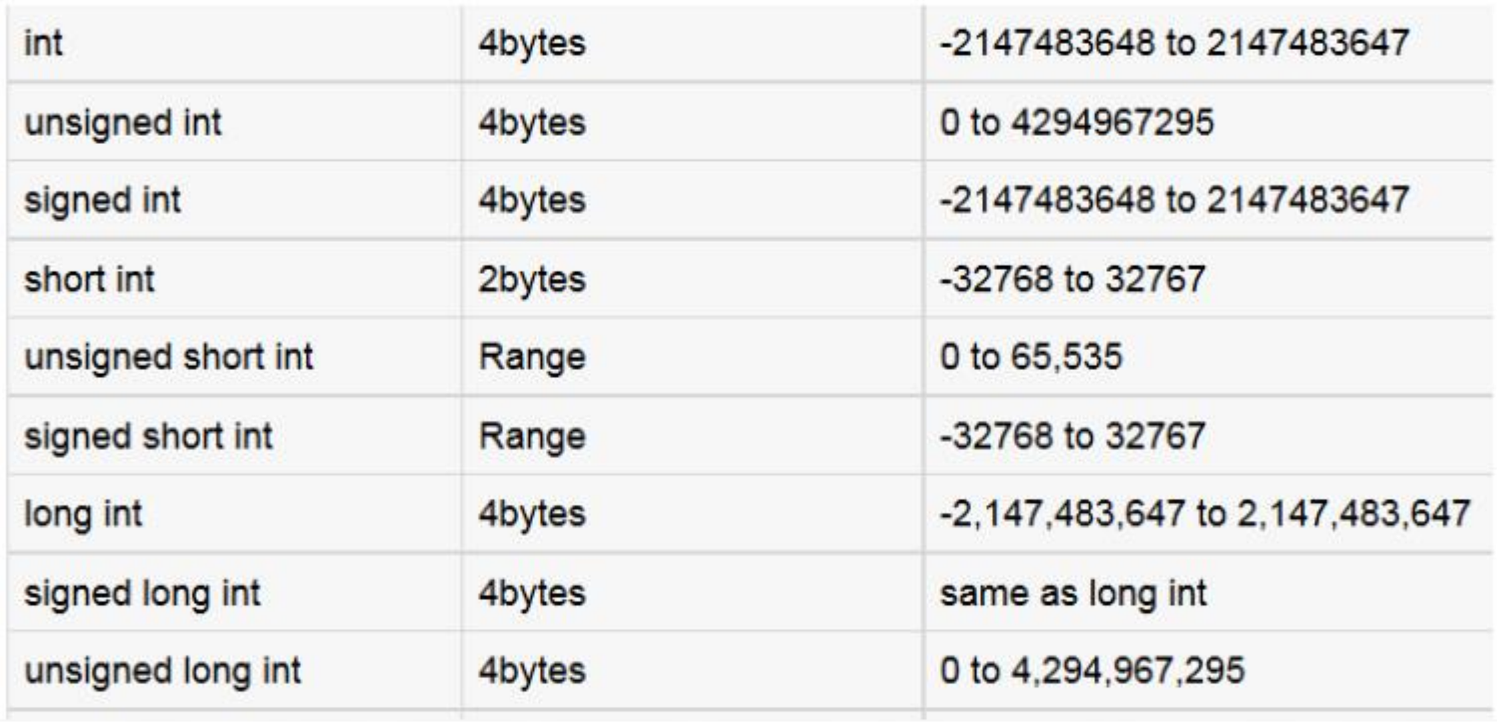
For example,

```
int salary = 85000;
```

Numerical integer types

int

- They can store a whole number value, such as 7 or 1024.



int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	Range	0 to 65,535
signed short int	Range	-32768 to 32767
long int	4bytes	-2,147,483,647 to 2,147,483,647
signed long int	4bytes	same as long int
unsigned long int	4bytes	0 to 4,294,967,295

C++ float and double

float and double are used to store floating-point numbers (decimals and exponentials).

The size of float is 4 bytes and the size of double is 8 bytes. Hence, double has two times the precision of float

For example,

```
float area = 64.74;

double volume = 134.64534;
```

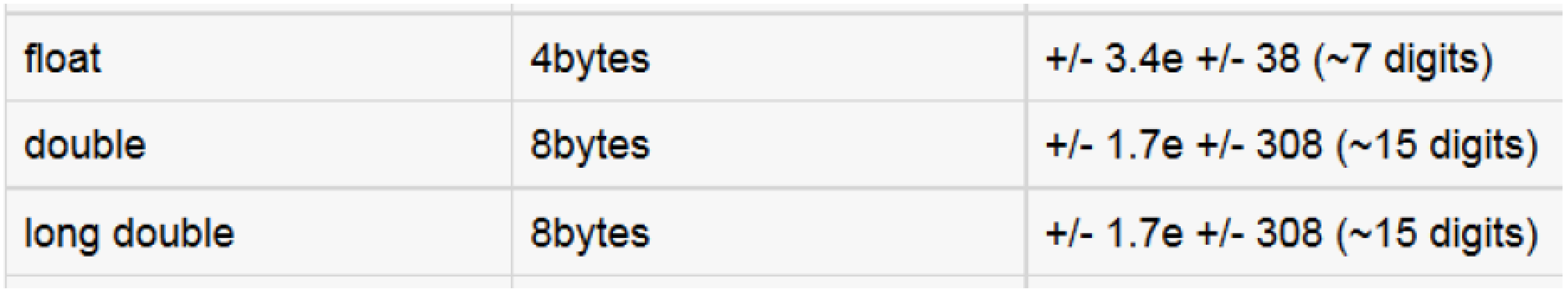
As mentioned above, these two data types are also used for exponentials. For example,

```
double distance = 45E12    // 45E12 is equal to 45*10^12
```

Floating-point types

float

- They can represent real values, such as 3.14 or 0.01



float	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	8bytes	+/- 1.7e +/- 308 (~15 digits)

C++ char

Keyword char is used for characters.

Its size is 1 byte.

Characters in C++ are enclosed inside single quotes ' '.

For example,

char test = 'h';

C++ wchar_t

Wide character wchar_t is similar to the char data type, except its size is 2 bytes instead of 1.

It is used to represent characters that require more memory to represent them than a single char.

For example,

```
wchar_t w  = L'A';
  cout << "Wide character value:: " << w << endl ;
  cout << "Size of the wide char is:: " << sizeof(w);
  return 0;
```

Output:

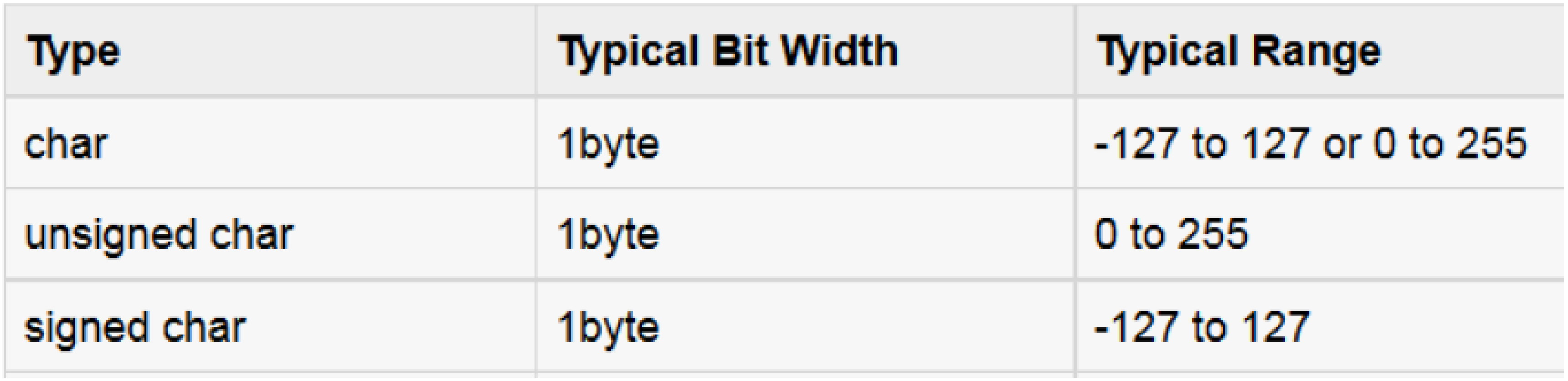
```
Wide character value:: 65 Size of the wide char is:: 2
```

•L is the prefix for wide character literals and wide-character string literals which tells the compiler that that the char or string is of type wide-char.

Character types

char

- They can represent a single character, such as 'A' or '$'. The most basic type is char, which is a one-byte character.



Type	Typical Bit Width	Typical Range
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127

C++ bool

The bool data type has one of two possible values: true or false.

Booleans are used in conditional statements and loops (which we will learn in later chapters).

For example,

```
bool cond = false;
```

Example:

```
bool isCodingFun = true;
bool isFishTasty = false;
cout << isCodingFun;  // Outputs 1 (true)
cout << isFishTasty;  // Outputs 0 (false)
```

C++ void

The void keyword indicates an absence of data. It means "nothing" or "no value".

We will use void when we learn about functions.

Note: We cannot declare variables of the void type.

String

string - stores text, such as "Hello World". String values are surrounded by double quotes

```
string myText = "Hello";     // String (text)
```

Modifiers

- Several of the basic types can be modified using one or more of these type modifiers
  - signed
  - unsigned
  - short
  - long

We can modify the following data types with the above modifiers:

Int

double

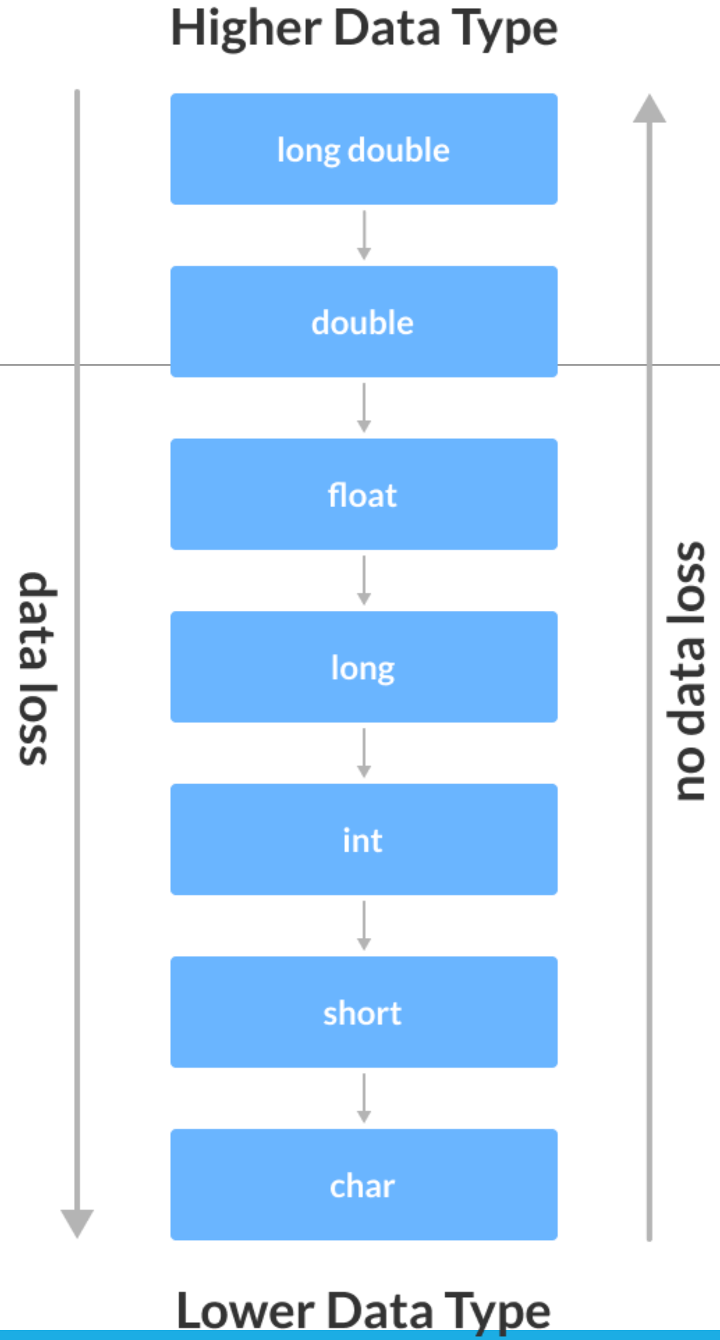
char

C++ Modified Data Types List

Data Type	Size (in Bytes)	Meaning
signed int	4	used for integers (equivalent to int)
unsigned int	4	can only store positive integers
short	2	used for small integers (range -32768 to 32767)
unsigned short	2	used for small positive integers (range 0 to 65,535)
long	at least 4	used for large integers (equivalent to long int)
unsigned long	4	used for large positive integers or 0 (equivalent to unsigned long int)
long long	8	used for very large integers (equivalent to long long int).
unsigned long long	8	used for very large positive integers or 0 (equivalent to unsigned long long int)
long double	12	used for large floating-point numbers
signed char	1	used for characters (guaranteed range -127 to 127)
unsigned char	1	used for characters (range 0 to 255)

The above mentioned modifiers can be used along with built in datatypes to make them more precise and even expand their range.

- **long** and **short** modify the maximum and minimum values that a data type will hold.
- A plain int must have a minimum size of **short**.
- Size hierarchy : short int < int < long int
- Size hierarchy for floating point numbers is : float < double < long double
- **long float** is not a legal type and there are no **short floating point** numbers.
- **Signed** types includes both positive and negative numbers and is the default type.
- **Unsigned**, numbers are always without any sign, that is always positive.

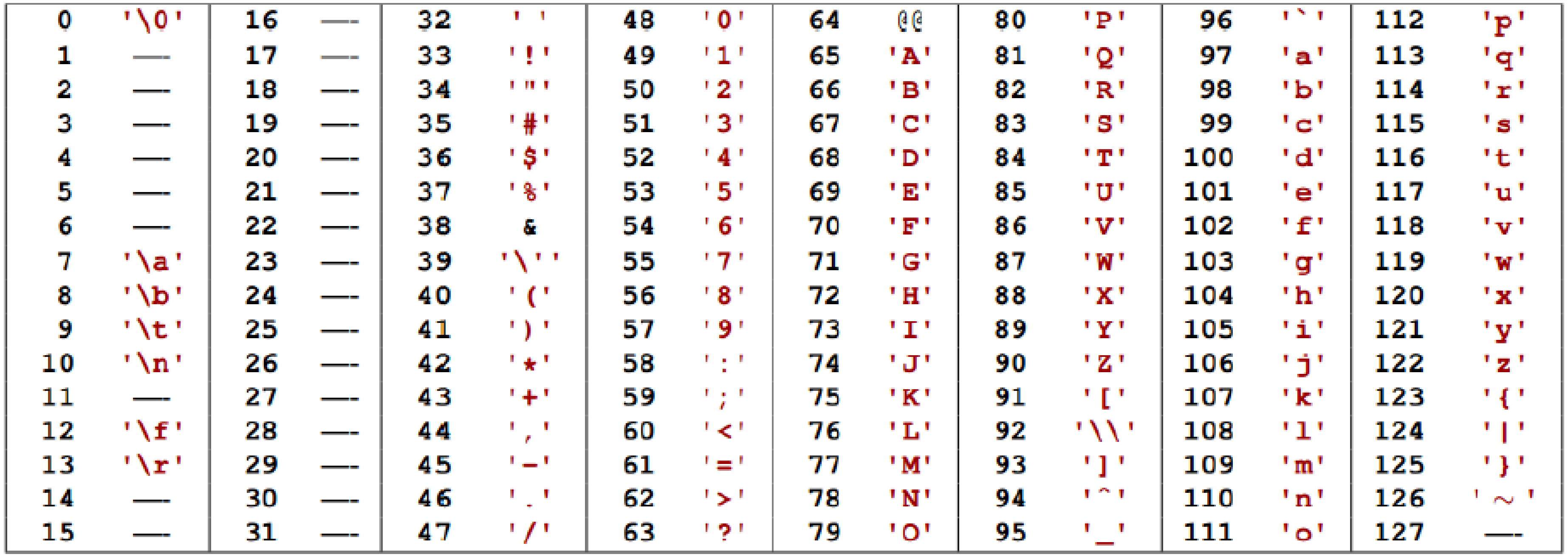

Higher Data Type
long double
double
float
long
int
short
char
data loss
no data loss
Lower Data Type

Examples:

Let's see a few examples.

```
long b = 4523232;

long int c = 2345342;

long double d = 233434.56343;

short d = 3434233; // Error! out of range

unsigned int a = -5;    // Error! can only store positive numbers or 0
```

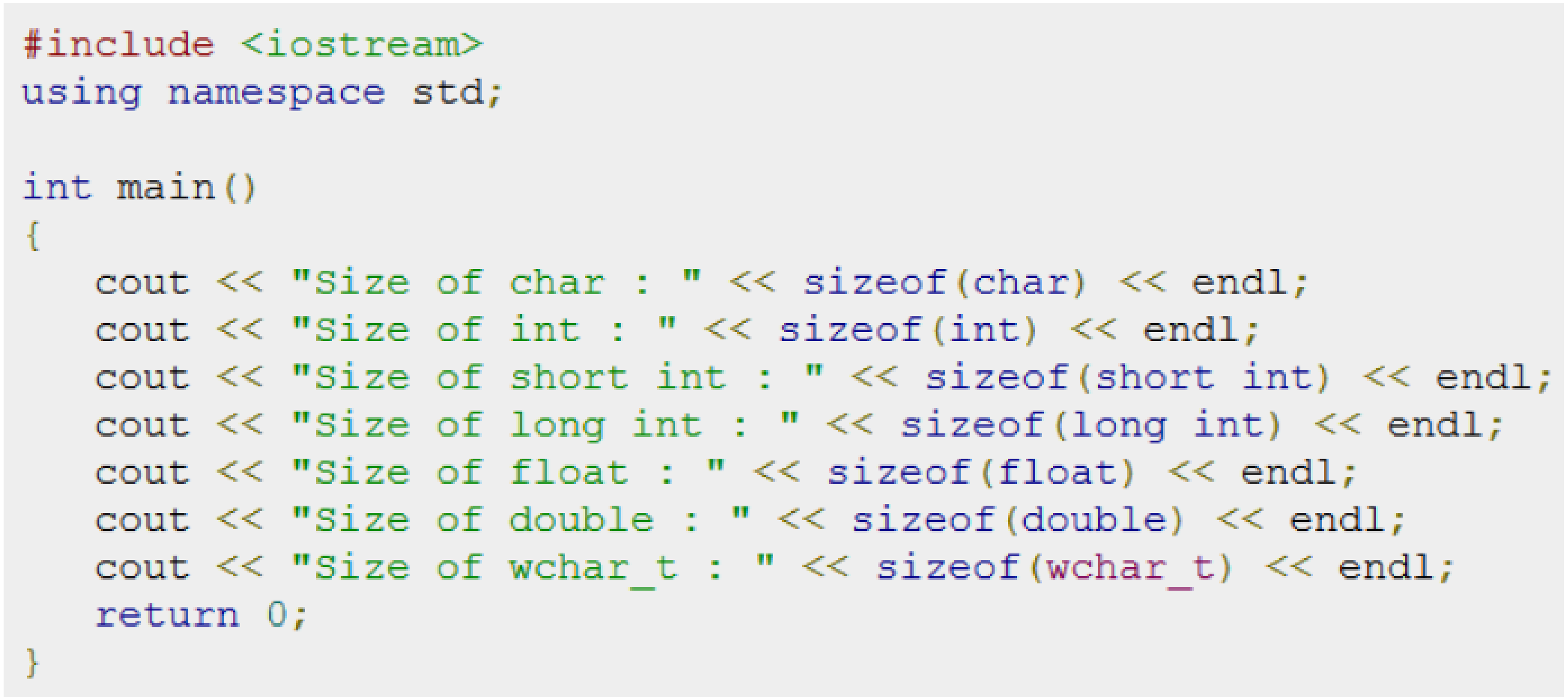
ASCII Code for the characters



0	'\0'	16	—	32	' '	48	'0'	64	@@	80	'P'	96	'`'	112	'p'
1	—	17	—	33	'!'	49	'1'	65	'A'	81	'Q'	97	'a'	113	'q'
2	—	18	—	34	'"'	50	'2'	66	'B'	82	'R'	98	'b'	114	'r'
3	—	19	—	35	'#'	51	'3'	67	'C'	83	'S'	99	'c'	115	's'
4	—	20	—	36	'$'	52	'4'	68	'D'	84	'T'	100	'd'	116	't'
5	—	21	—	37	'%'	53	'5'	69	'E'	85	'U'	101	'e'	117	'u'
6	—	22	—	38	&	54	'6'	70	'F'	86	'V'	102	'f'	118	'v'
7	'\a'	23	—	39	'\''	55	'7'	71	'G'	87	'W'	103	'g'	119	'w'
8	'\b'	24	—	40	'('	56	'8'	72	'H'	88	'X'	104	'h'	120	'x'
9	'\t'	25	—	41	')'	57	'9'	73	'I'	89	'Y'	105	'i'	121	'y'
10	'\n'	26	—	42	'*'	58	':'	74	'J'	90	'Z'	106	'j'	122	'z'
11	—	27	—	43	'+'	59	';'	75	'K'	91	'['	107	'k'	123	'{'
12	'\f'	28	—	44	','	60	'<'	76	'L'	92	'\\'	108	'l'	124	'|'
13	'\r'	29	—	45	'-'	61	'='	77	'M'	93	']'	109	'm'	125	'}'
14	—	30	—	46	'.'	62	'>'	78	'N'	94	'^'	110	'n'	126	'~'
15	—	31	—	47	'/'	63	'?'	79	'O'	95	'_'	111	'o'	127	—

Size of a data type

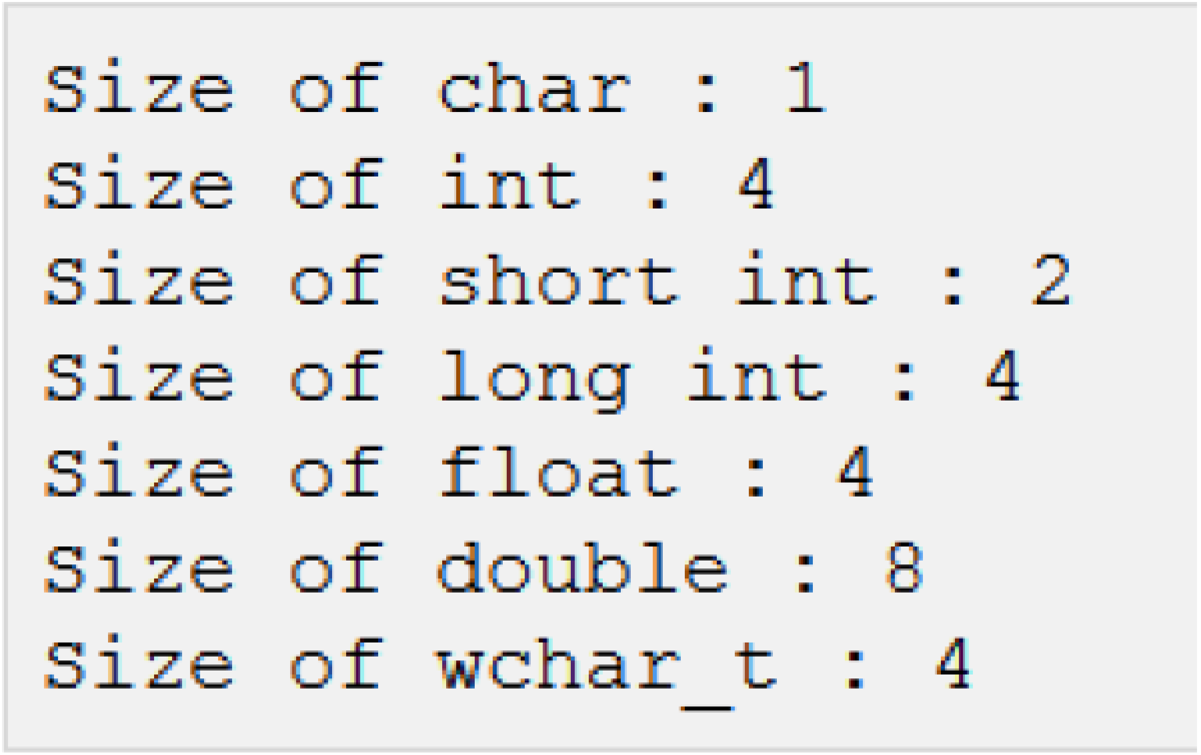
- Use **sizeof()** function to check the size



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

int main()
{
   cout << "Size of char : " << sizeof(char) << endl;
   cout << "Size of int : " << sizeof(int) << endl;
   cout << "Size of short int : " << sizeof(short int) << endl;
   cout << "Size of long int : " << sizeof(long int) << endl;
   cout << "Size of float : " << sizeof(float) << endl;
   cout << "Size of double : " << sizeof(double) << endl;
   cout << "Size of wchar_t : " << sizeof(wchar_t) << endl;
   return 0;
}
```

Answer

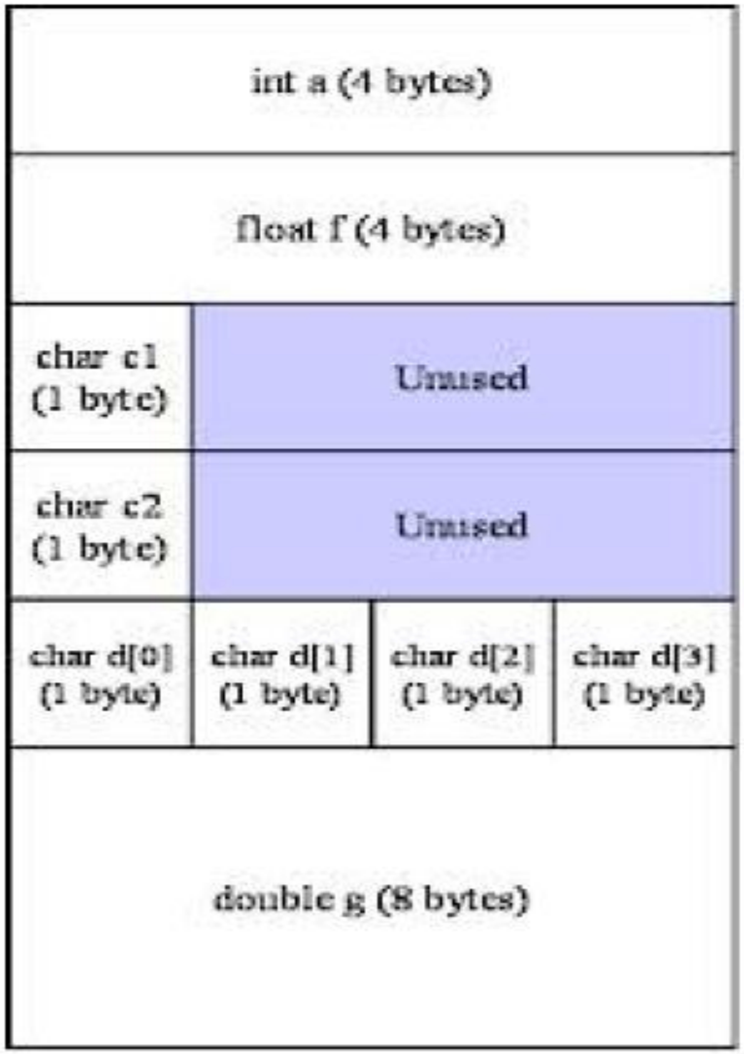
- Result which can vary from machine to machine/ Compilers



```
Size of char : 1
Size of int : 4
Size of short int : 2
Size of long int : 4
Size of float : 4
Size of double : 8
Size of wchar_t : 4
```

Data type selection

Memory allocation



Define Variables in C++

- Tell the compiler where and how much to create the storage for the variable

- Data Type <space> Variable Name
  - int number;
  - char letter;
  - float emp_salary

Variable name(rules)

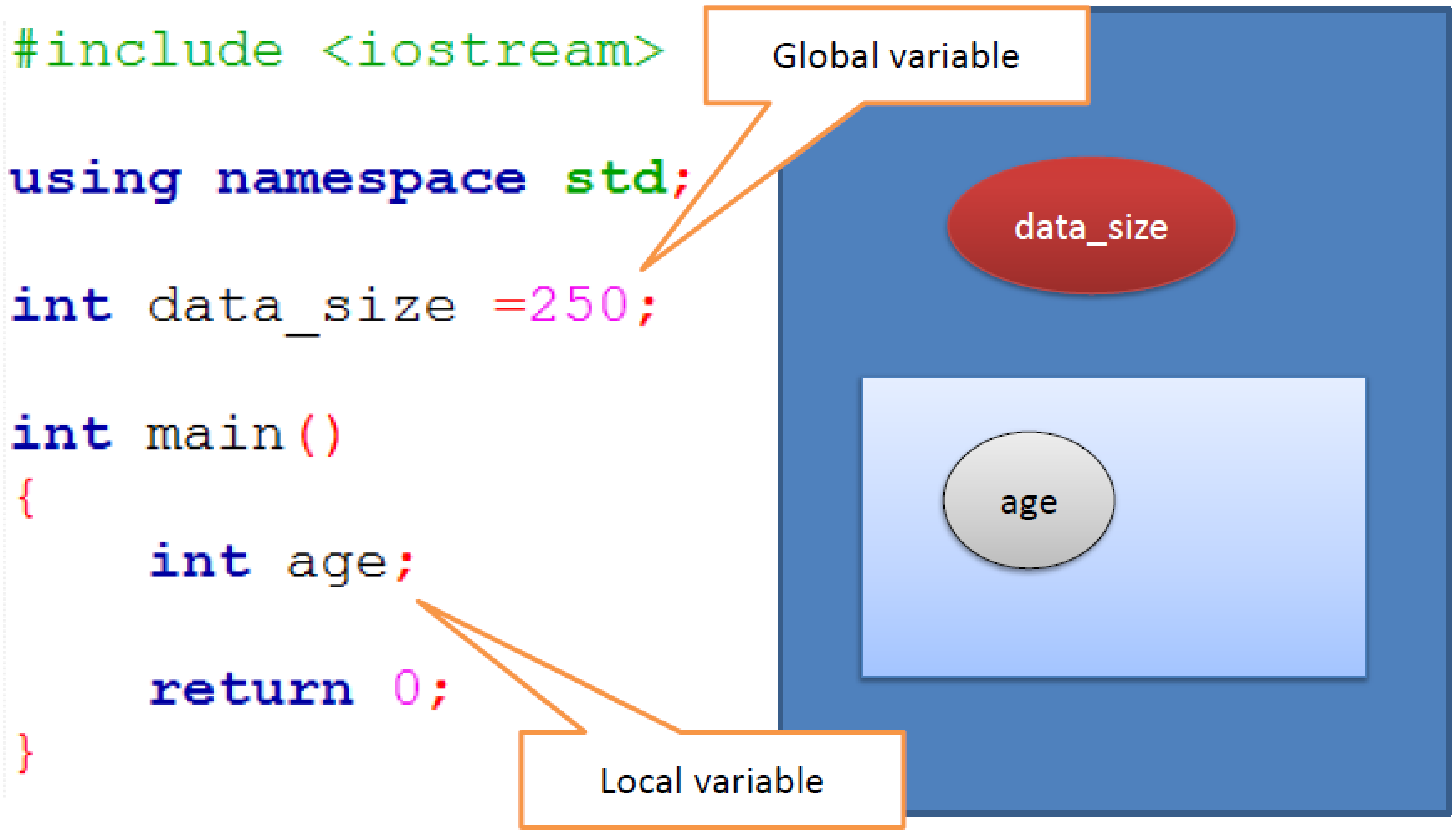
- Can use only letters, digits and underscore
- The first character must be a letter or underscore
- Case sensitive
- Cannot use keywords
- No limits on length

C++ is case sensitive. In other words, uppercase and lowercase letters are considered to be different. A variable named age is different from Age, which is different from AGE

C++ Variables

- Better not to begin a variable name with underscore.
- To form a name from two or more words, separate them with underscore
- Example
  - int student_age;
  - float employe_salary;
  - char grade ;

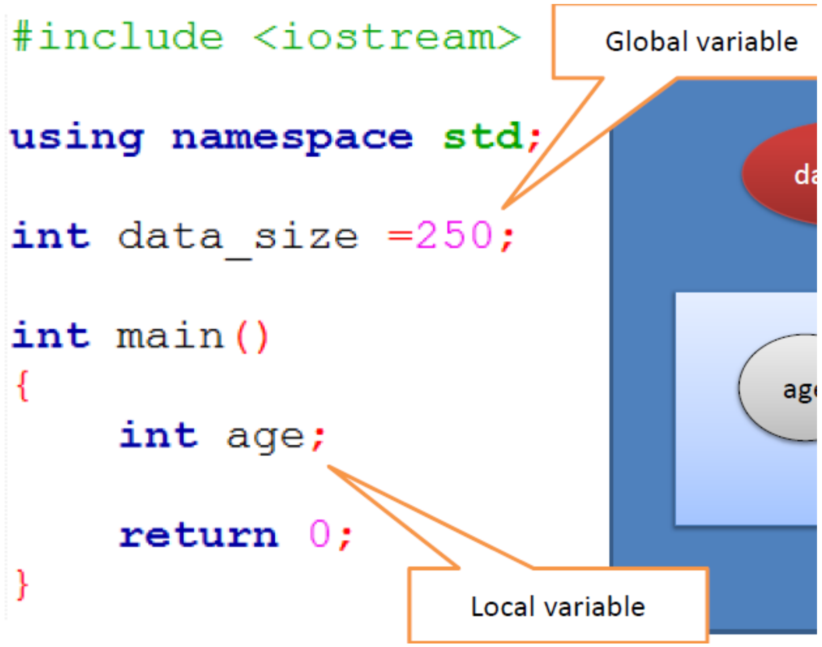
Local & Global Variables



```
#include <iostream>

using namespace std;

int data_size =250;

int main()
{
     int age;

     return 0;
}
```

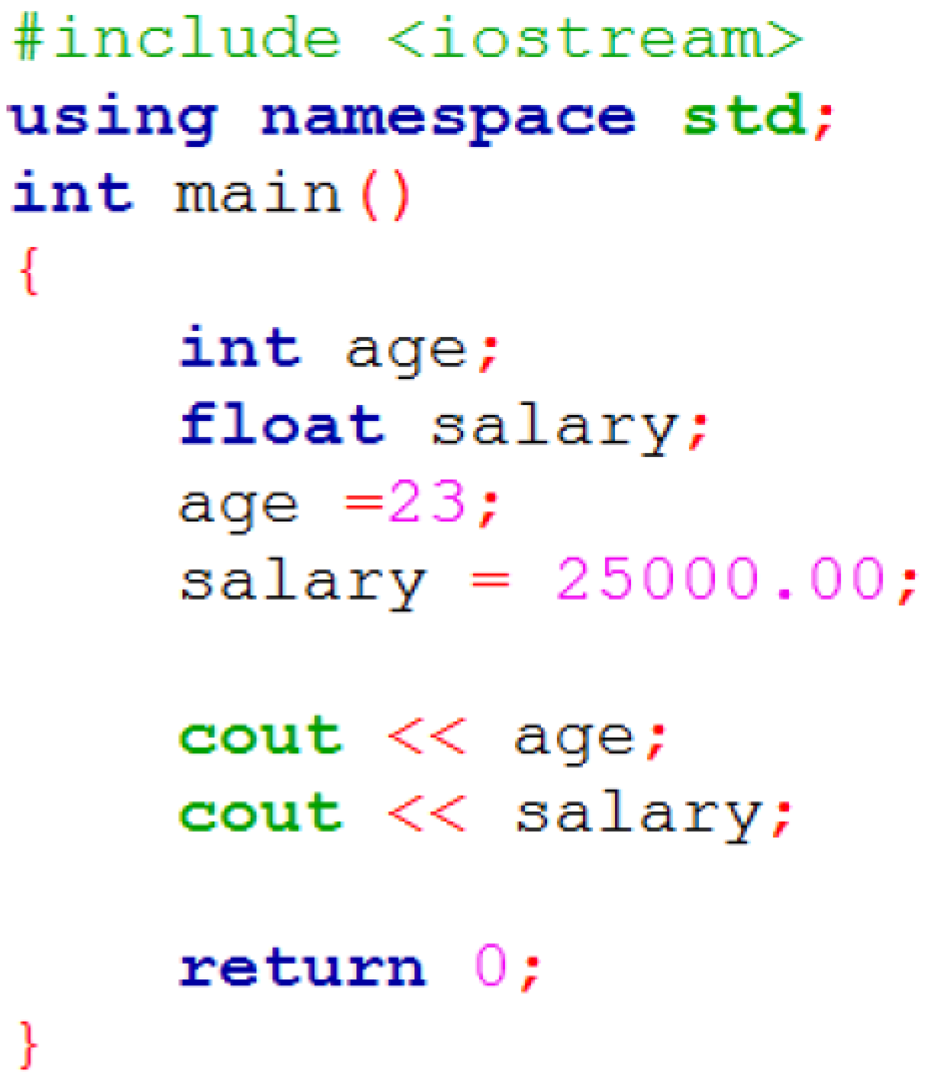
Local & Global Variables

Global variables are those which are not defined inside any function and have a global scope whereas local variables are those which are defined inside a function and its scope is limited to that function only



```
#include <iostream>

using namespace std;

int data_size =250;

int main()
{
    int age;

    return 0;
}
```

Variables in a program

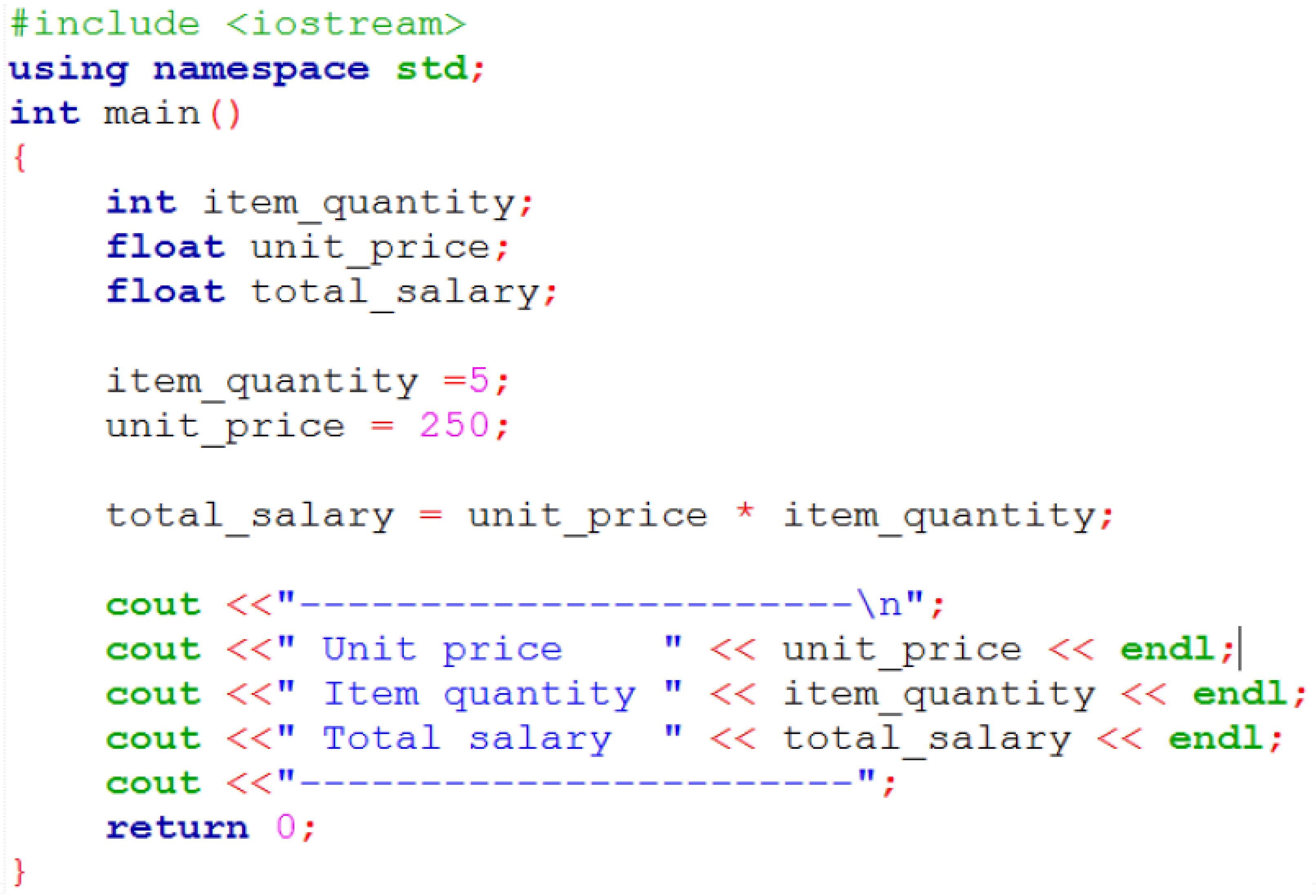


```cpp
#include <iostream>
using namespace std;
int main()
{
    int age;
    float salary;
    age =23;
    salary = 25000.00;

    cout << age;
    cout << salary;

    return 0;
}
```

Exercise 3.3

- Write a C++ program to calculate and display total amount of the given unit price and quantity of an item.

Answer



```
#include <iostream>
using namespace std;
int main()
{
    int item_quantity;
    float unit_price;
    float total_salary;

    item_quantity =5;
    unit_price = 250;

    total_salary = unit_price * item_quantity;

    cout <<"-----------------------\n";
    cout <<" Unit price    " << unit_price << endl;
    cout <<" Item quantity " << item_quantity << endl;
    cout <<" Total salary  " << total_salary << endl;
    cout <<"-----------------------";
    return 0;
}
```

Constant variables

- Constants are declared like variables with the addition of the **const** keyword

```
const double PI = 3.14159;
```

  Once declared and initialized, a constant can be used like a variable
- A constant may not be reassigned

Type Casting

- A way to convert a variable from one data type to another data type
- Use **cast operator**
  - (type_name) expression
  - (int) float_value

For example, suppose the given data is an float type, and we want to convert it into integer type. So, we need to manually cast float data to the int type, and this type of casting is called the Type Casting in C++.

Ex1:

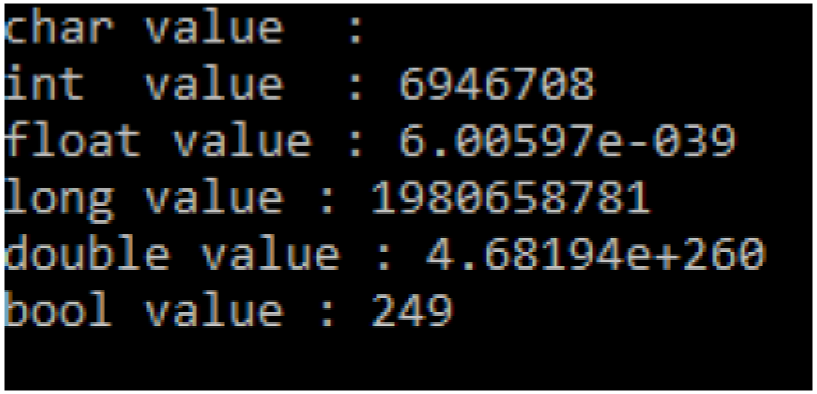
```
float num = 5.25;
int x;
x = int(num);
Output: 5
```

C++ Memory concept

- Variable names correspond to location in the computer’s memory
- Every variable has a name, a type, a size and a value
- A memory cell is never empty. But its initial contents may be meaningless to your program.
- The current contents of a memory cell are destroyed whenever new information is placed in that cell.

Garbage values

```
char ch;
int intv;
float floatv;
long longv;
double doublev;
bool boolv;
```



```
char value   :
int  value   : 6946708
float value : 6.00597e-039
long value : 1980658781
double value : 4.68194e+260
bool value : 249
```

```
cout << "char value   : "<< ch << endl;
cout << "int  value   : "<< intv << endl;
cout << "float value : "<< floatv << endl;
cout << "long value : "<< longv << endl;
cout << "double value : "<< doublev << endl;
cout << "bool value : "<< boolv << endl;
```

Input / Output

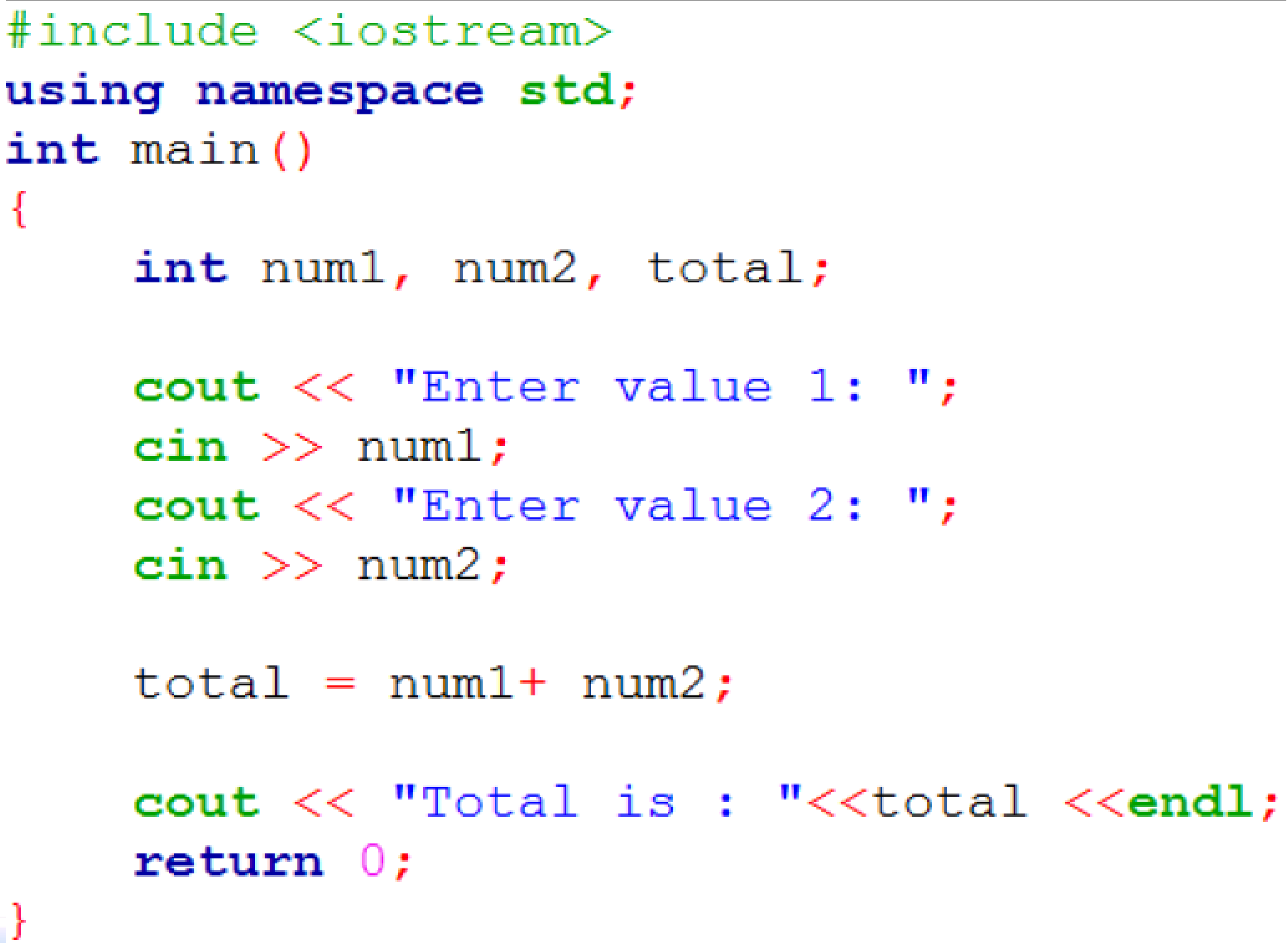
Standard Streams

- **cin** is the standard input, normally the keyboard.
- To input data from keyboard use the word '**cin**', followed by the 'Extraction' operator (>>)
  - cin >> x;
- Wait for a value to be entered at the keyboard and (when enter is pressed) will put that value into variable 'x'.

Exercise 3.4

- Write a C++ program to read two numbers from keyboard and display the total.

Answer

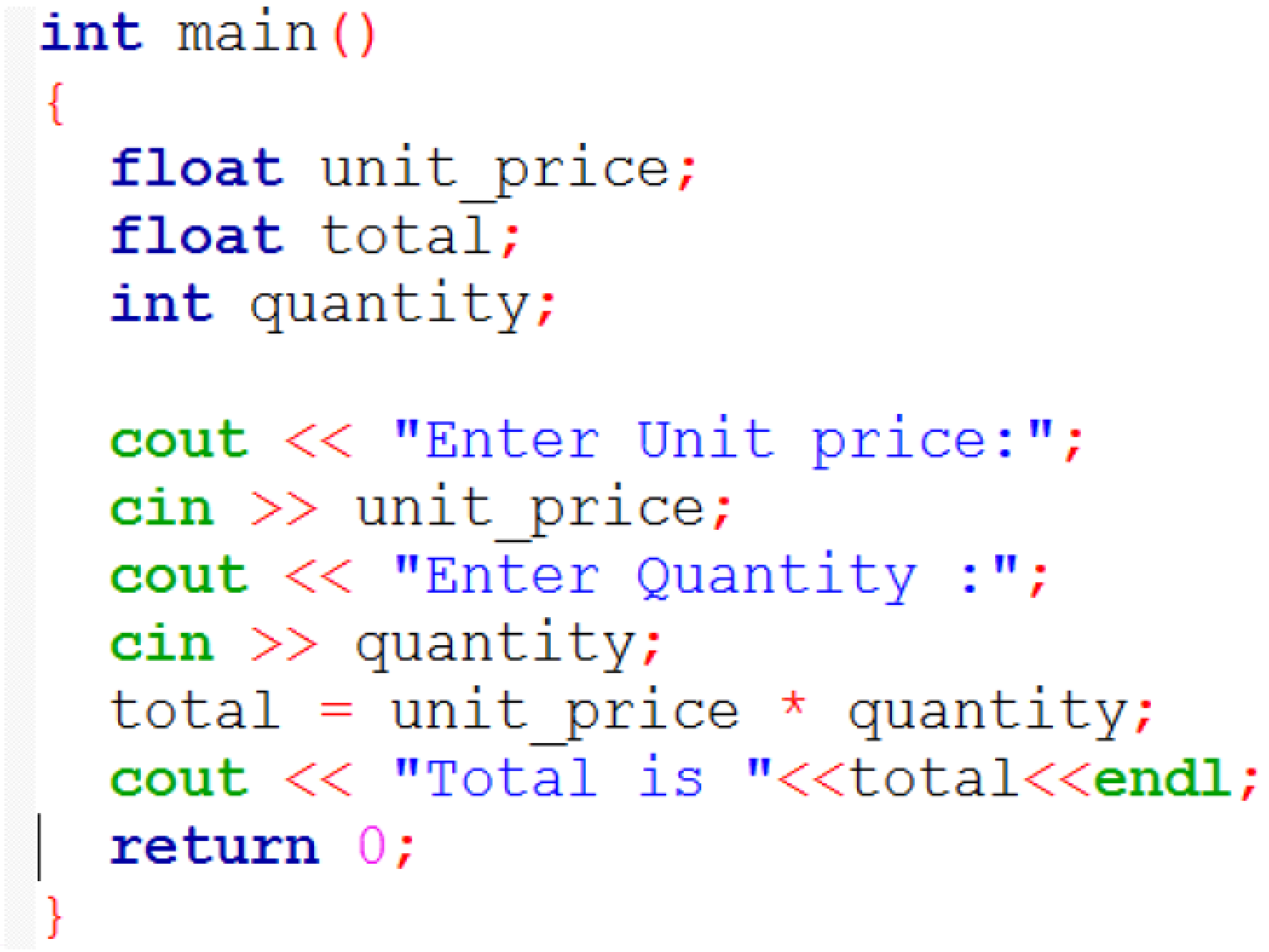


```
#include <iostream>
using namespace std;
int main()
{
    int num1, num2, total;

    cout << "Enter value 1: ";
    cin >> num1;
    cout << "Enter value 2: ";
    cin >> num2;

    total = num1+ num2;

    cout << "Total is : "<<total <<endl;
    return 0;
}
```

Exercise 3.5

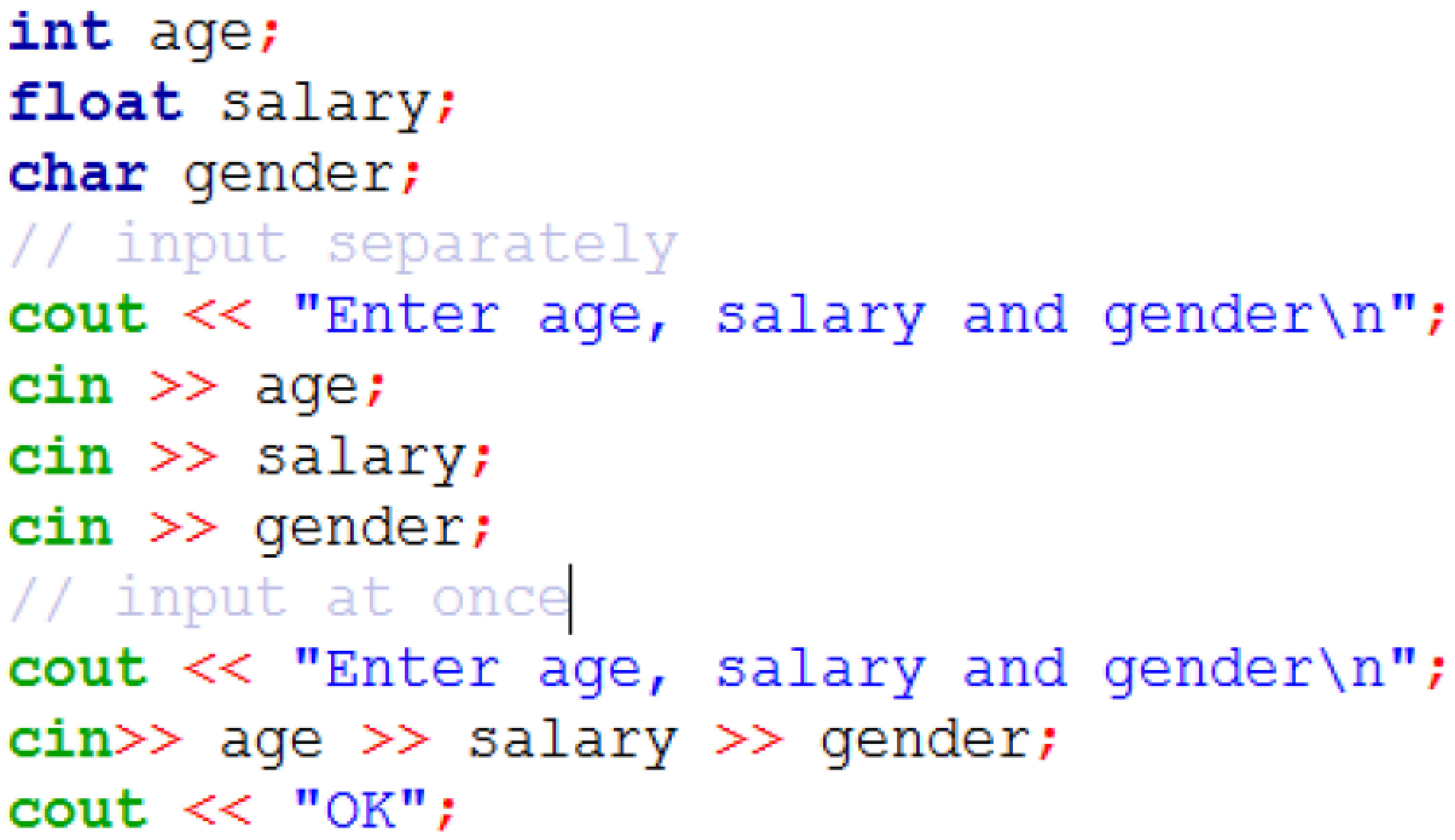
Create a C++ program to calculate and display total amount of given unit price and quantity of the some item.

Answer



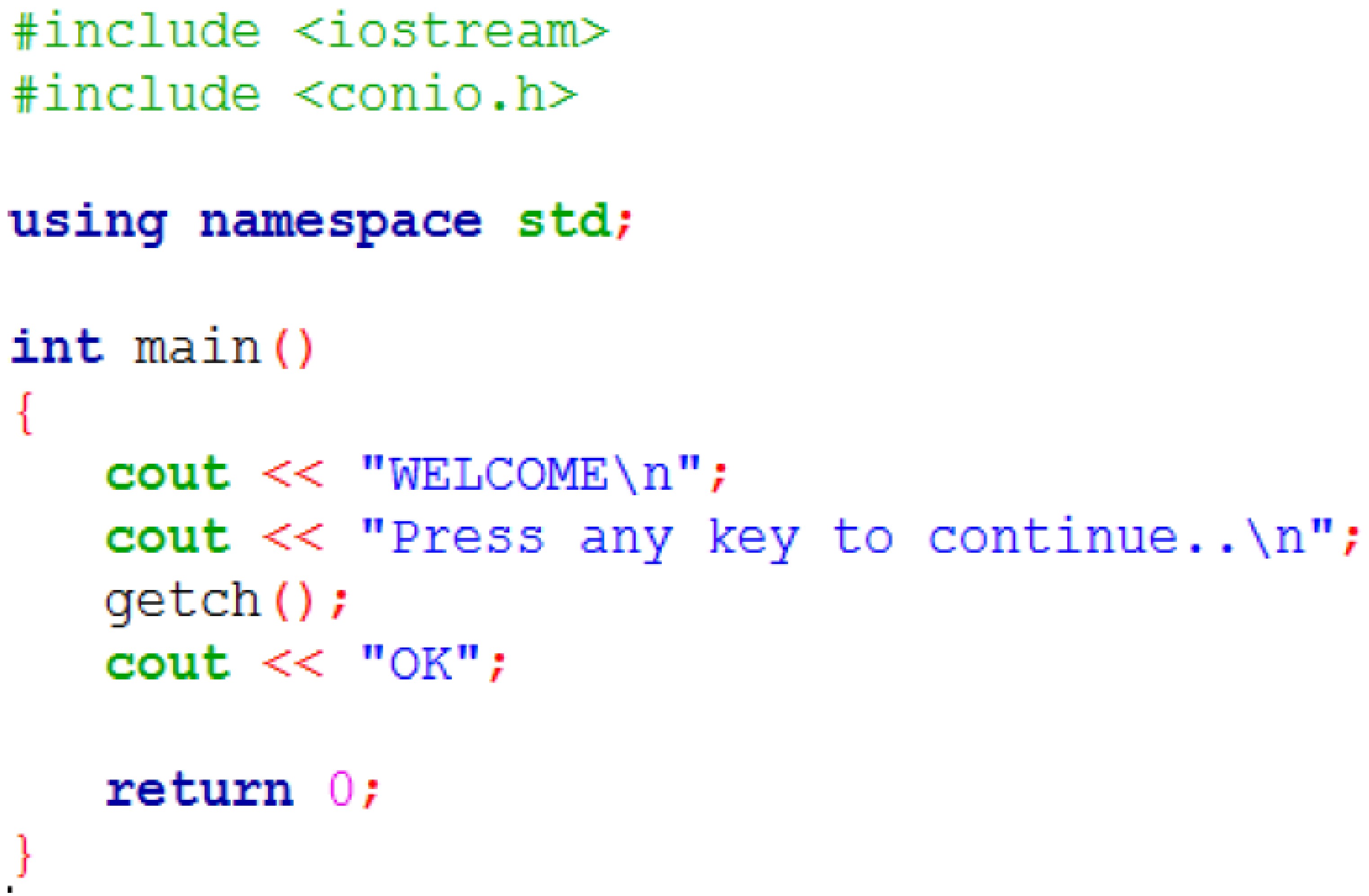
```
int main()
{
   float unit_price;
   float total;
   int quantity;

   cout << "Enter Unit price:";
   cin >> unit_price;
   cout << "Enter Quantity :";
   cin >> quantity;
   total = unit_price * quantity;
   cout << "Total is "<<total<<endl;
   return 0;
}
```

Input



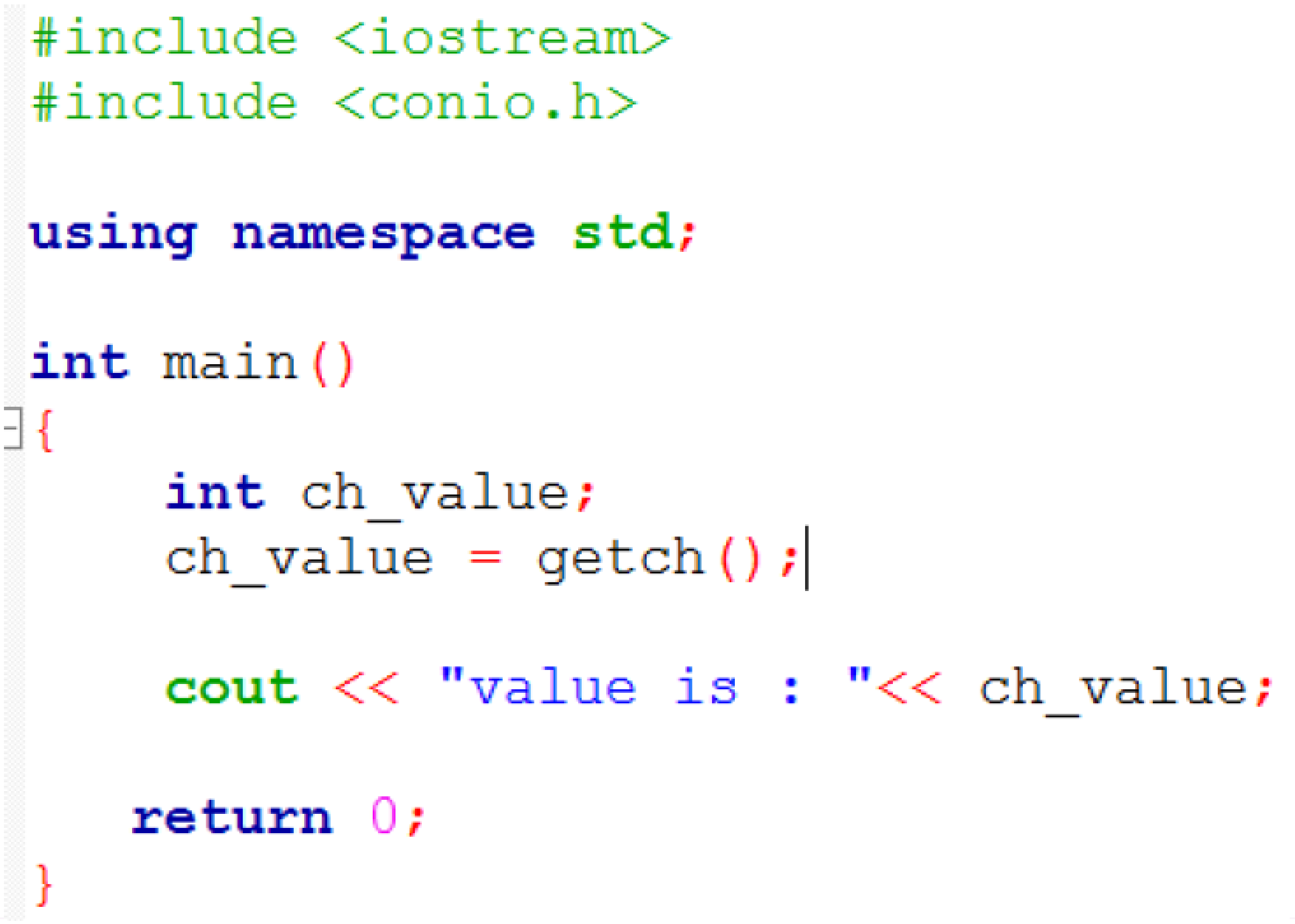
```
int age;
float salary;
char gender;
// input separately
cout << "Enter age, salary and gender\n";
cin >> age;
cin >> salary;
cin >> gender;
// input at once
cout << "Enter age, salary and gender\n";
cin>> age >> salary >> gender;
cout << "OK";
```

Wait until key press



```
#include <iostream>
#include <conio.h>

using namespace std;

int main()
{
    cout << "WELCOME\n";
    cout << "Press any key to continue..\n";
    getch();
    cout << "OK";

    return 0;
}
```

Read value of the key press

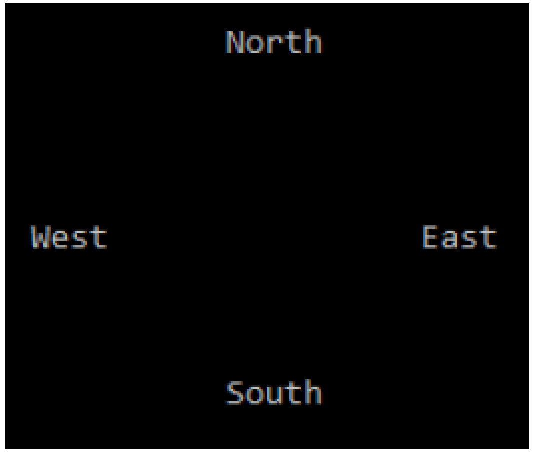


```
#include <iostream>
#include <conio.h>

using namespace std;

int main()
{
    int ch_value;
    ch_value = getch();

    cout << "value is : "<< ch_value;

  return 0;
}
```

Change the courser position

```
#include <windows.h>

using namespace std;

COORD coord = {0, 0};
void gotoxy (int x, int y)
{
    coord.X = x; coord.Y = y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), coord);
}
```

```
        gotoxy(15,1);
        cout<<"North";
        gotoxy(5,6);
        cout<<"West";
        gotoxy(25,6);
        cout<<"East";
        gotoxy(15,10);
        cout<<"South";
        getch();
```



```
          North




West                East



          South
```

Output

- To output data onto the screen, use the word ‘`cout`’, followed by the ‘insertion’ operator (<<).

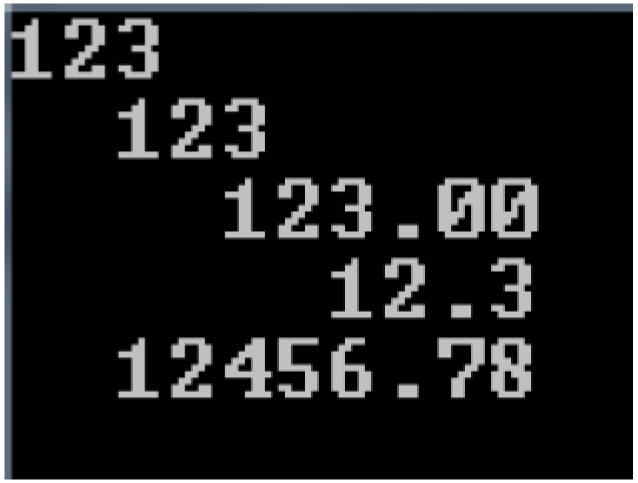
```
cout << "this is a String";
cout << "Value "<< total;
```

Change the output Format

- **Adjusting field widths**
- Use the width() member function to set the field width to display a value.
  - cout.width(10)

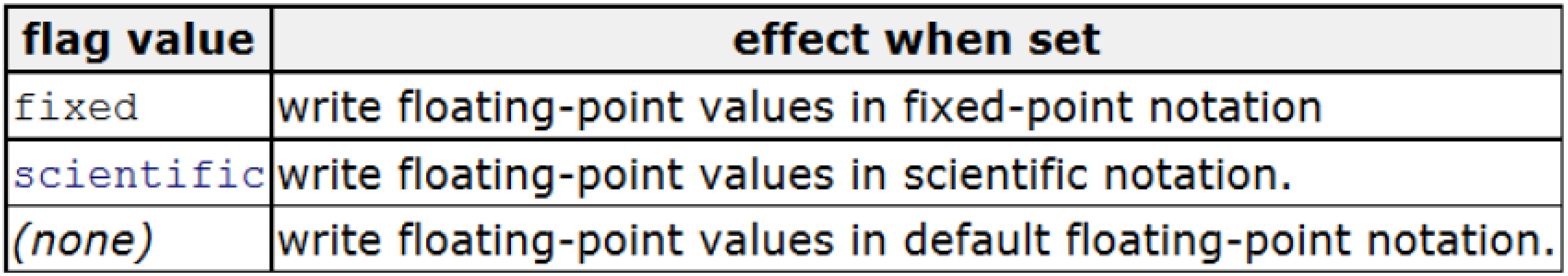
Example

```
#include <iostream>
using namespace std;
int main()
{
    cout << "123"<<endl;
    cout.width(5);
    cout << "123"<<endl;
    cout.width(10);
    cout << "123.00"<<endl;
    cout.width(10);
    cout << "12.3"<<endl;
    cout.width(10);
    cout << "12456.78"<<endl;
    return 0;
}
```

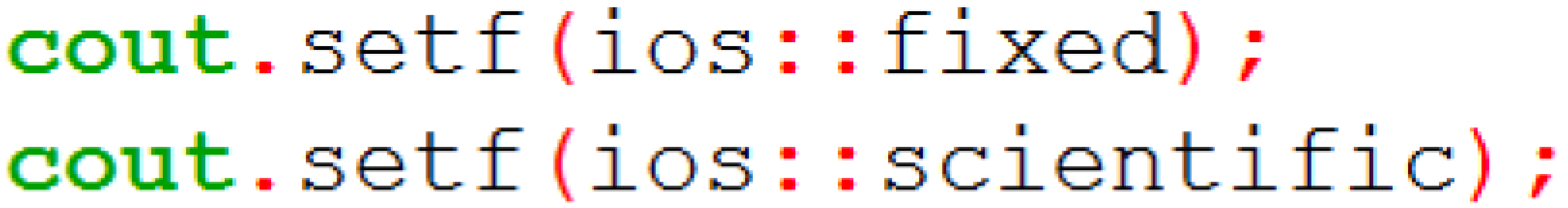


```
123
  123
    123.00
      12.3
  12456.78
```

Floating-point notation



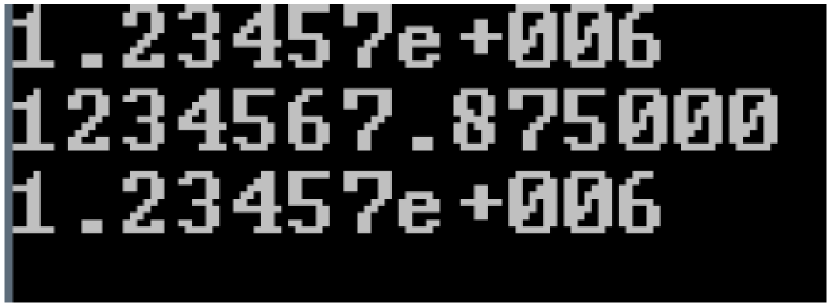
flag value	effect when set
`fixed`	write floating-point values in fixed-point notation
`scientific`	write floating-point values in scientific notation.
(none)	write floating-point values in default floating-point notation.



```
cout.setf(ios::fixed);
cout.setf(ios::scientific);
```

Example

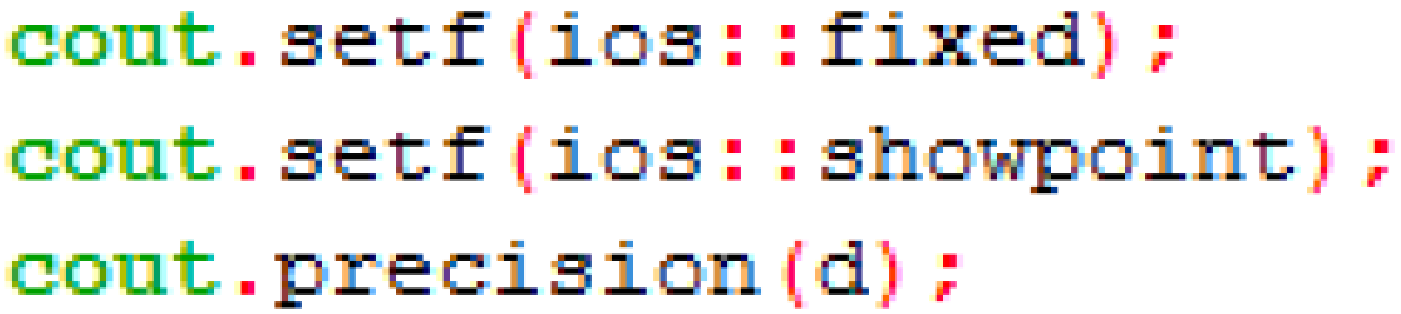
```
float num = 1234567.8977;
cout << num <<endl;
cout.setf(ios::fixed);
cout << num <<endl;
cout.setf(ios::scientific);
cout << num <<endl;
```



```
1.23457e+006
1234567.875000
1.23457e+006
```

Floating-point Precision

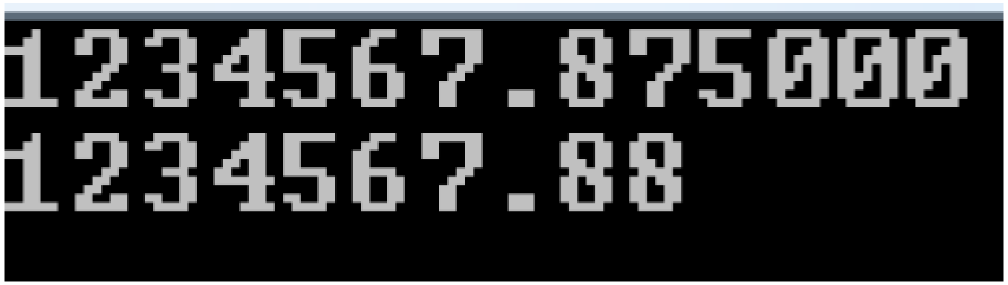
- Sets the ***decimal precision*** to be used to format floating-point values on output operations.
  - precision(number);



```
cout.setf(ios::fixed);
cout.setf(ios::showpoint);
cout.precision(d);
```

Example

```
float num = 1234567.8977;

cout.setf(ios::fixed);
cout << num <<endl;
cout.precision(2);
cout << num <<endl;
```



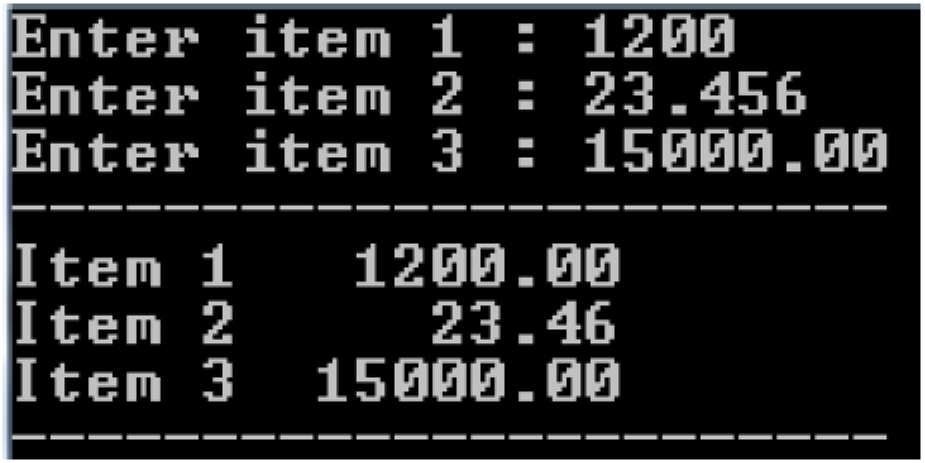
```
1234567.875000
1234567.88
```

Exercise 3.6

Write a C++ program to read price of the 3 items and print the total

Answer

```
float item1, item2, item3,total;
cout << "Enter item 1 : ";
cin >> item1;
cout << "Enter item 2 : ";
cin >> item2;
cout << "Enter item 3 : ";
cin >> item3;
cout.setf(ios::fixed);
cout.precision(2);
cout << "------------------------"<<endl;
cout << "Item 1";
cout.width(10);
cout << item1 <<endl;
cout << "Item 2";
cout.width(10);
cout << item2 <<endl;
cout << "Item 3";
cout.width(10);
cout << item3 <<endl;
cout << "------------------------"<<endl;
```



```
Enter item 1 : 1200
Enter item 2 : 23.456
Enter item 3 : 15000.00
------------------------
Item 1   1200.00
Item 2     23.46
Item 3  15000.00
------------------------
```

Examples

1. Write a C++ program to read 3 integer numbers and find the total and average

2. Write a C++ program which will convert a weight in KG to pounds and ounces.

   (1 Kg = 2.2046 pounds)

Example

- Write a C++ program which will convert a Celsius temperature into Fahrenheit

$$C = \frac{5}{9}(F - 32)$$

$$F = \frac{9}{5}C + 32$$

Summary

- Data types (char, int, float, long, double, bool)
- Variables (local , global and constant)
- Input fro keyboard (cin >>)
- Output (cout <<)
- Format your output
  - setwidth(x)
  - cout.setf()
  - cout.precision(x)

Thank you