Programming In C++

lecture 3 introduction about programming

Prepared By Dr. Ali Al-Sabaawi

Outlines

- Parts of C+++ program
- Language elements
- Keywords
- Data types
- Variables
- Backslash character
- Operators

Parts of C++ program

```
// A simple C++ program
#include <iostream>
using namespace std;

int main()
{
    cout << "Programming is great fun!";
    return 0;
}
```

The iostream file contains code that allows a C++ program to display output on the screen and read input from the keyboard.

The statement using namespace std; declares that the program will be accessing entities whose names are part of the namespace called std.

The reason the program needs access to the std namespace is because every name created by the iostream file is part of that namespace.

Parts of C++ program

This marks the beginning of a function. A function can be thought of as a group of one or more programming statements that collectively has a name.

Int stands for integer, it indicates that the function sends an integer value back to the operating system when it is finished executing.

Although most C++ programs have more than one function, every C++ program must have a function called main.

NOTE: C++ is a case-sensitive language. That means it regards uppercase letters as being entirely different characters than their lowercase counterparts. In C++, the name of the function main must be written in all lowercase letters.

Language Elements

Language Element	Description
Key Words	Words that have a special meaning. Key words may only be used for their intended purpose. Key words are also known as reserved words.
Programmer-Defined Identifiers	Words or names defined by the programmer. They are symbolic names that refer to variables or programming routines.
Operators	Operators perform operations on one or more operands. An operand is usually a piece of data, like a number.
Punctuation	Punctuation characters that mark the beginning or ending of a statement, or separate items in a list.
Syntax	Rules that must be followed when constructing a program. Syntax dictates how key words and operators may be used, and where punctuation symbols must appear.

Some of keywords

alignas	const	for	private	throw
alignof	constexpr	friend	protected	true
and	const_cast	goto	public	try
and_eq	continue	if	register	typedef
asm	decltype	inline	reinterpret_cast	typeid
auto	default	int	return	typename
bitand	delete	long	short	union
bitor	do	mutable	signed	unsigned
bool	double	namespace	sizeof	using
break	dynamic_cast	new	static	virtual
case	else	noexcept	static_assert	void
catch	enum	not	static_cast	volatile
char	explicit	not_eq	struct	wchar_t
char16_t	export	nullptr	switch	while
char32_t	extern	operator	template	xor
class	false	or	this	xor_eq
compl	float	or_eq	thread_local	

Data Types

There are many different types of data. Variables are classified according to their data type, which determines the kind of information that may be stored in them.

Data Type	Typical Size	Typical Range
`short int`	2 bytes	−32,768 to +32,767
`unsigned short int`	2 bytes	0 to +65,535
`int`	4 bytes	−2,147,483,648 to +2,147,483,647
`unsigned int`	4 bytes	0 to 4,294,967,295
`long int`	4 bytes	−2,147,483,648 to +2,147,483,647
`unsigned long int`	4 bytes	0 to 4,294,967,295
`long long int`	8 bytes	−9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
`unsigned long long int`	8 bytes	0 to 18,446,744,073,709,551,615

Identifiers

The names of variables, functions, labels, and various other user-defined objects are called identifiers. These identifiers can vary from one to several characters.

- The first character must be one of the letters a through z, A through Z, or an underscore character (_).
- After the first character you may use the letters a through z or A through Z, the digits 0 through 9, or underscores.
- Uppercase and lowercase characters are distinct. This means `ItemsOrdered` is not the same as `itemsordered`.

Variables

Variable Name	Legal or Illegal?
`dayOfWeek`	Legal.
`3dGraph`	Illegal. Variable names cannot begin with a digit.
`_employee_num`	Legal.
`June1997`	Legal.
`Mixture#3`	Illegal. Variable names may only use letters, digits, or underscores.

Variable Initializations

Some examples are

```
char ch = 'a';
int first = 0;
float balance = 123.23;
```

Local Variables

Variables that are declared inside a function are called local variables.

```
#include <iostream>
using namespace std;
int main()
{
    int x=44;
   cout << x << endl;
   return 0;
}
```

Global Variables

Global variables are known throughout the program and may be used by any piece of code.

These variables are declared before all program's functions.

```
#include <iostream>
using namespace std;
int x=44;
int main()
{
   cout << x << endl;
   return 0;
}
```

Const Variables

Variables of type const may not be changed by your program.

The compiler is free to place variables of this type into read-only memory (ROM).

```
#include <iostream>
using namespace std;
const int x=44;
int main()
{
   cout << x << endl;
    return 0;
}
```

Backslash Character Constants

```
#include <iostream>
using namespace std;
const int x=44;
int main()
{
   cout << “The value of x \n”<<x;
    return 0;
}
```

```
The value of x
44
```

Backslash Character Constants

Code	Meaning
\b	Backspace
\f	Form feed
\n	New line
\r	Carriage return
\t	Horizontal tab
\"	Double quote
\'	Single quote
\0	Null
\\	Backslash
\v	Vertical tab
\a	Alert
\?	Question mark
\N	Octal constant (where N is an octal constant)
\xN	Hexadecimal constant (where N is a hexadecimal constant)

Operators

- There are four main classes of operators: arithmetic, relational, logical, and bitwise.

- Generally, there are three types of operators: unary, binary, and ternary. These terms reflect the number of operands an operator requires.

- Unary operators only require a single operand.

  -, ++, --

  for example: -5

  ++, it is increment operator

  --, it is decrement operator

Arithmetic Operators

- Binary operators work with two operands.

Operator	Meaning	Type	Example
+	Addition	Binary	`total = cost + tax;`
-	Subtraction	Binary	`cost = total - tax;`
*	Multiplication	Binary	`tax = cost * rate;`
/	Division	Binary	`salePrice = original / 2;`
%	Modulus	Binary	`remainder = value % 3;`

Arithmetic Operators

- Binary operators work with two operands.

x++;
 is the same as
x = x + 1;

x--;
is the same as
x = x – 1;

Arithmetic Operators

The precedence of the arithmetic operators

highest	++ −−
	– (unary minus)
	* / %
lowest	+ −

Expression	Value
5 + 2 * 4	13
10 / 2 - 3	2
8 + 12 * 2 - 4	28
4 + 17 % 2 - 1	4
6 - 3 * 2 + 7 - 1	6

Relational and Logical Operators

Relational refers to the operator that tests or defines some kind of relation between two entities.

Logical refers to the ways these relationships can be connected.

p	q	p && q	p || q	!p
0	0	0	0	1
0	1	0	1	1
1	1	1	1	0
1	0	0	1	0

Relational and Logical Operators

Relational Operators

Operator	Action
>	Greater than
>=	Greater than or equal
<	Less than
<=	Less than or equal
= =	Equal
!=	Not equal

Logical Operators

Operator	Action
&&	AND
||	OR
!	NOT

Table 2-5. *Relational and Logical Operators*

Relational and Logical Operators

Highest	!
	> >= < <=
	== !=
	&&
Lowest	||

Combined Assignment Operators

The combined assignment operators, also known as compound operators, and arithmetic assignment operators.

Operator	Example Usage	Equivalent to
+=	`x += 5;`	`x = x + 5;`
-=	`y -= 2;`	`y = y - 2;`
*=	`z *= 10;`	`z = z * 10;`
/=	`a /= b;`	`a = a / b;`
%=	`c %= 3;`	`c = c % 3;`

Combined Assignment Operators

Example Usage	Equivalent to
`x += b + 5;`	`x = x + (b + 5);`
`y -= a * 2;`	`y = y - (a * 2);`
`z *= 10 - c;`	`z = z * (10 - c);`
`a /= b + c;`	`a = a / (b + c);`
`c %= d - 3;`	`c = c % (d - 3);`

The End