

Programming Language-II

Lecture # 3

Lecture Content

- Identifiers
- Keywords
- Data Types
- Variables(Rules, Declaration, Definition)
- Constants
- White space character
- Escape sequence character

Identifiers

- A C++ identifier is a name used to identify a user define element.
- C++ does not allow punctuation characters such as @ and % within identifiers.
- C++ is a case-sensitive programming language.
- Thus, **Age** and **age** are two different identifiers in C++.

Identifiers(Cont...)

- Here are some examples of acceptable and un acceptable identifiers,

1. Age
2. _age
3. 1_age
4. %age
5. myname50_temp
6. j
7. a23b9
8. retVal

Note: Identifiers name should be meaningful.

Keywords

- The following list shows the reserved words in C++. These reserved words may not be used as constant or variable or any other identifier names.

asm	else	new	this
auto	enum	operator	throw
bool	explicit	private	true
break	export	protected	try
case	extern	public	typedef
catch	false	register	typeid
char	float	reinterpret_cast	typename
class	for	return	union

Keywords(Cont...)

const	friend	short	unsigned
const_cast	goto	signed	using
continue	if	sizeof	virtual
default	inline	static	void
delete	int	static_cast	volatile
do	long	struct	wchar_t
double	mutable	switch	while
dynamic_cast	namespace	template	

Variables

- While coding, you need to use various variables to store various information.
- Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.
- You may like to store information of various data types like character, integer, floating point, double floating point, Boolean etc. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory.

Rule for declaring variables

- Variable may include letters, numbers and underscore(_).
- The first character of variable must be a letter or underscore .
- Blank spaces are not allowed in variable names.
- Both upper and lower cases are allowed. A user-defined variable is conventionally written in lower case.
- The constants are conventionally written in upper case.
- Special symbols cannot be used as variable name.
- Reserved word cannot be used as variable name. The names int, void and while are invalid variables.

Variables(Declaration)

- The process of specifying variable name and its type is called **variable declaration**.
- **Syntax:**
 - data_type variable_name;
- **Examples:**
 - int age;
 - float CGPA;
 - char grade;

Variables(Definition)

- The process of assigning a value to a variable is called **variable definition**.
- **Syntax:**
 - data_type variable_name=value;
 - **Examples:**
 - int age=25;
 - float CGPA=3.8;
 - char grade='A';

Datatypes

- There are two type of data types,
 - Pre defined (primitive)
 - User defined


Pre-defined data types

- C++ offers the programmer a rich assortment of built-in as well as user defined data types. Following table lists down six basic C++ data types

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

Pre-defined data types

- **bool**(Stores either value true or false)
- **char**(Store single character either it is special or normal)
- **int**(The most natural size of integer for the machine)
- **float**(A single-precision floating point value)
- **double**(A double-precision floating point value)
- **void**(Represents the absence of type)

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

Data types(Memory concern)

- The following table shows the variable type, how much memory it takes to store the value in memory, and what is maximum and minimum value which can be stored in such type of variables.

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
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295

Data types(Memory concern)

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	-2,147,483,648 to 2,147,483,647
signed long int	4bytes	same as long int
unsigned long int	4bytes	0 to 4,294,967,295
float	4bytes	+/- 3.4e +/- 38 (~7 digits)
double	8bytes	+/- 1.7e +/- 308 (~15 digits)
long double	8bytes	+/- 1.7e +/- 308 (~15 digits)

Data types(Memory concern)

- The size of variables might be different from those shown in the above table, depending on the compiler and the computer you are using.
- Example in next slide, which will produce correct size of various data types on your computer.

Data types(Memory concern)

```
#include<iostream>
using namespace std;
int main()
{
cout << "Size of char : " << sizeof(char) << endl;
cout << "Size of int : " << sizeof(int) << endl;
cout << "Size of signed int : " << sizeof(signed int)
    << endl;
cout << "Size of unsigned int : " << sizeof(unsigned
    int) << endl;

system("pause");
return 0;
}
```

Constants

- Constants refer to fixed values that the program may not alter and they are called **literals**.
- Constants can be of any of the basic data types and can be divided into Integer Numerals, Floating-Point Numerals, Characters, Strings and Boolean Values.
- Again, constants are treated just like regular variables except that their values cannot be modified after their definition.

Constants(cont...)

- There are two simple ways in C++ to define constants
 - Using **#define** preprocessor.
 - Using **const** keyword.

#define Preprocessor(Example)

```
#include <iostream>
using namespace std;
#define LENGTH 10
#define WIDTH 5
#define NEWLINE '\n'
int main()
{
    int area;
    area = LENGTH * WIDTH;
    cout << area;
    cout << NEWLINE;
    return 0;
}
```

The const Keyword(Example)

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

int main()
{
    const int LENGTH = 10;
    const int WIDTH = 5;
    const char NEWLINE = '\n';
    int area;
    area = LENGTH * WIDTH;
    cout << area;
    cout << NEWLINE;
    return 0;
}
```

White space characters

- In C++, whitespace refers to spaces, tabs and newlines.
- In some places in our code, whitespace is necessary whereas, in other places, it is just given to improve readability.
- For example, while writing '**int main()**', it is necessary to give a space between `int` and `main()` (as they are two different words).

Escape sequence character

Escape sequence	Description
\'	single quote
\"	double quote
\?	question mark
\\	backslash
\a	audible bell
\b	backspace
\f	form feed - new page
\n	line feed - new line
\r	carriage return
\t	horizontal tab
\v	vertical tab