CS-1002 Programming fundamentals (CYSEC)

Instructor : Jawad Hassan

jawad.hassan@nu.edu.pk

Fall 2022

September 6, 2022

FAST National University of Computer and Emerging Sciences, Islamabad

Lecture outline

- Variable and Literal
- Data Types
- C++ Special Characters
- Escape sequences / Control Characters
- Appendix-1: Programming style

Variable and Literal

- **Variables represent storage locations in the computer s memory.**
- **Literals are fixed values that are assigned to variables.**
- **Literals are also called constants**

Literal	Type of Literal
20	Integer literal
"Today we sold "	String literal
"bushels of apples.\n"	String literal
0	Integer literal

Variable

- Variables stores data at particular memory location:
  - Name
  - Type
  - Value
  - Size
  - Lifetime + Variable scope + Variable visibility
  - Polarity
  - Address

1. Variable Name

- Also referred to as **identifiers**
- **Rules:**
  - You cannot use a C++ keyword (reserved word) as a variable name.
  - Variable names in C++ can range from 1 to 255 characters.
  - All variable names must begin with a letter of the alphabet (a-z, A-Z) or an underscore(_).
  - After the first initial letter, variable names can also contain letters and numbers.
  - No spaces or special characters are allowed.
  - C++ is case Sensitive. Uppercase characters are distinct from lowercase characters.

C++ Reserved Keywords

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

2. 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.
- C++ recognizes following built-in data types which are designated by reserved words:
  - **char**
  - **short**
  - **float**
  - **long double**
  - **unsigned int**
  - **unsigned long int**
  - **long**
  - **double**
  - **unsigned char**
  - **unsigned short**
  - **boolean**

3. Size of Variable

- Memory occupied by a variable.
- Express in bytes [1 byte = 8 bits]
- Depends upon the type of variable
- **The sizeof operator is used to determine the size of a variable.**

4. Polarity

- **Signed or Unsigned**
- **Only integer or character can be signed**

Integer Data types

Integer variables can only hold whole numbers.

Data Type	Size	Range
`short`	2 bytes	-32,768 to +32,767
`unsigned short`	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`	4 bytes	-2,147,483,648 to +2,147,483,647
`unsigned long`	4 bytes	0 to 4,294,967,295

The char Data Type

- Use to store ASCII characters

Data Type	Size	Range
Unsigned Character	1 byte	0 to 255
Signed Character	1 byte	-128 to +127

Floating-Point Data Types

Floating-point data types are used to define variables that can hold real numbers.

Data Type	Key Word	Description
Single precision	`float`	4 bytes. Numbers between ±3.4E-38 and ±3.4E38
Double precision	`double`	8 bytes. Numbers between ±1.7E-308 and ±1.7E308
Long double precision	`long double*`	8 bytes. Numbers between ±1.7E-308 and ±1.7E308

*Some compilers use 10 bytes for long doubles. This allows a range of ±3.4E-4932 to ±1.1E4832

The bool Data Type

- **Boolean variables are set to either “true” or “false”.**

5. Variable Lifetime: **The lifetime of a variable the time period in which the variable is in memory.**

6. Variable scope: **A variable s scope is the part of the program that has access to the variable.**

7. Variable visibility: **Availability of a variable.**

8. Address of variable: **Where a variable reside in memory. Can be displayed using “&” operator.**

Variable deceleration and Initialization/Assignment

```
int  a;  //Declaration
a = 15; // Initialization
a = 30; //Assignment

int a=15; // Declaration + Initialization
int a, b, c, d; // Multiple Declaration
a=b=c=d=10; // Multiple Assignment

int a=b=c=d=10;  //
Int a,b,c,d=10;   //
```

Named Constants

- **Literals may be given names that symbolically represent them in a program.**

- **const qualifier is used.**

```
const int a =10;
```

Remember when a value is declared const it is assigned a value immediately and that value is Read only. i.e. we can not change it.

C++ Special Characters

Character	Name	Description
//	Double slash	Marks the beginning of a comment.
#	Pound sign	Marks the beginning of a preprocessor directive.
< >	Opening and closing brackets	Encloses a filename when used with the `#include` directive.
( )	Opening and closing parentheses	Used in naming a function, as in `int main()`
{ }	Opening and closing braces	Encloses a group of statements, such as the contents of a function.
" "	Opening and closing quotation marks	Encloses a string of characters, such as a message that is to be printed on the screen.
;	Semicolon	Marks the end of a complete programming statement.

Escape sequences / Control Characters

Escape Sequence	Name	Description
\n	Newline	Causes the cursor to go to the next line for subsequent printing.
\t	Horizontal tab	Causes the cursor to skip over to the next tab stop.
\a	Alarm	Causes the computer to beep.
\b	Backspace	Causes the cursor to back up, or move left one position.
\r	Return	Causes the cursor to go to the beginning of the current line, not the next line.
\\	Backslash	Causes a backslash to be printed.
\'	Single quote	Causes a single quotation mark to be printed.
\"	Double quote	Causes a double quotation mark to be printed.

Questions (???)

Thanks ☺

Programming style

- **Programming style refers to the way a programmer uses identifiers, spaces, tabs, blank lines, and punctuation characters to visually arrange a programs source code. these are some, but not all, of the elements of programming style.**
- Programming style refers to the way source code is visually arranged. Ideally, it is a consistent method of putting
  - **spaces** and **indentions** in a program so visual cues are created.
- These cues quickly tell a programmer important information about a program.

Appendix-1: Programming style

- **Every programmer has its own programming style and by time it can be improve.**
- **Every programmer is comfortable with his/her own programming style.**
- **Remember a programmer is not just writing code for him/her self. Other people (instructors or other peers) have to read or check code. Even a programmer have to read his/her code later.**
- **Good programming style gives clarity and understanding of code or program.**
- **Therefore, it is necessary to use good coding or programming practices.**
- **There are some common good programming styles or practices that should follow.**

Good programming practices

- **Although you are free to develop your own style, you should adhere to common programming practices. By doing so, you will write programs that visually make sense to other programmers.**

These are some Good programming guidelines for beginners

- Use **comments** when and where required. Simple rule use comments after three line or at least with a new scope.
- Use **indentation** whenever you enter in a new block. Use consistent indentation.
- Avoid **unnecessary parentheses**.
- Use **whitespace** for clarity for code.
- Use good **naming conventions**. Use reasonable and understandable names (identifiers) for variable and functions
- **Initialize variables** (Specially which are used for some calculations) when declared. Rule is that if a variable is used to calculate product initialize it with 1 and if a variable is used to calculate sum initialize it with 0. Some of variables can also be initialized with some other value.
- Use **small and understandable** instructions or statements
- Always **print out an appropriate message** before taking an input

Home work

- Number systems (Decimal, Octal, Binary and Hexadecimal) and their inter conversion
- ASCII character Table
- Attempt Exercise questions of Chapter 2.

References

- **Book:** STARTING OUT WITH **C++**

  {BOOK-1-CPP}

- **Chapter No.2**
- **Author: Tony Gaddis**