NOTES

Bachelor of Science (First Year)
CSC-101-Programming fundamentals using C
Title of the Unit : V – Overview of C
Module Name : Structure of a C program, data types, constants and variables
Module Number : 19

History of C

C was invented and first implemented by **Dennis Ritchie** on DEC PDP11 that used the UNIX operating system. C is the result of a development process that started with an older language called BCPL. BCPL was developed by Martin Richards, and it influenced a language called B, which was invented by Ken Thompson. B led to development of C in the 1970. ANSI established a committee in the beginning of 1983 to create a standard for C, which was implemented in 1987.

ASCII code: ASCII is an acronym for the **American Standard Code for Information Interchange**. ASCII is a code for representing English characters as numbers, with each letter assigned a number from 0 to 127. Eg. the ASCII code for uppercase M is 77. Most computers use ASCII codes to represent text, which makes it possible to transfer data from one computer to another. The standard ASCII character set uses just 7 bits for each character. There are several larger character sets that use 8 bits, which gives them 128 additional characters. Text files stored in ASCII format are sometimes called ASCII files. Text editors and word processors are usually capable of storing data in ASCII format, although ASCII format is not always the default storage format.

The extra characters are used to represent nonEnglish characters, graphics symbols, and mathematical symbols.

Character set: A character set is the mapping of characters to binary values. In 8 bit character sets, the values range from 0 to 255 and one character will be mapped to each of these values. If the users terminal is set to support a particular character set; every time he press a certain key or a combination of keys a corresponding character will be invoked.

Characters that C recognizes comprise of

Letters (upper case and lower case)

A B C D E .... a b c d e …

Digits 0 1 2 3 … and so on.

Special Characters

′ ″ ( ) * + / : = ! & $; <> % ? , . ˆ # @ ˜ ‘ { }

Helloworld program: The best way to learn a computer language is to start writing short programs that work and then gradually add complexity.

The traditional first C program prints out "Hello, World!" and looks something like this:

```
#include <stdio.h>
int main()
{
  /* my first program in C */
  printf("Hello, World! \n");
  return 0;
}
```

#include<stdio.h> includes a header file **stdio.h** into our program. This file contains definitions of printf() and scanf() that are the basic input output functions in C. We must include the file in every C program that has input and output.
int main() is the beginning of the main() function of the program. A C program is a collection of functions.main() is a compulsory function in every C program. The execution of a C program always begins with main(). The statements in main() are encloses within { and }.
/* my first program in C */ is a comment. A comment is written in English and provides some details about a statement or a program.Comments begin with /* and end with */. Comments are not compiled for errors and can be given anywhere in a C program.
printf("Hello, World! \n"); Displays a message **Hello, World!** on the output screenThe special character \n is a newline character that takes cursor to the next line.
return 0; This is always the last executed statement in a C program. It informs the Operating system that the program has executed successfully. The program returns an integer number 1 to the operating system denoting successful completion

Built in data types : Data types are a way of representing data storage formats. Two types of builtin data types: Fundamental/Basic data types (char, int, float, double, void, pointer).

Derived data types (array, string, structure).

char : character type values that are restricted to the defined ASCII characters.

int : positive and negative whole numbers.

float : positive and negative floating point numbers .

double : double precision floating point numbers.

void : used to explicitly declare a function as returning no value. void cannot be used to define a variable.

Pointer: used to store memory address of variables, is denoted by *.

Array – a finite sequence (or table) of variables of the same data type.

String – an array of character variables.

Structure – a collection of related variables of the same and/or different data types.The structure is called a record and the variables in the record are called members or fields.

Integer data type: Stores integer numbers : Positive integers, negative integers and a 0

Example int a=10;int num=-543;

Memory allocated = **2 bytes = 16 bits**

Min number = $(0000000000000000)_2 = (0)_{10}$

Max number = $(1111111111111111)_2 = 2^{16}-1 = (65535)_{10}$

Integer numbers can be positive, negative, and 0

unsigned int : Only positive integer numbers : Range is 0 to 65535

signed int: Positive and negative numbers: Range is divided into two parts: -32768 to +32767 including 0

unsigned long int : can store larger integers: 4 bytes : 0 to $2^{32}-1$

signed long int : $-(2^{31})$ to $+(2^{31} - 1)$ including 0

float and double data types

float: Stores floating point numbers: Positive & negative floating point numbers including 0.000000

Example: float pi=3.14159; float val=5.5;

Memory allocated = **4 bytes = 32 bits**

Min number = $(00000000000000000000000000000000)_2 = (0)_{10}$

Max number = $(11111111111111111111111111111111)_2 = 2^{32}-1$

Float numbers can be positive, negative, and 0.

So this range is divided into two parts from $-(2^{31})$ to $+(2^{31} - 1)$ including 0

double: Higher precision float numbers : **8 bytes**

char data type

Stores single character

'A'-'Z','a'-'z','0'-'9','*','$','>','@'……..

Example: char ch='#'; char gender ='M';

Memory allocated = **1 byte = 8 bits**

Min number = $(00000000)_2 = (0)_{10}$

Max number = $(11111111)_2 = 2^8-1 = (255)_{10}$

The ASCII values of characters range from 0 to 255

n binary bits give us a range of decimal numbers from 0 to $2^n - 1$

constants and literals The value stored in a constant can′t be changed during program execution. C has two types of constants, each with its own specific uses.**Literal constants**:20 and 'R' are the examples for literal constant: int count = 20; char name= 'R';**Symbolc constants**:A symbolic constant is a constant that is represented by a name (symbol) in the program. Like a literal constant, a symbolic constant can′t change. Whenever the User needs the constant′s value in the program, he can use its name as would use a variable name. The actual value of the symbolic constant needs to be entered only once, when it is first defined. A program that performs a variety of geometrical calculations frequently needs the value of π (PI) ie 3.14159 for its calculations. The value of π is constant. To calculate the circumference and area of a circle with a known radius, the user could write the following code:

Circumference = 3.14159 * (2 * radius);

Area = 3.14159 * (radius) * (radius);

If, however, the user define a symbolic constant with the name PI (π) and the value 3.14, he could write the following code:

Circumference = PI * (2 * radius);

Area = PI * (radius) * (radius);

C constants can be divided into two major categories:

(a) Primary Constants

(b) Secondary Constants

Rules for Constructing Integer Constants

An integer constant must have at least one digit. It must not have a decimal point. It can be either positive or negative. If no sign precedes an integer constant it is assumed to be positive. No commas or blanks are allowed within an integer constant. The allowable range for integer constants is -32768 to 32767Ex.: 426, +782, -8000

Rules for Constructing Real Constants

Real constants are often called Floating Point constants. The real constants could be written in two forms—Fractional form and Exponential form. A real constant must have at least one digit. It must have a decimal point. It could be either positive or negative. Default sign is positive. No commas or blanks are allowed within a real constant. Ex.: +325.34, 426.0, -32.76

Rules for Constructing Character Constants

Single special symbol enclosed within single inverted commas. Both the inverted commas should point to the left. For example, 'A' is a valid character constant whereas 'A' is not. The maximum length of a character constant can be 1 character. Ex.: 'A', ' ', '5', '='

V**ariables in C**

A variable is a way of referring to a memory location used in a computer program. This memory location holds values perhaps numbers or text or more complicated types of data like a payroll record.The value of a variable can change in course of program.Before the users can use a variable in a C program, it must be declared.

A variable declaration tells the compiler the **name and type** of a variable and optionally **initializes** the variable to a specific value. If the program attempts to use a variable that hasn't been declared, the compiler generates an error message. A variable declaration has the following form:

typename varname;

typename specifies the variable type and must be one of the keywords.

varname is the variable name, which must follow a set of rules.

User can declare multiple variables of the same type on one line by separating the variable names with commas:

```
int count, number, start;       /* three integer variables */
float percent, total;           /* two float variables */
char firstname;                 /* character type variable */
```

Rules for Constructing Variable Names

A variable name is any combination of 1 to 31 alphabets, digits or underscores. Some compilers allow variable names whose length could be up to 247 characters. Still, it would be safer to stick to the rule of 31 characters. Do not create unnecessarily long variable names as it adds to your typing effort.

The first character in the variable name must be an alphabet or underscore. No commas or blanks are allowed within a variable name. No special symbol other than an underscore (as in gross_sal) can be used in a variable name. Keywords cannot be used as variable names

Ex.: si_int

m_hra

pop_e_89

A character is a single letter, numeral, punctuation mark, or other such symbol. A string is any sequence of characters. Strings are used to hold text data, which is comprised of letters, numerals, punctuation marks, and other symbols. Using Character Variables, like other variable the users must declare chars before using them, and he can initialize them at the time of declaration. Here are some examples of character variables:

```
char a, b, c;            /* Declare three uninitialized char variables */
char code = 'x';  /* Declare the char variable named code and store the character x there */
code = '!';              /* Store ! in the variable named code */
```

To declare a variable as integer, follow the below syntax: **int variable_name;**

int is the type of the variable named variable_name.

'int' denotes integer type.

To declare a variable as float, follow the below syntax: **float variable_name;**

A float is a single precision floating point value.

To declare a variable as double, follow the below syntax: **double variable_name;**

A double is a double precision floating point value.

Examples: int i, j, k;

float f, salary;

double d;

C keywords

Keywords are the words whose meaning has already been explained to the C compiler (or in a broad sense to the computer). The keywords cannot be used as variable names because if we do so we are trying to assign a new meaning to the keyword, which is not allowed by the computer.The keywords are also called '**Reserved words'.** There are only 32 keywords available in C

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while