10

FUNDAMENTALS OF 'C'

10.1 INTRODUCTION

In this lesson you will be aware with the basic elements used to construct simple C statements. These elements include the C character set, keywords and identifiers, constants, data types, variables, arrays, declarations, expressions and statements. These basic elements are used to construct more comprehensive program components. Some of the basic elements needs very detailed information, however, the purpose of this type of basic elements is to introduce certain basic concepts and to provide some necessary definitions for the topics that follow in next few lessons.

10.2 OBJECTIVES

After going through this lesson you would be in a positions to

- recognize 'C' character set
- recognize keywords and identifiers
- define constants, data types, variables and arrays
- explain the concept of declaration
- describe the expressions and statements

10.3 THE C CHARACTER SET

C uses the uppercase letters A to Z, the lowercase letters a to z, the digit 0 to 9, and some special characters to form basic program elements (e.g., constants variables, operators, expression) as we discussed above. The special characters are listed below:

!	*	+	\	”	<
#	)	=	;	}	>
^	(	]	,	{	?
&	-	[	:	‘	/
%	_	= =	~	.	(blank)<>

C also uses some combinations of these characters such as \t, \n to represent special conditions such as horizontal tab and newline respectively. These character combinations are called escape sequences. Each escape sequence represents a single character, although it is a combination of two or more characters.

10.4 IDENTIFIERS AND KEYWORDS

Identifiers are names given to various elements of a program, such as variables, functions and arrays. Identifiers consist of digits and letters, in any order but the rule is first character should be a letter. Anyone can use both uppercase and lowercase letters. But uppercase and lowercase letters are not having same meaning e.g. RAM & ram are not same. The underscore character (_) can also be included and is considered to be a letter. Any identifier can start with underscore character, though this is not a good programming practice. Some of the following names are valid identifiers:

She, Z02, computer_02, _wind

Classes, areal, interest_rate, PERIOD

Now, we list out the following names which are not valid identifiers:

2^{nd}, “y”, ship-no, flag error

The first identifier name starting with 2 (i.e. a digit) is not valid, because first character must be a letter, illegal character “ ” is present in the “y” identifier, similarly ship-no, flag error are also invalid identifier name as hyphen & blank character are not allowed in the naming convention of identifiers.

An identifier can be arbitrarily long. Some implementation of C recognize only the first eight characters, though ANSI standard recognizes 31 characters.

Unlike identifiers, keywords are certain reserved words that have, standard, predefined meanings in C. These keywords can be used only for their intended purpose, they cannot be used as programmer-defined identifiers. The standard keywords are listed below:

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

It should be kept in mind that keywords are all in lowercase.

INTEXT QUESTIONS

1. What is the purpose of /t and /n character?
2. What is the basic difference between keyword and identifier ?
3. Is student-id is correct for identifier's name or not?

10.5 CONSTANTS

Constant in C refers to fixed values that do not change during the execution of a program. C supports four types of constants. They are integer, character, string & floating-point constants. As the name suggests integer & Floating point constants represent numbers. They are often referred to collectively as numeric-type constants The numeric-type constants must follow the following rules:

- Commas and blank spaces cannot be included within the constant.
- The constant can be preceded by a minus(-) sign if desired.
- The value of a constant cannot exceed specified minimum and maximum bounds. For each type of constant, these bounds will vary from one C compiler to another.

An integer constant refers to a sequence of digits. There are three types of integers, namely, decimal, octal and hexadecimal. A decimal integer constant can consist of any combination of digits taken from the set 0 through 9, proceded by an optional - or + sign. Various examples of decimal integer constants are:

123, – 345, 0, 5324, +62

An octal integer constant can consist of any combination of digits taken from the set 0 through 7, with a leading 0. Some examples of octal integers are:

0, 01, 0456, 05555,

A hexadecimal integer constant must begin with either 0x or 0x. It can then be followed by any combination of digits taken from the sets 0 through 9 and a through 7 (either upper or lowercase). Note that the letters A through F represent the numbers 10 through 15 respectively. Some examples of hexadecimal integers are:

0x2, 0xgf, 0xbc5, 0x

The largest integer value that can be stored is machine dependent. It is 32767 on 16-bit machines and 2147483647 on 32-bit machines. It is also possible to store larger integer constants on these machines by appending qualifiers such as U, L and UL to the constants. For example

56789U (unsigned integer), 876543217UL (unsigned long integer), 9765467L (Long integer)

Floating point constants is a base-10 number that contains either a decimal point or an exponent (or both). Some of the valid floating point constants are

0.   1.   0.3.   1.555E+8   .121265e18

If an exponent is present, its effect is to shift the location of the decimal point to the right if the exponent is positive or to the left if the exponent is negative. Floating point constants have a much greater range than integer constants. Typically the magnitude of a floating point constant might range from a minimum value of app. 3.4E-38 to a maximum of 3.4E+38. These constants are normally represented as double precision quantities in C. Each floating point constant will typically occupy 2 words (8 bytes) of memory.

Unlike integer constants floating point constants are approximations.

Character constant is a single character, enclosed in apostrophes (single quotation marks). Some of the valid character constants are 'A', 'y' '4' '$'

Each character constant has its equivalent ASCII value like 'A' has 65, 'y' has 121 '4' has 52 & so on.

Certain nonprinting characters as well as the double quote("), the apostrophe (') , the question mark (?) and the backslash(\) can be expressed in terms of escape sequences. An escape sequence always begins with a backward slash and is followed by one or more special characters. Several character constants, expressed in terms of escape sequences are:

'\n' '\t' '\b' '\\' '\'

The escape sequence \ϕ represents the null character which is used to indicate the end of a string.

String constants consist of any number of consecutive characters enclosed in double quotation marks. Some of the string constants are

"Red", "Mary Marry quite contrary", "2*(J+3)/J" and " "

Sometimes a backslash or a quotation mark must be included as a part of a string constant. These characters must be represented in terms of their escape sequences. The compiler automatically places a null character at the end of every string constant, as the last character within the string. This character is invisible when the string is displayed. One point must be noted here that a character constant 'A' and the corresponding single-character string constant "A" are not equivalent.

10.6 DATA TYPES

C supports different types of data, each of which may be represented differently within the computer's memory. The various basic data types are listed below in tabular form:

Data type	**Description**	**Typical memory requirements**
int	integers quantity	2 bytes or 1 word
char	simple character	1 byte
float	floating-point number	1 word (4 bytes)
double	double-precision floating -point number	2 words (8 bytes)

Each data type requires different memory requirements which may vary from one C compiler to another.

C compilers that are written for personal computers represent a word as 4 bytes. Some basic data types can be augmented by using the data type qualifiers short, long, signed & unsigned as discussed above. For example, integer quantities can be defined as short int, long int or unsigned int.

If short int and int both have the same memory requirements (e.g, 2 bytes), then long int will generally have double the requirements. (e.g., 4 bytes) Similarly if int & long int both have the same memory requirements (e.g., 4 bytes) then short int will have half the memory requirements(e.g. 2 bytes).

An unsigned int means all the bits are used to represent the numerical value unlike in the case of ordinary int in which the leftmost bit is reserved for the sign. Thus the size of an unsigned int can be approximately twice as large as an ordinary int. For example if an ordinary int can vary from -32,768 to +32,767 then an unsigned int can vary from 0 to 65,535.

As discussed above that floating point numbers have a decimal point. The C compiler differentiates between floating point numbers & integers because they are stored differently in the computer. Floating point number sometimes are referred to as real numbers. They include all the numbers between the integers. Some of the differences are listed below between floating point numbers and integer.

1. Integer include only whole numbers, but floating point numbers can be either whole or fractional.

2. Integers are always exact, whereas floating point numbers sometimes can lead to loss of mathematical precision.

3. Floating point operations are slower in execution and often occupy more memory than integer operations.

Floating point numbers may also be expressed in scientific notation. For example, the expression 2345.34e6 represents a floating point number in scientific notation. The letter e stands for the word exponent. The exponent is the whole number following the letter e; the part of the number before the letter e is called the mantissa.

The number 2345.34e6 should be interpreted as: 2345.34 times 10 to the 6th power.

The char type is used to represent individual characters. Hence, the char type will generally require only1 byte of memory. With most compilers, a char data type will permit a range of values extending from 0 to 255.

Some compilers permit the qualifier long to be applied to float or to double. e.g long float or long double.

Every identifier that represents a number or a character within a C program must be associated with one of the basic data types before the identifier appears in an executable statement. This is accomplished via a type declaration as described later on.

INTEXT QUESTIONS

4. What are the basic types of constants?
5. Define hexadecimal integer constant briefly.
6. What are two parts of floating point numbers?

10.7 VARIABLES AND ARRAYS

A quantity which may vary during program execution is called a variable. Each variable has a specific storage location in memory where its value is stored. The variable is given a name, the variable name is the “name tag” for the storage location. The value of the variable at any instant during the execution of a program is equal to the number stored in the storage location identified by the name of the variable. The variable name shown in Fig. 10.1 is Sum and its contents is 12.5

Fig 10.1: variable name and its contents

In C, the word identified is used as the general terminology for names given to variables, functions, constants, structures etc.

A variable names may consist of letters, digits and underscore (_) character, subject to the following conditions:

The programmer must follow the rules for naming variables. These are as follows:

1. They must begin with a letter
2. ANSI standard recognizes a length of 31 characters. However, the length should not be normally more than eight characters, since only first eight characters are treated as significant by many compilers.
3. The uppercase & lowercase letters are different e.g., Marry, MARRY , marry are different variables names.
4. Variable name should not be a keyword.
5. No special characters, such as period, comma, semicolon, blanks are permitted in variable names.

The variable name should be chosen to be descriptive of the role that the variable is designed to play. Variable can be assigned any value depending upon their data types. For example,

Sum = 12.5 as shown in the fig 1.1

declares a variable called 'Sum' and assigns a value 12.5 to it. The assignment statement has the following general format

variable = expression.

Notice that the variable name appears to the left side of the equal sign which is called the assignment operator. Only one variable name may appear to the left of the equal sign. A statement such as

```
C+d=e;
```

is not acceptable in C language

The array is another kind of variable that is extensively used in C. An array is an identifier that refers to a collection of data items which all share the same name. The data items must all be of the same type (i.e., all integers, all characters). The individual data items are represented by their corresponding array elements. The individual array elements are distinguished from one another by the value that is assigned to a subscript. For example student is a 10-element array. The first element is referred to as student[0], the second as student[1], and so on. The last element will be student[9].

The number shown in the square brackets is called subscript. For first element the value of subscript is 0, for second element it is 1 & so on. For an n-element array, the subscripts always range from 0 to n-1.

Arrays can be categorized into integer arrays, character arrays, one-dimensional arrays, multidimensional arrays. For now, we will confine our attention to only one type of array: the one dimensional character array, often called a char type array. This type of array is generally used to represent a string. Each array element will represent one character within the string. Thus, the entire array can be thought of as an ordered list of characters.

Note that an n-character string will require an (n+1) element array, because of the null character(\0) automatically placed at the end of the string as stated earlier.

For example string "National" is to be stored in a one dimensional character array called school. Since National contains 8 characters, school will be an 9-element array. Thus, school[0] will present the letter N, school[1] represent[a] & so on. Their subscript values are given below:

0	N
1	a
2	t
3	i
4	o
5	n
6	a
7	l
8	\0

INTEXT QUESTIONS

7. What is a variable?
8. What is the format of the assignment statement?
9. How many variables may appear on the left of the equal sign in an assignment statement ?
10. Define an array.

10.8 DECLARATIONS

A declaration associates a group of variables with a specific data type. In 'C' language all variables must be declared before they appear in executable statements.

A declaration has following format to declare a variable

```
<data type> <variable name>;
```

Each array variable must be followed by a pair of square brackets containing a positive integer which specifies the size of the array.

e.g

```
int x,y, z;

float  sq_root, sq_root1;

char student_name[10];
```

Thus x,y,z are declared to be integer type variables, sq_root & sq_root1 are floating point variables & student_name is character type array whose size is 10 elements.

The programmer can write above declarations in the following manner:

```
int x;

int y;

int Z;

float sq_root;

float sq_root1;

char student_name[10];
```

As stated above, integer type variables can be declared to be short integer for smaller integer quantities or long integer for larger integer quantities. They are declared by writing short int and long int or simply short and long respectively. e.g. short int x,y,z and short x,y,z both are same.

Similarly you can write unsigned int or unsigned as the type indicator.

Initial values can be assigned to variables within a type declaration. The declaration must have a format

<data type><variable name....>=constant of appropriate type;

e.g. int x=10;

char school ='N' ;

float sq_root =0;

From these declarations it is clear that x is an integer type variable and has initial value as 10, school is a character type variable and has initial value as 'N', and float sq–root is a floating point variable and has initial value as 0.

A character-type array can also be initialized within a declaration. For this the square brackets of an array name must be empty. It has following format:

<data type> <array name[ ]> = "<string>";

e.g.char school[] = "National";

This statement means 'school' will be an 9-element character array. The first 8 elements will represent the 8 characters within the word National & the 9th element will represent the null character(\0) which is automatically added at the end of the string.

This can also be written as

Char school[9]="National";

The size of the array must be specified correctly for this type of statements because if the size is small say char school[8] ="National" the characters at the end of the string (in this case, the null character) will be lost and if the size is too large say

char school[18] ="National"

the extra array elements may be assigned spaces, or they may be filled with meaningless characters.

INTEXT QUESTIONS

11. How do you declare a integer of short data type?
12. Can you initialize a char-type array within a declaration ?

10.9 EXPRESSIONS AND STATEMENTS

An expression represents a single data item, such as a number or a character. The programmers can use operators in the expressions in C like other programming languages. Expressions can also represent logical conditions that are either true or false.e.g.

```
x+y
y=z
x=y+z
```

The first expression involves use of addition operator(+), the second expression involve use of assignment operator(=) and third expression involves use of both addition operator & assignment operator.

Statement causes the computer to carry out some action. There are three types of statements in 'C' language. They are expression, compound & control statements.

An expression statement consists of an expression followed by a semicolon. For example, following two expression statements cause the value of the expression on the right of the equal sign to be assigned to the variable on the left.

```
x=5;
x=y+z;
```

A compound statement consists of several individual statements enclosed within a pair of braces ({ and}).

Unlike an expression statement, a compound statement does not end with a semicolon.

Control statements are used to create special program features, such as logical tests, loops and branches.

Symbolic constants are names for a sequence of characters. The characters may represent a numeric constant, a characters constant or a string constant. So, symbolic constant allows a name to appear in place of a numeric or character, or string constant. At the time of compiling each occurrence of a symbolic constant get replaced by its corresponding character sequence.

These are usually defined at the beginning of a program. For example:

```
#define student class
```

Where students represents a symbolic name, and class represents the sequence of characters associated with the symbolic name. It should be kept in mind that class does not end with a semicolon, since a symbolic constant definition is not a like C statement. Further if you include semicolon with class, this semicolon would be treated as a part of the numeric, character or string constant that is substituted for the symbolic name.

INTEXT QUESTIONS

13. What is an expression?
14. What is symbolic constant?

10.10 WHAT YOU HAVE LEANRT

In this lesson you have learnt about 'C' character set, keywords and identifier. Now you are well aware about constants and its different types such as int, char, float, signed, unsigned, long etc. There are some rules to declare variable names and arrays that you have learnt in this lesson. The concept of declarations, expression and statements have also been discussed for the benefit of the learners.

10.11 TERMINAL QUESTIONS

1. State the rules for naming identifiers.
2. What are the unsigned integer constants?
3. Explain an escape sequence?
4. Define a subscript? What range of values is permitted for the subscript of a one dimensional n-element array?

10.12 KEY TO INTEXT QUESTIONS

1. /t for tab character and /n for newline character.
2. Identifier are names given to various elements of a program, whereas keywords are reserved words that have standard, pre-defined meaning in C.
3. No, because 'hyphen' is not allowed.
4. Integer, character, string & floating,
5. A hexadecimal integer constant must begin with either x or X. It can then be followed by any combination of digits through 9 and A to F (lowercase or uppercase).
6. Mantissa & exponent part.
7. It is a symbolic name for a memory location in which value is stored.
8. Variable=expression;
9. Only one
10. An array is an identifier refers to a collection of data items which all have the same name .
11. Short int<variable names or short <variable name>
12. Yes
13. Expression represents a single data item, such as a number or a character.
14. Symbolic constants are names for a sequence of characters.