Unit 2
Elements of C

C Character Set

- The set of characters that are used to words, numbers and expression in C is called character set.
- The combination of these characters form words, numbers and expression in C.
- The characters in C are grouped into the following four categories.
  i) Letters or alphabets
  ii) Digits
  iii) Special Characters
  iv) White space

Letter or Alphabets

- Uppercase alphabets – A…Z
- Lowercase alphabets – a…z

Digits

- All decimal digits – 0 1 2 3 4 5 6 7 8 9

Special Characters

- Some special characters are: Comma (,), Period (.), Semicolon (;), Asterisk (*), Ampersand (&) etc.

White Spaces

i) Blank space
ii) Horizontal tab
iii) Vertical tab
iv) Carriage return
v) New line or line feed
vi) Form feed

C Tokens

- Token is the smallest individual element in C.
- For example, we cannot create a sentence without using words; similarly, we cannot create a program in C without using tokens in C.
- Therefore, we can say that tokens in C is the building block or the basic component for creating a program in C language.
- Tokens in C language can be divided into the following categories:
  - Keywords
  - Identifiers
  - Strings
  - Operators

- Constant
- Special Symbols

Keywords

- Keywords are predefined or reserved words that have special meanings to the compiler.
- These are part of the syntax and cannot be used as identifiers in the program.
- The following are the keywords or reserved words in the C programming language:

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

Identifiers

- C identifiers represent the name in the C program, for example, variables, functions, arrays, structures, unions, labels, etc.
- An identifier can be composed of letters such as uppercase, lowercase letters, underscore, digits, but the starting letter should be either an alphabet or an underscore.

Rules for constructing C identifiers

- The first character of an identifier should be either an alphabet or an underscore, and then it can be followed by any of the character, digit, or underscore.
- It should not begin with any numerical digit.
- In identifiers, both uppercase and lowercase letters are distinct. Therefore, we can say that identifiers are case sensitive.
- Commas or blank spaces cannot be specified within an identifier.
- Keywords cannot be represented as an identifier.
- The length of the identifiers should not be more than 31 characters.
- Identifiers should be written in such a way that it is meaningful, short, and easy to read.

Example of valid identifiers

total, sum, average, _m _, sum_1, etc.

Escape sequence

- An escape sequence in C language is a sequence of characters that doesn't represent itself when used inside string literal or character.
- It is composed of two or more characters starting with backslash \.

- For example: \n represents new line.

Delimiters

- A delimiter is a unique character or string that marks the start or stop of a particular statement, string, or function body set.
- Examples of delimiters include:
  - o Using parentheses or round brackets: ( )
  - o Using curly brackets: { }
  - o Escape code or comments: /*
  - o Double quotes are used to define string literals: " "

Variables

- A variable is a name of the memory location.
- It is used to store data.
- Its value can be changed, and it can be reused many times.

Syntax:

```
data_type variable_name;
```

Rules for defining variables

- A variable can have alphabets, digits, and underscore.
- A variable name can start with the alphabet, and underscore only. It can't start with a digit.
- No whitespace is allowed within the variable name.
- A variable name must not be any reserved word or keyword, e.g. int, float, etc.

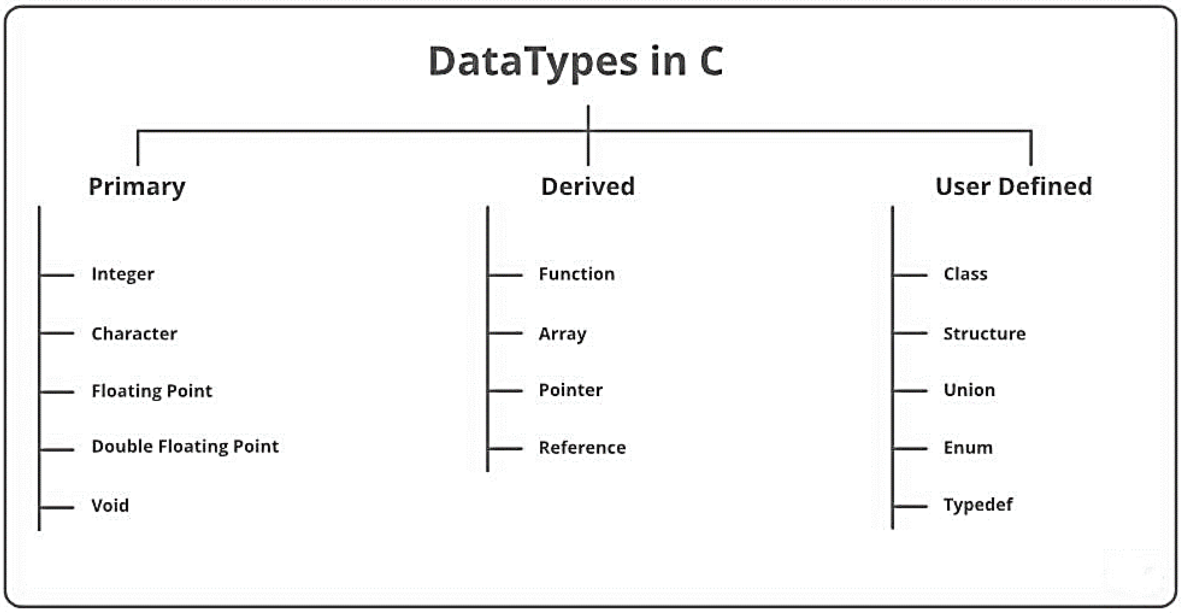
Types of Variables in C

There are many types of variables in c:

i) Local variable
ii) Global variable
iii) Static variable
iv) Automatic variable
v) External variable

Data types (Basic, Derived, and User Defined)

- A data type specifies the type of data that a variable can store such as integer, floating, character, etc.
- The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.
- The data types in C can be classified as follows:



Constants/ Literals

- Constants refer to fixed values that the program may not alter during its execution.
- These fixed values are also called literals.
- Constants can be of any of the basic data types like an integer constant, a floating constant, a character constant, or a string literal.

Expressions

- An expression is a formula in which operands are linked to each other by the use of operators to compute a value.
- An operand can be a function reference, a variable, an array element or a constant.
- There are four types of expressions exist in C:
    - Arithmetic expressions
    - Relational expressions
    - Logical expressions
    - Conditional expressions

Statements

- A statement is a command given to the computer that instructs the computer to take a specific action, such as display to the screen, or collect input.
- A computer program is made up of a series of statements.
- Some type of statements are:
    - Expression Statements
    - Compound Statements
    - Selection Statements

- Iterative Statements
- Jump Statements

Comments

- Comments in C language are used to provide information about lines of code.
- It is widely used for documenting code.
- There are 2 types of comments in the C language.
    - Single Line Comments (//)
    - Multi-Line Comments (/*…*/)

Structure of a C program

- The sections of a C program are listed below:
    1. Documentation section
    2. Preprocessor section
    3. Definition section
    4. Global declaration
    5. Main function
    6. User defined functions

Documentation section

- It includes the statement specified at the beginning of a program, such as a program's name, date, description, and title.
- It is represented as:

```
//name of a program

        OR

/*

        Overview of the code

*/
```

Preprocessor section

- The preprocessor section contains all the header files used in a program.
- It informs the system to link the header files to the system libraries.
- Example:

```
#include<stdio.h>
#include<conio.h>
```

Define section

- The define section comprises of different constants declared using the define keyword.

- It is given by:

```
#define PI 3.14
```

Global declaration

- The global section comprises of all the global declarations in the program.
- It is given by:

```
float num = 2.54;
 int a = 5;
char ch ='z';
```

Main function

- main() is the first function to be executed by the computer.
- It is necessary for a code to include the main().
- It is like any other function available in the C library.

User defined functions

- The user defined functions specified the functions specified as per the requirements of the user.
- For example, color(), sum(), division(), etc.

Exercise

1. Differentiate between constant and literals. Why do we need to define the type of data?(5) [TU 2079]
2. Discuss structure of a C Program with suitable example. (10)(5) [TU 2074, 2078]
3. What is variable? How is it different from constant? How do you write comments in C? (5) [TU 2078]
4. What is data type? Why do we need it in programming? Explain any three basic data types with example. (5) [TU 2077]
5. What is preprocessor directives? Discuss # define directive with example. (5) [TU 2075]