C Language Fundamental

1) **Character Set:** The character set of C represent alphabet, digit, special symbol and whitespace character.

Types	Character set
Alphabet	A to Z, a to z
Digit	0 to 9
Special Symbol	` ~ ! @ # $ % ^ & * ( ) - _ = + \ | { } [ ] ; : " ' . , ? / > <
Whitespace character	Blank space, Horizontal tab, new line (Enter), Backspace etc.

2) **Identifier:** The name given to programming element is called as Identifier. The programming element is like variable name, label name, constant name, function name, array name etc.

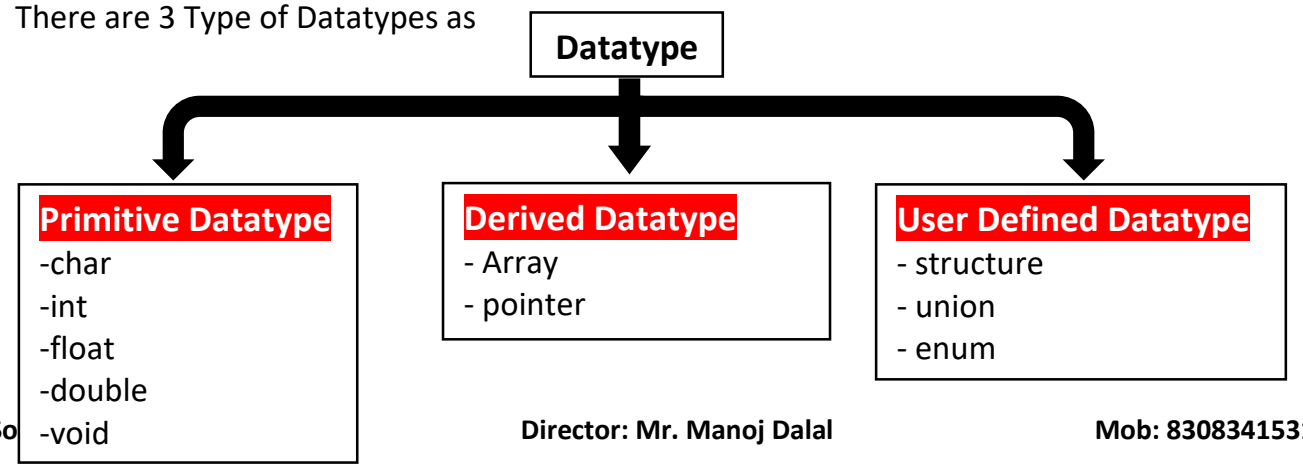
Rules for naming identifier

1. It should be start with alphabet.
2. It should not be start with digit.
3. No special symbol allowed except underscore (_).
4. Keyword is also not allowed.
5. Whitespace not allowed
6. The maximum length of identifier is up to 31 character allowed but recommended up to 14-15 character.

3) **Keyword:** Keyword is also called as Reserved words. Each and every keywords has predefined meaning define by compiler in c language. There are 32 keywords as

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

4) **Datatype:** Datatype represent the type of value which we can stored on memory using variable. Datatype has 3 properties as 1) Type of value 2) Memory size 3)Range
There are 3 Type of Datatypes as



Primitive Datatype: It is also called as Basic or in built datatype which is provided by compiler.

Datatypes	Size	Range	Usage
char	1 byte	-128 to +127	This datatype used when we want to store single character.
int	4 byte	-2147483648 to +2147483647	This datatype used when we want to store numerical value which does not contain decimal point.
float	4 byte	1.175494e-038 to 3.402823e+038	This datatype used when we want to store numerical value which contain decimal point.
double	8 byte	2.225074e-308 to 1.797693e+308	This is similar to float. Only Difference between float and double is range.
void	0 byte	No Range	This datatype used to return nothing from function.

5) Variable: The name given to memory area is called as variable. By using name of variable we can access the content on memory, address of variable and memory size occupied by variable.

Syntax to declare variable

```
datatype var1, var2,……varn;
```

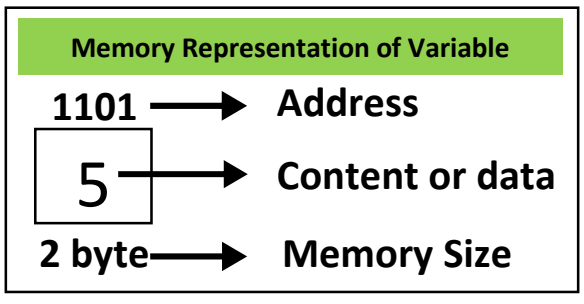
datatype → It represent the type of value we can store in variable.
var1, var2 → The name given to variable according to rule of identifier.

Example:
```
int x;
float p;
```
In above example we declare variable x of type int and p of type float. **x** will be only int type of value can be stored and **p** will be store float type of value.

Initialization of Variables:

Assigning initially value to variable using Assignment operator is called as initialization of variable. There are two way we can initialize value to variable.

1) At the time of declaration of variable
   Example:
   ```
   int x=5;
   Float p=7.13;
   ```
2) After the declaration of Variable
   Example:
   ```
   int x;
   float p;
   x=5;
   p=7.13;
   ```



Char type Variable: When we want to assign character value to character variable then it can be written character value in single quote. Internally it does not store character value on memory instead of that ASCII code will be stored.

Example:
```
char ch;
ch='M';
```

In above example character value 'M' will not be stored instead of that ASCII code 78 will be stored in the form Binary on memory.

Assignment Operator(=): This operator used to assign value to variable or assign another variable value or result of expression or value return by function.

Example:
```
int x, y=13;
x=7;        //assign value 7 to x
x=y;        //assign value of y i.e. x=13
x=x*y-13; // assign result of expression. i.e. x=78
x=sqrt(169); // assign value return by sqrt function. i.e. x=13
```

Types of Variable:

1) **Local Variable:** A variable which is declared within any function is called as local variable. It can be accessible only within function in which function they can be declared.
2) **Global Variable:** A variable which is declared all above the function and after #include pre-processor is called as global variable. It can be accessible in the program.
3) **Formal Parameter:** A variable which is declared within any function header is called as formal parameter. It can be accessible only within function in which function header they can be declared.

   Example:

```
#include<stdio.h>
int y=13; // Global Variable
int main()
{
    void demo();
    int x=7; //local variable
    printf("\n x=%d",x);
    printf("\n y=%d",y);
}
void demo()
{
    printf("\n x=%d",x); // this statement gives error because x is local variable of main
                                function which is not accessible outside the function.
    printf("\n y=%d",y);
}
```

6) **Constant:** If we want to define a variable whose value cannot be changed then can use the concept constant. There are 2 way to define constant in c language.
   1. Using #define pre-processor
      Syntax: #define constantname constantvalue
      Example:
      ```
      #define X 13
      #define PI 3.14
      ```
   2. Using Keyword const
      Syntax: datatype const constantname = constantvalue;
      Example:
      ```
      int const X=13;
      Const float PI=3.14;
      ```

Note:

1. If we define constant variable using #define pre-processor then its scope is compulsory global.

2. If we define constant variable using const keyword then its scope is local as well as gobal.

7) **Enumeration Constant:** We can create user defined integer constant datatype using **enum** keyword.
Syntax: enum datatypename {var1, var2, ……., varn};
Example:
1. **enum color {red, black, white, yellow, green, blue};**
In this example we create user defined datatype named color and create variable red, black, white, yellow, green, blue. This all variable having default constant integer value as red=0, black=1, white=2, yellow=3, green=4, blue=5.
2. **enum color {red, black, white=11, yellow, green, blue};**
Now, the value of red=0, black=1, white=11, yellow=12, green=13, blue=14.

8) **Type modifiers:** Type Modifiers are keywords used to change the properties of primitive datatypes. There are 4 modifiers in c language as **short, long, signed and unsigned**. Type modifiers are classified in to 2 types
1) Size modifiers – short and long
2) Range modifiers – signed and unsigned

Datatypes	Size	Range
char	1 byte	-128 to +127
signed char	1 byte	-128 to +127
unsigned char	1 byte	0 to 255
int	4 byte	-2147483648 to +2147483647
signed int	4 byte	-2147483648 to +2147483647
unsigned int	4 byte	0 to 4294967295
short int	2 byte	-32768 to +32767
signed short int	2 byte	-32768 to +32767
unsigned short int	2 byte	0 to 65535
long int	4 byte	-2147483648 to +2147483647
signed long int	4 byte	-2147483648 to +2147483647
unsigned long int	4 byte	0 to 4294967295
long long int	8 byte	-9223372036854775808 to 9223372036854775807
signed long long int	8 byte	-9223372036854775808 to 9223372036854775807
unsigned long long int	8 byte	0 to 18446744073709551615
float	4 byte	1.175494e-038 to 3.402823e+038
double	8 byte	2.225074e-308 to 1.797693e+308
long double	10 byte	3.4E-4932 to 1.1E+4932
void	0 byte	No Range

9) Operator and Expression:
Operator: A symbol that represent certain operation is called as operator.

Expression: The combination of operator and operand is called as expression.
c=a+b
Types of Operator: The type of operator is depend on how much operand required operator to perform operation.
1. **Unary Operator:** The operator which required one operand is called as unary operator.

2. **Binary Operator:** The operator which required two operand is called as Binary operator.
3. **Ternary Operator:** The operator which required three operand is called as ternary operator.

Categories of Operators:

1) **Arithmetic Operator:** An arithmetic operator performs mathematical operations such as addition, subtraction, multiplication, division etc on numerical values (constants and variables). This all operators are Binary Operator.

Operator	Meaning of Operator
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	modulus division (remainder after division)

Example: int x=5, y=3;

x+y →8
x-y → 2
x*y →15
x/y → 1
x%y → 2

In above example x/y expression gives result 1 which is incorrect result because x and y of type int then result also in int. But correct result of expression x/y is 1.66. we can achieve correct result in this situation by using concept typecasting as

(float)x/y → 1.66

2) **Relational Operator:** A relational operator specify relationship between two operands or expression. This operator used to create conditional expression in program and the conditional expression gives result in the form of 1(true) or 0(false). Relational operators are used in decision making and loops.

Operator	Meaning of Operator	Example
==	Equal to	5 == 3 is evaluated to 0
>	Greater than	5 > 3 is evaluated to 1
<	Less than	5 < 3 is evaluated to 0
!=	Not equal to	5 != 3 is evaluated to 1
>=	Greater than or equal to	5 >= 3 is evaluated to 1
<=	Less than or equal to	5 <= 3 is evaluated to 0

3) **Logical Operator:** This operator used when we want to combine two condition to test at a time. This operator also gives result in the form of 1(true) or 0(false).

Operator	Meaning	Example
&&	Logical AND. Return 1(true) only if all conditions are true.	int x = 5, y = 2; ((x==5) && (y>5))→ 0.
||	Logical OR. Return 1(true) only if at least one condition is true	int x = 5, y = 2; ((x==5) || (y>5)) → 1.
!	Logical NOT. Return 1(true) if the condition is 0(false) or vice versa.	int x = 5; !(x!=5) → 1.

4) **Assignment Operator:** we have already learn about assignment operator in above.

5) Increment and Decrement Operator

Increment Operator: The ++ symbol is called as increment operator. This operator increase value by 1 of given operand. We can use two different way increment operator.

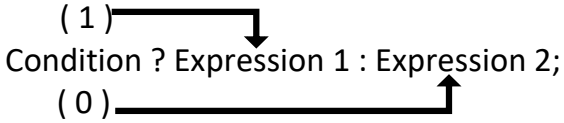
Post Increment	Pre Increment
1) If we placed increment operator after operand is called as post increment operator. 2) In this first assign then increment. Ex - int m=1,n; n=m++; output: m=2, n=1	1) If we placed increment operator before operand is called as pre increment operator. 2) In this first increment then assign. Ex - int m=1,n; n=++m; output: m=2, n=2

Decrement Operator: The -- symbol is called as decrement operator. This operator decrease value by 1 of given operand. We can use two different way decrement operator.

Post Decrement	Pre Decrement
1) If we placed decrement operator after operand is called as post decrement operator. 2) In this first assign then decrement. Ex - int m=1,n; n=m--; output: m=0, n=1	1) If we placed decrement operator before operand is called as pre decrement operator. 3) In this first decrement then assign. Ex - int m=1,n; n=--m; output: m=0, n=0

6) **Conditional Operator (? :) :** This operator used when we want to handle 2 or more than 2 decision in program. This is the only ternary operator in any language.

Syntax:



Example: int n=5;

n%2==0 ? printf("\n Even") : printf("\n Odd");

Output: Odd

7) **Bitwise Operator:** The Bitwise operators are used to perform bit-level operations on the operands. The operators are first converted to bit-level and then the calculation is performed on the operands. The mathematical operations such as addition, subtraction, multiplication, etc. can be performed at bit-level for faster processing.

Operators	Meaning of operators
&	Bitwise AND
|	Bitwise OR
^	Bitwise XOR

~	Bitwise NOT or Complement
<<	Bitwise Left Shift
>>	Bitwise Right Shift

Example: short int x=7, y=13;

x = 0000 0000 0000 0111

y = 0000 0000 0000 1101

---

x&y = 0000 0000 0000 0101 = 5 → Bitwise AND

x|y = 0000 0000 0000 1111 = 15 → Bitwise OR

x^y = 0000 0000 0000 1010 = 10 → Bitwise XOR

Bitwise left shift Operator (<<)

Example: x<<2

x = **00**00 0000 0000 0111

Remove 2 Zero's

x = 0000 0000 0001 11**00 = 28**

Add 2 Zero's

Bitwise right shift Operator (<<)

Example: x>>2

x = 0000 0000 0000 01**11**

Remove 2 Zero's

x = **00**00 0000 0000 0001 **= 1**

Add 2 Zero's

8) Other Operator
   1. comma
   2. typecast
   3. sizeof: This operator gives memory size(int) of variable or value or datatype.

10) Escape Sequences character: Some characters, such as backspace and control characters, have no visible image. Such characters are known as non-printable characters. Other characters (single and double quotation marks, question mark, and backslash) have special meaning in the many programming languages. Our programs are not able to use any of these characters directly. Instead, we can use an escape sequence to represent such char. An escape sequence begins with a backslash.

Escape sequence	Description
\a	Bell Alert – A beep (sound) is generated indicating the execution of the program to alert the user.
\b	Backspace – To move cursor left by one position.

\f	Form feed - It is a page-breaking ASCII control character. It forces the printer to eject the current page and to continue printing at the top of another.
\n	New line – It moves cursor on next line.
\r	Carraige return – It moves cursor at the beginning of same line.
\t	**Horizontal tab** - We use it to shift the cursor to a couple of spaces to the right in the same line.
\v	Vertical tab -
\'	Single quote – To print single quote
\"	Double quote – To print double quote
\\	Backslash – To print backslash
\000	To represent Octal number
\xhh	To represent Hexadecimal number
\0	Null character

11) **Format Specifier:** The format specifier is used during input and output. It is a way to tell the compiler what type of data is in a variable during taking input using scanf() or printing using printf().

Format Specifier	Type
%c	character
%d	signed integer
%e or %E	scientific notation of floats
%f	float values
%g or %G	similar as %e or %e
%hi	signed integer (short)
%hu	unsigned integer (short)
%i	unsigned integer
%l or %ld or %li	long
%lf	double
%Lf	long double
%lu	unsigned int or unsigned long
%lli or %lld	long long
%llu	unsigned long long
%o	octal representation
%p	pointer
%s	string
%u	unsigned int
%x or %X	hexadecimal representation
%n	prints nothing
%%	prints % character

12) Standard Input and Output Function:

When we are working on computer we need to perform input output operations using standard input device (keyboard) and output device (screen). Similarly we need to perform input output operation in program then c language provide input output functions in header file **stdio.h** to interact or communicate with input output device.

1. **printf( ):** This function used to print message or output on screen.

   Syntax: int printf(const char*, [Argument list]);
   Where
   int – it is return type to specify total no. of character print on screen.
   printf – This is name of function and f stands for formatted.

   This function having two parameter one is control string (const char *) which contains Text, escape sequence character and format specifier. Second parameter is called argument list which is optional.
2. **scanf( ):** This function used to read any type of input from keyboard.

   Syntax: int scanf(const char*, Argument list);
   Where
   int – It returns the number of fields that were successfully converted and assigned.
   printf – This is name of function and f stands for formatted.

   This function having two parameter one is control string (const char *) which contains escape sequence character and format specifier. Second parameter is called argument list which is compulsory.