Ch-1 Introduction to C

1.1 What is programming ?

-> First of all what is Program ?

-> Program is set of instructions or command which give to the computer to complish certain task.

-> this process is called programming.

-> means process to doing program is called as programming.

-> History of C

C is programming language, developed in 1972.

-> C was developed by Dennis Ritchie at AT & T bell laboratories located in USA.

-> Important of C

-> C is mother of all language and when we want to enter in programing field then we must learn C language

-> C is Basic programming language.

→ Compiler and Interpreter.

→ Compiler and Interpreter are one type of translator which convert code High language to low level language.

→ Code → Compiler/Interpreter → PC

(human language) (process). (0,1) (Binary language)

C, C++, English

output (0,1)

Compiler/Interpreter ← output (user).

(0,1 → H.L)

→ Difference Between Compiler & Interpreter

* Compiler	* Interpreter.
→ Read code as all at same time.	→ Read code (line by line)
→ Fast	→ Slow

~~Ch 2 datatype, constant, variable.~~

1.2 printf() function.

-> printf function is used print message in output.

-> printf(" .... message .... ");

-> what will be write in double " ", it will output / show in the output screen.

-> \n <- new line, to break line.

-> \t <- tab, to print 4 space at one time.

-> Basic program syntax.

```
header file  #include<stdio.h> <- standard input output
             #include<conio.h> <- console  "  "

main() <- main method / function
{
    clrscr();
    printf("Hello world");

    getch();
}
```

body.

Ch-2 : Datatype, Constant & variable.

2.1 What is datatype ?

-> datatype is simply type of data.

-> means which type of data we have, we can divide into some types according to their nature.

-> Example.

like water, juice, soda etc are one type of substance but we consider it in 'liquid' form, so liquid is datatype for water, juice etc.

same we cannot consider stone as a liquid substance, because there are some rules which types of data consider in the liquid.

-> like we have multiple type of data, but we will learn in c, mentioned bellow.

1. Integer : -1, 0, 1, 2 ..... (Natural value)
2. float : -0.1, -2.3, 4.5 ... (decimal value)
3. char : 'A', 'a', '#', (any single character)
4. double : 75.85.. , (decimal value + Big value, ex. mobile NO, Adhar NO.

-> what is variable:

-> where we can store some value.

- Syntax to declare variable.

datatype variableName;

- Syntax to initiallize variable

datatype variableName = value;

Examples.

```
int a;                      int a = 18;
float percentage;           float per = 98.75;
double NO;                  double NO = 9825486060
char x;                     char x = 'A';
```

-> format specifier.

-) it specifies datatype when we print data or get data from the user.

ex. printf("%d", varName);

%d Formate specifier.

-> where we put format specifier in " ", value will be print there..

-> For Integer - int - %d
float - float - %f
char - char - %c
double - double - %lf

Extra

When we print point value and output comes like.

95.750000 , like this unnecessary zeros.

-> then how to remove unnessary zeros or fix the number of digit after . point.

-> use. "%.2lf" , "%.2f"

output for -> 95.75

it specify digit after point.

<code>

-> printf("%d", integerVariable);

-> int a; // 18
double b; // 9825486060
char c; // A
float f; // 18.81

printf("%d %lf %c %f", a, b, c, f);

2.2 keywords : pre-reserved word in language.

→ ex. int, float, for, if, else..

constant : it is used to fix value of variable.

ex. if i want to fix value of variable name pi, which is 3.14

X int pi ; // pi variable create in float because 3.14 is decimal value.

* const float pi = 3.14 ;

↑ using this keyword we can set or fix value of variable for entire program.

scanf()

→ scanf function is used to scan / get value from the user.

→ to take input from the user.

→ Syntax:

scanf("%d", &varname);

↑ current specifier of variable.

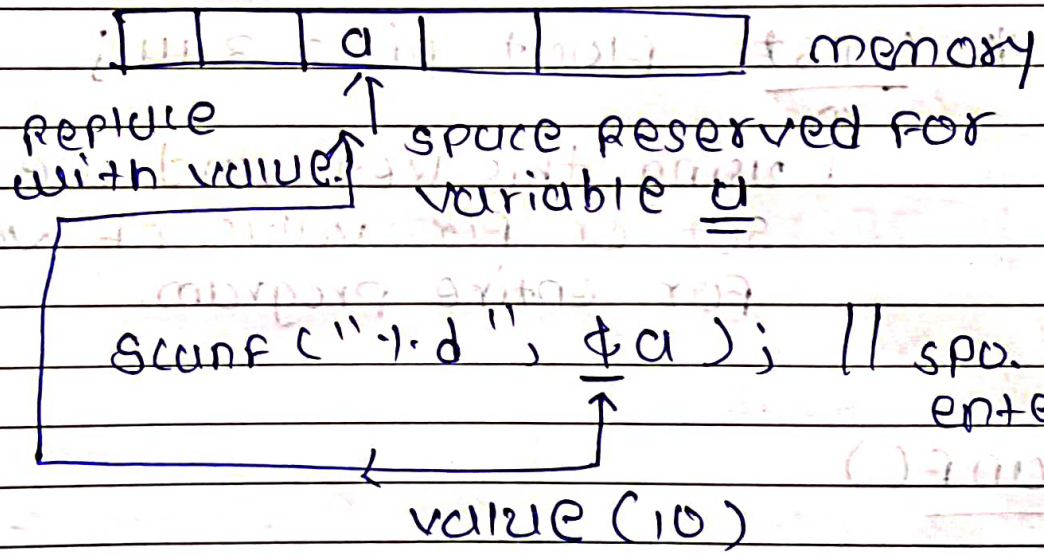
→ scanf("%d", &varname)

here this is sign of address.

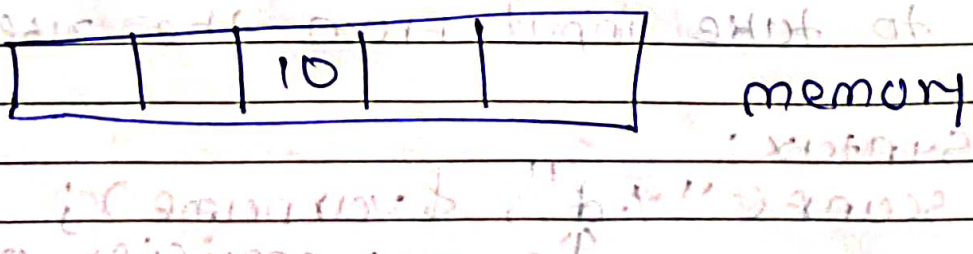
→ its indicate which value will be entered by user, it will be store at which location.

flow of scanf.

```
int a;
```



// after scanf:



Ch-3: Operator & Expression

Date: / /

3.1 Type of operator

(1) Unary operator

→ which have/want only one value

→ ex. a++, x--

(1) ++ increament.

→ int x = 0;
x++;
→ x = 1 ← output

(2) -- decrement

int x = 1;
x--;
x = 0 ← output

(2) Binary operator.

→ which have/want minimum two value.

→ a+b, a>b, x=10

* types of Binary operator.

(1) Arithmatic operator (+, -, *, /, %)

(2) logical operator (&&, ||, !)

(3) Relational Operator (>, <, >=, <=, ==)

(4) assignment operator (=, +=, -=, *=, /=).

→ if we store deimal value in integer output will be integer.

int x = 3.14;

→ output 3

(1) Arithmatic Operator.

-> Arithmatic Operator is used to perform mathematical operation on the values.

(2) Logical operator.

-> Logical operator is used to connect multiple condition.

&&

Condition-1	C-2	Output
0	0	0
0	1	0
1	0	0
1	1	1

||

C-1	C-2	Output
0	0	0
0	1	1
1	0	1
1	1	1

(3) Relational operator.

-> Relational operator is used to compare two values.

(4) Assignment Operator.

-> Assignment operator is used to assign value to the variable.

-> x = 10;

-> x += 10 -> x = x + 10

-> x++ -> x = x + 1

→ Now you got it what is operator

→ Operator is used to perform operation / process on the values.

4. control structure.

4.1 → if <- when we have only one condition
if else <- " " two condition
else if ladder <- step by step
Nested if <- one into another.

→ when we have condition that time we will use if else to solve this conditions.

→ Simple if.

```
if (condition)
{
    // true
}
else
{
    // false.
}
```

→ ladder. step by step.

```
if (condition)
{ // true
}
else if (condition)
{
}
else if (condition)
{
}
;
```

~~6.2~~ → Nested (one into another)

6.2

```
if( )
{
    if( )
    {
    }
    else
    {
    }
}
else
{
    if( )
    {
    }
    else
    {
    }
}
```

Nested and ladder works same, just way of writing is different.

→ when we have multiple conditions then we will use Nested / ladder.

Extra

* Algorithem.

→ step by process or flow of code.

ex. WAP to print sum of two number.

→ Algorithem for this code.

step 1 : start
step 2 : declare two varicble a, b
step 3 : take input from user in a, b
step 4 : sum = a+b
step 5 : print sum
step 6 : end.

→ it will help you to understand every complex code.

→ Note: make ruff algorithem for every program in book before you start coding.

4.3
~~5.3~~

switch case:

→ when we have one input and multiple output / cases that time we will use switch case.

→ ex. mcq → question - 1, answer = 4.
but we can select only - 1

```
switch (variableName)
{
    case expected : printf(" ");
          input :      statement.
          :
          :
          :
    multiple cases can be write
    default : when user enter value
}             out of created cases.,
              for that we will print
              error message in default.
```

-> default always written in the last of all cases.

-> break; <- to break switch on true case.

<- write at end of every cases.

-> Ternary Operator.

? :

-> same as if else.
-> ternary operator used when we want to write all conditional statement in single line.

→ we can't use ; in between ternary operator.

→ we will put at the end of ternary line ;

5. Looping.

5.1 What is loop ?

-> When we want to print some line of code multiple time, then we will use loop to reduce code.

Types of Loop.

(1) Entry controlled loop	(2) Exit controlled loop.
-> while loop -> for loop	-> do..while loop.
-> which loop check condition in entry of loop, its called entry controlled loop.	-> which loop check condition at end of loop its called exit controlled loop.

-> Syntax.

```
declaration/initialization
while(condition)
{
  increment/
  decrement
}
```

while loop.

-> for(start; condition; ++, --)
{
statement
}

-> do
{
statement
} while(condition);

Note: which line ~~of~~ is end with ();
there compolsury put ;
or
which line is end with { }
there never print ;

// different between while and do while loop

while	do while
-> Entry control	-> Exit control
-> For every time first condition checks then going to print output if condition is true.	-> For first time output will be print then check condition.
	-> minimum one output will be print if condition is satisfied or not

-> Jump statement

(1) break;
(2) goto;
(3) continue;

(1) break : this keyword terminate execution of particular block.

(2) goto : this keyword used to jump our execution anywhere to anywhere.

(3) continue : this keyword used to skip execution of particular cycle of loop.