

Ch-1 Introduction to C

Page:

Date: / /

1.1 What is programming?

→ First of all what is program?

→ Program is set of instructions or command which give to the computer to accomplish certain task.

→ this process is called programming.

→ means process to do program is called as programming.

→ History of C

C is Programming language, developed in

1972.

time

→ C was developed by Dennis Ritchie at AT & T bell laboratories located in

USA. colour

→ Importance of C

→ C is mother of all languages and whenever we want to enter in programming 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 readable language) (process). (Binary language)

C, C++, Java, Python
English

Output
(0,1)

Compiler
Output
(User).

Interpreter

→ Difference Between Compiler & Interpreter

* Compiler

? * Interpreter

→ Read code as all → Read code line by line at same time.

→ fast processing → slow processing

→ good performance and efficient

~~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 I show in the output screen.

→ \n - newline, to break line.

→ \t - tab , to print a space at one time.

→ Basic program syntax.

header [#include <stdio.h>] - standard input output file [#include <conio.h>] - console I/O

main() ← main method/function

{ ← opening brace for main function
 ↳ end brace for main function . }

cirscr();

printf("Hello world");

getch();

} ← closing brace for main function .

↑ pressing enter key → Run → start → execute → run

↓ exit

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. ~~stimated min find it~~

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

Some 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 types of data, but we will learn in C, mentioned below.

1. Integer : -1, 0, 1, 2... (natural value)

2. Flout : -0.1, -2.3, 4.5... (decimal value)

3. char : 'A', 'Z', '#' (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;

function here giving thing something

- Syntax to initialize variable

int a; datatype variableName = value;

datatype variableName = value;

Examples. Answer at next chapter.

int a; datatype variableName = value;

float percentage; datatype variableName = value;

double NO; datatype variableName = value;

char x; datatype variableName = value;

→ format specifier.

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

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

int a; datatype variableName = value;

(7) (7) 7

format 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 this
 95.75000 , like this unnecessary zeros.

→ Then how to remove unnecessary zeros or fix the number of digit after point.

→ Use. "%.nf" , "1.2f"
 output for → 95.75

→ It specifies digit after point.

printf("%d", integerVariable);
 → printf("%d", integerVariable);

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

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

→ Output : 18.9825486060A18.81

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 created in float because 3.14 is decimal value.

* const float pi = 3.14;

If 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:

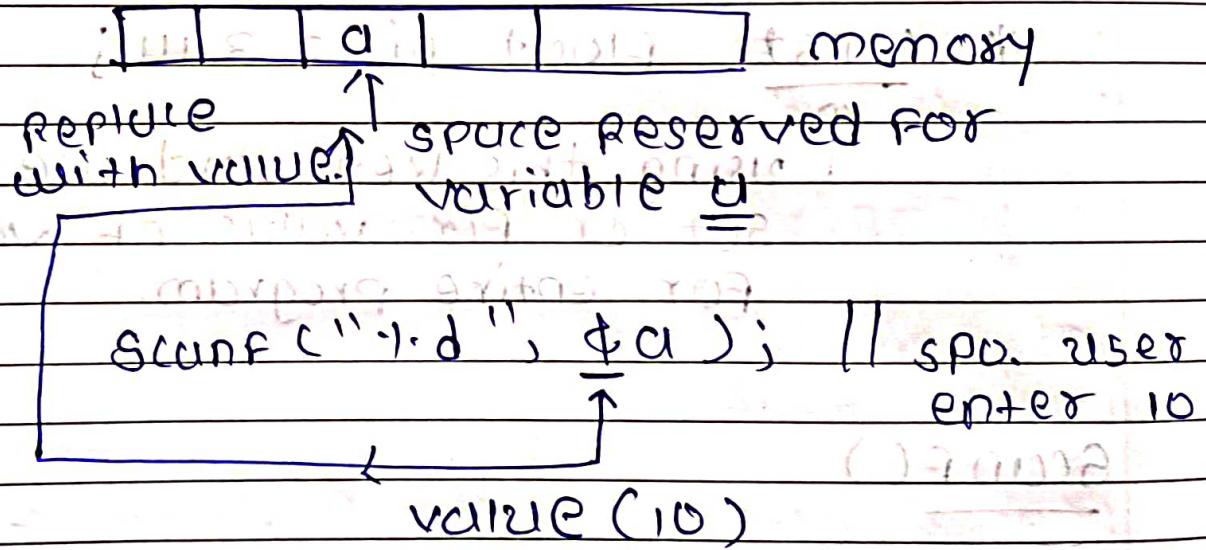
scanf("%d", &varname);

%d = format specifier of variable.

→ `scanf("%d", &varname)`
here this is sign of address.

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

flow of `scanf`.
`int a;`



note: can't be done in arithmetic operation
|| after `scanf` it must be used `&`



so it is used to store memory

ch-3: Operator & Expression

3.1

Type of operator

① Unary operator
operator which have minimum one value.

→ which have / want only one value → which have / want minimum two value.

→ ex. $a++$, $x++$ → $a+b$, $c>b$, $x=10$

① $++$
increment.

→ int $x=0;$

$x++;$

→ $x=1$ ← output

* types of Binary operator.

(1) Arithmetic operator

(+, -, *, /, %)

② $--$
decrement

int $x=1;$

$x--;$

→ $x=0$ ← output

(2) Logical operator

(&&, ||, !)

(3) Relational Operator

(>, <, >=, <=, ==)

(4) Assignment operator

(=, +=, -=, *=, /=).

→ if we stores decimal value in integer output will be integer.

int $x = 3.14;$

→ output $\boxed{3}$

(1) Arithmetic Operator:

→ Arithmetic 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

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

C-1 & C-2 i.e Output

0	+	0	0
1	+	0	1
1	-	0	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.

→ $x = 10;$

→ $x += 10 \rightarrow x = x + 10$

→ $x++$

→ $x = x + 1$

→ Now you got it what is operation

operator don't do it to value

→ Operator is used to perform operational process on the values.

With operators you can make a program which does addition, subtraction, multiplication, division etc.

Let's see how we can do this.

For example, if we want to add two numbers.

Then we will use + operator.

Now let's see how we can do this.

For example, if we want to subtract two numbers.

Then we will use - operator.

Now let's see how we can do this.

For example, if we want to multiply two numbers.

Then we will use * operator.

Now let's see how we can do this.

For example, if we want to divide two numbers.

Then we will use / operator.

Now let's see how we can do this.

For example, if we want to find remainder after division.

Then we will use % operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use == operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use != operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use < operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use > operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use <= operator.

Now let's see how we can do this.

For example, if we want to compare two numbers.

Then we will use >= operator.

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)

{

 if true

}

 else

{

 if false.

}

→ ladder. if (condition)

 { if true

 STEP BY

 STEP.

{

 else if (condition)

{

 y

 else if (condition)

{

 y

 ;

→ → NESTED (One into another)

4.2 if()
 {
 // code block
 }

In if() we can add another if()
 {
 // code block
 }

example : else {
 }
 // code block
 }

→ when writing nested if()
 }
else {
 // code block
}

then condition must be same
 {
 if()
 }

→ so if() goes before if()
 {
 if()
 }
 else {
 if()
 }

means both condition must be same
 }
 {
 if()
 }

~~→~~ → condition must be same
 {
 if()
 }

Nested condition works same , just
way of writing is different.

→ when we have multiple conditions
then we will use nested ladder.

digitally how train runs and on condition
when train goes right then it moves right

when train goes left then it moves left

when train goes up then it moves up

when train goes down then it moves down

→ How to write nested ladder

Enter

* Algorithm.

→ Step by process or flow of code.

ex. WAP to print sum of two numbers.

→ Algorithm for this code.

Step 1: start

Step 2: declare two variable 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 rough algorithm for every program in book before you start coding.

4.3

~~switch case~~ switch cases are used when we want different action when some condition is true.

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

→ ex. mcq → question - I, answer = u.
but we can select only - I

```
switch (variableName)
```

{

cuse expected : printf(" "); /

input : statement.

;

;

;

multiple cuses can be write

} default : when user enter value
out of created cuses.,
for that we will print
error message in default.

→ default always written in
the last of all cuses.

→ break; ← to break switch on
true case.

← write at end of every
cuses.

→ 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 ;

Let's see how it works

Suppose we have two numbers

100 and 200 between the two

which one is greater

if first one is greater then print first one

if second one is greater then print second one

if both are equal then print both

conditions used or -> if condition is

then either the number is greater than

or equal to

or less than the given

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

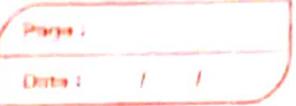
so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

so we can use if condition in if condition

5. Looping.



Q.1 what is loop?

→ when we want to print some line of code multiple times, then we will use loop to reduce code.

Types of Loop.

(1) Entry

controlled

(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)

{
 initialization /
 increment /
 decrement
}

while-

loop.

→ `for(start; condition; increment)`
{
 statement
}

→ `do`
{
 statement
}
 } `while(condition);`

NOTE: which line ~~of~~ is end with `);`
there compulsory put ;

which line is end with `{}
there never print`

~~different between while and do while~~

→ Entry control \rightarrow Exit control
(condition) ~~and~~

→ 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.