RTU ROLL NO. .....

ARYA GROUP OF COLLEGES

Checked

I MID TERM EXAMINATION 2018-19 (I Sem.)

1FY3-06_Programming for Problem Solving

BRANCH: Common to All

Pawan Sen

Max Marks:- 40 **Time:- 2 hrs.**

PART A (Attempt All)

Q.1	(a)	WAP in C to enter two numbers and print their multiplication.	
	(b)	Explain flowchart and algorithm with an example?	
	(c)	Explain the difference between Primary memory and Secondary memory?	5*2
	(d)	Explain formatted input and output function with syntax and example?	
	(e)	Explain the difference between compiler and interpreter?	

PART B (Attempt any Four)

Q.2	(a)	Explain the difference between high level and low level programming languages?	
	(b)	Explain access methods in detail.	
	(c)	Explain Switch case statement with syntax and example?	4*4
	(d)	What is Von –Neumann Architecture? Explain with diagram.	
	(e)	Draw a flowchart to check if a year is leap or not. (eg. 2004 is leap whereas 1900, 2017 are not)	
	(f)	WAP in C to enter 2 numbers & perform all Arithmetic operations and print output?	

PART C (Attempt any Two)

Q.3	(a)	Write a program to swap two numbers (i) Using third variable (ii) Without using third variable	2*7
	(b)	WAP in C to find the maximum number between the given 3 numbers.	
	(c)	What are Conditional Statements in C? Explain in detail.	

Part - A

Q.1 (a) WAP in C to enter two number and Print their multiplication?

Ans>
```
#include<stdio.h>
# include <conio.h>
void Main()
{
   int a,b,c;
Printf("Enter value of a & b \n");
Scanf("%d %d", &a, &b);
   c = a*b;
Printf("Multiplication = %d", c);
   getch();
}
```

Output:-
```
Enter value of a & b
10
10
Multiplication = 100
```

Q.1 (b) Explain flowchart and Algorithm with an example?

~~Algorithm~~ → flowchart →

A flowchart is graphical or Pictorial representation of an algorithm with the help of diffrent Symbol, shapes and arrows in order to demonstrate a Proces or a Bogram.

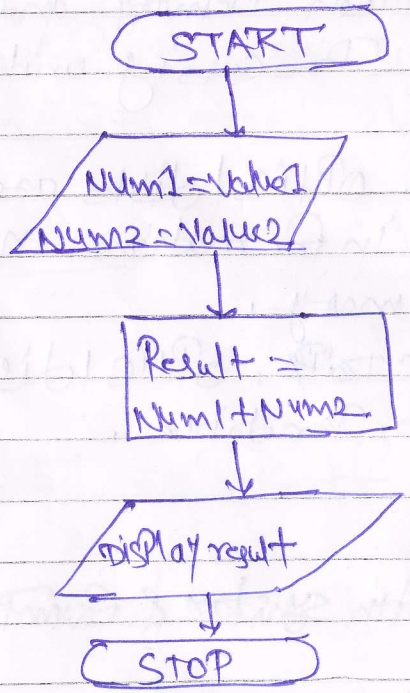
→ The Main Purbse of a flowchart is to analz diffrent Processes. several standard graphics are applied in a flowchart.

Algorithm →

to write a logical step by step method to solve the problem is called Algorithm. In other word, an algorithm is a procedure for solving problem.

→ in order to solve mathematical or computer problem, this is the first step of the procedure. An algorithm include calculation, reasoning and data processing. Algorithm can be represented by natural language, Pseudo code and flow chart.

Ex→



Q→ An algorithm to find out sum of two number

STEP 1: Input two number : a, b.
STEP 2: Calculate sum = a+b.
STEP 3: Print "Total =" sum.
STEP 4: STOP

Q.1(c) Explain the difference between Primary memory & Secondary Memory?

Ans→

Primary Memory	Secondary Memory
1. Primary Memory is also known as Main Memory or Internal Memory.	1. Secondary Memory is also known as External Memory or Auxiliary Memory.
2. In Primary Memory, data is directly accessed by the Processing unit	3. In Secondary Memory, data is first transferred to Main Memory and Then routed to Processing unit.
3. Semi Conductor chips are used to store information in Primary Memory. Ex→ RAM, ROM, Registers	3. Magnetic disk, optical disks are used to store information in Secondary Memory. Ex→ Magnetic Tapes, optical disks CD, DVD, Pendrive.

Q.1(d) Explain formatted input & output function with syntax & Example?

Ans→

The formatted input/output functions read & write, respectively, all types of data value.

Types of formatted Input/output function ⇒

They are two types.

1. Printf()
2. Scanf()

1. Printf() (formatted output function) →

→ This function is used to print any text as well as value of the variable on the standard output device (Console (Monitor, Screen).

→ Printf is very basic library function in c language that is declare in Stdio.h header file.

Syntax →

```
Printf(" Message Text ");
Printf(" formatCode ", Variable-list);
```

Ex→

```
Printf("hello good morning");
```

or

```
int a=10, b=20;
Printf("a=%d b=%d ", a, b);
```

```
a=10
b=20
```

Scanf() (formatted input function) →

This function is used to input value from keyboard.

Syntax→ `Scanf("format code", &v1, &v2 ... &vn);`

Ex→

```
#include<stdio.h>
#include<conio.h>
void Main()
{
   int a, b;
Printf("Enter value of a, & b \n");
Scanf("%d %d", &a, &b);
Printf("a = %d", a);
Printf("b = %d", b);
   getch();
}
```

output:-

```
Enter value of a & b
10
50
a = 10
b = 50
```

Q.1(e) Explain the difference between Compiler & Interpreter ?

Ans→

	Interpreter	Compiler
1.	Translate Program one statement at a time.	1. Scans the entire Program and translate it as a whole into Machine Code.
2.	it take less amount of time to analyze the Source Code but the overall execution time is slower.	2. it take large amount of time to analyze the Source Code but the overall execution time is comparatively faster.
3.	No intermediate Object Code is generated, hence are	3. Generates intermediate Object Code.

PART-B

Q.2(a) Explain the difference between high level and low level Programming language?

Ans→

Differences	high level language	low level language
1. Platform Dependencies	High level Programming language (HLL) are Platform independent That means Program written in high level language can run on different hardware with different Configuration.	Low level Programming lang. (LLP) are Platform dependent That means Program written in low level can run on the same hardware with same Configuration.
2. Speed	high level Program are slower Comparison to low level lang. Program because they need to Convert high level to low level	Low level language Program are faster than high level lang. Program as they do not need to Convert.
3. Performance	They are also better Performance	low level Program are faster so Performance are better than high level.

4.	Easiness	High level Program are easy to write read modify and understand.	Low level Program are not easy as high level language. They are only two low level Programming language Binary and Assembly. Binary has only 0's and 1's.
5.	Translation	High level language are translated by Compiler or interPreters	Low level language Binary does not need translation as Binary Codes are Machine Codes and Computer understand Them without any translations.

Q.2(b) Explain Access method in details.

Ans→

Each Memory is a Collection of various memory locations. Accessing The Memory means finding and reaching The desired location and Then reading information from That memory location.

→ Information from memory locations can be accessed in following ways.

1. Sequential Access method
2. Direct Access method
3. Random Access method

1. Sequential Access method →

→ In this mode the memory locations can be accessed sequentially only. To reach a particular locations, access mechanism will first of all go through all the locations prior to the desired location and then reach the required location and read it.

→ Some backup devices such as magnatic tape are the example of sequential Access method.

2. Direct Access method →

→ In Direct Access, information is stored on tracks and each track has a separate read/write head. Using this read/write head information of that could be accessed sequentially. This feature make it a semi random mode which is generally use in magnetic disks.

3. Random Access method →

→ In This Aecess mode where each memory location has a uniaue address. Using These uniaue address each memory location can be addressed independently in any order in eaual amount of time.

→ The desired location is in beginning of memory or it is in the end the access time of information from both the location would be same. Generally, main memory are random Access Memories.

Q.2 (c) Explain switch Case statement with syntax and example?

Ans→

Switch Case statement →

Switch Case Can be used to replace else if ladder. The switch stat. Provides a way of choosing between a set of alternatives, based on the value of expressions.

The switch Provides for a multiway branch. Thus, it enables a program to select among several alternatives.

Syntax:

```
switch(expression)
{
  Case 1:
    statement;
    break;

  Case 2:
    statement;
    break;

  Case 3:
    statement;
    break;

  default:
    statement;
}
```

Ex:

```
#include<conio.h>
#include<stdio.h>
void main()
{
  char ch;
printf("Enter a character");
scanf("%c", &ch);
 switch(ch)
{
  Case 'a':
printf("vowel");
  break;

Case 'e':
printf("vowel");
  break;

Case 'i':
printf("vowel");
  break;

Case 'o':
  printf("vowel");
  break;
```

```
Case 'U':
Printf ("Vowel");
  break;
default:
  Printf ("This is not Vowel");
}
  getch();
}
```

output :- I

Enter a Character

A

Vowel

output II

Enter a Character

Z

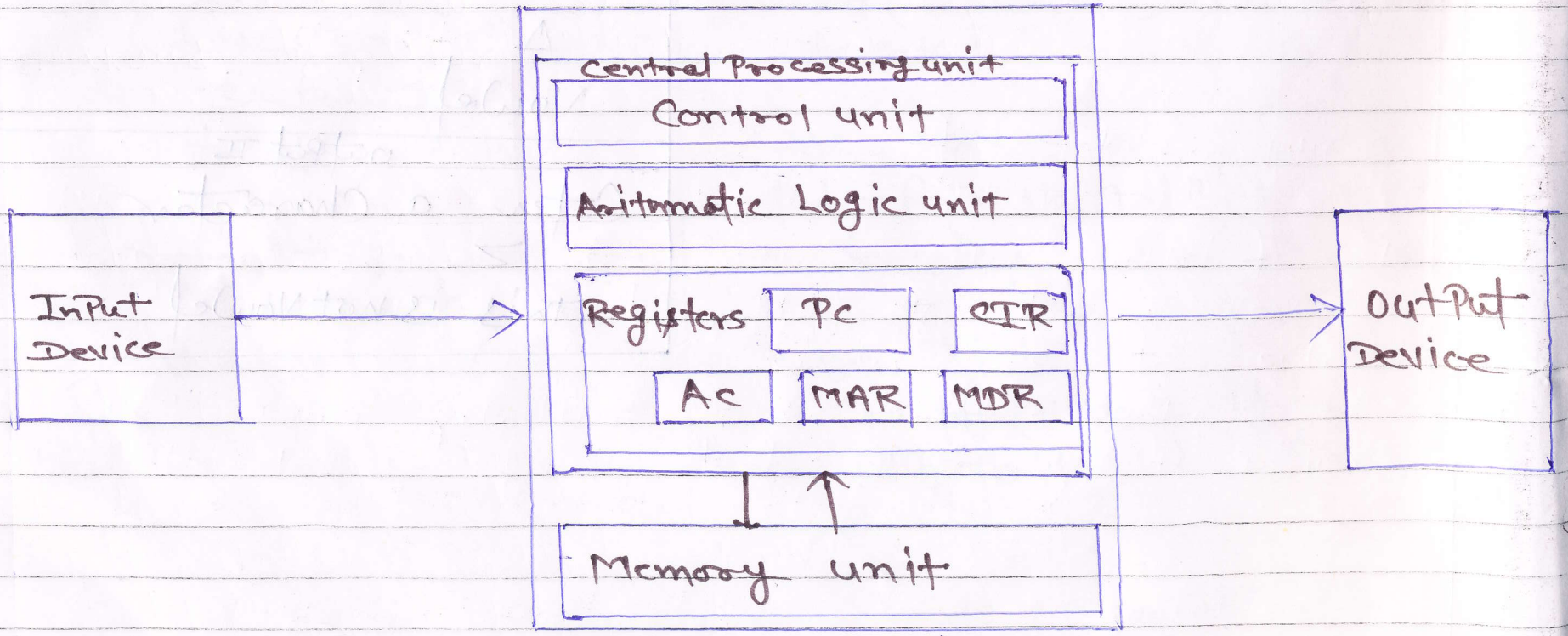
This is not Vowel

Q.2 (1) Explain The Von-Neumann Architecture ? Explain with diagram?

Ans→

→ Von Neumann Architecture Was first Published by John Von Neumann in 1945. his Computer Architecture design Consist of a Control unit, Arithmatic Logic unit (ALU), memory unit Registers and Inputs/outputs.

→ Von Neumann Architecture is based on Stored Program Computer Concept, where instruction data & Program data are Stored in The same memory.



Central Processing unit →

The CPU is the electronic circuit responsible for executing the instruction of a computer program. it is sometimes referred to as the microprocessor or processor. The CPU contains the ALU, CU and variety of registers.

Registers →

Registers are high speed storage areas in the CPU. All the data must be stored in a registers before it can be processed.

MAR	Memory address Registers	holds the memory locations of data need to be accessed.
MDR	Memory Data Registers	holds data that is being transferred to or from memory.
AC	Accumulators	Where intermediate arithmetic and logical results are stored.
PC	Program Counter	Contains the address of next instructions to be executed
CIR	Current instruction Registers	Contains the current instruction during processing.

Arithmetic logic unit (ALU) →

The ALU allows arithmetic (add, subtract etc) and logic (AND, OR, NOT) operations to be carried out.

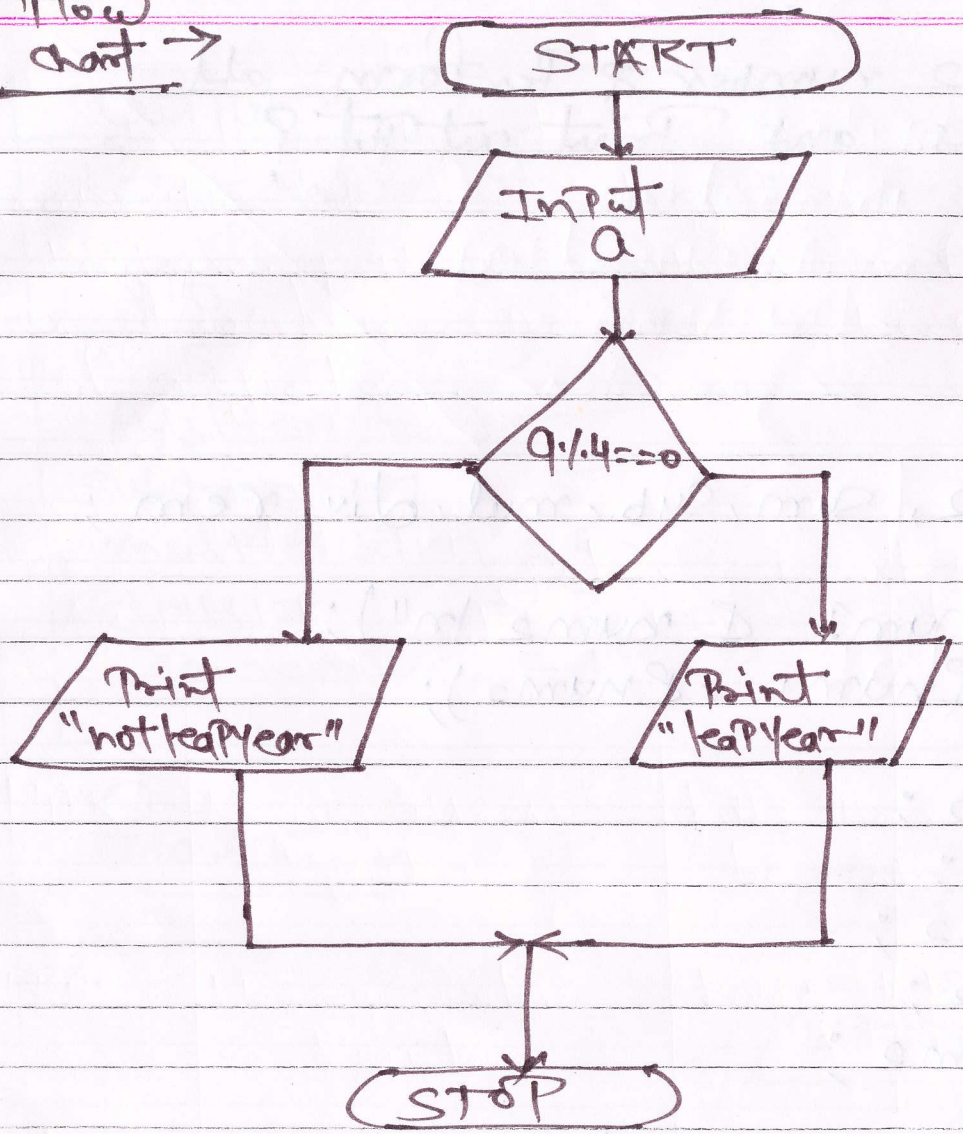
Control unit →

The Control unit controls the operation of the Computer's ALU, memory and input/output devices, telling them how to respond to the program instructions. it has just read and interepted from the memory unit. The Control unit also provides the timing and Control signals required by other Computer components.

Q.2(e) Draw a flow chart to check if a year is leap or not (eg.- 2004 is leap whereas 1900, 2017 are not)

Ans →

flow chart →




```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a, year;
printf("enter a year");
scanf("%d", &year);
if(year%4==0)
{
printf("leap year");
}
else if(year%400==0)
{
printf("leap year");
}
else if(year%100==0)
{
printf("leap year");
}
else
{
printf("not leap year");
getch();
}
```

Q.2(f) WAP in c to enter 2 number & Perform all Arithmetic Operations and Print out put?

Ans:

```
#include<stdio.h>
#include<conio.h>
void Main()
{
    int num1, num2, sum, sub, mul, div, rem;
    clrscr();
Printf("Enter value of num1 & num2 \n");
Scanf("%d %d", &num1, &num2);

    Sum = num1 + num2;
    Sub = num1 - num2;
    mul = num1 * num2;
    div = num1 / num2;
    rem = num1 % num2;

Printf("Addition = %d \n", sum);
Printf("Subtraction = %d \n", sub);
Printf("multiplication = %d \n", mul);
Printf("division = %d \n", div);
Printf("remender = %d", rem);
  getch();
}
```

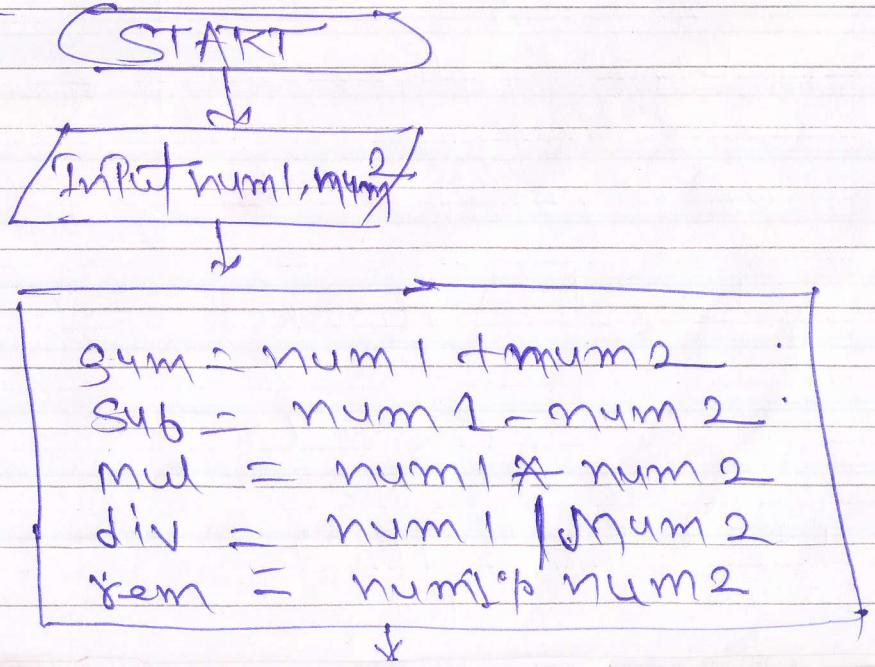
Output:-

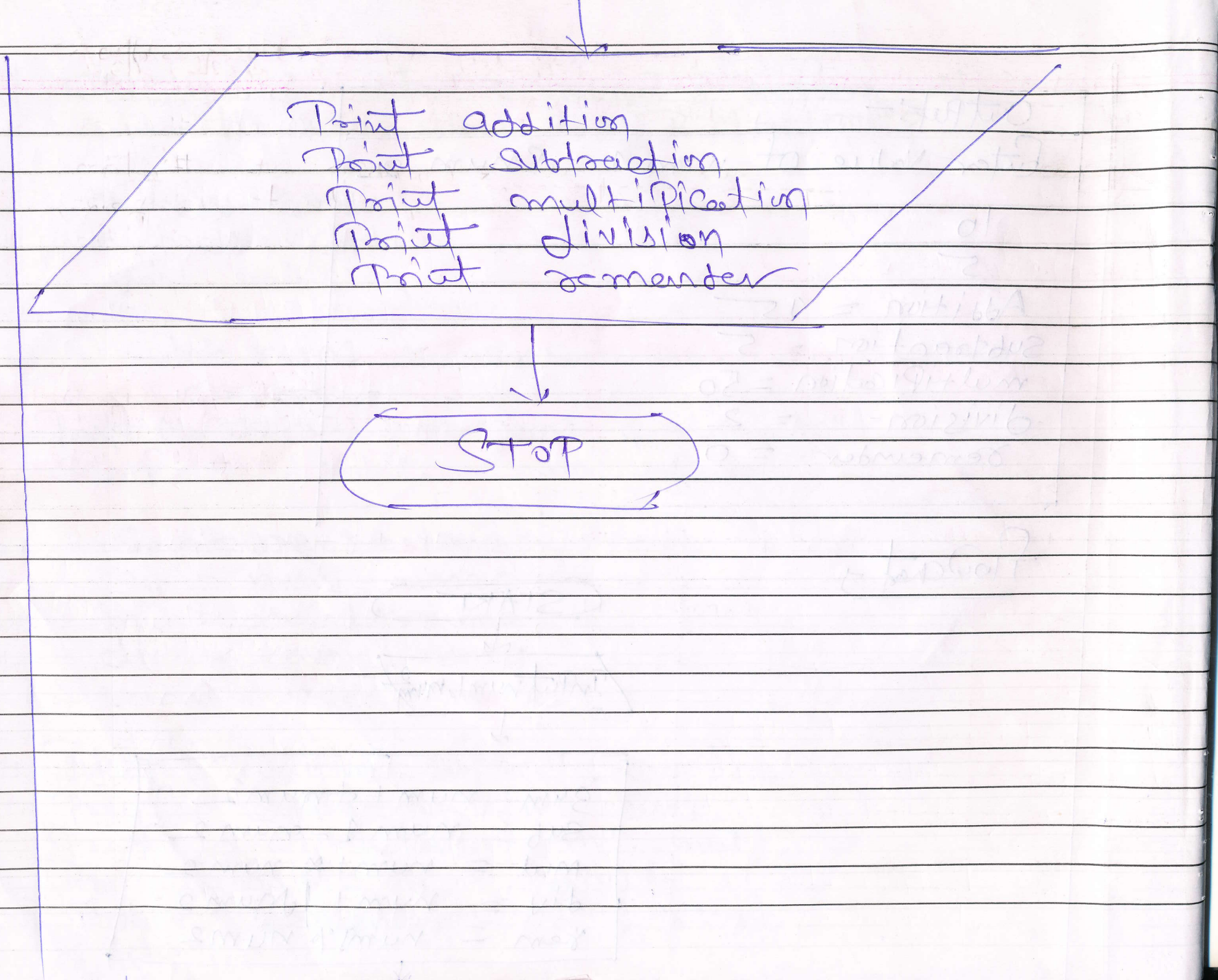
Enter Value of num 1 & num 2

10
5

Addition = 15
Subtraction = 5
multiplication = 50
division = 2
remender = 0

flowchart :-




Print addition
Print subtraction
Print multiplication
Print division
Print remender
STOP

Q.3(a) write a Program to swap two number
(i) using Third Variable (ii) without using Third Variable.

Ans → Using Third Variable Program →

```
#include <stdio.h>
# include <conio.h>
Void Main()
{
    int x, y, t;
Printf ("Enter two integer Value \n");
scanf ("%d %d", &x, &y);
Printf ("before swaping x=%d \n y=%d", x, y);
    t=x;
    x=y;
    y=t;
Printf ("After swaping x=%d \n y=%d", x, y)
    getch();
    }
```

Output :-

```
Enter Value of x & y
  40 , 30
before swaping    x = 40 , y = 30
After swaping     x = 30 , y = 40
```

swapping without using Third Variable →

```
#include <stdio.h>
#include <conio.h>
Void main()
{
   int a, b;
Printf ("Enter value of a&b\n");
  scanf ("%d%d", &a, &b);

Printf(" before swaping a = %d \n b = %d \n", a, b);

    a = a+b;
    b = a-b;
    a = a-b;
```

```
Printf(" After swaping a = %d \n b = %d ", a, b);
getch();
}
```

output :-

```
Enter value of a & b
10
20
before swaping
a = 10
b = 20
After swaping
a = 20
b = 10
```

Q.3(b) WAP in c to find the maximum number between the given 3 number.

Ans->

```
#include<conio.h>
#include<stdio.h>
  Void main()
{
   int a,b,c;
    clrscr();
Printf("Enter value of a,b,c \n");
scanf("%d%d%d",&a ,&b,&c);
if(a>b)
 {  if(a>c)
       Printf("maximum = %d", a);
    else
       Printf("maximum=%d", c);
 }
   else
      {   if(b>c)
```

```
        Printf("Maximum = %d", b);
    else
        Printf("maximum = %d", c);
    }
    getch();
}
```

Output

```
Enter Value of a, b, c
10
20
30
Maximum = 30
```

Q.3(c) What are Conditional statement in c? Explain in details.

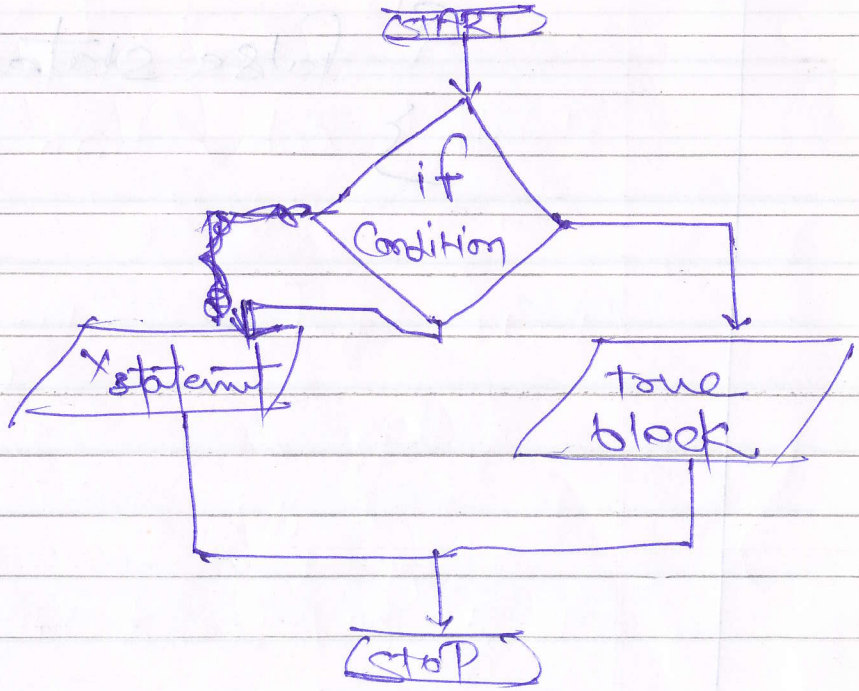
Ans→ Conditional statement helps us to make decision based on certain conditions. These conditions are specified by a set of conditional statements having boolean expressions which are evaluted to a boolean value true or false. These are following types of Conditional statement in c.

1. if statement
2. if - else statement
3. nested if - else statement
4. ladder else - if statement
5. switch Case statement.

1. **Simple if statement** → The single if statement in c language is used to execute the code if condition is true.

Syntax →

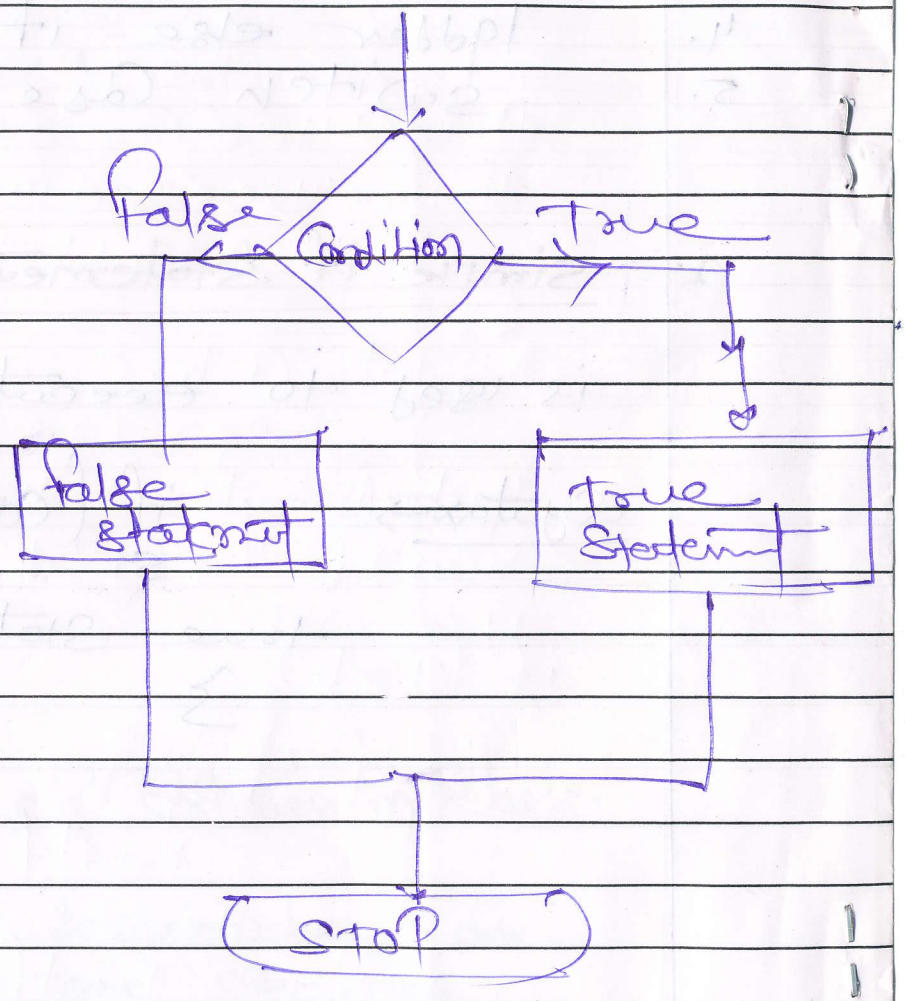
```
if (condition)
{
true Statement ;
}
```



2. if else statement →

Syntax →

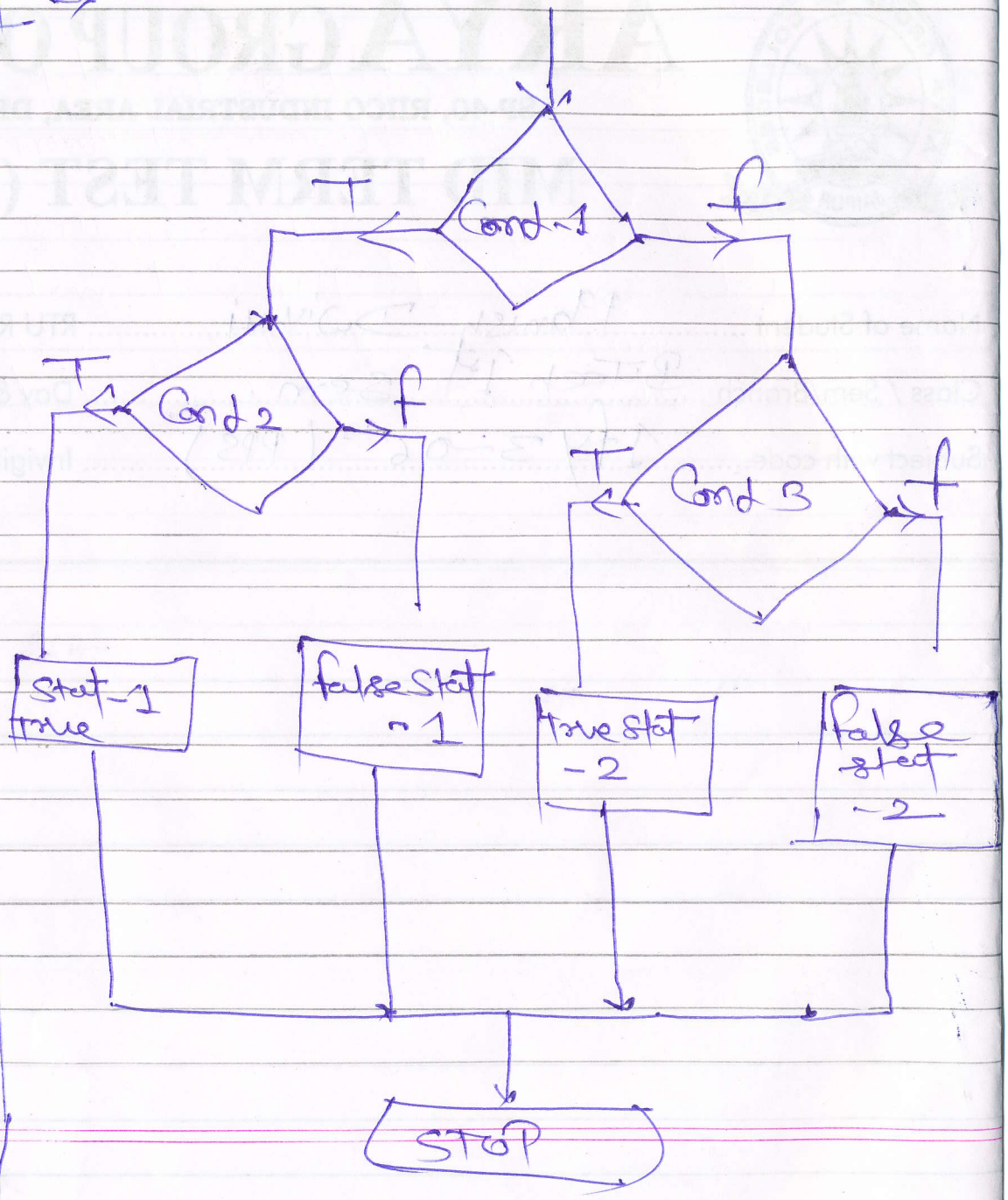
```
if(condition)
{
True statement;
}
else
{ false statement;
}
```



3. nested if else statement →

Syntax →

```
if (condition1)
{
  if (condition 2)
    {
  true stat - 1 ;
    }
  else
    { false stat 1 ;
    }
else
    { if (cond - 2)
        {
      true stat - 2 ;
        }
      else
        {
          false stat - 2 ;
        }
}
```



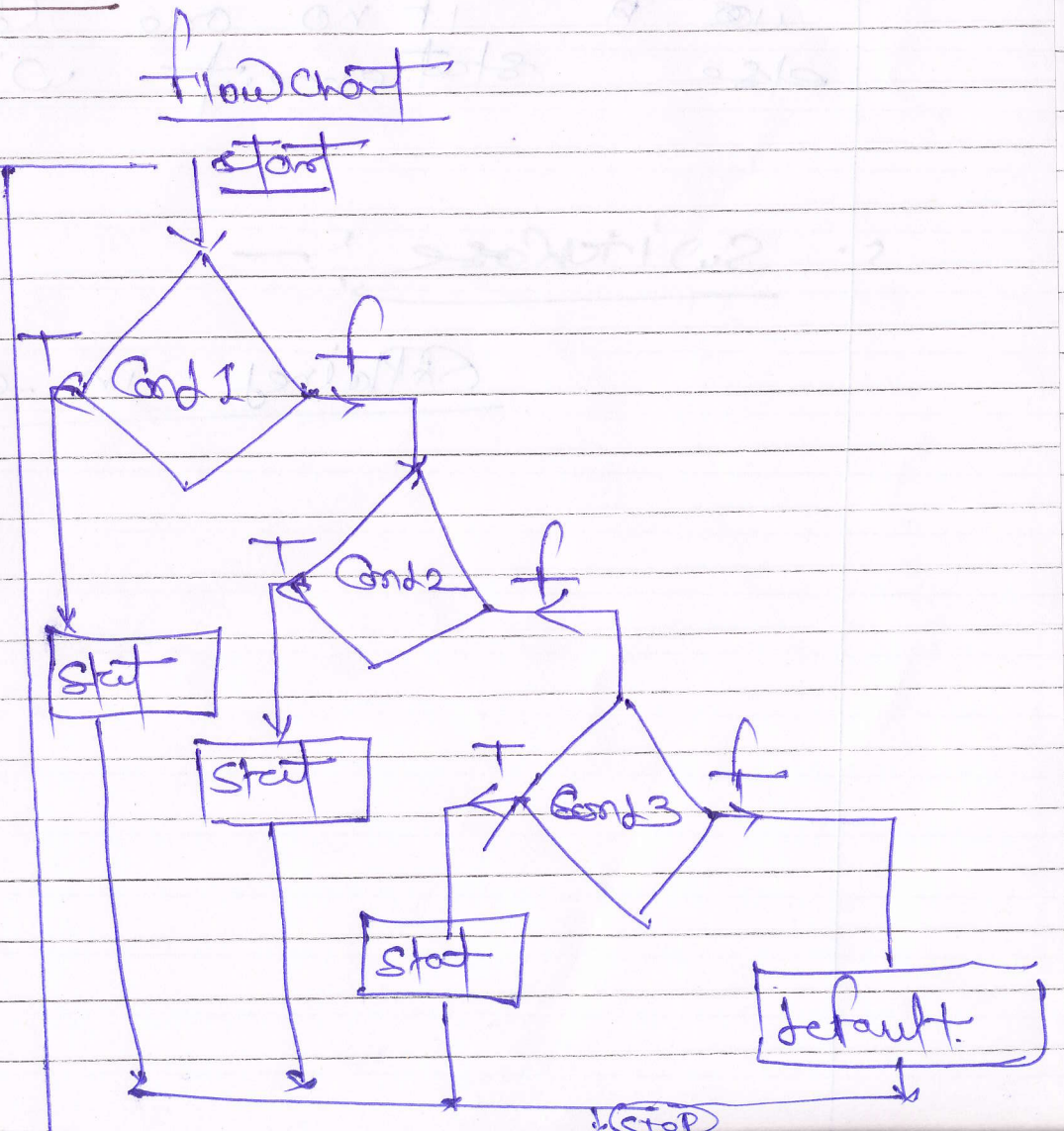
The nested if-else statement is used when program requires more than one test expressions.

4. **Ladder else-if statement =>**

Syntax =>

```
if (condition 1)
  statement;
else if (condition-2)
    statement;
else if (condition-3)
    statement;
  :
  :
elseif (condition-n)
    statement n;
else
    statement;
```

Flowchart



The ladder else-if statement used to execute one code from multiple conditions.

There is many conditions, if any condition is true then statement will be executed if no one condition is true then else statement will be execute.

5. Switch case :-

Explained in earlier