Assignment - 1 (Unit - 1)

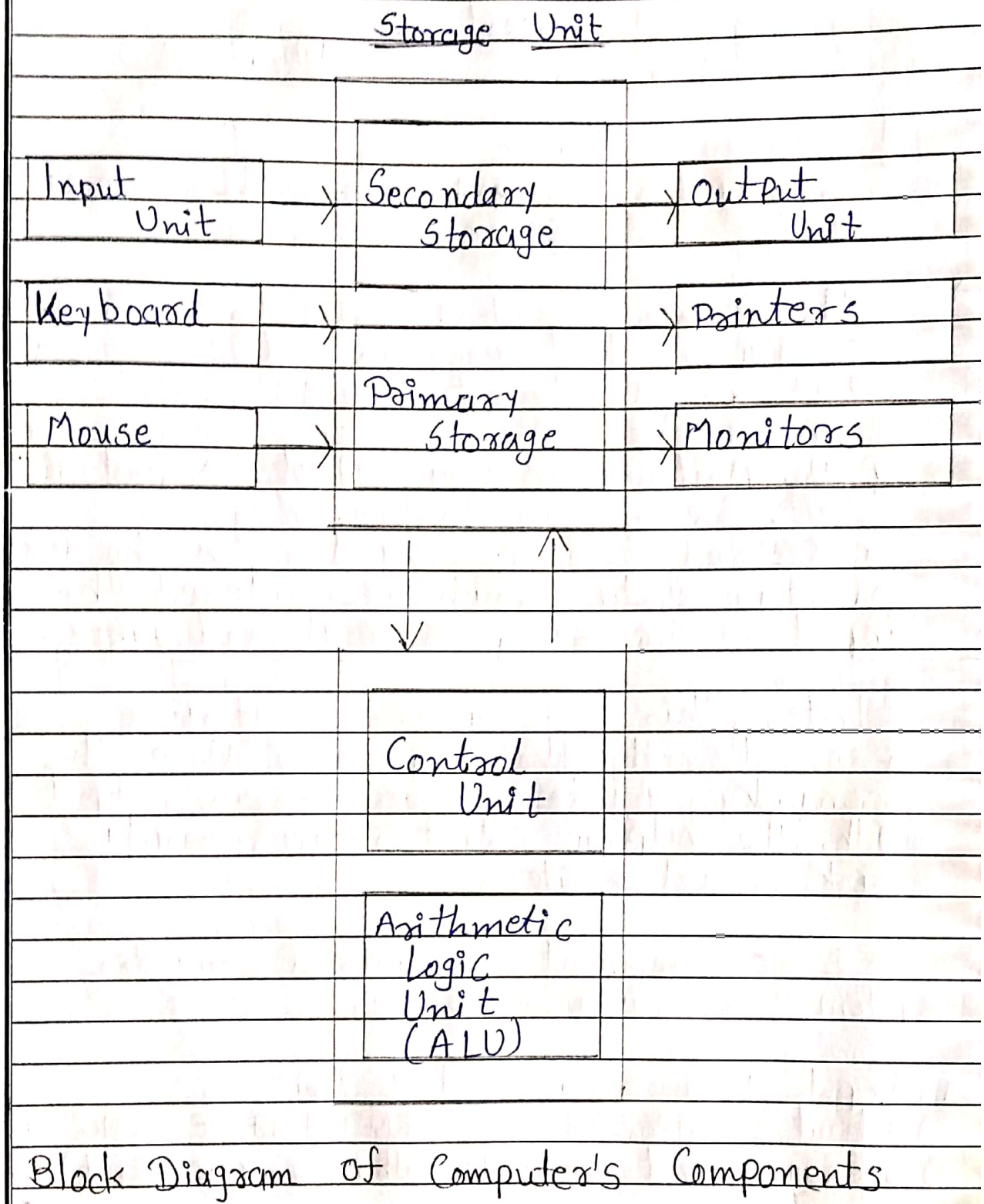
Q-1] Discuss basic component of computer system with the help of block diagram.

→ The computer system consists of mainly 3 types that are central Processing Unit [CPU], Input Devices, & Output Devices. The 'CPU' again consists of ALU (Arithmetic Logic Unit) & control Unit. The set of instructions is presented to the computer in the form of raw data which is entered through input device such as keyboard / mouse.

Later this set of instructions is proceed with the help of CPU & the computer system produce an output with the help of Output Devices mainly Printers and moniters

* Basic components & parts of computer system are given below:

1) Input Devices {4) Control Unit
2) Output Devices} (5) CPU (CENTRAL PROCESSING UNI
3) Storage Unit } (6) ALU (Arithmetic Logic Unit)



Block Diagram of Computer's Components

Q-2 Differentiate between system software and application software.

→

	System Software	Application Software
1.	System software is used for operating computer hardware.	Application software is used by user to perform specific task.
2.	It is installed on the computer when operating system is installed.	It is installed according to users requirement.
3.	In general, the user does not interact with system software because it works in the background.	In general, the user interacts with application software
4.	System software can run independently. It provides platform for running application softwares.	It can't run independently. They can't run without the presence of system software.
5.	e.g. Compiler, assembler, debugger, driver, etc.	e.g. Word processor, web browser, etc.

Q3. How many levels of programming languages are available for computing task within computer?

→

Programming languages can be broadly classified into 3 categories:-

(1.) Machine languages
(2.) Assembly languages
(3.) High level languages

(1) Machine language:- It is native language of computer, the language closests to the hardware itself. Each unique computer has a unique machine language. A machine program is made up of a series of binary patterns which represents simple operations that can be accomplished by the computere. (e.g. add 2 operands, move data to a memory location) Programming in machine language requires memorization of the binary codes & can be difficult for human programmer.

(2.) Assembly language:- They represent an effort to make programming easier for the human. The machine language instructions are replaced with simple mnemonic abbreviations. Thus assembly languages are unique to a specific

computer. An assembly languages program requires translation to machine language. This translation is accomplish by a computer program known as Assembler. Assemblers are written for each unique machine language.

3) High level languages :- High level languages like C++, C, JAVA, etc., are more English-like & therefore, make it easier for programmers to "think" in the programming language. High level languages also require translation to machine language before execution. This translation is accomplished by either a compiler or an interpreter.

→ Compilers translate the entire source code program before execution (e.g. C++, Java)

→ Interpreter translate source code programs one line at a time. (Eg. python)

Q-4. What is language translator? Explain different type of translator.

→

A program written in a high-level language is called as source code. To convert the source code into machine code, translators are needed.

Role of translator are :-

1.} Translating the high level language program into an equivalent machine language program.

2.} Providing diagonstic message whenever the programmer violats specification of the high-level language program.

The different type of translators are :-

1.} Compiler :- It is used to convert high-level language to low-level lenguge. It translators the entire program & also reports the errors in source program encountered during the translation.

2.} Interpreter :- It is also used to convert High-level language to low-level language. It translates line by line and reports the error once it encountered during the translation process.

It directly executes the operations specified in the source program when the input is given by user.

Q-5 What is the need to draw flowchart / algorithm.

→

An algorithm is a step-by-step analysis of the process while a flowchart explains the steps of a program in a graphical way.

Algorithm :- An algorithm is a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.

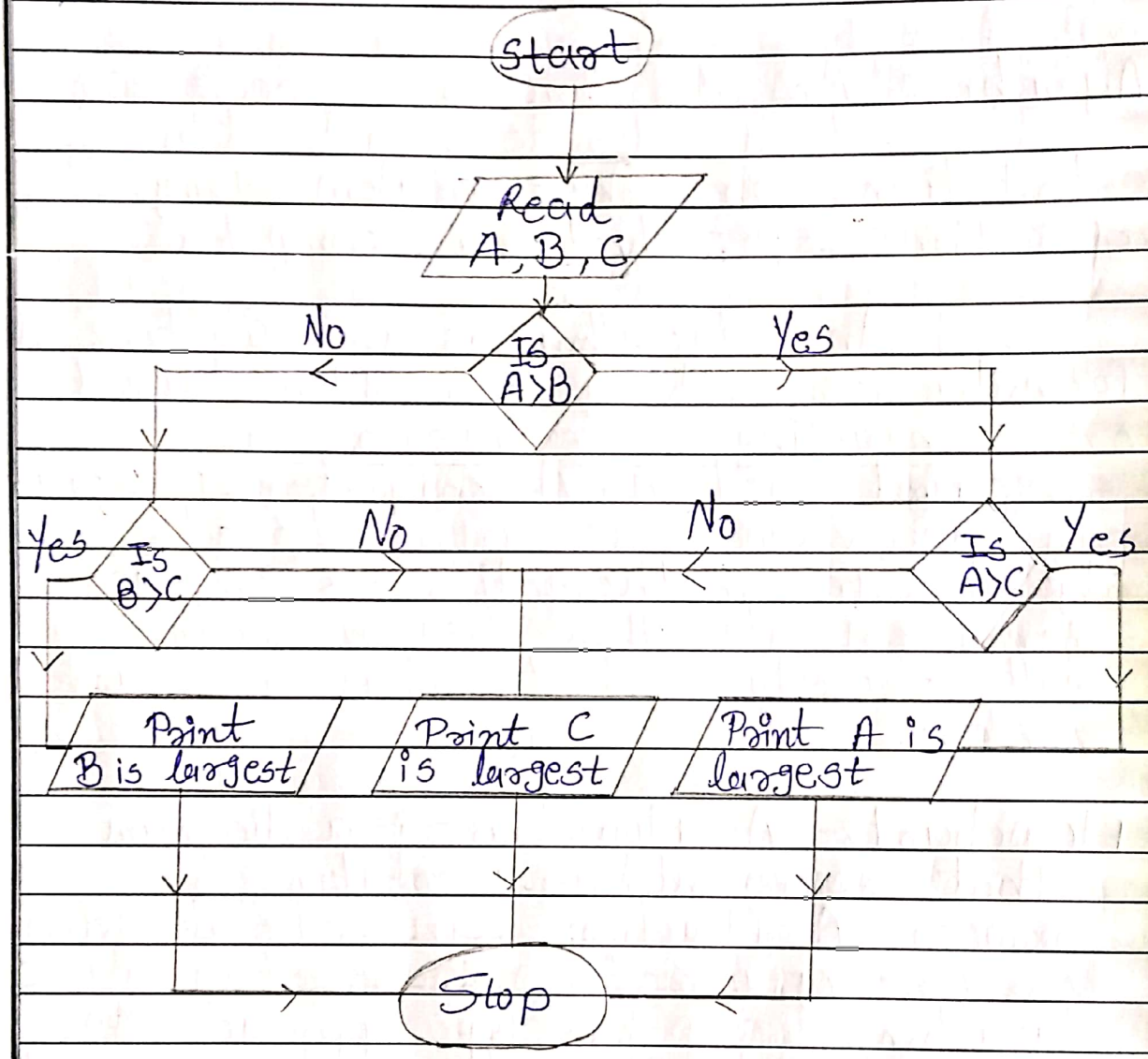
An algorithm is a procedure of formulas for solving a problem based on conducting a sequence of specified actions. A computer program can be viewed as an elaborate algorithm. In mathematics & computer science, an algorithm usually means a small procedure that solve a recurrent problem.

Flowchart :- A flowchart is a diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the

steps as boxes of various kinds, and their order by connecting the boxes with arrows.

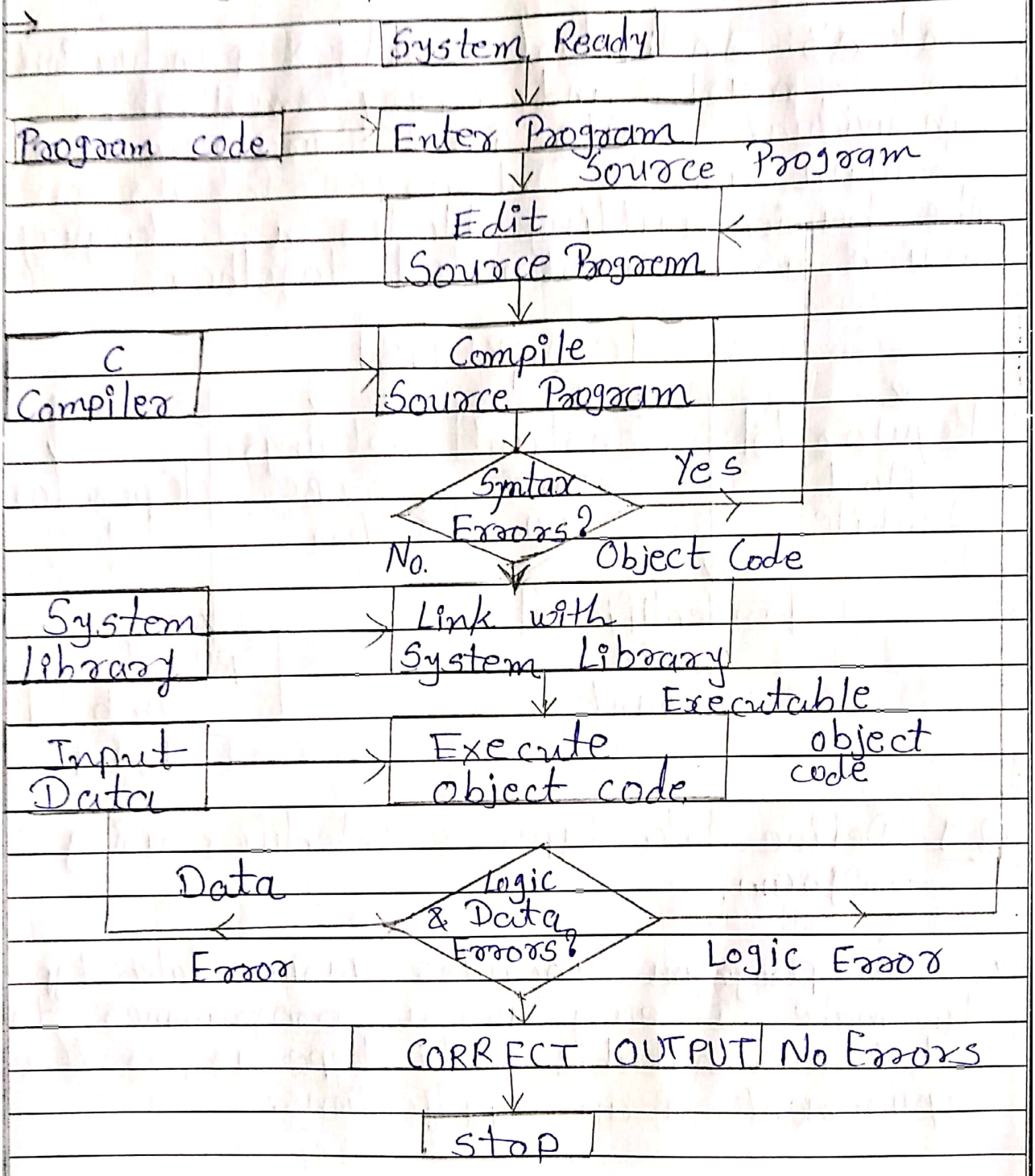
Q-6. Write down a flowchart to identify greatest number within 3 numbers.

→



Q-1. Write the process to compile and running a program.

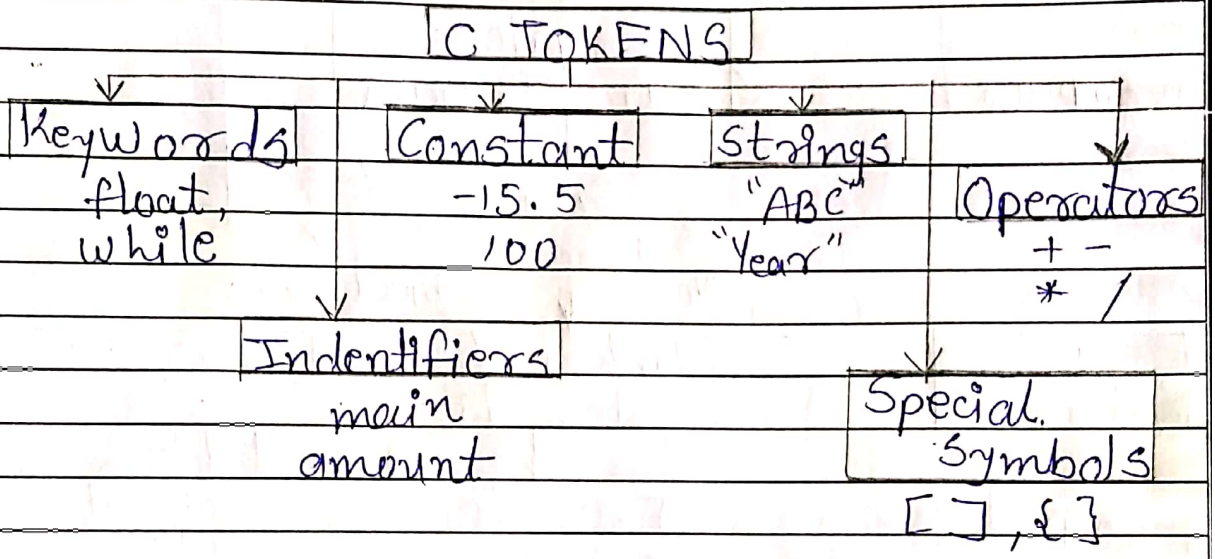
→



Q-8 Define tokens which are used in C Programming.

→

Token's:- Individual words & punctuation marks and every smallest individual units in C program are known as "C Tokens". C tokens are the basic building blocks in C language which are constructed together.



Q-9 Define data types & indentifier using C program.

→ C program provides various data types to make it easy for a programmer to select a suitable data type as per the requirement of an applications.

- C program supports 3 classes of data types:

1.) Primary (or Fundamental) data types.

2.) Derived data types.

3.) User-defined data types.

The primary data types and their extensions are discussed in this section. The user-defined data types are defined in the next section while the derived data types such as arrays, functions, structures and pointers are discussed as and when they are encountered.

- C compilers support five fundamental data types :-

(1) int for Integer data
(2) char for character
(3) float for floating point numbers
(4) double for double precision floating point
(5) void

Identifier: It is a string of alphenumeric characters that begins with an alphabetic character or an underscore character that are used to represent various programming elements such as variable, functions, arrays, structure unions & so on. An identifier is a user defined word.

Assignment ⇨ 2
Unit ⇨ 2

Q-1 Write down basic structure of C program.

→

A C program can be viewed as a group of building blocks called functions. A function is a subroutine that may include one or more statement designed to perform a specific task.

Documetation Section
Link Section
Defination Section
Global Declaration Section
main ( ) Function Section
{
Declaration Part
Executable Part

}
Subprogram section.

Function 1
Function 2
-
-
Function n

(User-defined functions)

The documentation section consists of a set of comment lines giving the name of the program, and other details.

The link section provides instructions to the compiler to link functions from the system library.

The definition section defines all symbolic constants.

There are some variables that are used in more than one function. Such variables are called global variables & are declared in the global declaration section that is outside of all the functions.

Every C program must have one main() function section. This section contains 2 parts :- Declaration part
:- Executable part

The declaration part declares all the variables used in the executable part. There is at least one statement in the executable part.

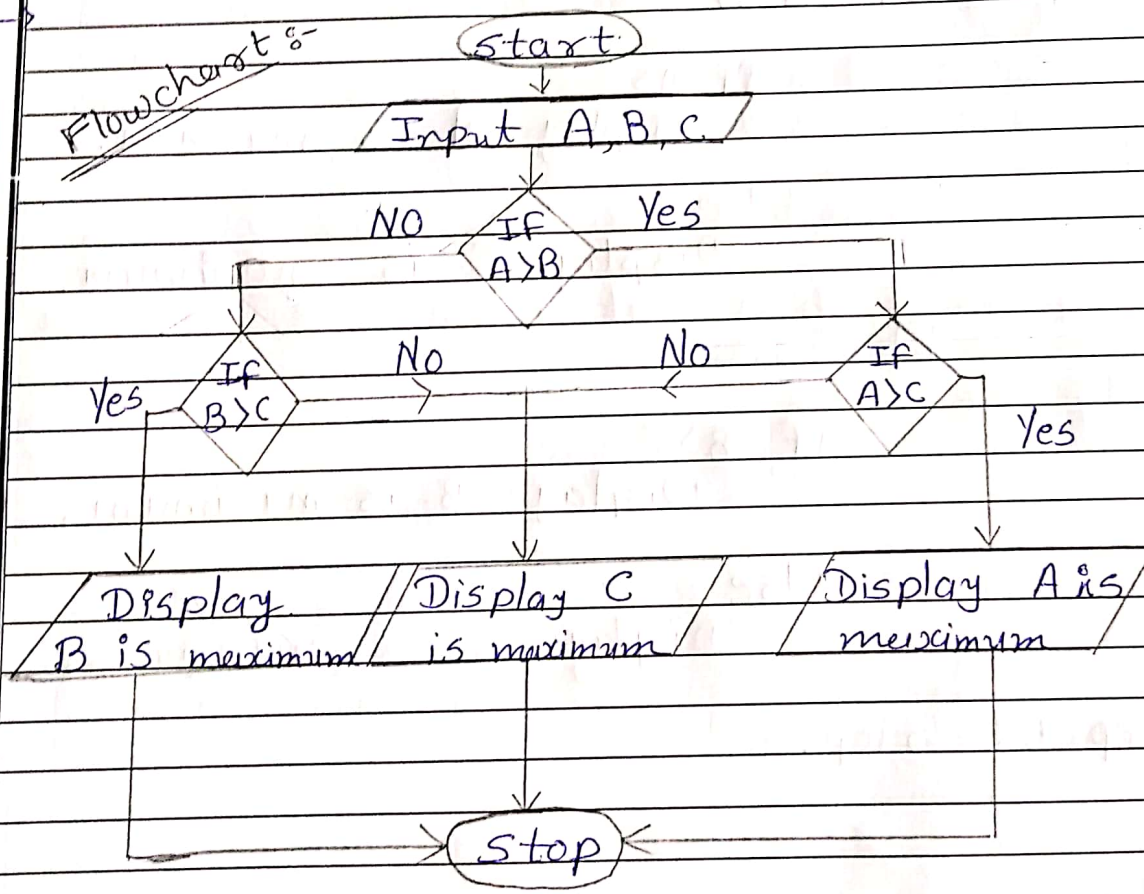
The subprogram section contains all the user-defined functions that are called in the main function. User-defined functions

are generally placed immediately after the main function.

Q.2 Draw a flow chart and write algorithm for the following:-

(a) To find the maximum number out of 3 given numbers.

Flowchart:-



Algorithm:-

Step-1 : Start

Step 2: Declare variable A, B and C.

Step 3: Read variables A, B & C.

Step 4: If A>B

If A>C

Display A is maximum.

Else

Display C is maximum.

Else.

If B>C

Display B is maximum.

Else

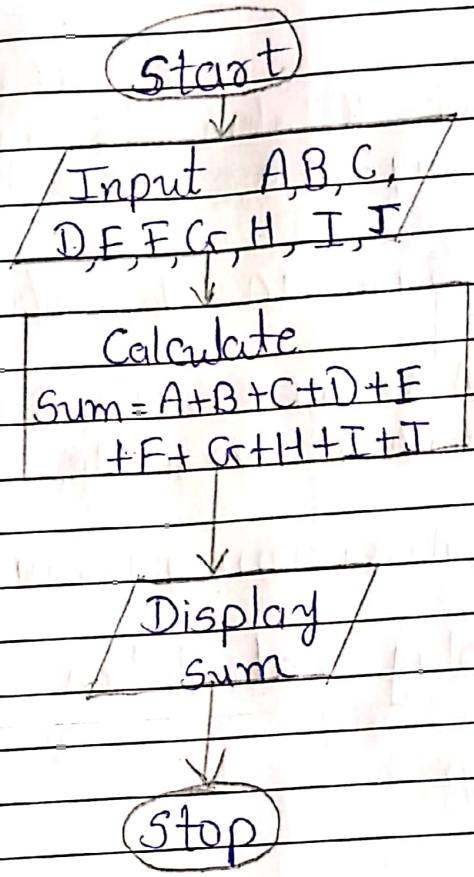
Display C is maximum.

Step 5: Stop.

(b) To perform the summation of 10 elements read from the user.

→

Flowchart:-



Algorithm :-

Step 1 :- Start

Step 2 : Accept 10 elements from user
say A,B,C,D,E,F,G,H,I,J

Step 3 : Add /Sum 10 elements.

Add A,B,C,D,E,F,G,H,I & J and store in another variable

Sum = A+B+C+D+E+F+G+H+I+J

Step 4 :- Display calculation value ie/ sum

Step 5 :- Stop

Q-3 Explain logical & relational operators used in C. Explain using proper examples.

→

Relational Operators :-

A relational operator is termed as a relational expression. The value of a relational expression is either one or zero. It is one if the specified relation is true and zero if the relation is false.

Ex; 10 < 20 is True

but,

20 < 10 is False

Relational operator is a programming language construct or operator that tests or defines some kind of relation between two entities.

C supports 6 relational operators.

Operator	Meaning
<	is less than
<==	is less than or equal to
>	is greater than
>=	is greater than or equal to
==	is equal to
!=	is not equal to

A simple relational expression contains only one relational operator and takes the following form:-

ae-1 relational operator ae-2

ae-1 and ae-2 are arithmetic expression which may be simple constants, variables or combination of them.

Ex: 4.5 <= 10 True
4.5 < -10 False
-35 >= 0 False.
10 < 7+5 True.

Asithmetic operators have a higher priority over relational operators.

Relational expressions are used in decision statements such as if & while to decide the course of action of a running program

Logical Operators :-

C has following 3 logical operators :-

&& → AND
|| → OR
! → NOT

The logical operators && and || are used when we want to test more than one condition and make decisions.

Ex :-

a>b && x==10

An expression of this kind which combines 2 or more relational expressions is termed as a logical expression or a compound relational expression. A logical expression also yields a value of 1 or zero. Where a>b is true & x==10 is also true.

Truth Table.

op-1	op-2	Value of the expression op-1 && op-2	op-1 || op-2
Non-Zero	Non-Zero	1	1
Non-Zero	0	0	1
0	Non-Zero	0	1
0	0	0	0

Q-4 Explain the ternary operator with appropriate example.

The ternary operator is known as the conditional operator.

The ternary operator is an operator that takes 3 arguments. The first argument is a comparison argument, the second is the result upon a true comparison and the third is the result upon a false comparison.

A ternary operator pair "?:" is available in C to construct conditional expressions of the form

exp1 ? exp2 : exp3

where exp1, exp2 and exp3 are expressions.

Note th Only one of the expressions (either exp2 or exp3) is evaluated.

Ex.:- To display minimum of 2 values.

```
min = x<y ? x : y
printf("%d", min);
```

Q-5 Explain C tokens? Briefly explain each token.

→ In a C program the smallest individual units are known as C tokens. C has 6 types of tokens.

1. Keywords :- Every C word is classified as a keyword / identifier. All keywords have fixed meanings & these meanings cannot be changed. Keywords serve as building blocks for program statements.

Ex break, char, else, while, long, switch, case, do, float, void, if.

2. Identifiers :- Every C word is classified as identifier / keyword.

Identifiers refer to the names of variables functions and arrays. These are user-defined names and consist of a sequence of letters and digits, with a letter as a first character. Lowercase letters are comonly used in indentifiers.

Rules for Identifiers :-

(1.) First character must be an alphabet.

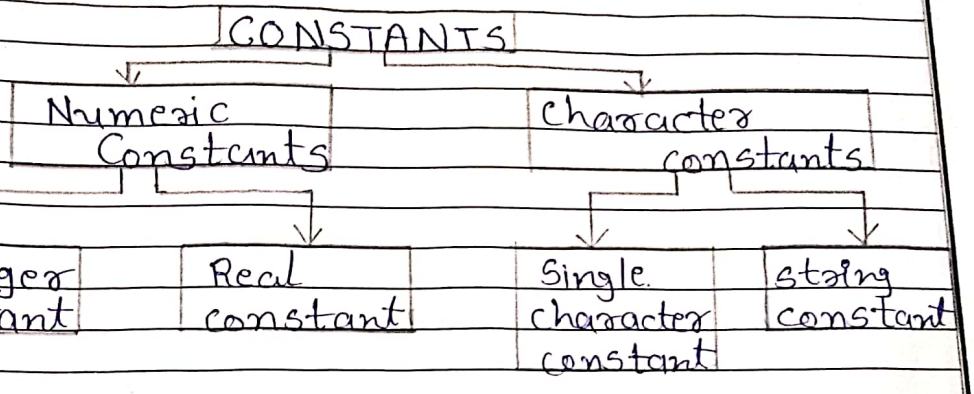
(2.) Must consist of only letters, digits or underscore.

(3.) Only first 31 characters are significant.

(4.) Can't use a keyword.

(5.) Must not contain white space.

3. Constants :- Constants in C refer to fixed values that do not change during the execution of a program. C supports several types of constants as illustrated in following figure



Embedded spaces, commas, and non-digit characters are not permitted between digits.

Ex 15 750 20,000 $1000 are illegal numbers.

4. String :- A string is an array of characters ended with a null character (\0). This null character indicates that string has ended. strings are always enclosed with double quotes(" ").

5. Special symbols :-

→ Special symbol are already listed by compile. We can't genrate out own symbols.

→ All special symbols have their own special meaning.

→ Ex :- < > ; [ ] ; ( ) ; { } ; #

6. Operators :- An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. C is rich in built-in operators and provide the following types of operators :-

1.} Arithmetic operators
2.} Relational operators
3.} Logical operators
4.} Assignment operators
5.} Increment & Decrement Operators
6.} Conditional operators
7.} Bitwise operators
8.} Special Operators

Q-6 What is the need of initialization of a variable & assignment statement used in C program ?

→

Initialization plays a key role in programming as the variables that are used for writing the code occupy a certain

amount of memory in the CPU. If a garbage value is set for a variable, then the whole logic of the program changes and will result in an incorrect value as the output.

Initializing a variable means specifying an initial value to assign to it before it's used. Notice that a variable that isn't initialized does not have a defined value, hence it can't be used until it is assigned such a value.

In Computer programming, an assignment statement sets and resets the value stored in the storage location denoted by a variable name; in other words, it copies a value into the variable. In most imperative programming languages, the assignment statement is a fundamental construct.

Q-7 Explain the bit-wise operators.

→

C has a distinction of supporting special operators known as bitwise operators for manipulation of data at bit level. These operators are used for testing the bits, or shifting them right

or left. Bitwise operators may not be applied to float or double.

Bitwise Operators

Operator	Meaning
&	bitwise AND
|	bitwise OR
^	bitwise exclusive OR
<<	shift left
>>	shift right

Q-8 What is the use of comma ',' operator?

→ The comma operator can be used to link the related expressions together. A comma-linked list of expressions are evaluated left to right and the value of right-most expression is the value of the combined expression.

Ex `value = (x=10, y=5, x+y);`

First assigns the value 10 to x, then

assigns 5 to y, and finally assigns 15 to value.

In for loops:

```
for (n=1, m=10, n<=m; n++, m++)
```

In while loops:-

```
while (c=getchar(), c!='\0')
```

Q-9 How can you read the data from keyboard and display on the screen?

→ Reading data from keyboard:-

Another way of giving values to variables is to input data through keyboard using the scanf function. It's a general input function available in C & is very similar in concept to the printf function. It works much like an INPUT statement in Basic.

scanf's general format is-

```
scanf("Control string", &variable1, &variable2,
```

The control string contains the format of data being received. The ampersand symbol & before each variable name is an operator that specifies the variable name's address.

Ex scanf ("%d", &no1);

When this statement is encountered by the computer the execution stops and waits for the value of the variable number to be typed in. Since the control string "%d" specifies that an integer value is to be read from the terminal.

Q1. Explain
diagram

->

Th
when m
multipat
which th
else is
general

if

el

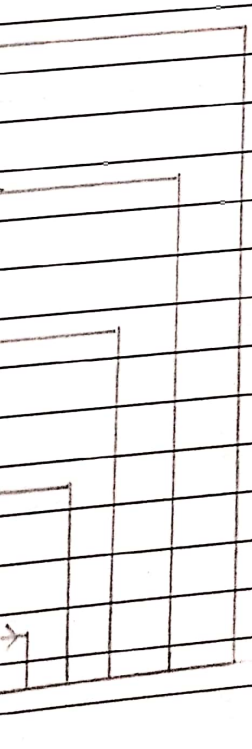
Assignment ⇒ 3
Unit ⇒ 3

Q1. Explain "if...else...if" ladder of C with neat diagram and a brief program code.

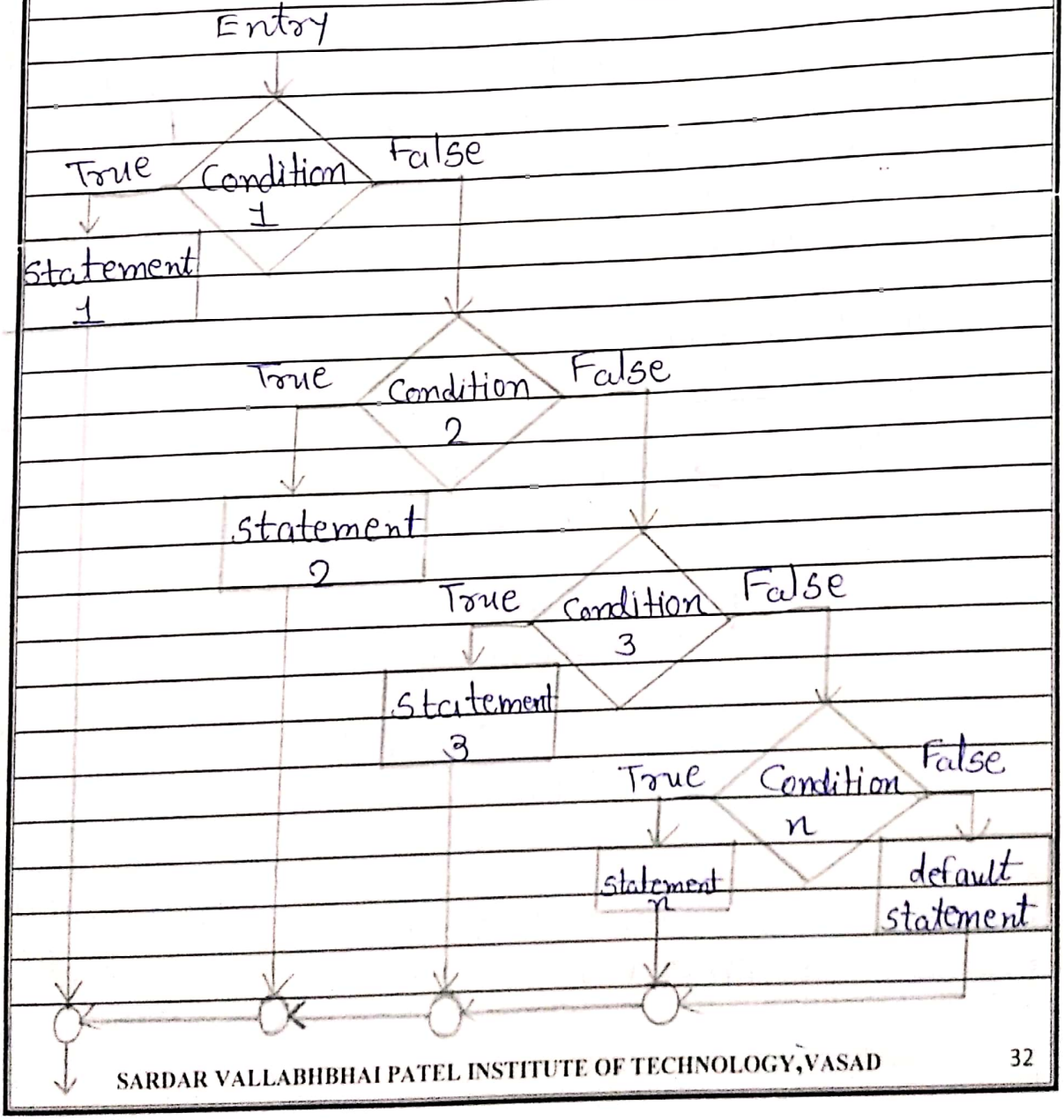
→

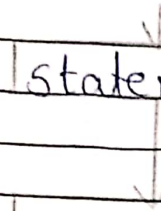
There is a way of putting ifs together when multipath decisions are involved. A multipath decision is a chain of ifs in which the statement associated with each else is an if. It takes the following general form:

```
if (condition 1)
    statement-1;

else if (condition 2)
    statement-2;

else if (condition 3)
    statement-3;

else if (condition n)
    statement-n;

else
    default-statement;
statement-x;
```



This construct is known as the else if ladder. The conditions are evaluated from the top, downwards. When all the n conditions become false, then the final else containing the default-statement will be executed.





Q-2 Compare for-loop and while-loop with the help of program to solve addition of 10 odd numbers.

→

For loop :-

```
#include <stdio.h>
  void main()
{int i, n;
  int sum;

  printf("Enter the value: ");
  scanf("%d", &n);
  printf("\n\n");

  for (i=1; i<=10; i++)

  {   if (i%2 != 0)

        {printf("%d", i); printf(", ");
        sum = sum+i;
        }

  }
```

```
printf("\n The sum of shown 10
        odd number is=%dd",sum);
}
```

→ While - loop :-

```
# include <stdio.h>
  void main()
{
   int i,n, sum;
   printf("Enter the value : ");
   scanf("%d",&n);
   printf("\n\n");
    i=1;
    while (i<=10)
{     if (i%2 !=0)
        { printf(" %d", i);
          printf(" , ");
          sum = sum + i;
          i++;
        }
}
```

```
printf("\n The sum of shown 10 odd
        number is = %d", sum);
}
```

Output:-

Enter the value: 20.

1, 3, 5, 7, 9, 11, 13, 15, 17, 19,

The sum of shown 10 odd number is = 100.

Q-3 Explain the syntax of "switch...case" statement. Define rules for "switch" statement. Write a program using "switch ... case" statement.

→

```
switch (expression)
{
   case value-1:
                block-1
                break;
   case value-2:
                block-2
                break;
   - - -
   - - -
```

```
default:
        default-block
        break;
}
statement-x;
```

- This is the syntax of "The switch statement".

- The expression is an integer expression or characters.
- Value-1, Value-2 ... are constants or constant expressions and are known as case labels.
- Each of these values should be unique within a switch statement.
- block-1, block-2 ... are statement lists and may contain zero or more statements. case lables end with a colon (:).

- Rules for Switch statement:-

→ The switch expression must be an integral type.

→ Case labels must be constants or constant expressions.

→ Case labels must end with colon.

→ The break statement transfer the control out of the switch statement.

→ The break statement is optional.

→ The default label is optional. If present, it will be executed when the expression doesnot find a matching case label.

→ There can be at most one default label.

→ It is permitted to nest switch statements. The default may be placed anywhere but usually placed at the end.

→

```
#include <stdio.h>
void main()
{
    char ch;
    int a,b,c,d;
```

```
printf("Enter any two value: ");
scanf("%d %d", &a, &b);
printf("\n\n");
printf(" + is for addition \n");
printf(" - is for substraction \n");
printf("Now, Enter your choice: ");
    ch=getch();
  switch(ch)
{
  case '+':
     { c=a+b;
    {printf("Sum is %d", c);}
    }

  case '-':
     { d=a-b;
    {printf("Your answer is %d", d);}
    }
```

```
default:
{ printf("Entered choice is invalid");
}
}
}
```

Output:-

```
Enter any two value: 5  3

+ is for addition
- is for substraction
Now, Enter your choice: +
    Sum is 8.
```

3-4 Differentiate in between "if...else" ladder with "switch...case".

	The else-if ladder	The switch-case statement
1.	In else-if ladder, the control goes through the every else if statement until it finds true value.	1. In switch statement, the value of the switch the control jumps to the corresponding case.
2.	There is no use of break statement in else-if ladder.	2. The break statement is mainly used in switch statement.
3.	It is more flexible than switch statement.	3. It is less flexible in compares else-if ladder.
4.	The code needs to be processed in the order determined by the user.	4. Each case is independent of the previous one (case).
5.	It works on the basis of true-false basis.	5. It works on the basis of equality operator.

Q-5 Write a C program to read 10 numbers from user and find sum and average of them.

→

```
#include <stdio.h>

void main()

{ float ave;
  int n, i, max, sum=0;

  max=0;

  for (i=1; i<=10; i++)

  {
    printf("Enter ten no : ");
    scanf("%d", &n);

      if (max<n)
      {
        max=n;
      }
      printf("maximum = %d", max);

    sum = sum + n;

    ave = sum / 10;

}
```

```
printf ("\n The sum of given numbers
        = %d", sum);
printf ("\n The average of given numbers
        is = %f", ave);
}
```

Output:-

```
Enter ten no : 1 2 3 4 5 6 7 8 9 10

maximum = 10
The sum of given numbers = 55
The average of given numbers is = 5.5
```

Q-6 Explain the various conditional statements.

→

If statement :-

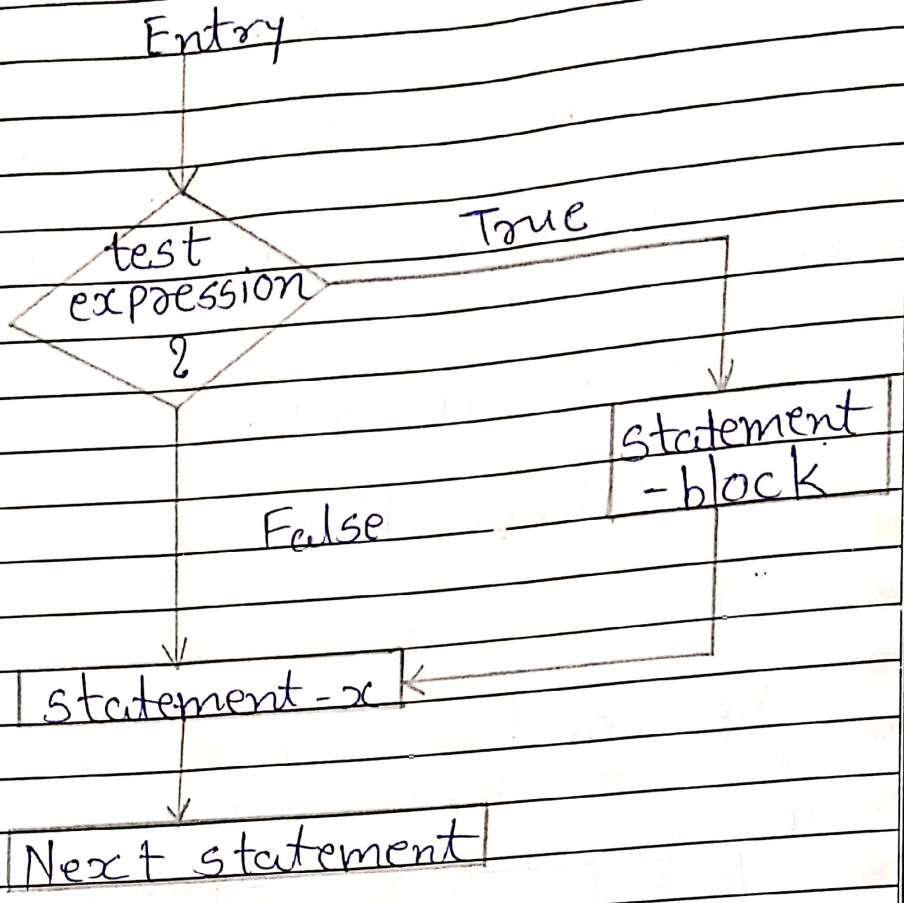
The general form of a simple if statement is

```
if (test expression)
 {
    statement-block;
 }
 statement-x;
```

The 'statement-block' may be a single statement or a group of statements. If the test expression is true, the statement-block will be executed; otherwise the statement-block will be skipped and the execution will jump to the statement-x. When the condition is true both the statement-block and the statement-x are executed in sequence.

Ex :-

```
if (code==1)
    boy=boy+1;

if (code==2)
    girl=girl+1;
```



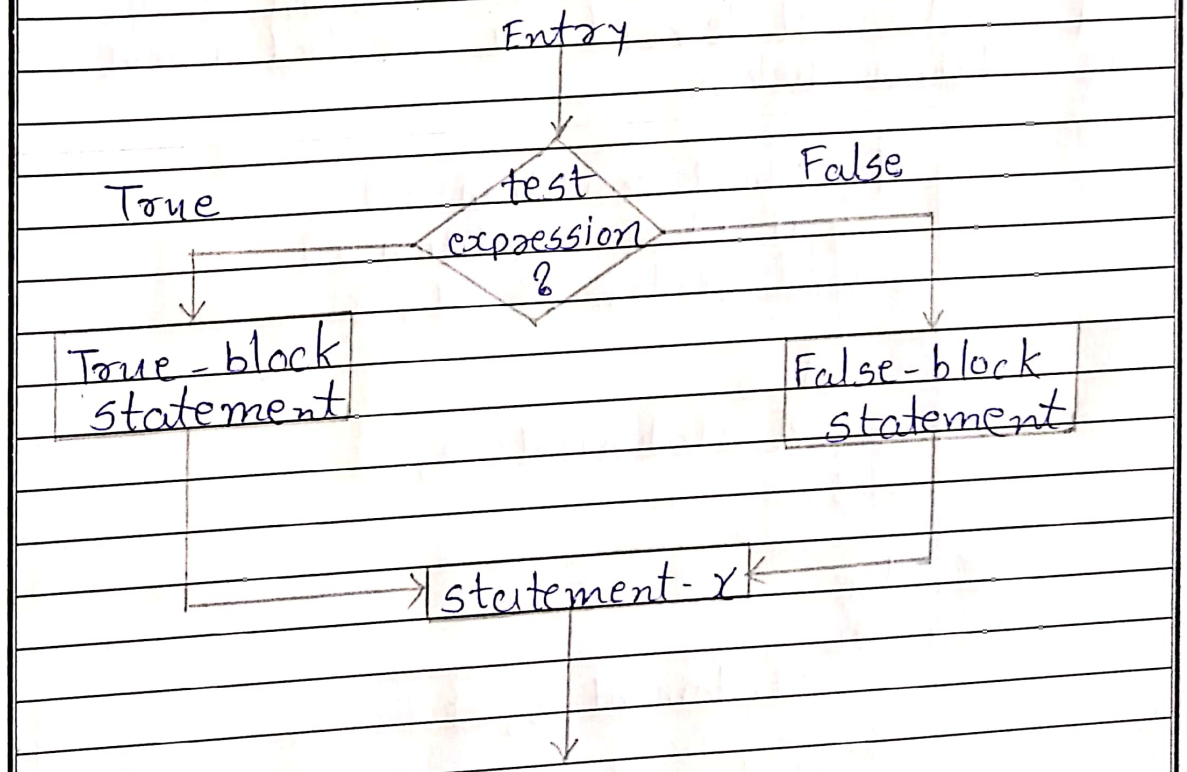
The if---else statement :-

The if--else statement is an extension of the simple if statement. The general form is,

```
if (test expression)
  {
    True-block statement(s)
  }
```

```
else
  {
    False-block statement(s)
  }

statement-x
```

If the test expression is true, then the true-block statement immediately following the if statements are executed; otherwise, the false-block statement are executed. In either case, either true-block or false-block will be executed, not both.



Ex :-

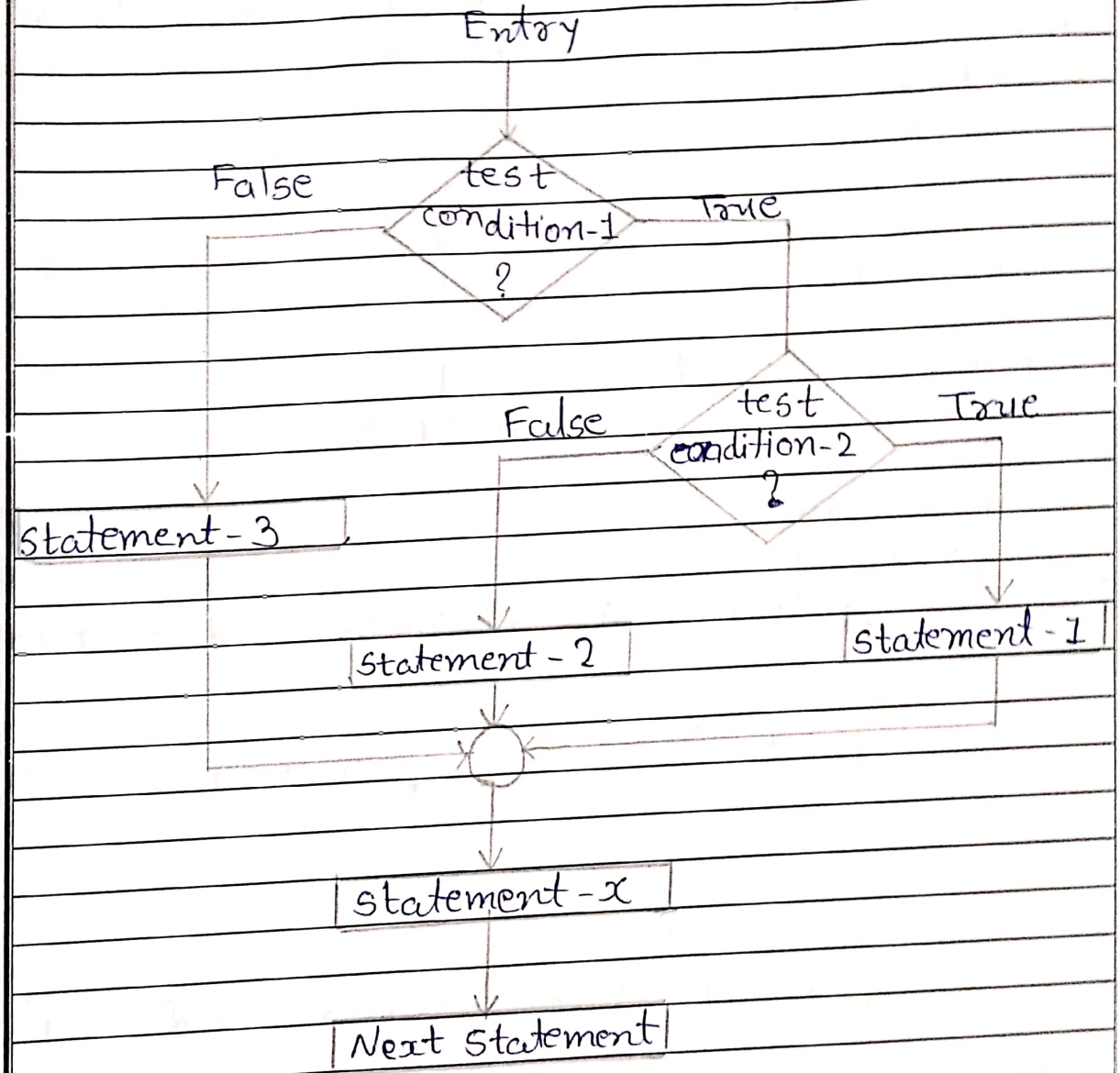
```
if (code == 1)
    boy = boy + 1;
else {
    girl = girl + 1;
```

Nesting of if...else statements :-

→ When a series of decisions are involved, we may have to use more than one if...else statement in nested form as shown below.

```
if (test condition-1)
  { if (test condition-2)
      { statement-1; }
    else
      { statement-2; }
  }
else
  {
    statement-3; }
Statement-x;
```

The logic of execution is illustrated in following figure.



If the condition-1 is false, the Statement-3 will be executed; otherwise it

continues to perform the second test. If the condition-2 is true, the statement-1 will be evaluated; otherwise the Statement-2 will be evaluated and then the control is transferred to the statement-x.

```
#include <stdio.h>
void main()
{
  float A,B,C;

printf("Enter the 3 numbers:");
scanf("%f %f %f", &A, &B, &C);

if (A>B)
  {if (A>C)
     {printf("A is maximum = %f\n", A);
     }
   else
     {printf("C is maximum = %f\n", C);
     }
  }

else if (B>C)
     {printf("B is maximum = %f\n", B);
     }
   else
     {printf("C is maximum = %f\n", C);
     }
}
```

Output :-

```
Enter any number : 66
25
41

66 is maximum.
```

Q-7 Differentiate between "while" & "do-while" looping construct.

→

While loop	Do while loop
1.) Syntax; `n=1;` `while (n<=10)` `{` `statement;` `n=n+1;` `}`	1.) Syntax; `n=1;` `do` `{` `statement;` `n=n+1;` `}` `while(n<=10);`
2.) Semicolon is not used in while loop.	2.) Semicolon is used in do-while loop.
3.) condition is checked first.	3) condition is checked later.

While loop	DO while loop
4.) Since condition is checked first, statements may or may not get executed.	4.) Since condition is checked later, the body statements will execute at least once.
5.) The main feature of the while loop is, its an entry controlled loop.	5.) The main feature of the do-while loop is it is an exit controlled loop.

Q-9 Explain "break" & "continue" statement used in C program.

→ Break statement:-

The break statement at the end of the each block signals the end of a particular case and causes an exist from the switch statement.

The break statement transfers the control out of the switch statement.

The break statement is optional. A loop can be accomplished by using the break statement. When a break statement is

e countered inside a loop, the loop is immediately exited and the program continues with the statement immediately following the loop. When the loops are nested, the break would only exit from the loop containing it. That is, the break will exit only a single loop.

Ex:-

```
#include <stdio.h>
void main()
{
  int num = 0;
  while (num <= 50)
  {
    printf("value of variable num is: %d\n", num);
    if (num == 2)
    {
      break;
    }
    num++;
  }
  printf("\n\n" finished \n");
}
```

Output:-

value of variable num is : 0
value of variable num is : 1
value of variable num is : 2

Finished.

Continue statement:

→ Like the break, C supports another similar statement called the continue statement. However, unlike the break which cause the loop to be terminated, the continue, as the name implies, cause the loop to be continued with the next iteration after skipping any statements in between.

The format is :- continue;

The use of this in while and do loops, continue causes the control to go directly to the test-condition and then to continue the iteration process.

Ex

```
#include <stdio.h>
void main()
{
  int j;
  for (j=0; j<=8; j++)
  {
    if (j==4)
    {
      continue;
    }
    printf("%d", j);
  }
}
```

Output :-

0 1 2 3 5 6 7 8