Khansa Bint-e- Tariq

- BSCS

First Semester

Programming Fundamentals.

BGNU.

Submitted to -Sir Waseem Khan.

- F240212405

Introduction to C++.

C++ is an object-Oriented Programming (OOP) language. It was developed by "Bjarne Stroustrup" in 1980. Bjarne Stroustrup wants to develope a language that support OOP features with powerful features of C language, and hence C++ evolved.

C++ consists of C language features along with class-object construct features of Simula67.

Initially, C++ is named as "C with classes." by Bjarne Stroustrup but later in 1983, it was renamed to C++ (indicating an increment to C) proposed by RRick Mascitti.

It is a superset of C language, i.e. C++ contains features of C language & adds some of its own. A all programs written in C are also C++ programs.

C++ programming language pro many essential features including poly

virtual and friend functions, templates namespaces and operator overloading et

Some Widely known products and applications that use C++

Game Engines :-

C++ is one of the best at the field of game development and man game engines including Visual Pinball (Developed Game – Pin Ball) Frostbite (Developed Game – Need for speed), REDengi (Developed Game – The Witcher), ShiVa (Developed Game – Prince of Persia 2) an many more are developed using C++.

GUI Based Applications :-

Most of the Adobe applicat including illustrator, photoshop, etc. a developed in C++. Windows media playe is also developed using C++.

Browsers: Mozilla Firefox is developed in C++. Also, Google's chrome browsers have some part written in C++.

Advance Graphics Software: Alias system's Maya 3D software that is used for the **animation**, 3D graphics, and virtual reality, is developed in C++.

Databases: The world's most popular databases Oracle Database, **MySQL**, etc. are coaded in C++ along with C.

Operating Systems: Most of the windows have three huge postions written in C++ (also with Visual C++). Some part of Apple OS also coaded in C++.

Programming Languages: Many mo programming languages including **C#**, **D**, **Java**, **PHP**, **Python**, **Limbo** e influenced by the C++ language and their initial and partial implementa written in C++.

Features and advantages of C++.

Simple: C++ is simpler to understand as the syntax is almost similar to the C language with very little changes.

Portable Language: C++ is a portable language. It means that C++ programs written for a computing environment can easi run on another computer having the same environment.

Object - Oriented: It follows the conc of OOPs like polymorphism, inheritance, encapsulation, abstraction. This makes the development and maintenance easier.

Structured Programming Language :-

C++ is a structured programm language, i.e. the whole code is divided into smaller parts (modules) with the h of functions, classes, objects. Such code are easy to understand and modify.

Mid-level Language: C++ is a middle -level language as it has features of both high-level (development of the program is straight-forward, simple and more understand

Rice Library:

C++ provides a lots of in-built functions. This saves times and makes the development faster.

Better Memory Management :-

C++ provides better memo management. It supports dynamic memor allocation and we can allocate and deallocate (free) the allocated memo at any time whenever needed.

Exception Handling :-

C++ provides Exceptu handling mechanism to deal wit the Runtime exceptions.

Some drawbacks of C++

- C++ provides data security but it up to the mark, hence there are issues with C++.

- Exception handling mechanism provided by C++ is not that much reliable.
- C++ doesnot provide automatic memory mangement mechanism.
- C++ doesnot provide **built-in multi-threading** mechanism.
- C++ is not a strictly typed language, i.e., we can pass an integer value to a floating-type variable or vice versa

ASCII code (in decimal)	Character / Symbol		
From (32–127) all are printing characters.			
32	SPACE	64	@
33	!	65	A
34	"	66	B
35	#	67	C
36	$	68	D
37	%	69	E
38	&	70	F
39	'	71	G
40	(	72	H
41	)	73	I
42	*	74	J
43	+	75	K
44	,	76	L
45	-	77	M
46	.	78	N
47	/	79	O
48	0	80	P
49	1	81	Q
50	2	82	R
51	3	83	S
52	4	84	T
53	5	85	U
54	6	86	V
55	7	87	W
56	8	88	X
57	9	89	Y
58	:	90	Z
59	;	91	[
60	<	92	\
61	=	93	]
62	>	94	^
63	?	95	

ASCII code (in decimal)	Character / Symbol		
From (32–127) all are Printing characters.			
32	SPACE	64	@
33	!	65	A
34	"	66	B
35	#	67	C
36	$	68	D
37	%	69	E
38	&	70	F
39	'	71	G
40	(	72	H
41	)	73	I
42	*	74	J
43	+	75	K
44	,	76	L
45	-	77	M
46	.	78	N
47	/	79	O
48	0	80	P
49	1	81	Q
50	2	82	R
51	3	83	S
52	4	84	T
53	5	85	U
54	6	86	V
55	7	87	W
56	8	88	X
57	9	89	Y
58	:	90	Z
59	;	91	[
60	<	92	\
61	=	93	]
62	>	94	^
63	?	95	

RSCII code (in decimal)	Character / Symbol
96	`
97	a
98	b
99	c
100	d
101	e
102	f
103	g
104	h
105	i
106	j
107	k
108	l
109	m
110	n
111	o
112	p
113	q
114	r
115	s
116	t
117	u
118	v
119	w
120	x
121	y
122	z
123	{
124	|
125	}
126	~
127	DEL

C++ Keywords

Keywords	Description
alginas	Used to specify the alignment requirement of a type / object.
alignof	Returns the alignment in terms of bytes.
and	Alternative to && operator.
and_eq	Alternative to && operator
asm	Used to insert an assembly instruction.
auto	Used to define a local (automatic) variable.
bitand	Alternative to bitwise & operator
bitor	Alternative to bitwise | operator
bool	Boolean datatype
break	Used to pass the control out of 'while', 'do', 'for', 'switch' statement.
case	Used in switch statement to define a particular case value.
catch	Used in Exception Handling, along with "try" block.
char	Basic data type. Used to indicate character type
char16-t	Datatype for UTF-16 character representation.
char32-t	Datatype for UTF-32 character representation.
class	Used to declare a class
compl	Alternative to ~ operator
concept	Used to declare a named set of requirements.
const	Used to make variable / function unmodified
constexpr	Used to declare a constant expression
const-cast	Used to cast constant variables
continue	Used to skip certain statement inside loop
decltype	Returns the type of the expression
default	Used in switch case. Default case get executed when the va doesnot match any of 'case' value
delete	Used to deallocate dynamic memory
do	do-while loop (iteration statement)
double	Datatype for floating type variables (double precisio
dynamic-cast	Used to implement runtime casting
else	Decision making statement (used with "if").
enum	Used to define enumerated type data type, i.e. a. of integer constants.
explicit	Used to define the constructors that donot a

static_assert	Used to perform compile-time assertion checking
static_cast	Type conversion. Used to cast a pointer to a base class to a pointer to a derived class
struct	Keyword which is used to create a structure i.e a collection of variables having different datatype
switch	Decision making statement which is used to test the value of expression against different "case" values
template	Used to create generic functions.
this	A special pointer that points to current object
thread_local	Used to define a thread local storage duration
throw	Used in exception Handling to throw an exception from try to catch block
true	Represents a boolean true value.
try	Used in Exception Handling. Contain code which may throw an exception
typedef	Used to define a new type. Basically, it enables us to provide a new name for existing datatypes
typeid	Used to describe an Object, i.e static type identification
typename	Used to define template argument keyword.
union	Used to create a Union which is a collection of variables of different datatype that shares common storage space.
unsigned	Type modifier. Used to alter the meaning of base data-type (in order to increase the positive range)
using	Used to import namespaces into current space
virtual	Used to declare a virtual function, i.e a function that can be overridden by a derived class
void	Means nothing. Commonly used as a function retu type and indicate that function does not return va
volatile	used to indicate that volatile entity. A volatile entity can altered in unspecified way by hardware or backg routine.
wchar_t	Used to declare a wide-charter variable
while	while loop (iteration statement)
xor	Alternative to ^ operator
xor_eq	Alternative to ^= operator

static_assert	Used to perform compile-time assertion checking
static_cast	Type conversion. Used to cast a pointer to a base class to a pointer to a derived class
struct	Keyword which is used to create a structure i.e a collection of variables having different datatype
switch	Decision making statement which is used to test the value of expression against different "case" values
template	Used to create generic functions.
this	A special pointer that points to current object
thread_local	Used to define a thread local storage duration
throw	Used in exception Handling to throw an exception from try to catch block
true	Represents a boolean true value.
try	Used in Exception Handling. Contain code which may throw an exception
typedef	Used to define a new type. Basically, it enables us to provide a new name for existing datatypes
typeid	Used to describe an Object, i.e static type identification
typename	Used to define template argument keyword.
union	Used to create a Union which is a collection of variables of different datatype that shares common storage space.
unsigned	Type modifier. Used to alter the meaning of base data-type (in order to increase the positive range)
using	Used to import namespaces into current space
virtual	Used to declare a virtual function, i.e a function that can be overridden by a derived class
void	Means nothing. Commonly used as a function return type and indicate that function does not return va
volatile	Used to indicate that volatile entity. A volatile entity can altered in unspecified way by hardware or backg routine.
wchar_t	Used to declase a wide-charter variable
while	while loop (iteration statement)
xor	Alternative to ^ operator
xor_eq	Alternative to ^= operator

Operator	Meaning of operator	Associativity
::	Scope resolution	left-to-right
++ -- type( ) type{ } ( ) [ ] . -> typeid const_cast dynamic_cast reinterpret_cast static_cast	Suffix/postfix increment and decrement. Function-style type cast Function call (Parentheses) Array subscripting Element selection by reference element selection through pointer. Run-time type information Cast away const. Run-time typed-checked cast Cast one type to another Compile-time type checked cast.	Left-to-right
++ -- + - ! ~ (type) * & sizeof( ) new, new[ ] delete, delete[ ]	Prefix increment Prefix decrement Unary plus Unary minus logical NOT Bitwise NOT C-style type cast Indirection (dereference) Address-of Size-of Dynamic Memory allocation Dynamic Memory	Right-to-left

.* ->*	Pointer to member	left-to-right	4
* / %	multiplication Division Remainder (modulo-division)	left-to-right	5
+ -	Addition Subtraction	left-to-right	6
<< >>	Bitwise left shift Bitwise right shift	left-to-right	7
< <= > >=	less than less than or equal Greater than Greater than or equal	left-to-right	8
== !=	Equal to Not equal to	left-to-right	9
&	Bitwise AND	left-to-right	10
^	Bitwise XOR (exclusive OR)	left-to-right	11
|	Bitwise OR (inclusive OR)	left-to-right	12
&&	logical AND	left-to-right	13
||	logical OR	left-to-right	14
?:	Conditional operator	Right-to-left	15
= *= /= %= += -= &= ^= |= <<= >>=	Assignment operator Short-Hand Assignment operators	Right-to-left	16
Throw	throw operator	Right-to-left	17
,	Comma	left-to-right	18

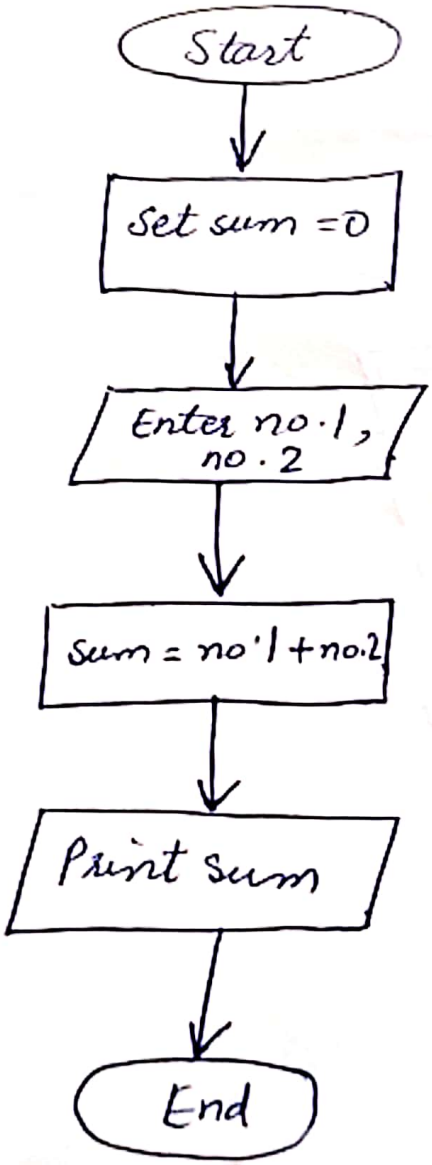
Q.NO.1

Add 10 and 20

Pseudocode

- Start
- Set sum = 0
- Enter no.1 and no.2
- Print sum
- End

Flow Chart



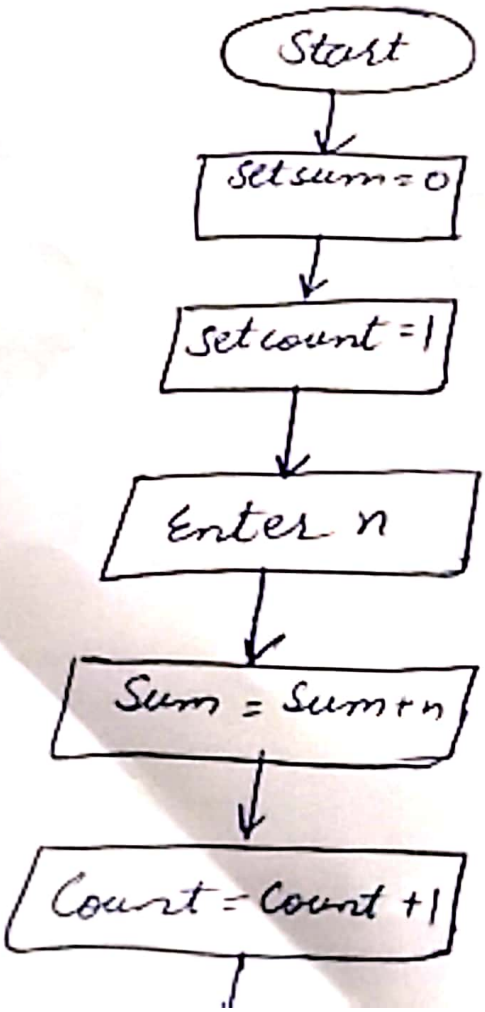
Q.NO.2

Find sum of 5 numbers

Pseudocode

- Start
- Set sum = 0
- Set count = 1
- Enter n
- Sum = Sum + n
- Count = Count + 1
- Check the conditions
- Print sum
- End

Flowchart



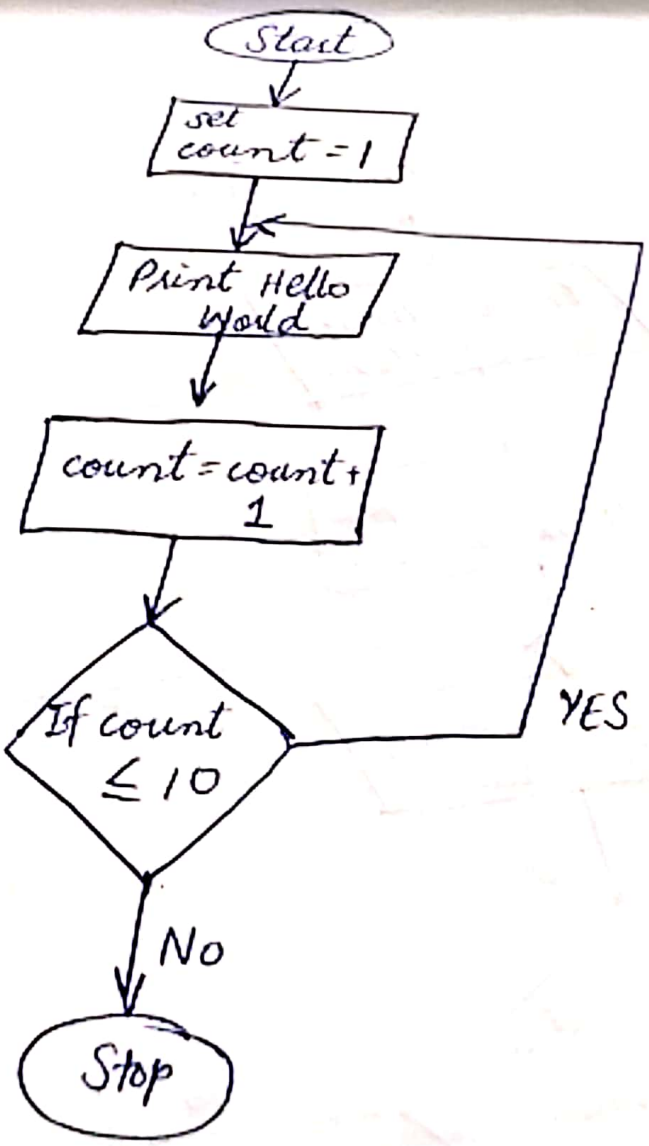
Q. No. 5

Print Hello World 10 times

Pseudocode

- Start
- Set count = 1
- Print Hello World
- Count = Count + 1
- Check conditions
- End

Flowcharts



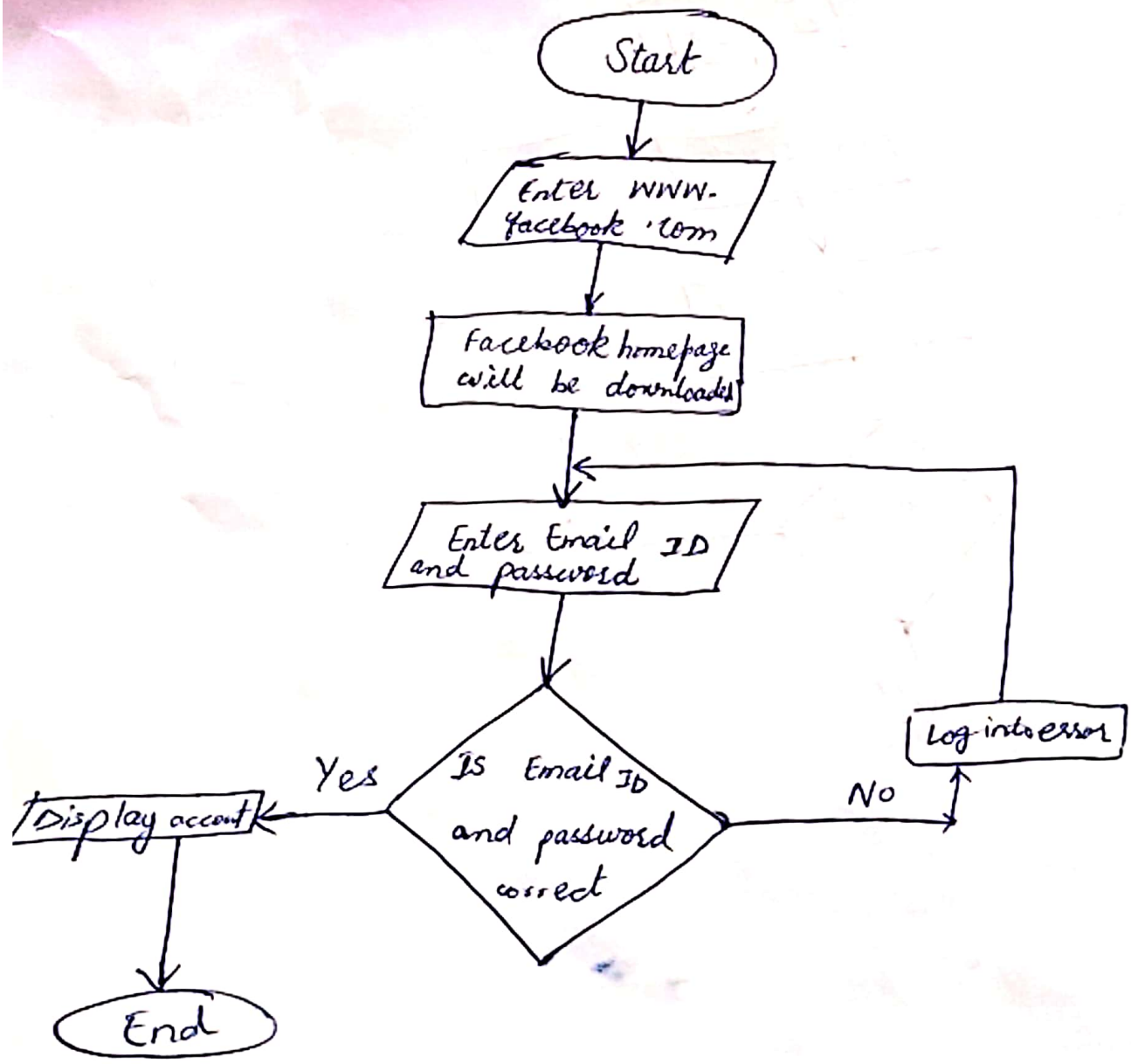
Q. NO. 4

Log into a facebook account

Pseudocode

- Start
- Enter www.facebook.com in browser
- Facebook homepage downloaded
- Enter Email ID and password
- Check condition
- Display account
- End

Flowchart

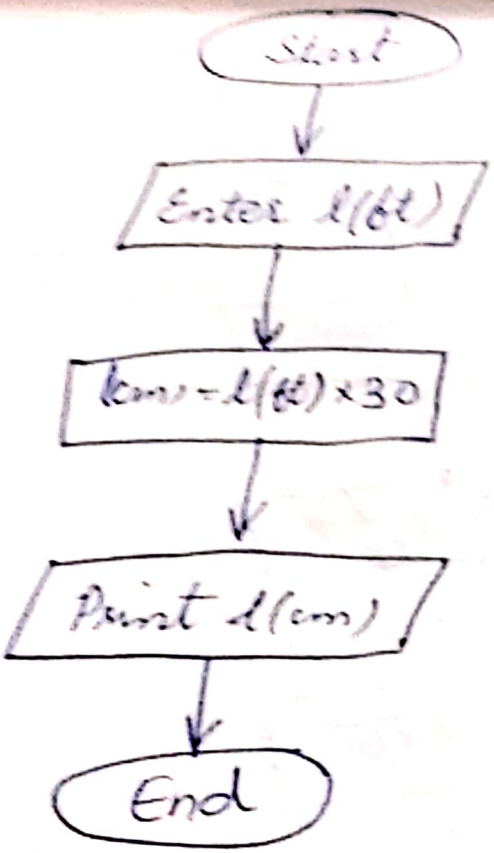


Convert l(ft) into l(cm)

Pseudocode

- Start
- Enter l(ft)
- l(cm) = l(ft) x 30
- Print l(cm)
- End

Flowchart



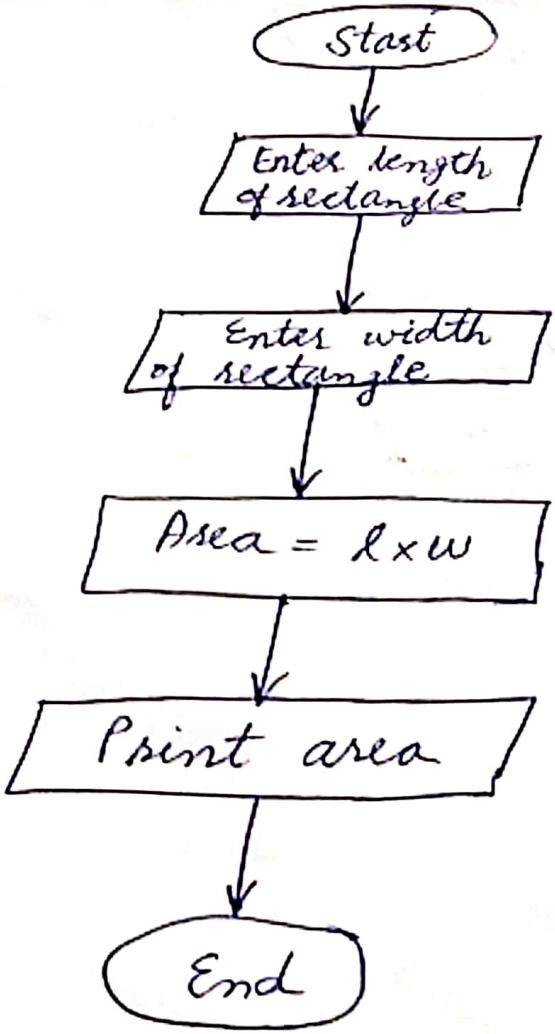
Q. NO. 6

Read 2 sides of rectangle calculate area

Pseudocode

- Start
- Enter length
- Enter width
- Area = l x w
- Print area
- stop

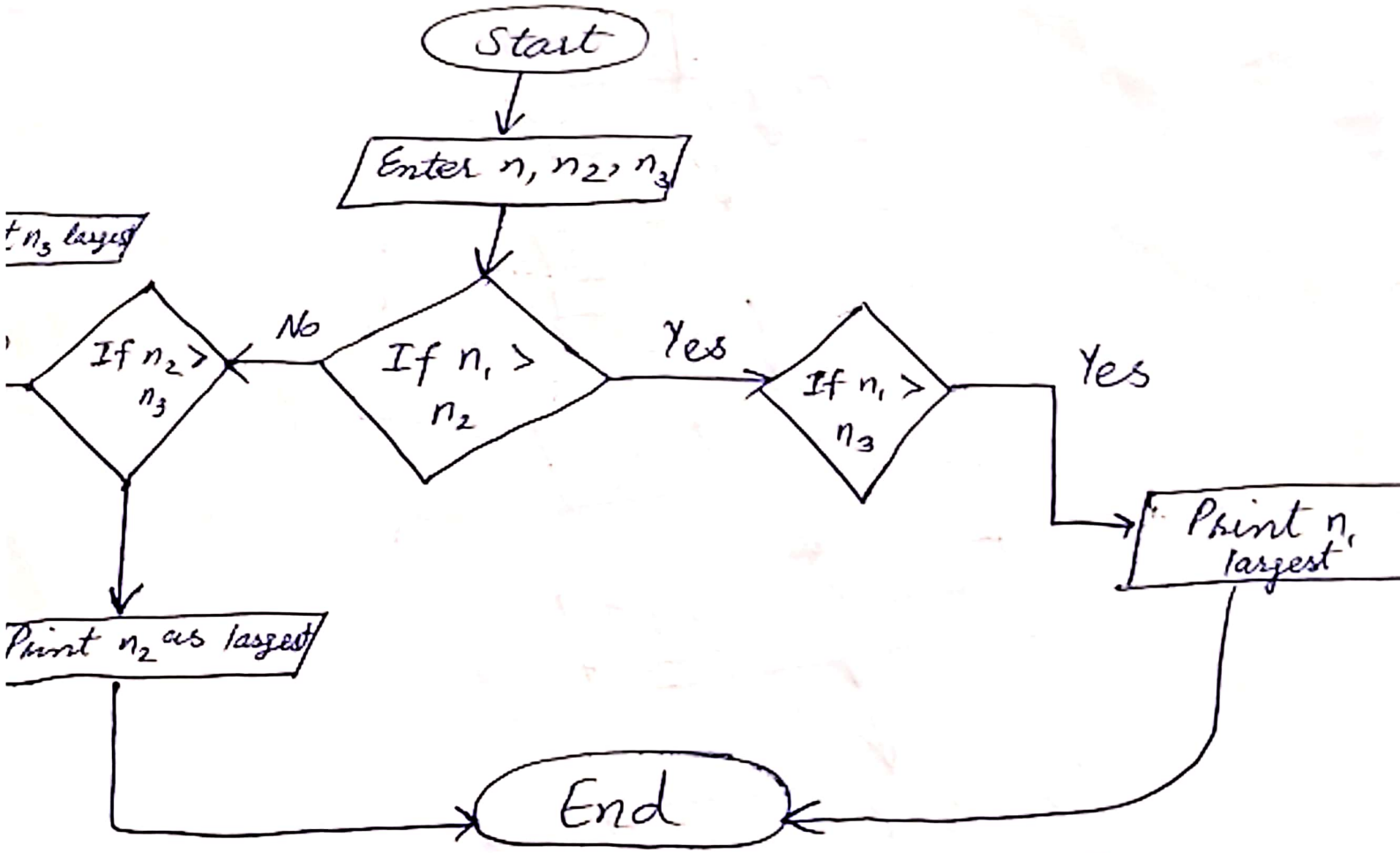
Flowchart



Pseudocode

- Start
- Enter n_1, n_2, n_3
- Check conditions
- Print largest value
- End

Flowchart



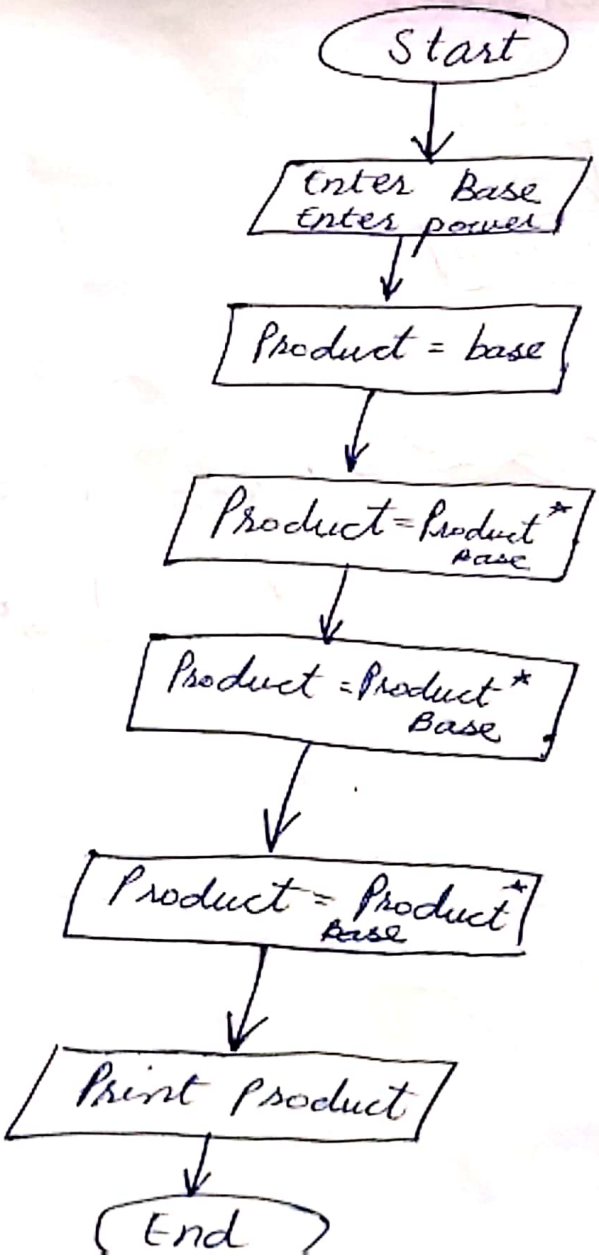
Q.NO.8

Calculate 2^4

Pseudocode

- Start
- Enter base
- Enter power
- Product = Base
- Product = product * Base
- Repeat process 2 times
- Print result
- End

Flow Chart

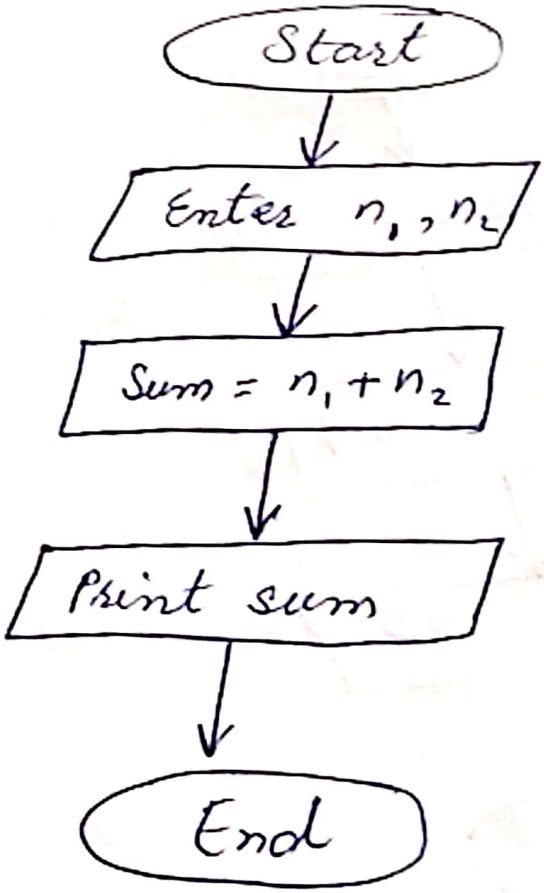


- Add 2 numbers entered by users

Pseudocode

- Start
- Enter n_1, n_2
- Sum = $n_1 + n_2$
- Print sum
- End

Flow Chart



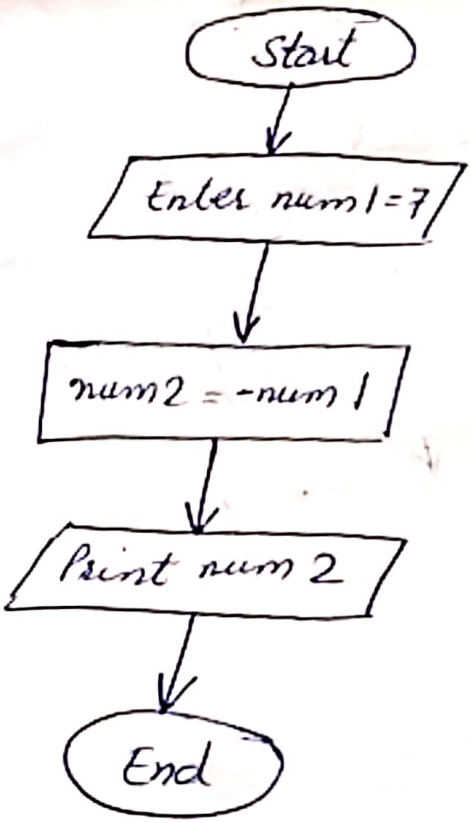
Q. NO. 10

Read num1 = 7 and store neg. value of it in num2

Pseudocode

- Start
- Enter num1 = 7
- Put num2 = -num1
- Print num2
- End

Flowchart



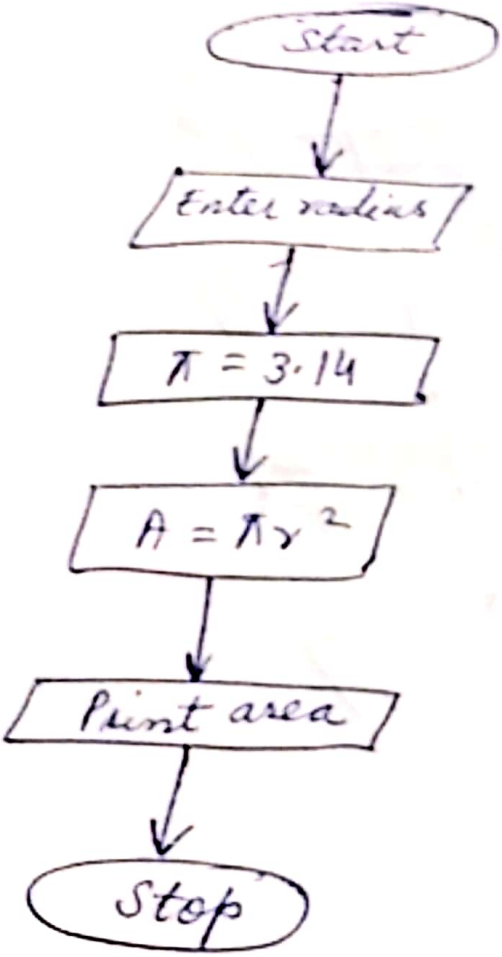
Q. No. 11

Compute area of circle

Pseudocode

- Start
- Enter r
- As $\pi = 3.14$ and area $= \pi r^2$
- Print area
- End

Flowchart



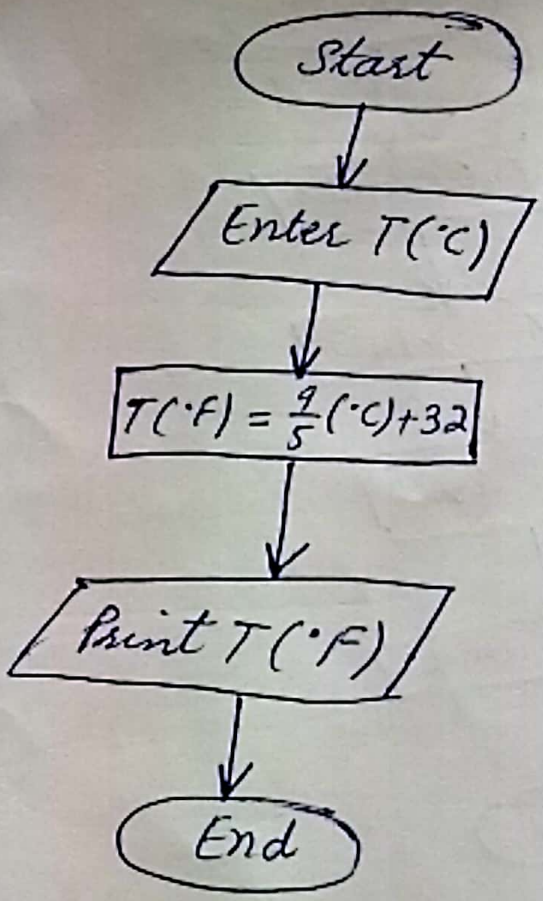
Q. NO. 12

Convert T(°C) into T(F)

Pseudocode

- Start
- Enter T(°C)
- Use formula $T(F) = \frac{9}{5}\,°C + 32$
- Print T(°F)
- End

Flowchart

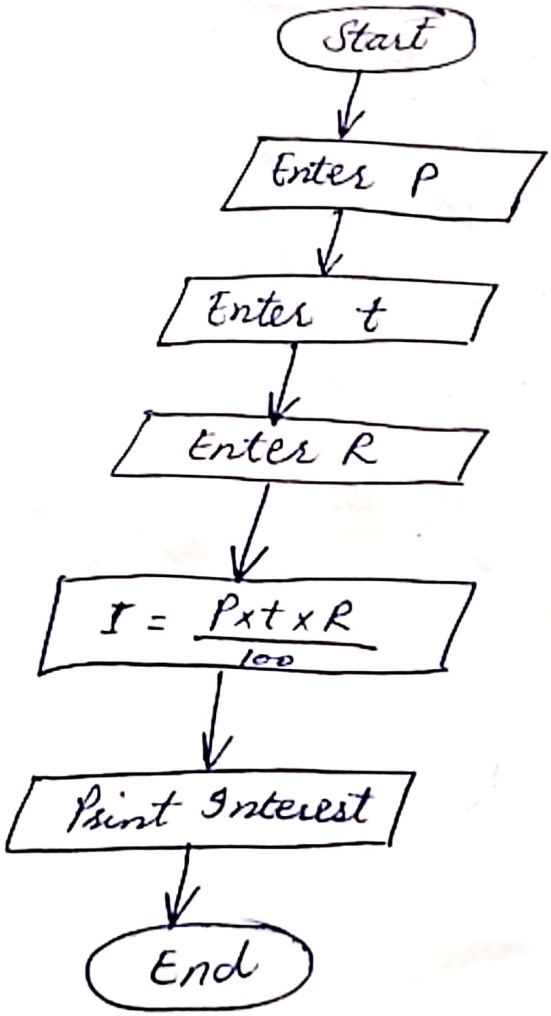


Calculate Interest

Pseudocode

- Start
- Enter P
- Enter t
- Enter R
- $I = \frac{P \times t \times R}{100}$
- Print interest
- End

Flowchart



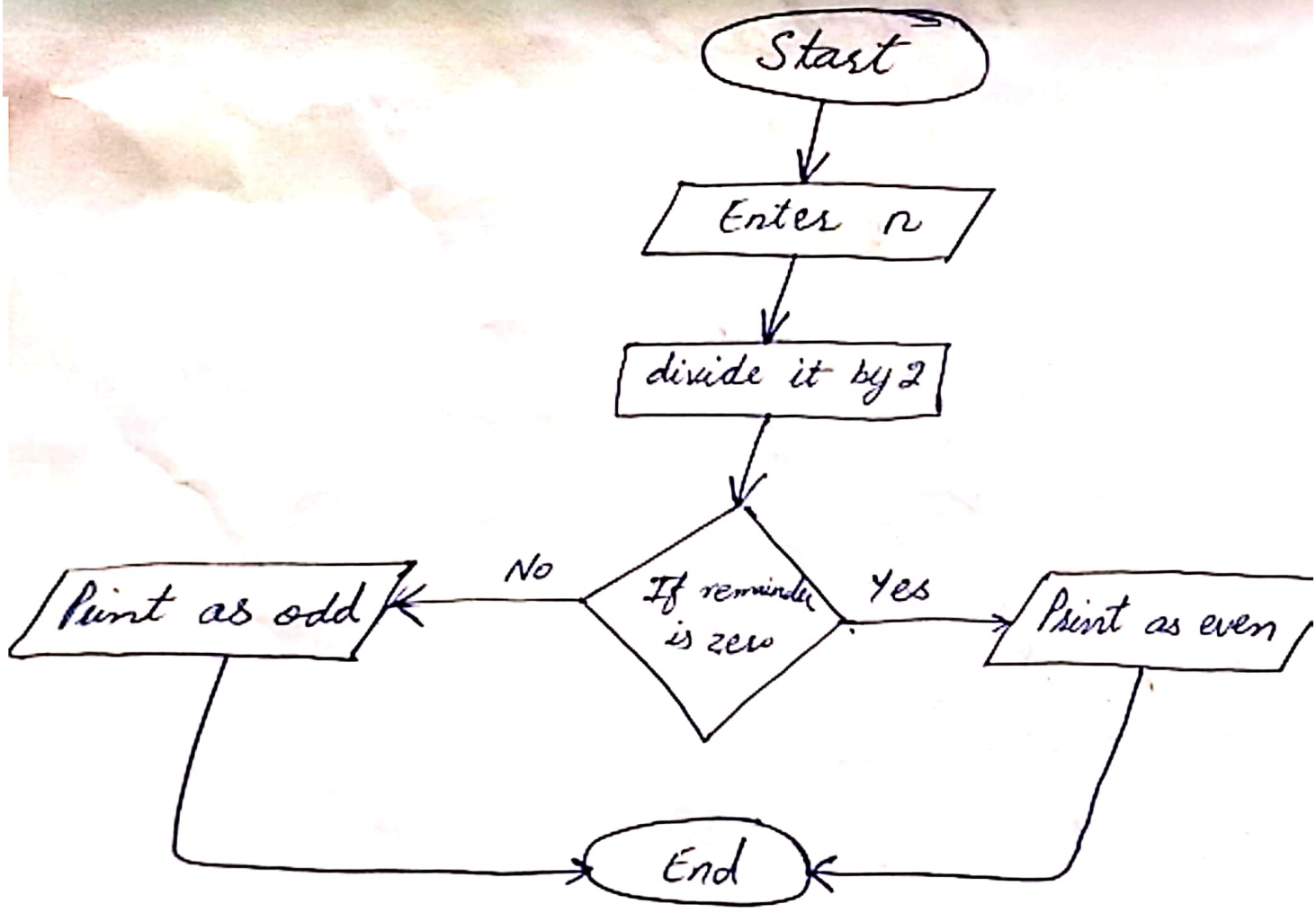
Q. NO. 14

Print a number as even or

Pseudocode

- Start
- Enter number (n)
- Divide no. by 2
- If remainder is zero it is even, otherw
- End.

Flowchart

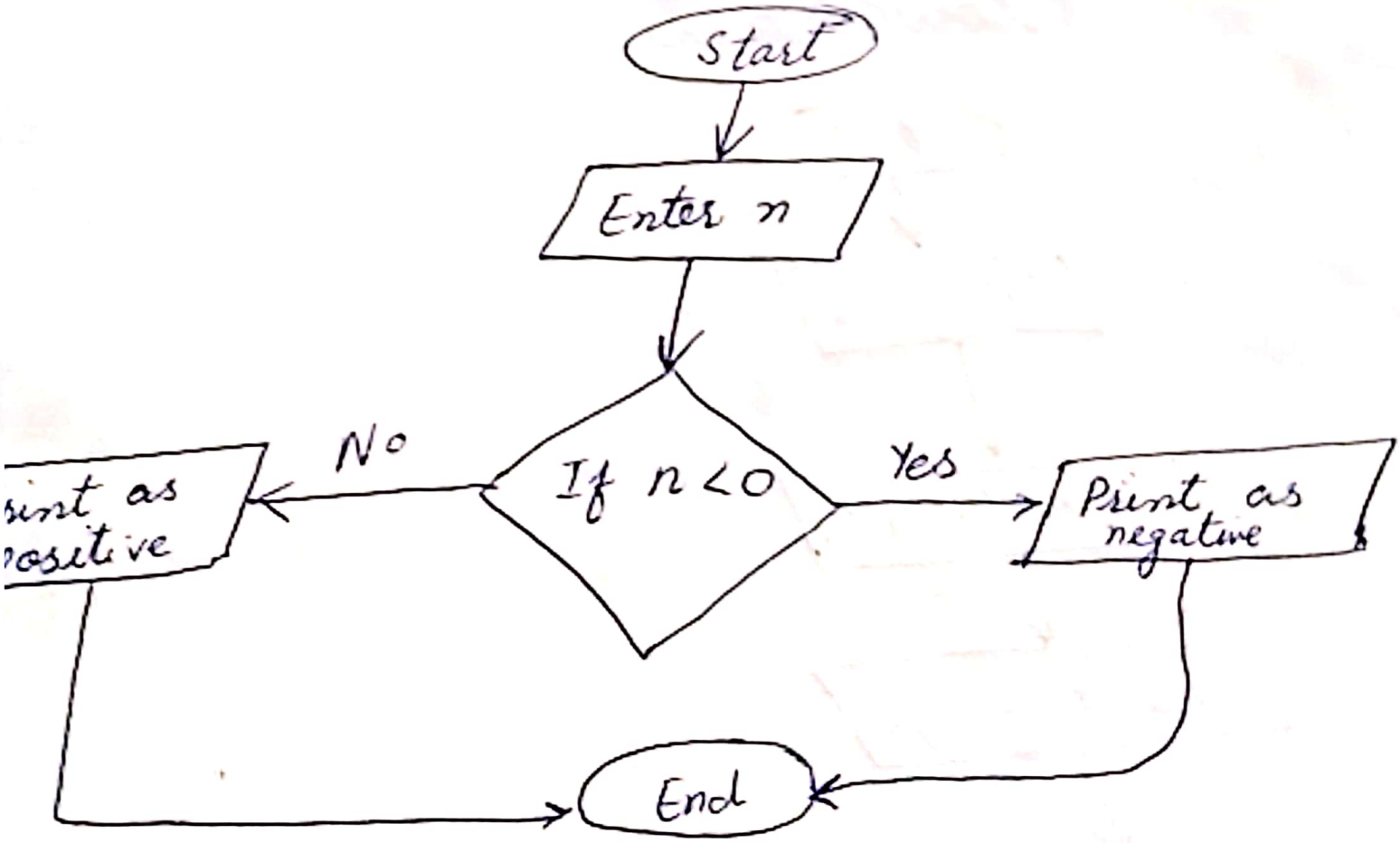


Show if a no. is positive ... negative.

Pseudocode

- Start
- Enter n
- If $n < 0$, it is negative otherwise, it is
- End

Flowchart

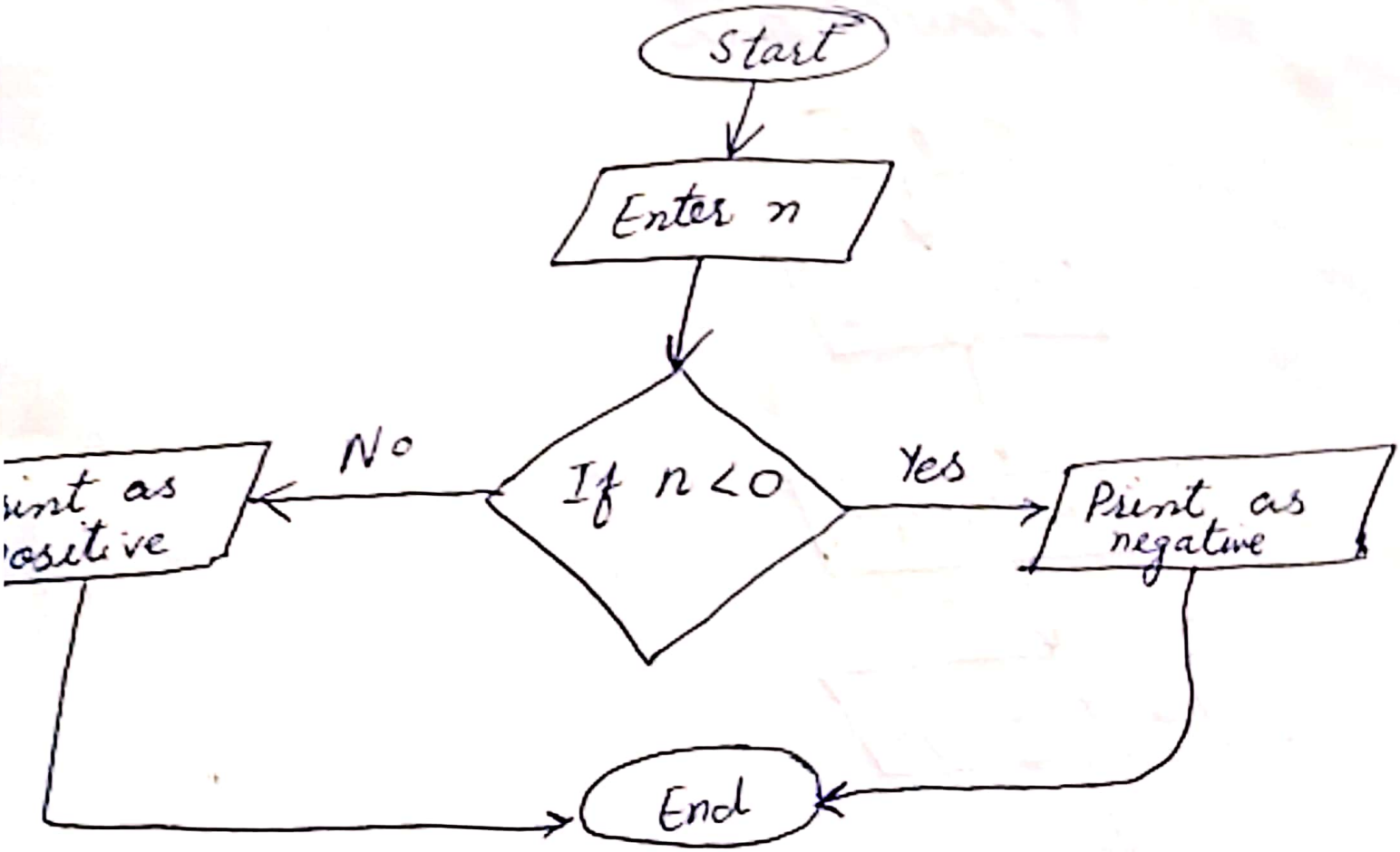


Show if a no. is positive or negative.

Pseudocode

- Start
- Enter n
- If $n < 0$, it is negative otherwise, it is
- End

Flowchart



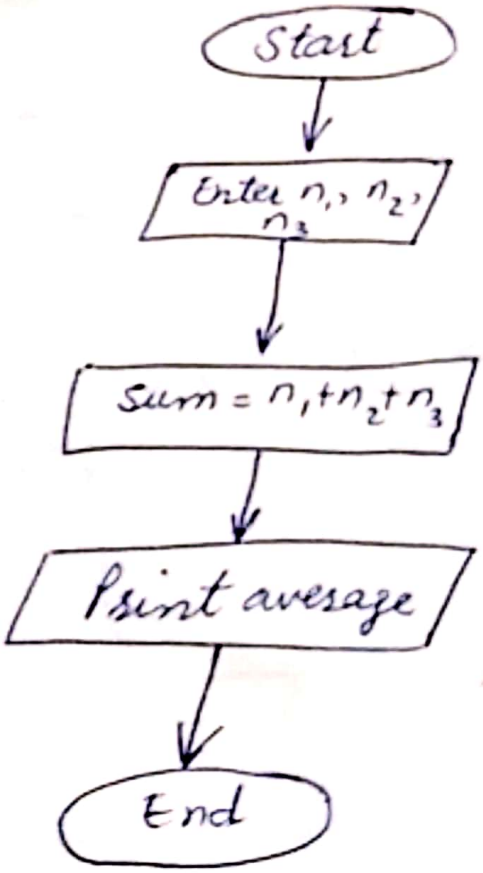
Q. NO. 16

Print average and sum of th numbers.

Pseudocode

- Start
- Enter n_1, n_2 and n_3
- Sum = $n_1 + n_2 + n_3$
- average = $\frac{n_1 + n_2 + n_3}{3}$
- Print average
- End

Flowchart



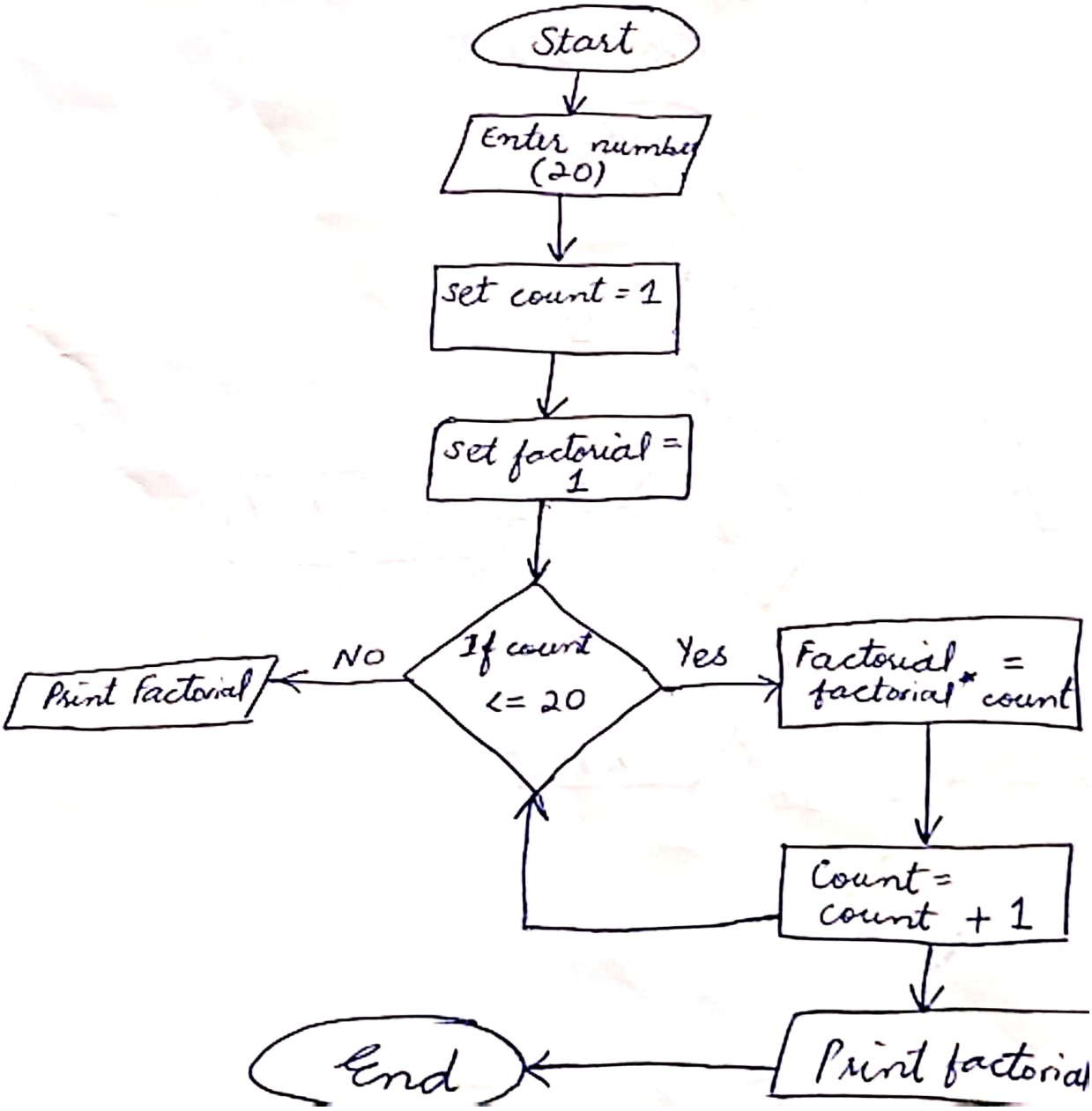
Q. NO. 18

Obtain factorial of 20

Pseudocode

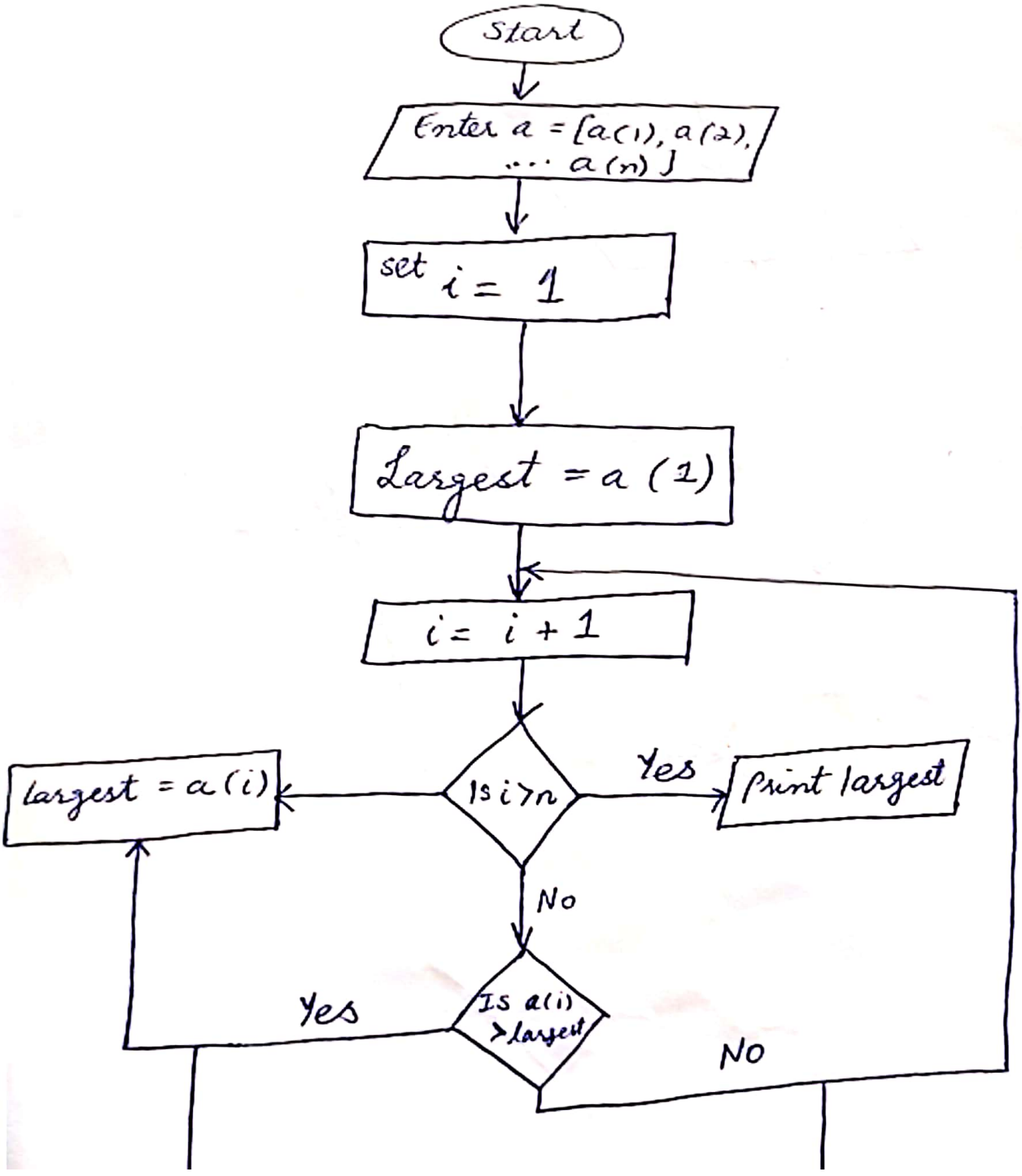
- Start the process
- Enter number (20)
- Set count to 1
- Set factorial to 1
- Multiply factorial by count
- Go to step no. 5
- Print factorial
- Check condition
- Increment cou
- End the pro

Flowchart



- Start the process
- Set $i = 1$
- Enter number
- Start by assuming that 1^{st} no. is largest.
- If any no. is greater than it upgrade the largest no.
- After checking all no, print largest no. found
- End the process.

Flowchart



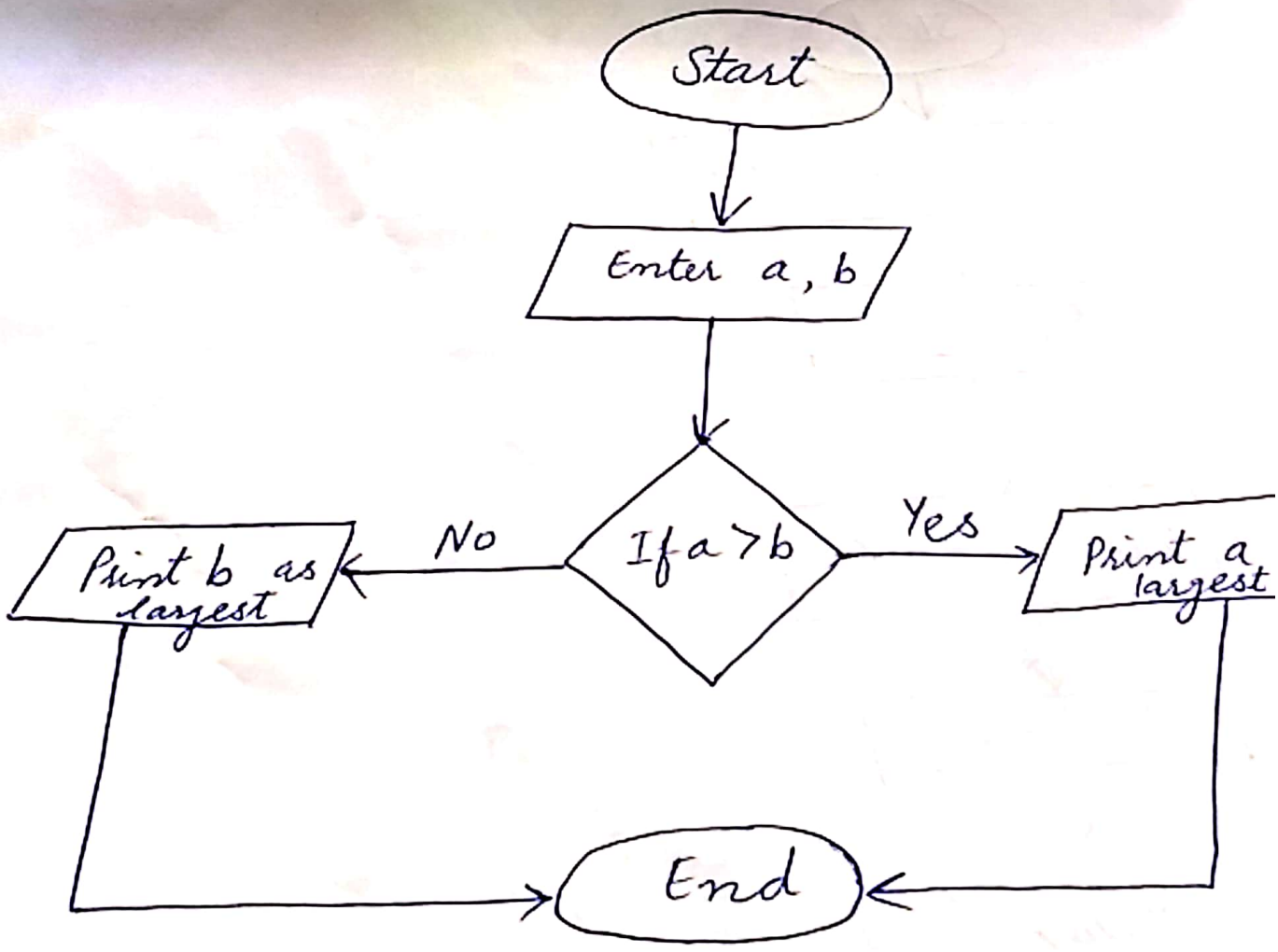
Q. NO. 20

Determine largest value among values.

Pseudocode

- Start the process
- Enter two values 'a' and 'b'
- If a > b, print 'a' as largest and vi
- Print result
- End the process

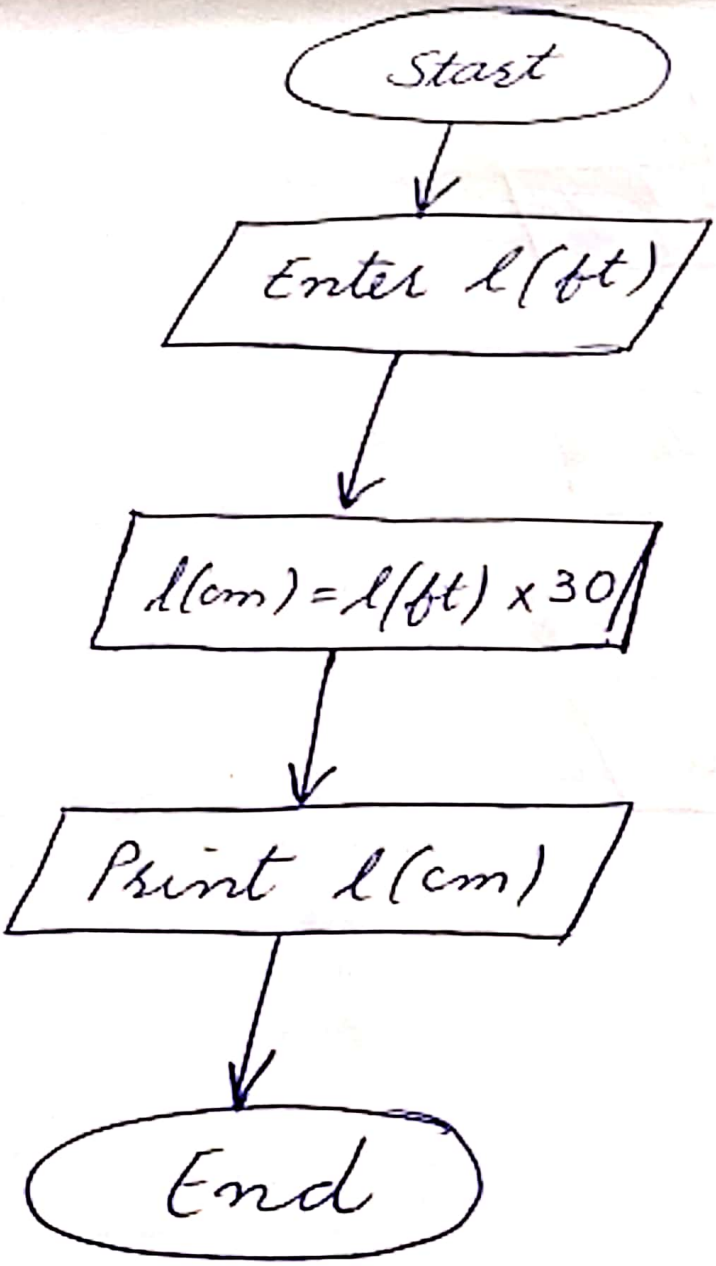
Flowchart



Pseudocode

- Start
- Enter l (ft)
- multiply it by 30
- Print l (cm)
- End

Flowchart



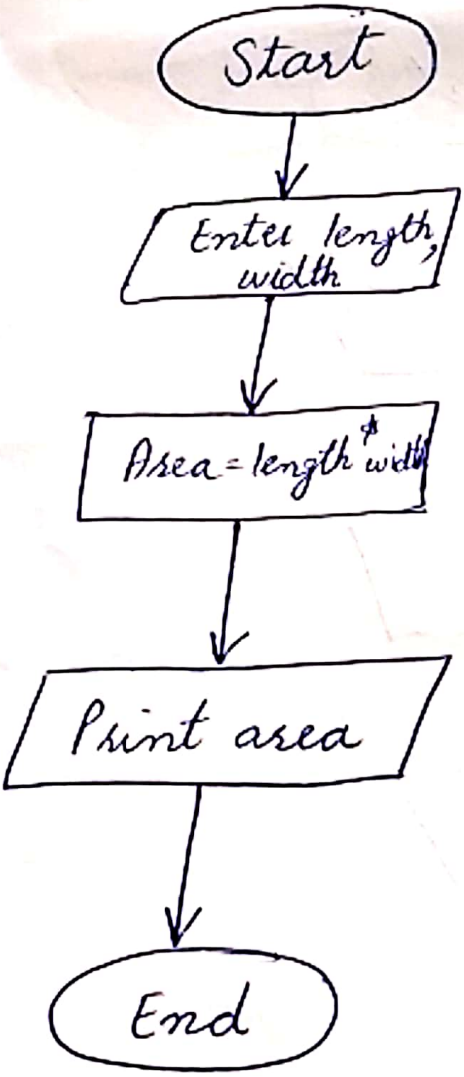
Q. NO. 22

Calculate area of rectangl

Pseudocode

- Start
- Enter length and width
- Multiply them
- Print area
- End

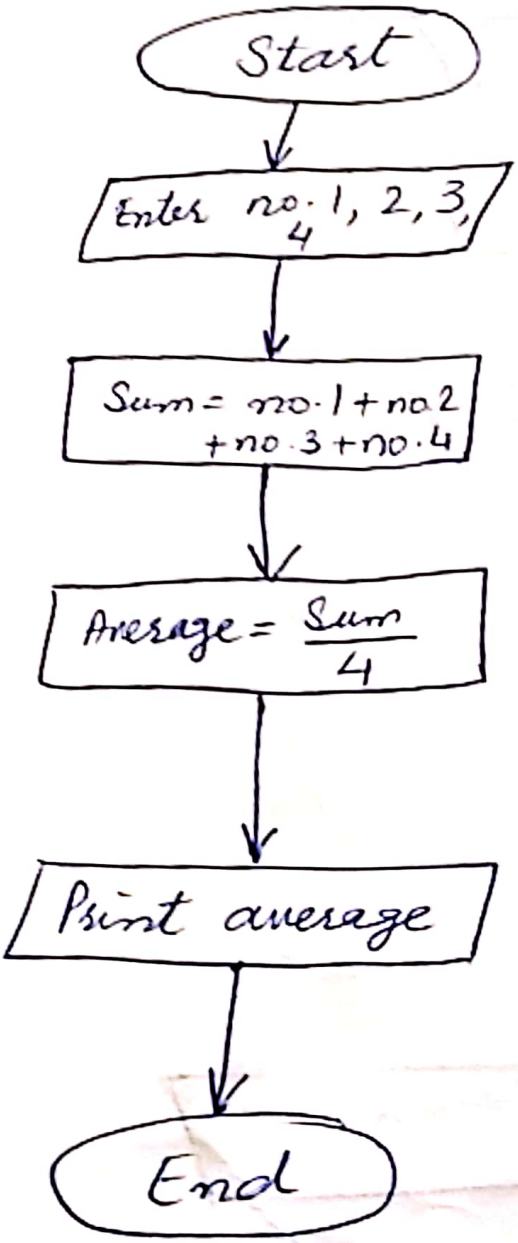
Flowchart



Pseudocode

- Start
- Enter no. 1, no. 2, no. 3 and no. 4
- Add numbers
- Divide sum by 4
- Print average
- End

Flowchart



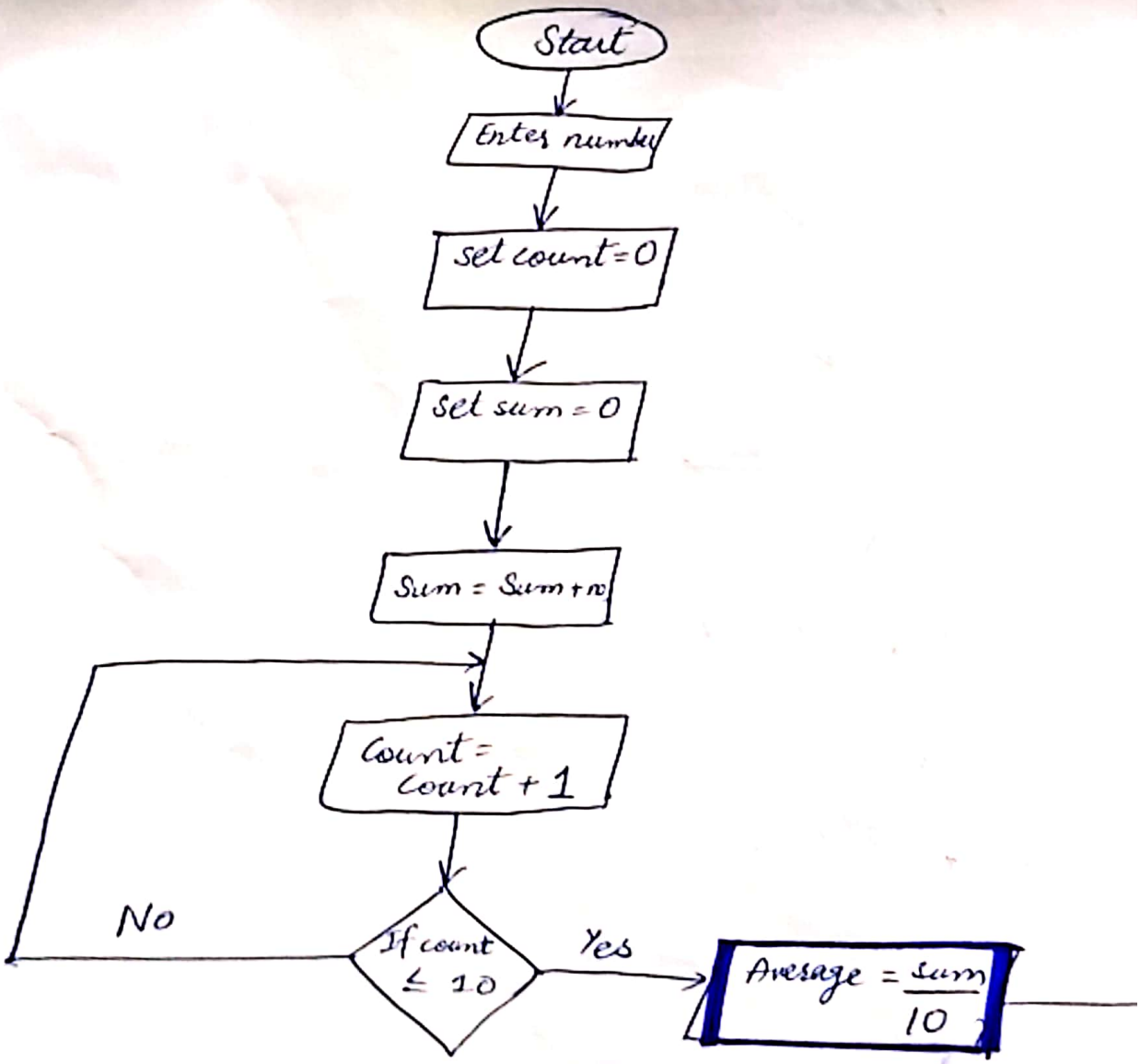
Q. NO. 24

Print average of 10 numbers.

Pseudocode

- Start
- Set count to 0
- Enter number
- Set sum to 0
- Sum = Sum + number
- Increment count by 1
- Check if count reaches 10, then average
- Declare average
- End

Flowchart

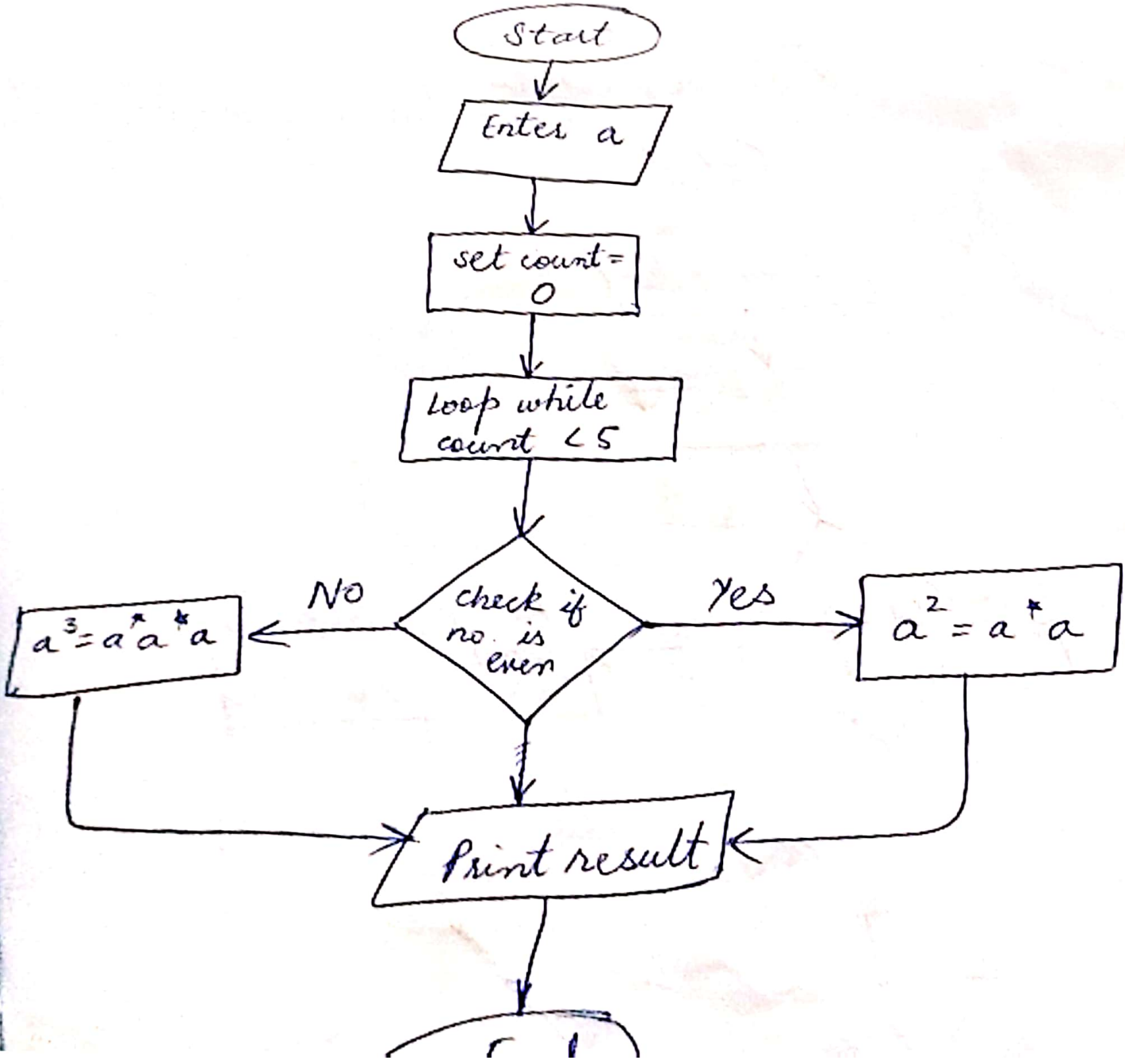


Print square of a no. if it is even and print cube if it's odd.

Pseudocode

- Start
- Enter no. (a)
- Check if it's even print square, if its odd print c
- Print result
- End

Flowchart



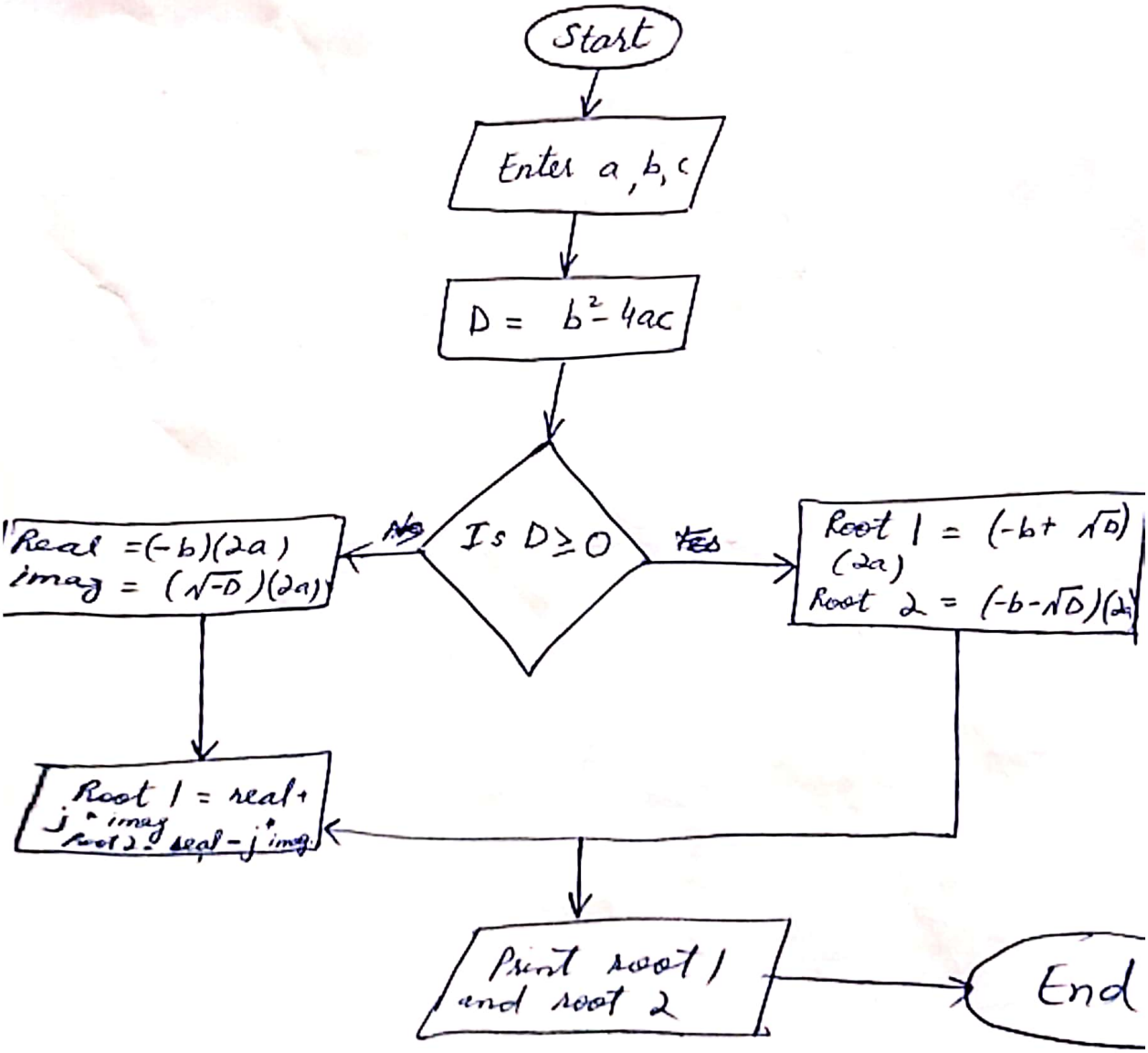
Q. NO. 26

Find all roots of quadratic equation.

Pseudocode

- Start
- Enter a, b, c
- $D = b^2 - 4ac$
- If $D \geq 0$ then, root 1 = $(-b+\sqrt{d})(2a)$
- Root 2 = $(-b-\sqrt{d})(2a)$
- Real = $(-b)(2a)$ and imaginary = $(\sqrt{-D})(2a)$
- Root 1 = real + j x imaginary
- Root 2 = real - j x imaginary
- Print root 1, root 2
- End

Flowchart



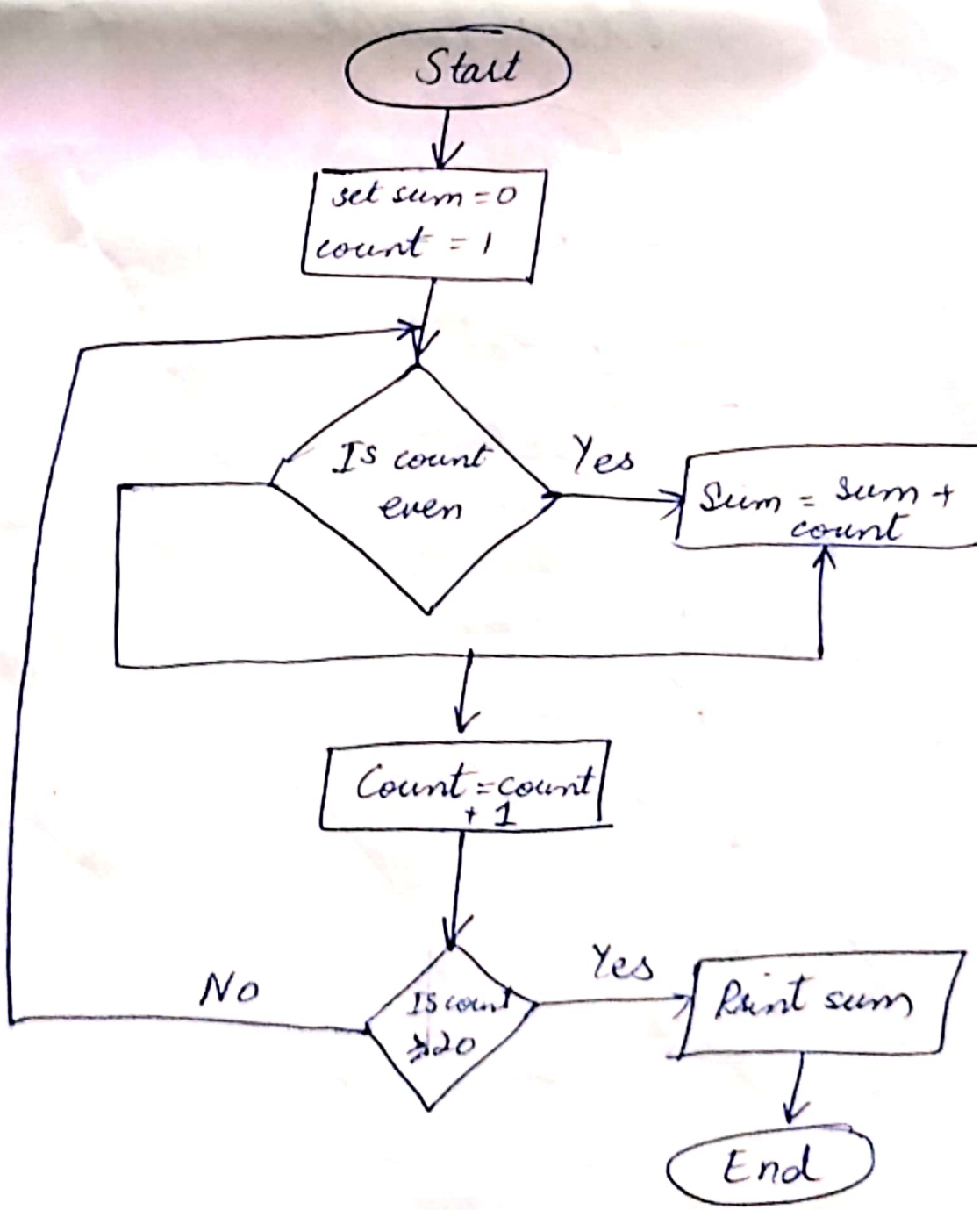
Q. NO. 28

Add even no. from 0 — 20

Pseudocode

- Start
- Set count = 1
- Count = Count + 1
- Check conditions
- End
- Set sum = 0
- Check if count is eve
  sum = sum + 1

Flowchart



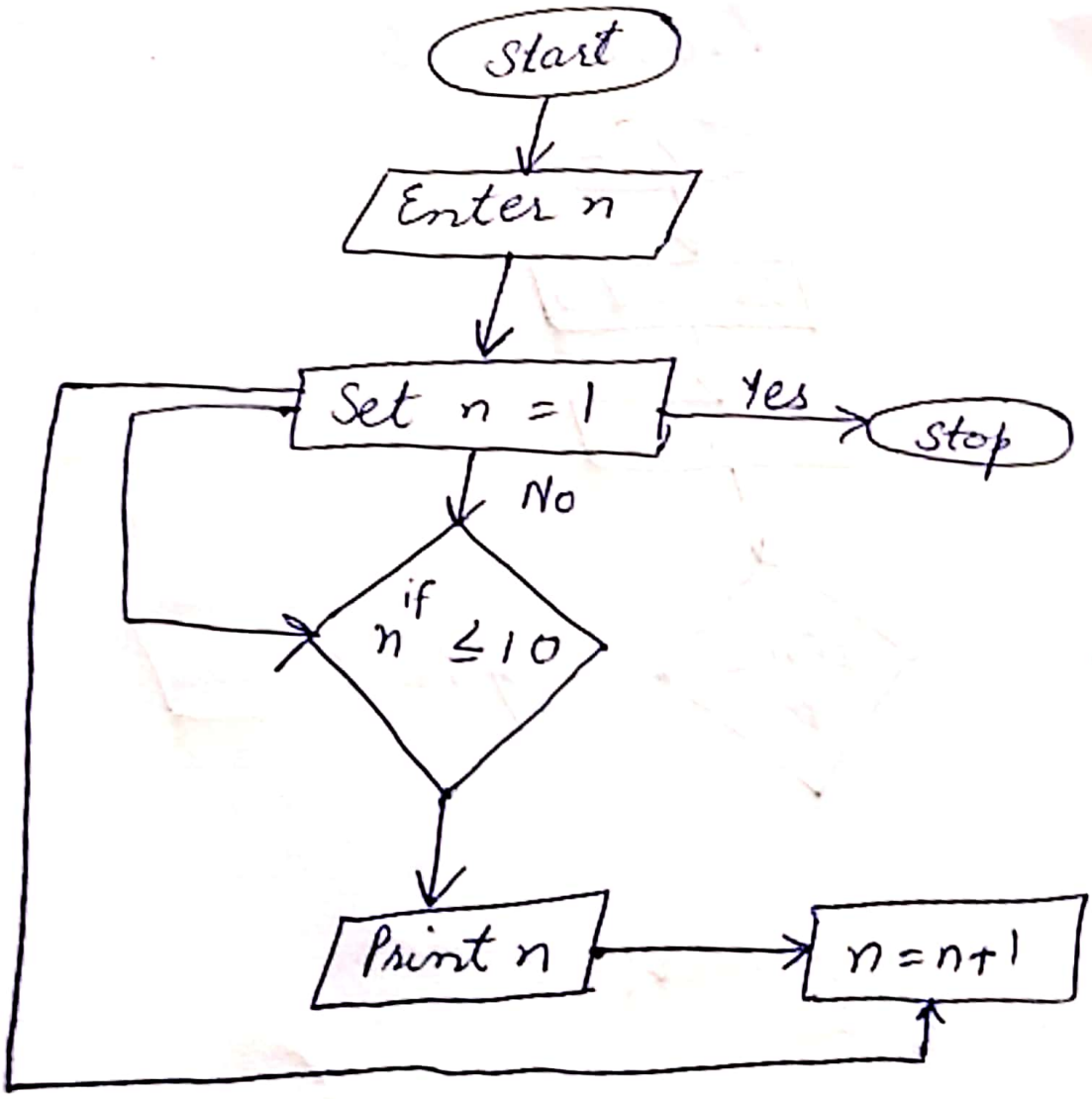
Q. No. 24

Print factorial of non-negative integer

Pseudocode

- Start
- Enter n
- If $i \leq n$ then factorial = fact · i
- otherwise print fact.
- Stop

Flow Chart



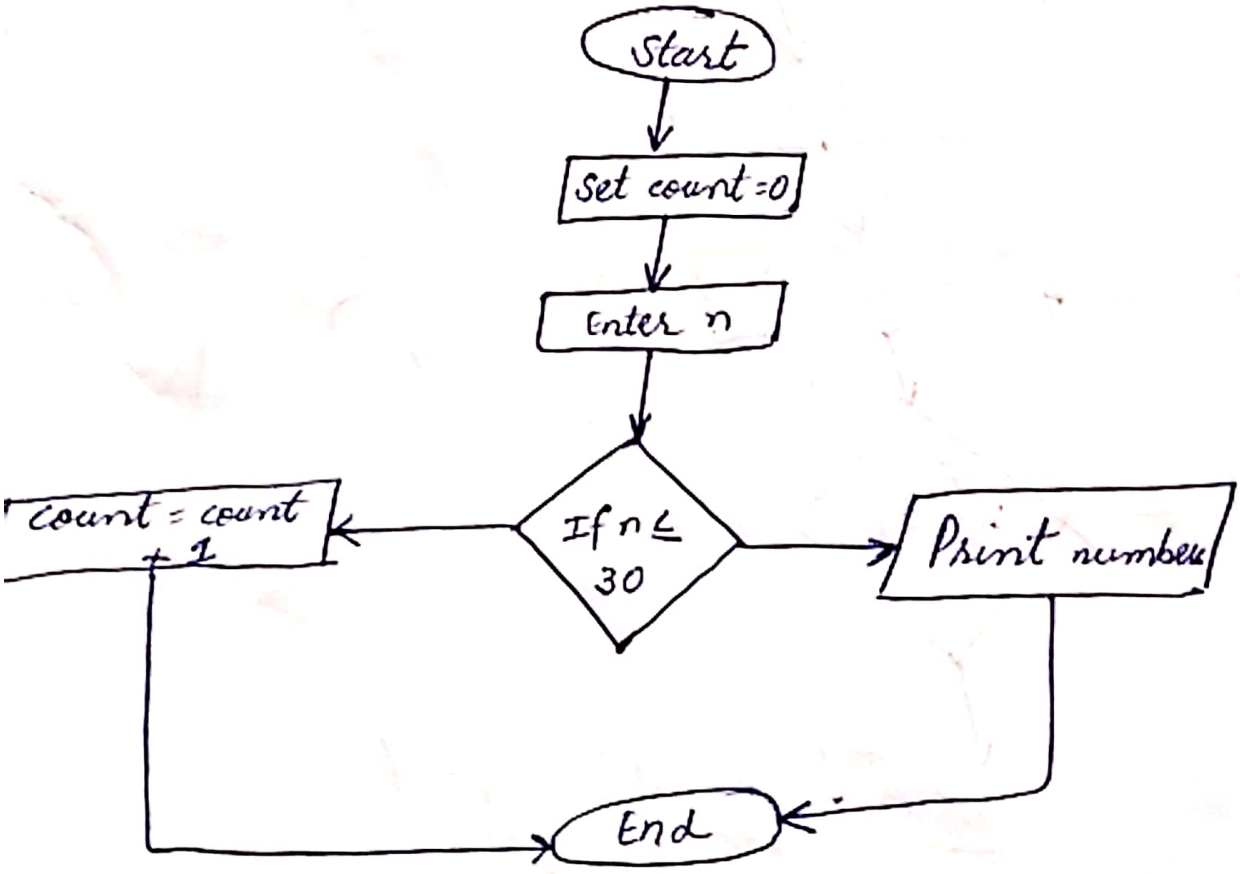
Q. NO. 30

Print numbers from 1 — 30

Pseudocode

- Start
- Set count = 0
- Enter n
- Check if $n \leq 30$, print numbers otherwise count = count + 1
- Print numbers
- End

Flowchart



Programming Fundamentals

A set of rules/instructions of words which are given to the ~~software~~ computer or a compound to solve the problems is called **programming**

→ **Computer** is an electronic machine that accepts data, processes it and produces output

→ The device or ~~anything~~ system that facilitate the human being is called a **machine**. Machine has many types

→ **Electronic machine** works on 2 states only. "ON" and "OF"

→ Computer can understand only binary language. Why?

Notes + 50 names of programming languages ❤ o

A4 size paper that is plain.

ture · # · 01

rogramming : A set of instructions or a pound of words which are given to the nputer to solve problems are called **programming.**

→ Programming is done in many languages like ava, Python, C++, Felix, E, Joy etc.

→ **Computer** is an electronic machine that cepts data, processes it and produces output.

→ **Components** of Computer are :-

ardware ① CPU (Central processing unit) ② Memory (RAM) ③ Storage ④ Motherboard ⑤ Input-evices ⑥ Output devices. ⑦ GPU ⑧ Cooling System. → These all are ncluded in hardware of computer.

ftware ① operating System (OS) ② Applications.

computer does not have only 2 components. All the bove mentioned are the components of computer ~~But software and hardware are its 2 major componen~~

· Basic Function of a Computer :

nput : Receiving data through input devices
rocessing : Performing operations on data using CP
Storage : Saving data for future access
utput : Display or presenting processed data thro
but devices.

Common

Some Types of Computers are :-

① Personal Computers (PCs) ② WorkStations ③ Servers

- Desktop
- Laptop
- Netbook / ultrabook

④ Main Frame Computers ⑤ Super Computers ⑥ Embedded Compute

⑦ Tablet Computers ⑧ Smartphones ⑨ Micro Com

→ Information about Hardware and Software :-

CPU:

Machine: The device or system that facilitate human beings is called machine.

→ Machine has many types.

Electro machine is a type of machine that converts electrical energy into mechenical energy or vice versa.

E.g: Generators (→ convert mechinical energy into electricity), Electric Motors (→ convert electrical energy into mechenical work)

→ Electromachine works on **2 states** only. It can only be **on** or **off**.

→ Computer can understand only **binary language** composed of 0s and 1s)

Lecture #: 02 Artificial Intelligence

No. of outer loops run (1 to 5) (no. of rows)

No. of space loop runs (inner loop) $j >= i$

No. of print loop runs (inner loop) $k <= i$

Outer loop run	Space loop run	Print loop
1st run	4 times	1 time
2nd "	3 "	2 times
3rd "	2 "	3 times
4th "	1 "	4 times
5th "	X	5 times

↳ dependent upon i → no. of stars

Code

```
n = 5; // size
for (i = 1; i <= n; i++) // outer loop
{ for (j = n - 1; j >= i; j--) // space loop
  { cout (" "); // whitespace
  }
  for (k = 1; k <= i; k++) // print loop
  { cout (" * ");
  }
```

(straight in 1st row)

جتنے stars ہیں اس کو n کہ کے اس کو minus کریں گے set '5' کیا ہے سے
اتنی space ڈال دیں گے۔ اس لیے یہ
" i ہے dependent ہے sp

lec. #. 04

Signs/Shapes Commonly used as symbols in Flowchats:-

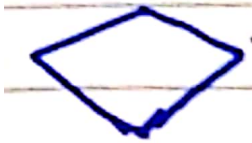
- A flowchat is a visual representation of a process, system or algorithm using different shapes (called symbols) to represent steps decisions or actions and arrows to ~~represent~~ indicate flow of control between them.

SYMBOLS AND THEIR USES


→ oval → represents start or end of a process.


→ rectangle → a process or operation step (جیسے کوئی کام یا steps)


→ Arrow → Indicate the flow between steps.


→ Diamond → use for indicating condition / Signifies a point requiring a yes or no


→ Parallelogram use for input and output operations.

Compiler :- Converting source code of the C language in machine code

not compiler
Visual studio is IDE.

IDE → Integrated Development Environment
↳ provides environment.
↳ IDE remains same and we change compiler

Single line coment ⟶ //

Whole line as coment ⟶ // At starting

Compiler ignores coment.

User Input :-

Decleration ⟶ int a
↳ to declare a variable

Initialization ⟶ a = 5
↳ Initialize ↳ value of variable

int a = 5 ⟶ Both decleration & Initialization

```
# include <iostream>
using namespace std;
```

* file name میں کوئی space نہیں رکھتے ۔
eg: Filename : Nothingspecial

* جب else if لگائیں گے تو else لگانا ضروری

e.g.: if( ){ } else if( ){ }else if( ){ }else{ }

```
#include <iostream>
using namespace std;
int main ( ){
```

* "nested if"
condition کے اندر condition ہو
e.g.: if( ){
if( ){

Loop in C++ → to repeat a specific sequence

For loop → 3 parts میں divide

① declaration/init
② Condition
کہ loop کب تک چلے گا ۔
③ sequence → incre / dec ↓ any formula

loop body → { }

i++ / i--] Post incr- / decr...

++i / --i] Pre incr- / decr-

While loop

• ... میں صرف [illegible]

Question # Switch case :

```
int a = 5;
switch (a) {
case 1:
cout << "This is case 1" << endl;
break;
Case 2:
 cout << "This is 2" << endl;
 break;
case 5 :
 cout << "You win" << endl;
 break;
default :
  cout << "This is default" << endl;
  break;
```

switch( ) int n num if, else char 'a'

if <- cases int num; num = 5

cin >> num;

switch(num){ 'A' char

case 1: case 1:

cout << " Not true"; Dev C++

break;

case 5:

cout << "true";

break;

default:

cout << "Invalid";

break; }

int a; cout << "Enter number" << endl;

cin >> a;

switch (a) { → something to be checked.

case 1:

cout << "Monday";

break;

case 2:

cout << " Tuesday";

break

case 3:

cout << "Wednesday";

break;

.

.

.

}

default:

cout << " Invalid";

low level language

high level language

0, 1

Variables → Containers to store us data

- int (-1, 0, 1, ···)
- float (-0.11, 1.22, ···)
- double (1.2312...)
- character / char (c,

Comments

These can be stored in it

→ use to write addition information

→ computer ignore کرتا ہے means understand کرنے کی کوشش نہیں کرتا۔ یہ صرف information دینے کے لیے ہوتا ہے۔

/* → کوئی بھی comments

*/

C++ main body:-

```
#include <iostream>
using namespace std;
int main ( ) {

return 0;
}
```

Simplest :-

```
#include <iostream>
using namespace std;
int main ( ){
cout << " Hello world! " << en

return 0;
}
```

we have to add comment in our code :-

```
#include<iostream>
using namespace std;
int main ( ) {
// int a = 14;
```

```
int a = 14, b = 15;
cout << " This is tutorial 4" <<
```

[1. Ques]: To print no.s from 0 to 3

```
#include <iostream>
using namespace std;
int main(){
/* Loops in C++:
```

multiline comment

These are 3 types of loops in C++:

1. For loop 2. while loop 3. do-while loop

*/

Second comment

```
/* For Loop in C++ */
// int i = 1;
// cout <<i;
// i++;
for (int i=0; i<4; i++)
{
/* code */
cout <<i<<endl;
}
return 0;
}
```

Syntax of for loop

initialization, condition, increment/decrement/formula

These brackets { } include loop body (C++ code) means cout.

How to use for loop:-

```
** main body
** for (int ; condition ; updation)
** {
      cout <<i<<endl;
   }
** return 0;
   }
```

for loop consists of three things which should be written in ( ). All things are initialization; condition; and updation

[2. Ques]: To print even numbers from 0 to 40.

increment میں ( ) bracket والی for صرف
increment گے تو ایک بار ہوگا لیکن اگر ہم نے دو بار
ے تو ہمیں cout کے

```
#include <iostream>
using namespace std;
int main(){
for(int i=0; i<=40
{
cout<<i<<endl;
i++;
}
return 0;
}
```

[3. Ques]: Print no.s 10, 13 and upto so on it reaches 25.

( ) for کرنا ہو تو increment بار
ہو - اگر دو بار کرنا ہو تو اسکے ساتھ
ے نیچے بھی i++ لکھو اور اگر اس
( ) کے for
فع کرنا ہو تو پھر
کر جتنا increment کرنا ہے اسکے آگے
(-: ہو۔ جیسے i increment کا '3'

at next page :-

Syntax :-

```
do
{
   C++statement;
} while (condition);
```

Ques :- What is d/w While & do-while loop?

Ans :- In while loop, ↑ statement is executed if and only if the condition is true but in do-while loop, if the condition is true or if condition is false, statement executes one time bcz condition is not checked in 1st step, it is checked after the first step.

Proper ans :- (The while-loop checks the condition first and then executes the statement. The do-while loop will execute the statement(s) atleast once and then check the condition)

Ques :- Print 1-40 using do-while loop :-

```
# include <iostream>
using namespace std;
int main ( ){
  int i = 1;
  do{
     cout << i << endl;
       i++;
  } while (i <= 40);
    return 0;
  }
```

Example :-

```
# include <iostream>
using namespace std;
 int main ( ){
    int i = 1;
while (false)
{
   cout << i << endl;
}
    return 0;
 }
```

while-loop

Program runs, but ans was not 1, it was blank as this:- ---------

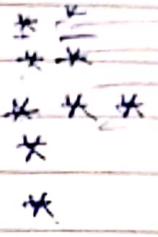
do-while loop.

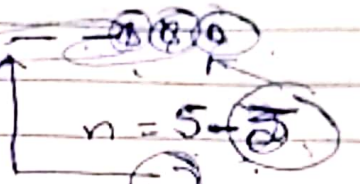
```
# include<iostream
using namespace std
int main ( ){
   int i = 1;
do{
   cout<<i<<endl;
     i++;
} while(false);
  return 0;
}
```

Program run, statement executed and print

code 1

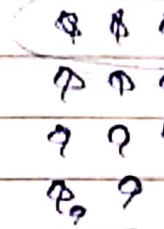
```
for(int i=1; i<=5; i++)
{ for(int j=1; j<=i; j++)
  { cout<<"*"; }
  cout<<endl;
}
```



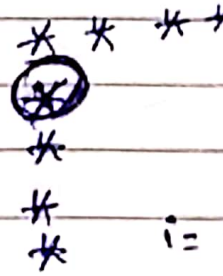


outer

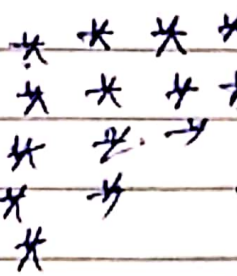
```
int i=1; i<=5; i++)
{ int j=1; j<=i; j++)
```



```
int i=5; i>=1; i--)
int j=1; j<=i; j++)
```



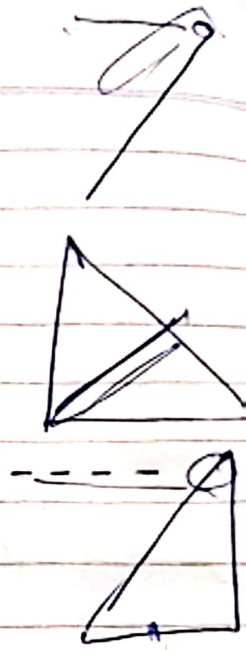

```
int n=3;
for(i=n; i>=n; i++)
{ for(j=1; j<=i; j++)
  { cout<<" "; }
for(k=n; k>=i; k--)
  { cout<<"*"; }
cout<<endl; }
```



abs(

i>=1 int n=5;

```
for( int i=1; i<=n; i++)
{
  for( int j=n-1; j>=i; j--)
  {
    cout << " ";
  }
  for( int k=1; k<=i; k++)
  {
    cout << "*"; }
  cout << endl;
}
```



```
int n=5
for( int i=1; i<=n; i++)
{
  for (int k=n-1; k>=i; k--)
  {
    cout << " "; }
  for( int j=1; j<=i; j++)
  {
    cout << "*";
  }
  cout << endl; }
```

```
                                  i<=5
        int   n=5                 i<=5
for ( int   i=1 ;   i<=n ;  i++)
  {                          4     4>=1
   for ( int k =  n-1;  k>=i ;  k--)
     {      cout << "  ";  }
   for  (int  j=1 ; j<=i ; j++)
     {    cout << "*" ;  }
          cout<< endl; }
for ( int  i= n-1 ;  i>=1 ;  i--)
{
 for (int  k= n-1 ; k>=i ;  k--)
   {    cout<<"  ";  }
for (int j =i ; j>=1 ; j--)
  {   cout <<"*" ;  }
        cout << end l;
                    }
   return 0;
```