ALGORITHMS AND FLOWCHARTS

Revision with Some Examples

Program Development Tools (Algorithm and flow charts)

Algorithm

An algorithm is a finite set of instruction written in a sequence that should be followed to solve the given problem. It must satisfy the following criteria:

- **Input**:
  One or more quantities are supplied for processing which is known as input.
- **Output**:
  After inputs are processed, it must produce at least one quantity as output.
- **Unambiguous**:
  Each instruction in the algorithm must be clear and it shouldn't have double meaning.
- **Finiteness**:
  The algorithm should terminate after the finite number of steps.

Flowchart

- A picture (graphical representation) of the problem solving process.
- Provides a step-by-step procedure for solution of a problem.

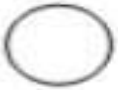
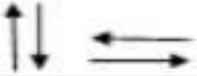
Elements of a flowchart:

- Various geometrical shaped boxes represent the steps of the solution.
- The boxes are connected by directional arrows to show the flow of the solution.

Uses of a flowchart:

- To specify the method of solving a problem.
- To plan the sequence of a computer program.

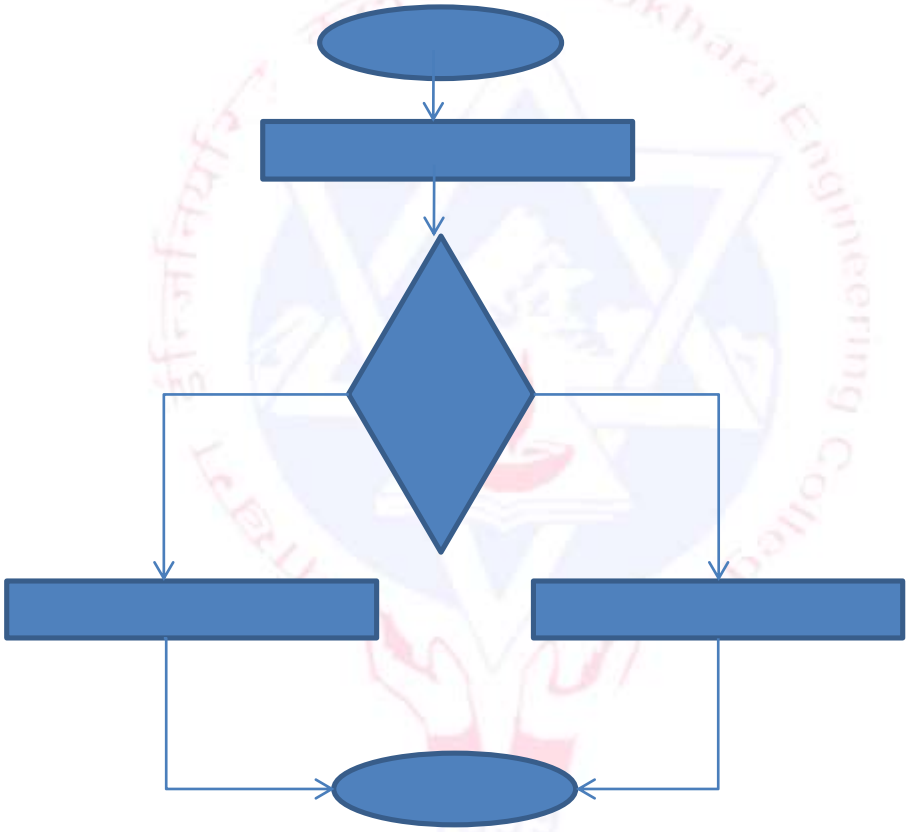
Symbols used in drawing flowchart

Symbol	Name	Function
	**Process**	Indicates any type of internal operation inside the Processor or Memory
	input/output	Used for any Input / Output (I/O) operation. Indicates that the computer is to obtain data or output results
	Decision	Used to ask a question that can be answered in a binary format (Yes/No, True/False)
	Connector	Allows the flowchart to be drawn without intersecting lines or without a reverse flow.
	Predefined Process	Used to invoke a subroutine or an Interrupt program.
	Terminal	Indicates the starting or ending of the program, process, or interrupt program
	Flow Lines	Shows direction of flow.

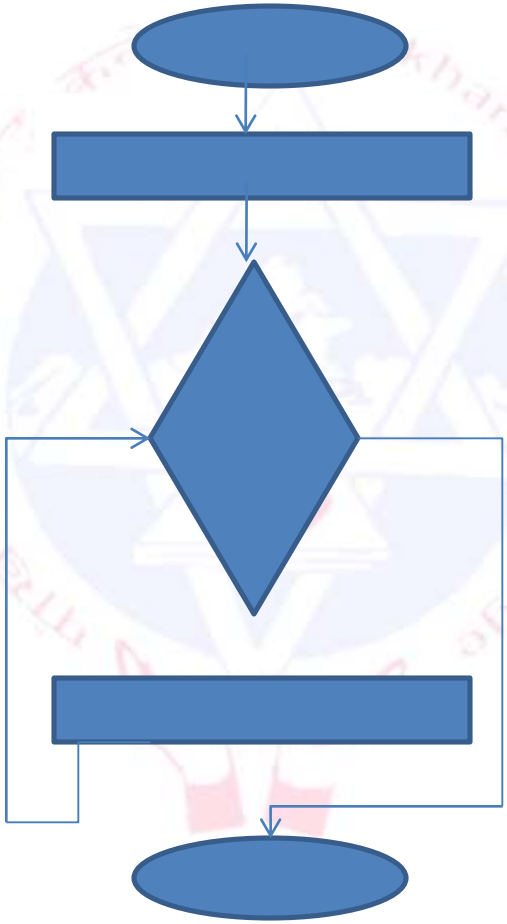
Flowchart structure

1. Sequential composition

2. Conditional structure



3. loop structure



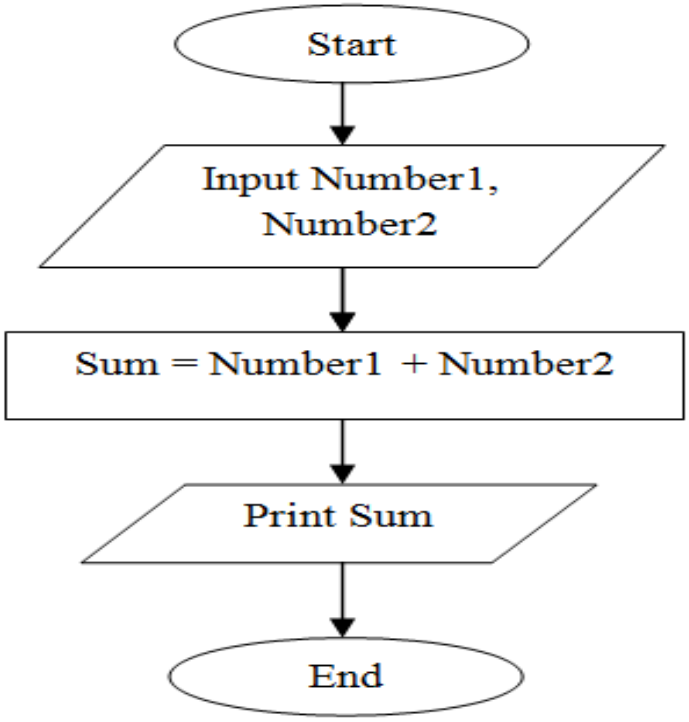
Some Examples of Algorithm & Flowchart

1. Sum of two numbers

Algorithm

- Step 1: Start
- Step 2: Declare variables number1, number2 and sum.
- Step 3: Read values for number1, number2.
- Step 4: Add number1 and number2 and assign the result to a variable sum.
- Step 5: Display sum
- Step 6: Stop

Flowchart



Largest among two numbers

Algorithm

Step 1: Start

Step 2: Read a, b

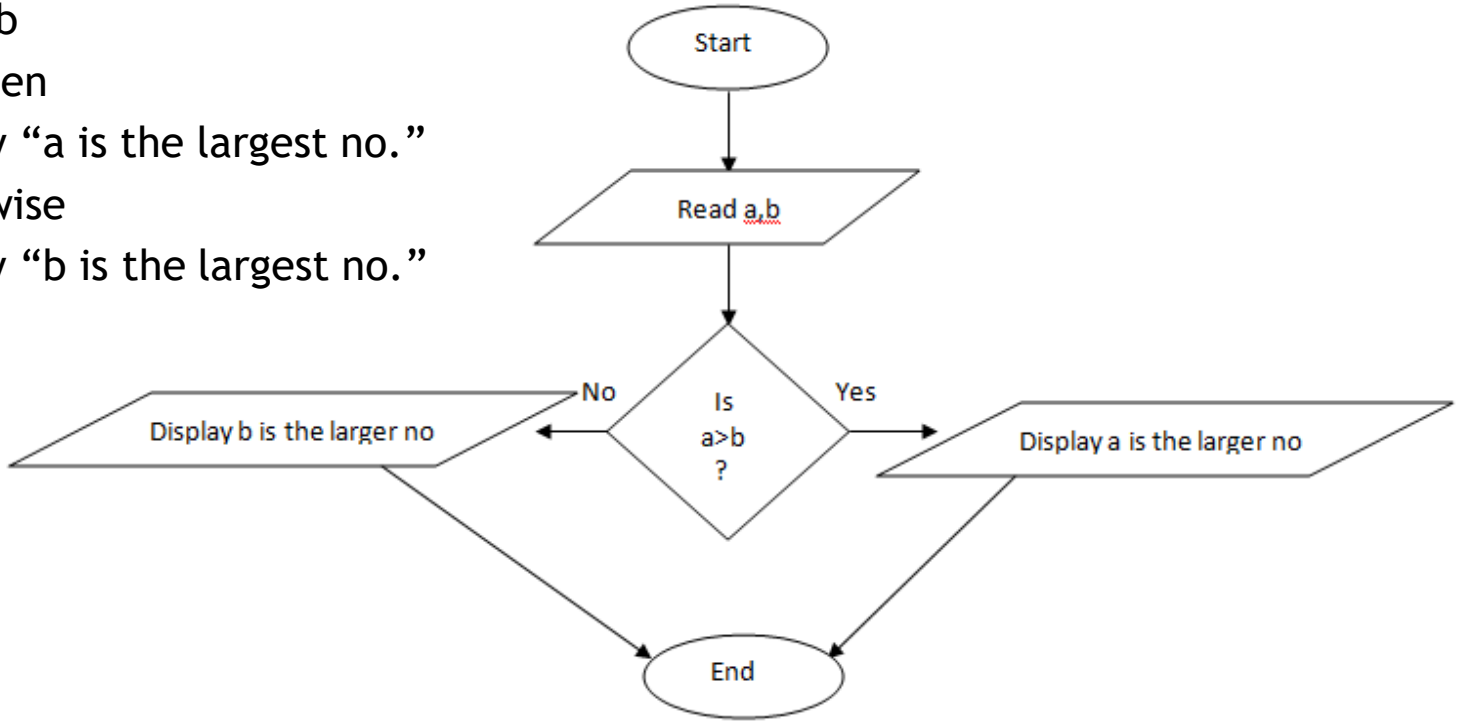
Step 3: If a>b then

Display “a is the largest no.”

Otherwise

Display “b is the largest no.”

Step 4: Stop

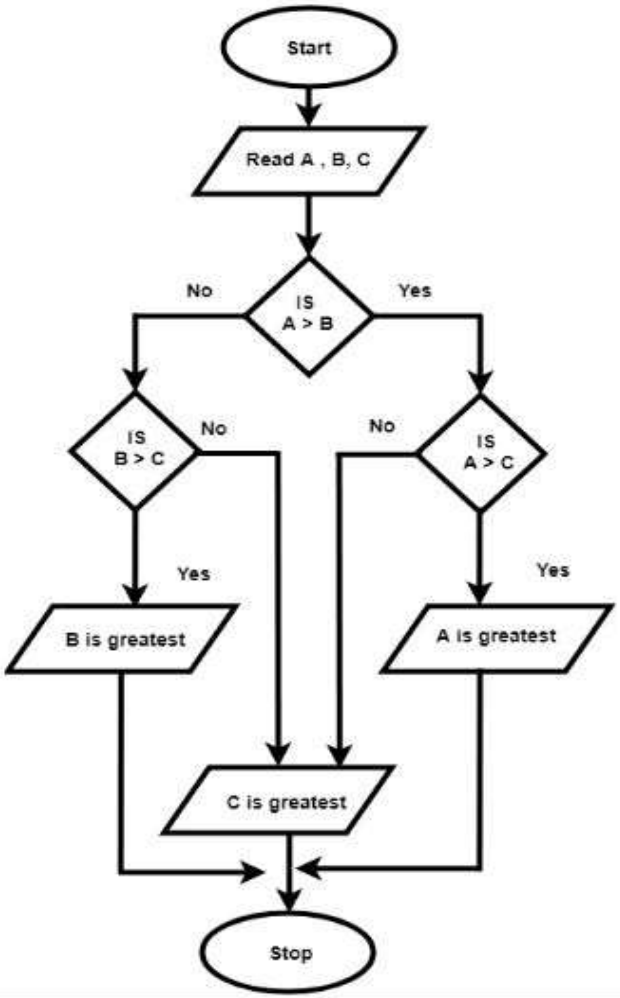


Check the greatest among 3 numbers

Algorithm

STEP 1: START
STEP 2: Read three numbers and store them in A , B, C
STEP 3: Is A > B
        If Yes: Go to Step 6,
        If No: Go to Step 4
STEP 4: Is B > C
        If Yes: Print B is greatest
        If No: Go to Step 5
STEP 5: Print C is greatest and Go to step 8
STEP 6: Is A > C
        If Yes: Print A is greatest
        If No: Go to Step 7
STEP 7: Print C is greatest and Go to Step 8
STEP 8: Stop

Flowchart


Start
Read A , B, C
IS A > B
No
Yes
IS B > C
No
Yes
B is greatest
IS A > C
No
Yes
A is greatest
C is greatest
Stop

4. Check whether a given number is negative or positive

Algorithm

Step 1 → Start

Step 2 → Declare integer variable Num

Step 3 → Assign value to the Num

Step 4 → Check if Num is greater than 0

If true print *Num is **Positive***

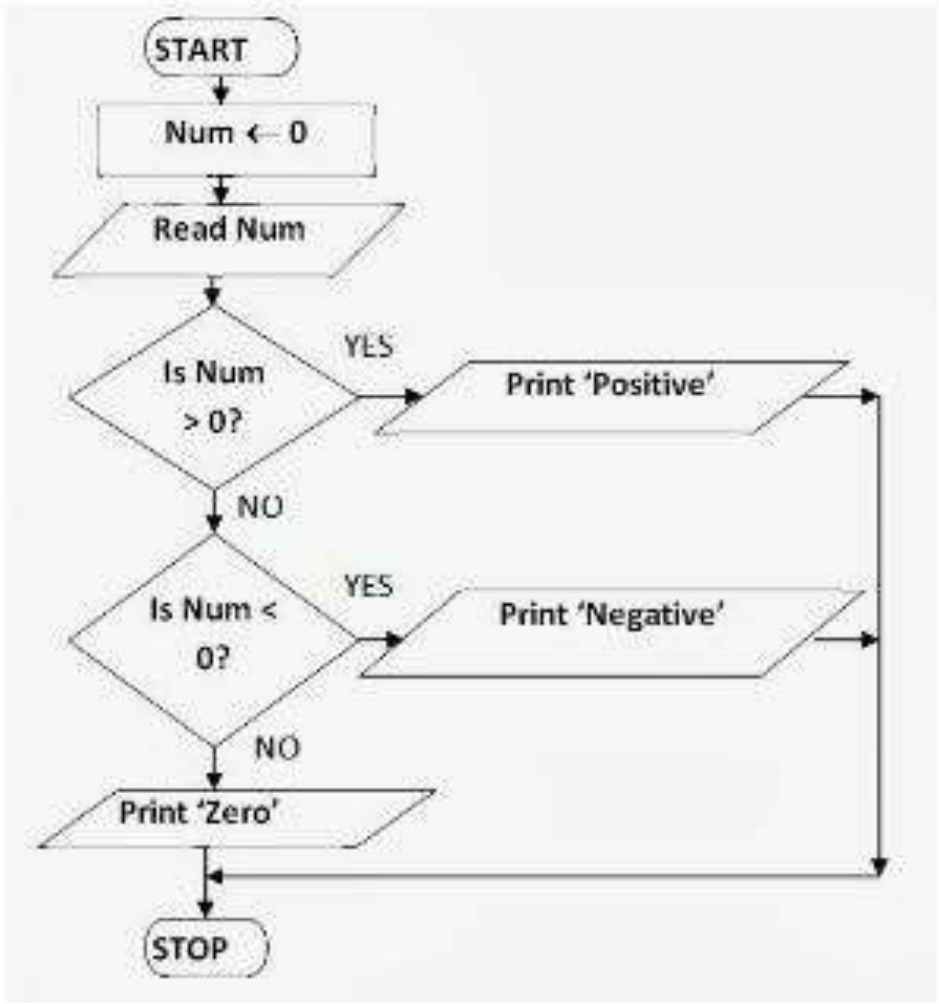
If false go to step 5

Step 5 → Check if Num is less than 0

If true print *Num is **Negative***

If false print *Num is 0*

Step 6 → Stop


START
Num ← 0
Read Num
Is Num > 0?
YES
Print 'Positive'
NO
Is Num < 0?
YES
Print 'Negative'
NO
Print 'Zero'
STOP

5. Check the given number is odd or even

Algorithm

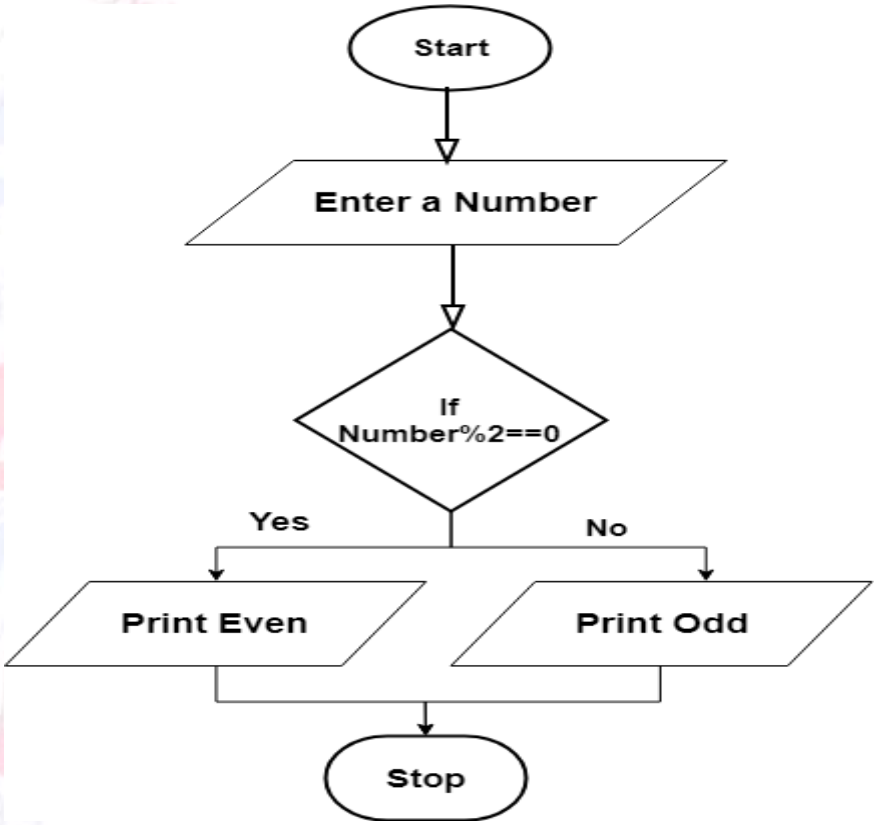
Step 1 : Start

Step 2 : Enter a number

Step 3 : If (number%2 == 0),

print " **even** " else print " **odd** "

Step 4 : Stop



6. Display the number from 1 to 10

Algorithm

STEP 1: Start.

STEP 2: Initialize the variable count to one.(Declare variable count = 1)

STEP 3: Display (Print) the variable count .

STEP 4: Increase variable count by one. (count=count+1)

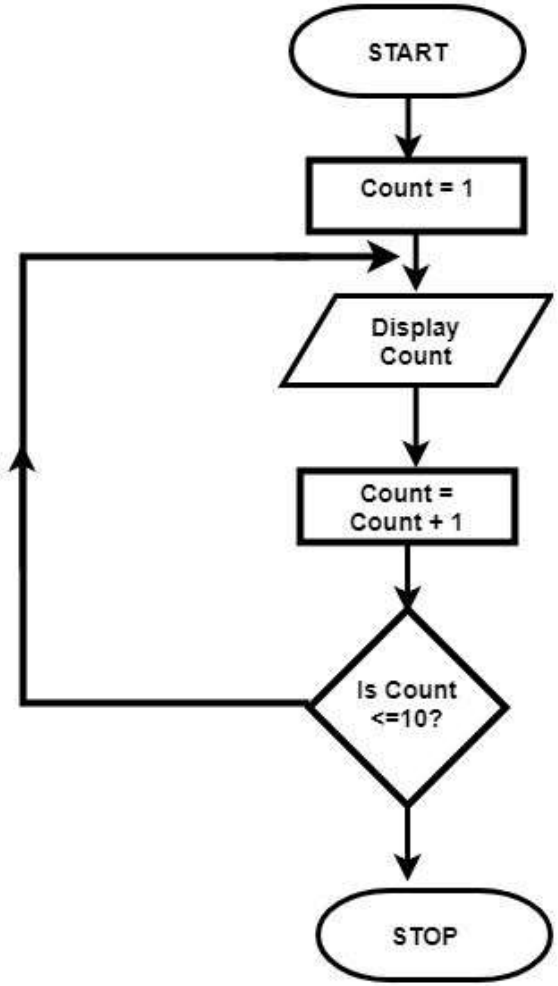
STEP 5: Check whether count variable exceeds 10.(count>10)

If Yes, Go to Step 6

If No Go to step 3

STEP 6: Stop

Flowchart



7. Area of Circle

Algorithm

STEP 1: Start

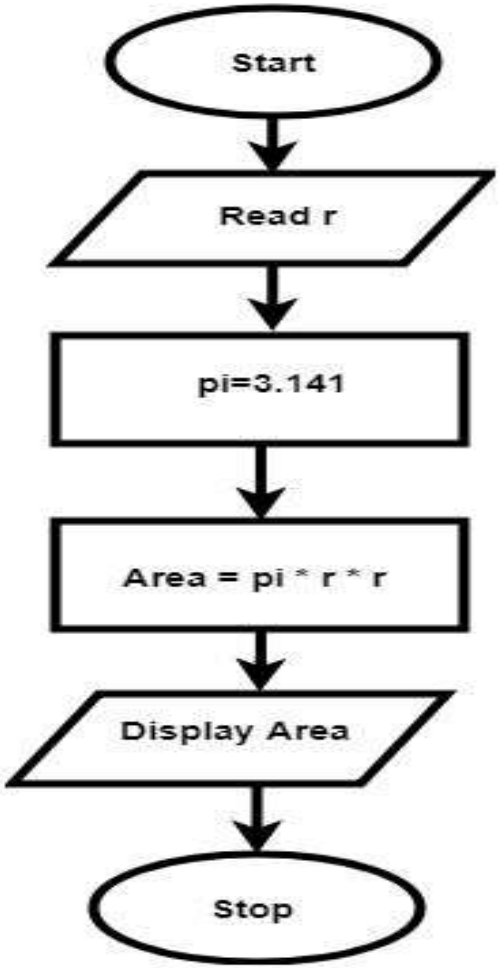
STEP 2: Read Radius ‘r’.

STEP 3: Store 3.14 in the value of pi.

STEP 4: Find area of circle (Area = pi * r * r)

STEP 5: Display Area.

STEP 6: Stop



Simple Interest

Algorithm

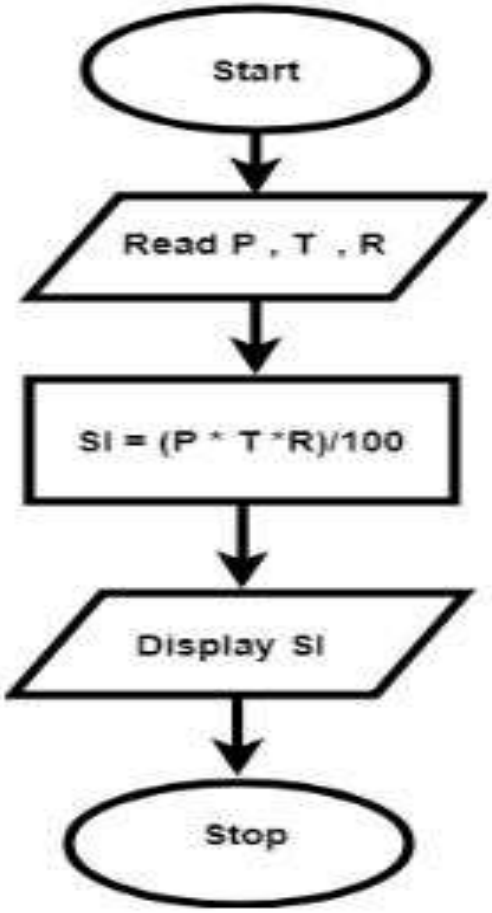
STEP 1: Start

STEP 2 : Read Principal, Time and Rate
(P , T, R)

STEP 3: Calculate the Simple Interest:
(SI = P * T * R / 100)

STEP 4 : Display SI

STEP 5 : Stop



Sum of first 50 natural numbers

Algorithm

STEP 1: Start.

STEP 2: Initialize the variable n to 0 and sum to 0.(n=0, sum=0)

STEP 3: Increase variable n by one.(n=n+1)

STEP 4: Add n to sum and store it on sum (sum=sum+n)

STEP 5: Check whether n variable exceeds 50.(n>50)

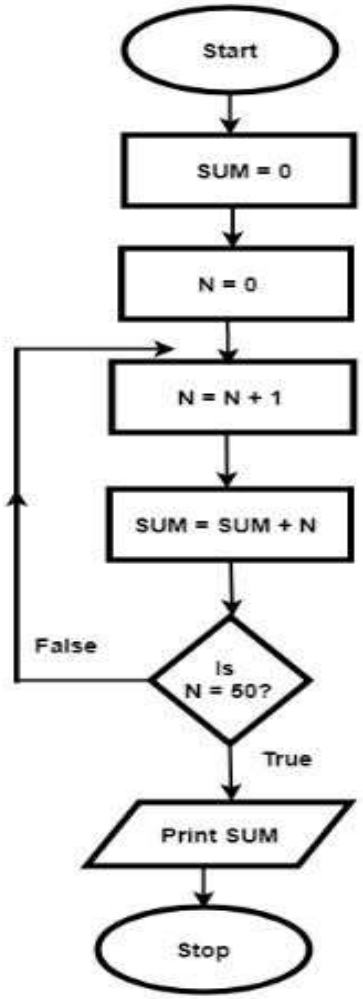
If Yes, Go to Step 6,

If No, Go to step 3

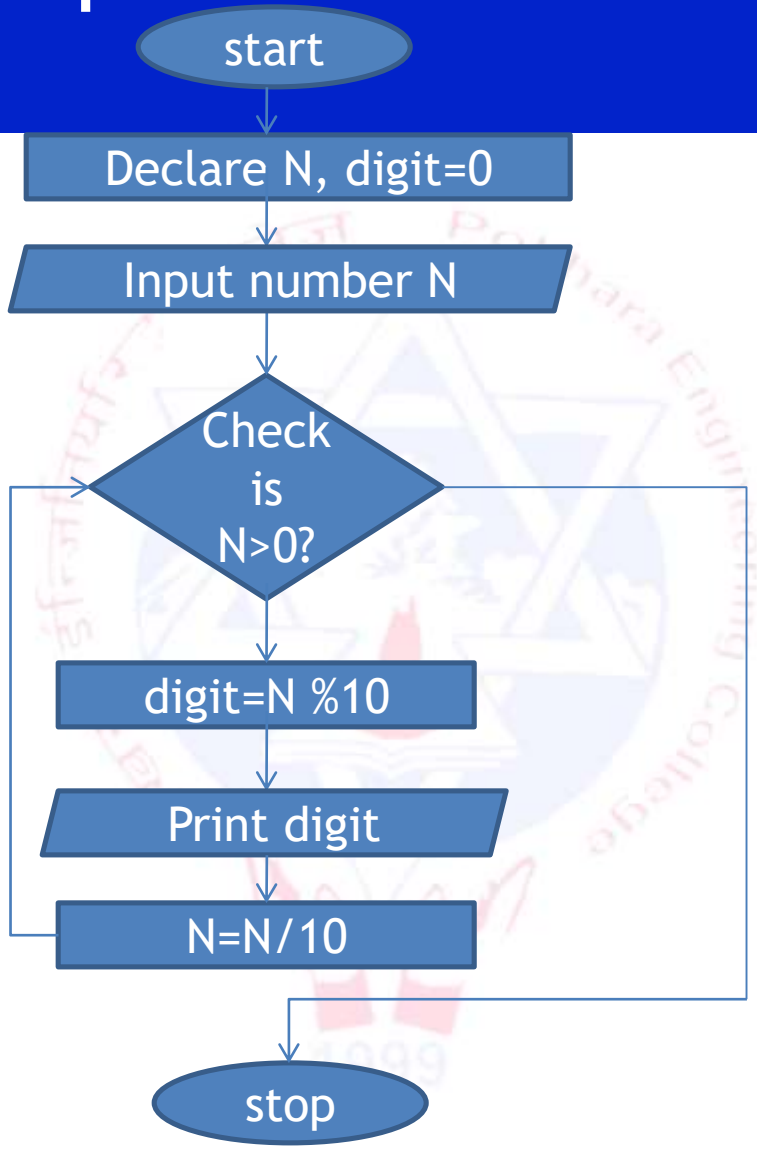
STEP 6: Print Sum

STEP 7: Stop

Flowchart

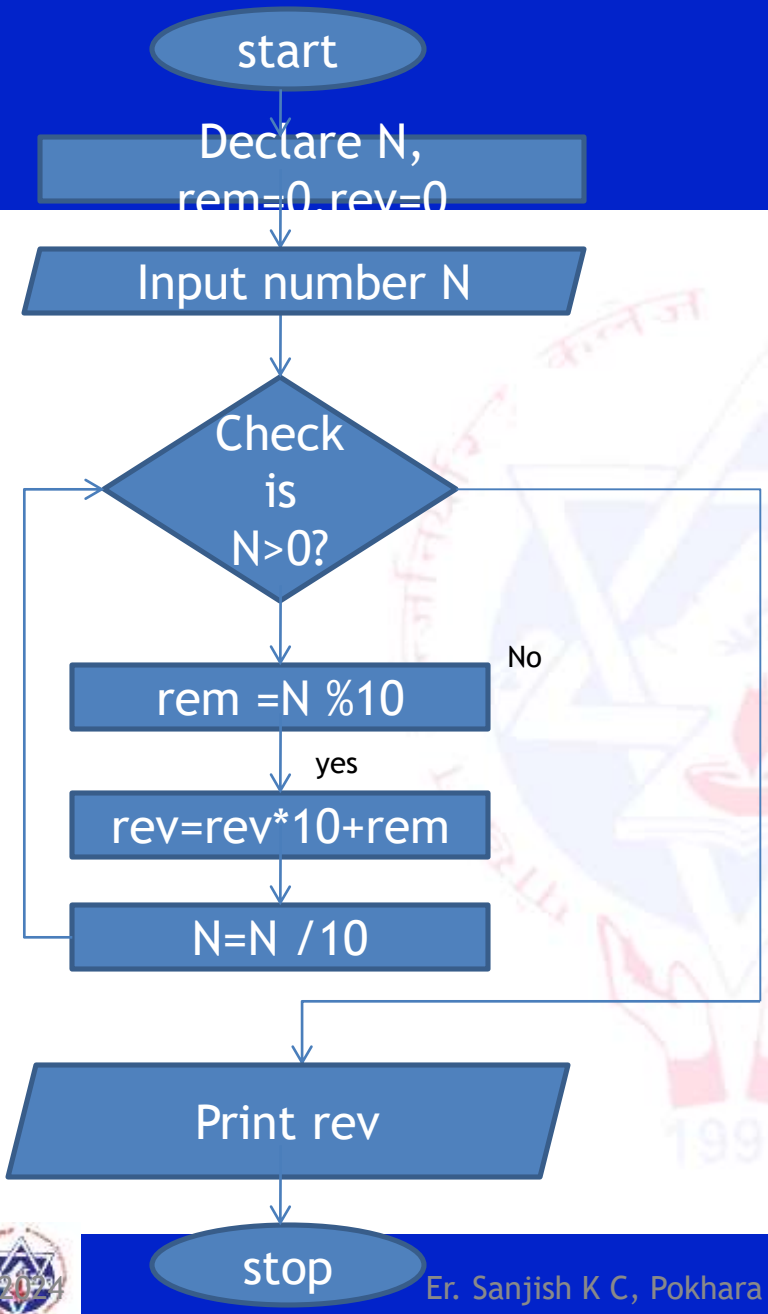


Flowchart to print the individual digit of a given number



10. Reverse of a given number

- **Step 1:** Start
- **Step 2:** Read number *N*
- **Step 3:** Set rev=0
- **Step 4:** Check is N>0?
  If yes, go to step 5
  if no, print rev and stop
- **Step 5:** Set rem =N %10
- **Step 6:** Set rev=rev*10+rem
- **Step 7:** Set N =N/10
- **Step 8:** Print rev
- **Step 9:** Stop



rem=0, rev=0
N=2507

Is 2507>0? , yes
rem= 2507%10= 7
rev= 0* 10+7=7
N=2507/10=250

1st Iteration

Is 250>0? , yes
rem= 250%10= 0
rev= 7* 10+0=70
N=250/10=25

2nd Iteration

Is 25>0? , yes
rem= 25%10= 5
rev= 70* 10+5=705
Iteration
N=25/10=2

3rd

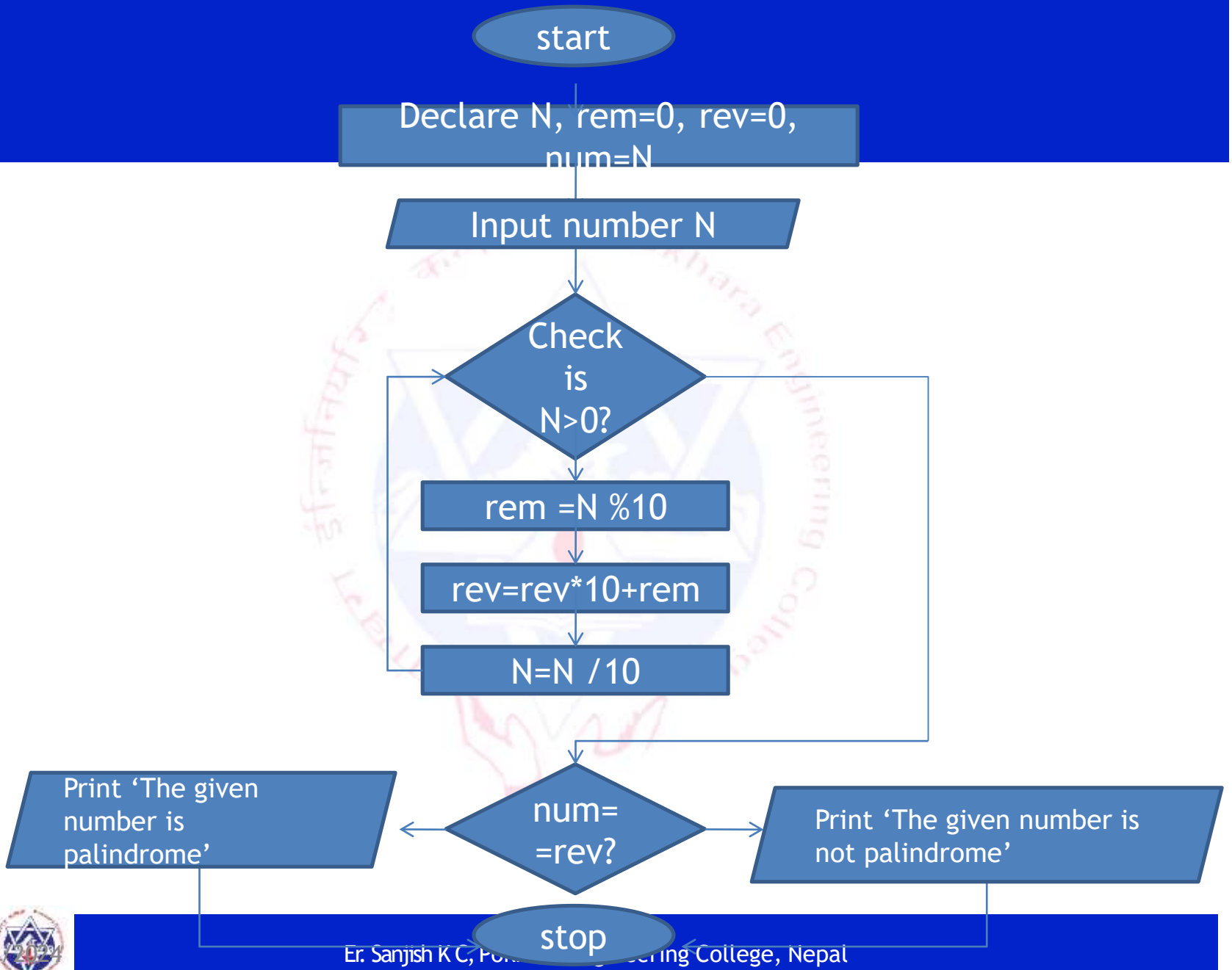
Is 2>0? , yes
rem= 2%10= 2
rev= 705* 10+2=7052
N=2/10=0

4rth Iteration

Is 0>0? , no
rev= 7052 will be printed which is reverse of 2507.

To check whether a given number is palindrome or not.

- **Step 1:** Start
- **Step 2:** Read number *N*
- **Step 3:** Set rev=0 ,rem= 0, num=N
- **Step 4:** Check is N>0?
  If yes, go to step 5
  if no, go to step 8
- **Step 5:** Set rem =N %10
- **Step 6:** Set rev=rev*10+rem
- **Step 7:** Set N =N/10 and go to step 4
- **Step 8:** Check is num==rev?
  If yes, go to step 9
  if no, go to step 10
- **Step 9:** Print 'The given number is palindrome' and go to 11
- **Step 10:** Print 'The given number is not palindrome' go to 11
- **Step 11:** Stop


start
Declare N, rem=0, rev=0, num=N
Input number N
Check is N>0?
rem =N %10
rev=rev*10+rem
N=N /10
num= =rev?
Print 'The given number is palindrome'
Print 'The given number is not palindrome'
stop

To check whether a given number is Armstrong or not.

Armstrong number is a **number** that is equal to the sum of cubes of its digits.

Eg: (1=1^1) (1634=1^4+6^4+3^4+4^4) (371=3^3+7^3+1^3)

1: Start

2: Declare num, n, no, sum, c, rem

3: Input number num

4: Set n=num,no=num,c=0,sum=0

5: Is no>0

If yes, go to step 6

If no, go to step 8

6: no=no/10

7: c=c+1 and go to step 5

8: Is n>0

If yes, go to step 9

If no, go to step 12

9: rem=n%10

10: sum=sum + rem^c

11: n=n/10 and go to step 8

12: Is sum==num?

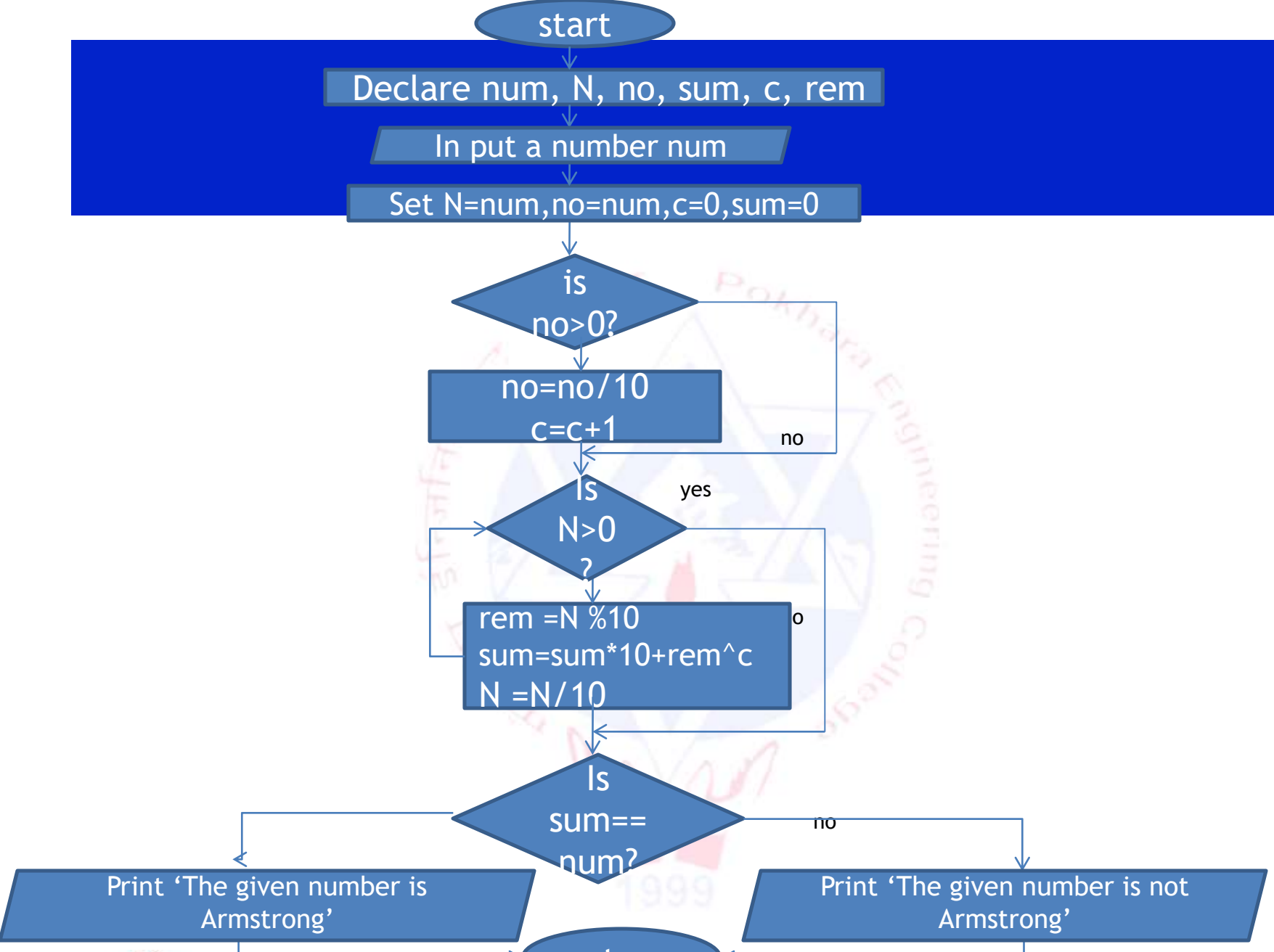
If yes, go to step 13

If no, go to step 14

13: Print 'The given number is Armstrong' and go to 15

14: Print 'The given number is not Armstrong' and go to 15

15:stop


start
Declare num, N, no, sum, c, rem
In put a number num
Set N=num,no=num,c=0,sum=0
is no>0?
no=no/10
c=c+1
no
Is N>0 ?
yes
rem =N %10
sum=sum*10+rem^c
N =N/10
o
Is sum== num?
no
Print 'The given number is Armstrong'
Print 'The given number is not Armstrong'
stop

Quadratic Equation

The standard form of a quadratic equation is:

```
ax² + bx + c = 0, where
a, b and c are real numbers and
a != 0
```

The term `b²: - 4ac` is known as the **discriminant** of a quadratic equation. It tells the nature of the roots.

- If the discriminant is greater than `0`, the roots are real and different.
- If the discriminant is equal to `0`, the roots are real and equal.
- If the discriminant is less than `0`, the roots are complex and different.

Roots of Quadratic Equation

If the discriminant > 0,

$$root1 = \frac{-b + \sqrt{(b^2 - 4ac)}}{2a}$$

$$root2 = \frac{-b - \sqrt{(b^2 - 4ac)}}{2a}$$

If the discriminant = 0,

$$root1 = root2 = \frac{-b}{2a}$$

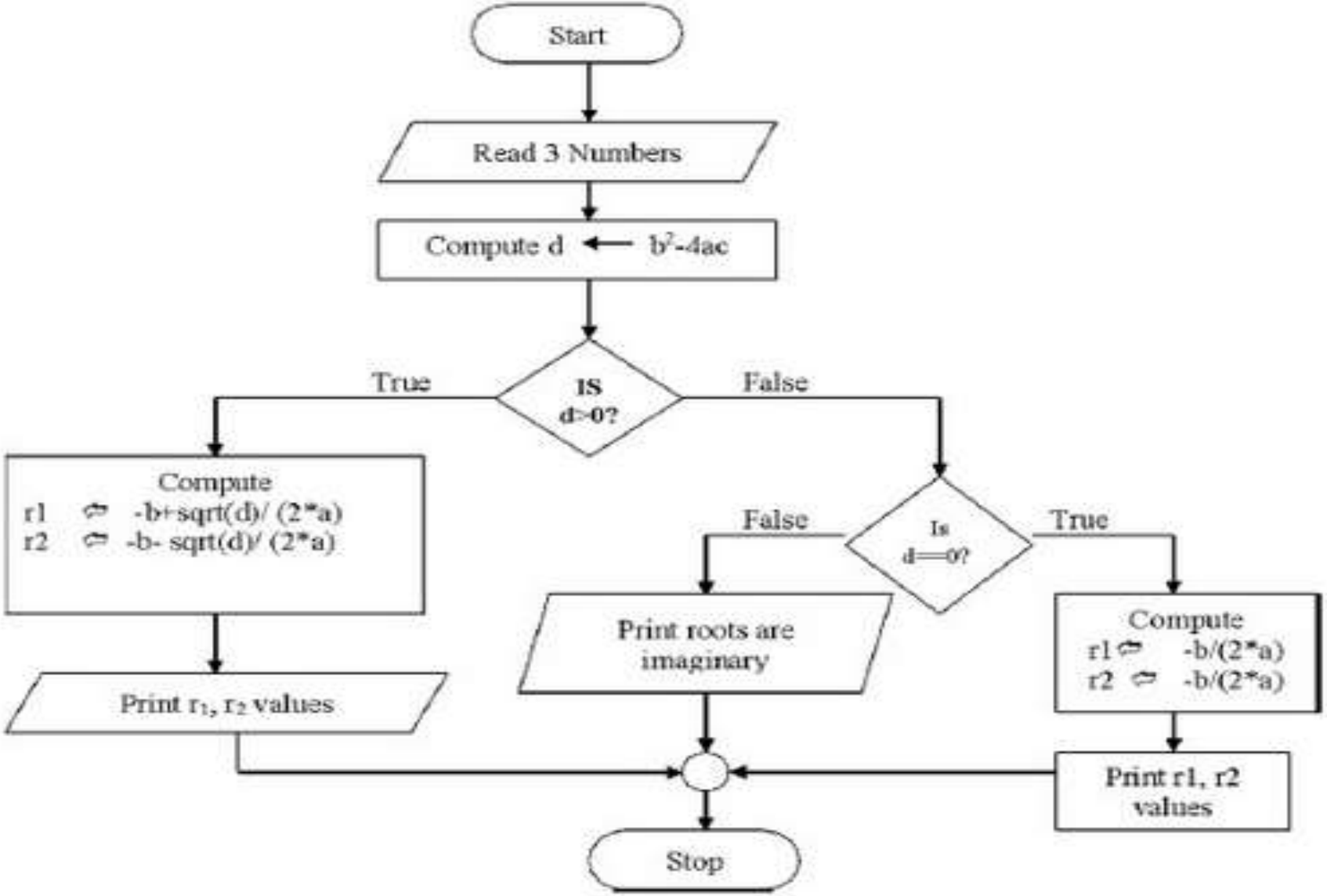
If the discriminant < 0,

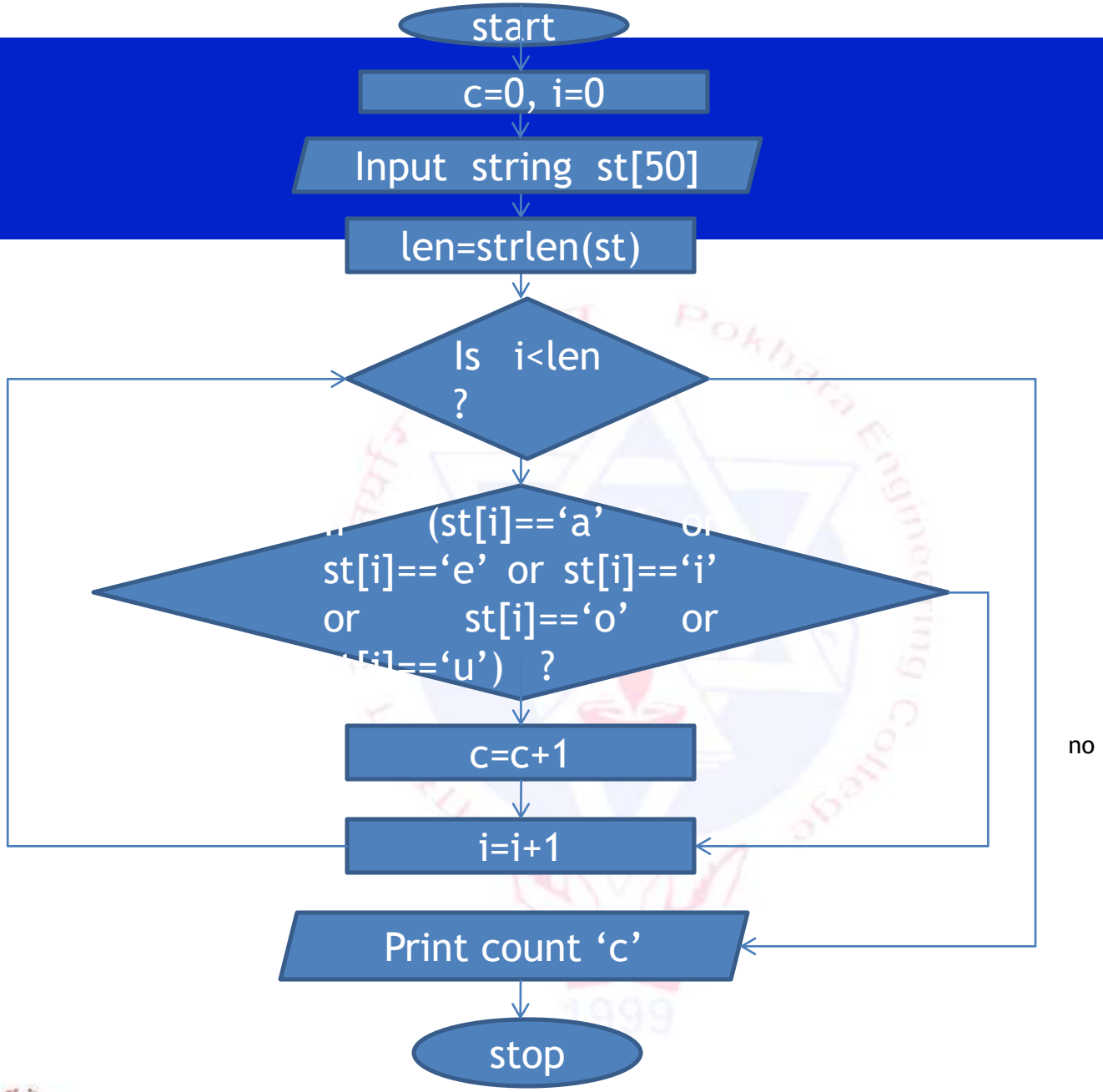
$$root1 = \frac{-b}{2a} + \frac{i\sqrt{-(b^2 - 4ac)}}{2a}$$

$$root2 = \frac{-b}{2a} - \frac{i\sqrt{-(b^2 - 4ac)}}{2a}$$

Figure: Roots of a Quadratic Equation

Flowchart


Start
Read 3 Numbers
Compute d ← b²-4ac
IS d>0?
True
False
Compute
r1 ⇦ -b+sqrt(d)/ (2*a)
r2 ⇦ -b- sqrt(d)/ (2*a)
Print r1, r2 values
Is d==0?
False
True
Print roots are imaginary
Compute
r1 ⇦ -b/(2*a)
r2 ⇦ -b/(2*a)
Print r1, r2 values
Stop


start
c=0, i=0
Input string st[50]
len=strlen(st)
Is i<len ?
(st[i]=='a' or st[i]=='e' or st[i]=='i' or st[i]=='o' or
u') ?
c=c+1
i=i+1
Print count 'c'
stop
no

THANK YOU