



SNS COLLEGE OF TECHNOLOGY

Coimbatore-35
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai



DEPARTMENT OF INFORMATION TECHNOLOGY

23ITT101-PROGRAMMING IN C AND DATA STRUCTURES

I YEAR - IISEM

UNIT 1 – Introduction to C

TOPIC 3 –Algorithm, pseudo code, flow chart, and programming language



NOTATIONS OF AN ALGORITHM



- Algorithm can be expressed in many different notations, including Natural Language, Pseudo code, flowcharts and programming languages.
- Natural language tends to be verbose and ambiguous.
- Pseudocode and flowcharts are represented through structured human language.
- A notation is a system of characters, expressions, graphics or symbols designs used among each others in problem solving to represent technical facts, created to facilitate the best result for a program
- In simple words Notations collectively represents the following:
 - Pseudo code
 - Flowcharts
 - Programming languages.



PSEUDOCODE



- Pseudocode is an informal high-level description of the operating principle of a computer program or algorithm.
- It uses the basic structure of a normal programming language, but is intended for human reading rather than machine reading.
- It is text based detail design tool.
- Pseudo means 'false' and code refers to 'instructions' written in programming language.
- Pseudocode cannot be compiled nor executed, and there are no real formatting or syntax rules.
- The pseudocode is written in normal English language which cannot be understood by the computer.

Example:

Pseudocode: To find sum of two numbers

```
READ num1,num2
```

```
sum=num1+num2
```

```
PRINT sum
```



BASIC RULES TO WRITE PSEUDOCODE



1. Only one statement per line.

Statements represents single action is written on same line.

For example to read the input, all the inputs must be read using single statement.

2. Capitalized initial keywords

The keywords should be written in capital letters.

Eg: READ, WRITE, IF, ELSE, ENDIF, WHILE, REPEAT

3. Indent to show hierarchy

Indentation is a process of showing the boundaries of the structure.

4. End multi-line structures

Each structure must be ended properly, which provides more clarity.

5. Keep statements language independent.

Pseudocode must never written or use any syntax of any programming language.

Example: 01

Pseudocode: Find the total and average of three subjects

```
READ name, mark1, mark2, mark3
```

```
Total=mark1+mark2+mark3
```

```
Average=Total/3
```

```
WRITE name, mark1, mark2, mark3
```

Example: 02

Pseudocode: Find greatest of two numbers

```
READ a, b
```

```
IF a>b then
```

```
    PRINT a is greater
```

```
ELSE
```

```
    PRINT b is greater
```

```
ENDIF
```



ADVANTAGES & DISADVANTAGES OF PSEUDOCODE



□ Advantages of Pseudocode

- Can be done easily on a word processor
- Easily modified
- Implements structured concepts well
- It can be written easily
- It can be read and understood easily
- Converting pseudocode to programming language is easy as compared with flowchart

□ Disadvantages of Pseudocode

- It is not visual
- There is no standardized style or format



COMMON KEYWORDS USED IN PSEUDOCODE







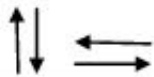


1. `//`: This keyword used to represent a comment.
2. `BEGIN,END`: Begin is the first statement and end is the last statement.
3. `INPUT, GET, READ`: The keyword is used to inputting data.
4. `COMPUTE, CALCULATE`: used for calculation of the result of the given expression.
5. `ADD, SUBTRACT, INITIALIZE`: used for addition, subtraction and initialization.
6. `OUTPUT, PRINT, DISPLAY`: It is used to display the output of the program.
7. `IF, ELSE, ENDIF`: used to make decision.
8. `WHILE, ENDWHILE`: used for iterative statements.
9. `FOR, ENDFOR`: Another iterative incremented/decremented tested automatically.













FLOWCHART

- A graphical representation of an algorithm.
- Flowcharts is a diagram made up of boxes, diamonds, and other shapes, connected by arrows.
- Each shape represents a step in process and arrows show the order in which they occur.

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 SYMBOLS

Name	Symbol	Description
Process		Process or action step
Flow line		Direction of process flow
Start/ terminator		Start or end point of process flow
Decision		Represents a decision making point
Connector		Inspection point
Inventory		Raw material storage
Inventory		Finished goods storage
Preparation		Initial setup and other preparation steps before start of process flow
Alternate process		Shows a flow which is an alternative to normal flow
Flow line(dashed)		Alternate flow direction of information flow

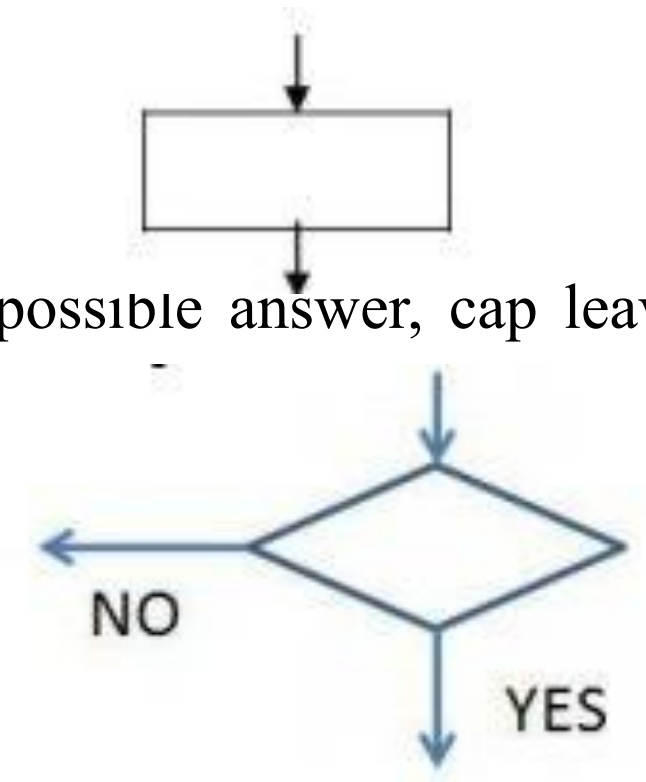


Rules for drawing flowchart

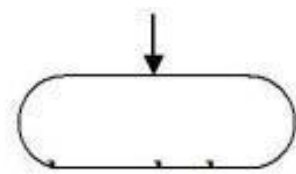


1. In drawing a proper flowchart, all necessary requirements should be listed out in logical order.
2. Flow chart should be clear, neat and easy to follow. There should not be any room for ambiguity in understanding the flowchart.
3. The usual directions of the flow of a procedure or system is from left to right or top to bottom.
Only one flow line should come out from a process symbol.

4. Only one flow line should enter a decision symbol, but two or three flow lines, one for each possible answer, can leave the decision symbol.



5. Only one flow line is used in conjunction with terminal symbol.



6. If flowchart becomes complex, it is better to use connector symbols to reduce the number of flow lines.
7. Ensure that flowchart has logical start and stop.



ADVANTAGES & DISADVANTAGES OF FLOWCHART



Advantages of Flowchart:

Communication:

Flowcharts are better way of communicating the logic of the system.

Effective Analysis

With the help of flowchart, a problem can be analyzed in more effective way.

Proper Documentation

Flowcharts are used for good program documentation, which is needed for various purposes.

Efficient Coding

The flowcharts act as a guide or blue print during the system analysis and program development phase.

Systematic Testing and Debugging

The flowchart helps in testing and debugging the program

Efficient Program Maintenance

The maintenance of operating program becomes easy with the help of flowchart.

It helps the programmer to put efforts more efficiently on that part.

Disadvantages of Flowchart

Complex Logic:

Sometimes, the program logic is quite complicated. In that case flowchart becomes complex and difficult to use.

Alteration and Modification:

If alterations are required the flowchart may require redrawing completely.

Reproduction: As the flowchart symbols cannot be typed, reproduction becomes problematic.



CONTROL STRUCTURES USING FLOWCHARTS AND PSEUDOCODE



□ Sequence Structure

- A sequence is a series of steps that take place one after another.
- Each step is represented here by a new line

<u>Pseudocode</u>	<u>Flow Chart</u>
General Structure	
Process 1 Process 2 ... Process 3	<pre>graph TD; Start(()) --> P1[Process 1]; P1 --> P2[Process 2]; P2 --> P3[Process 3];</pre>
Example	

READ a READ b Result <u>c=a+b</u> PRINT c	<pre>graph TD; Start([Start]) --> Input[/a=10, b=20/]; Input --> Process[c=a+b]; Process --> Output[/print c/]; Output --> Stop([Stop]);</pre>
--	--



CONDITIONAL STRUCTURE



Conditional Structure

- Conditional structure is used to check the condition.
- It will be having two outputs only (True or False)
- **IF** and **IF...ELSE** are the conditional structures used.

<u>Pseudocode</u>	<u>Flow Chart</u>
General Structure IF condition THEN Process 1 ENDIF	
Example READ a READ b IF a>b THEN PRINT a is greater	



CONDITIONAL STRUCTURE



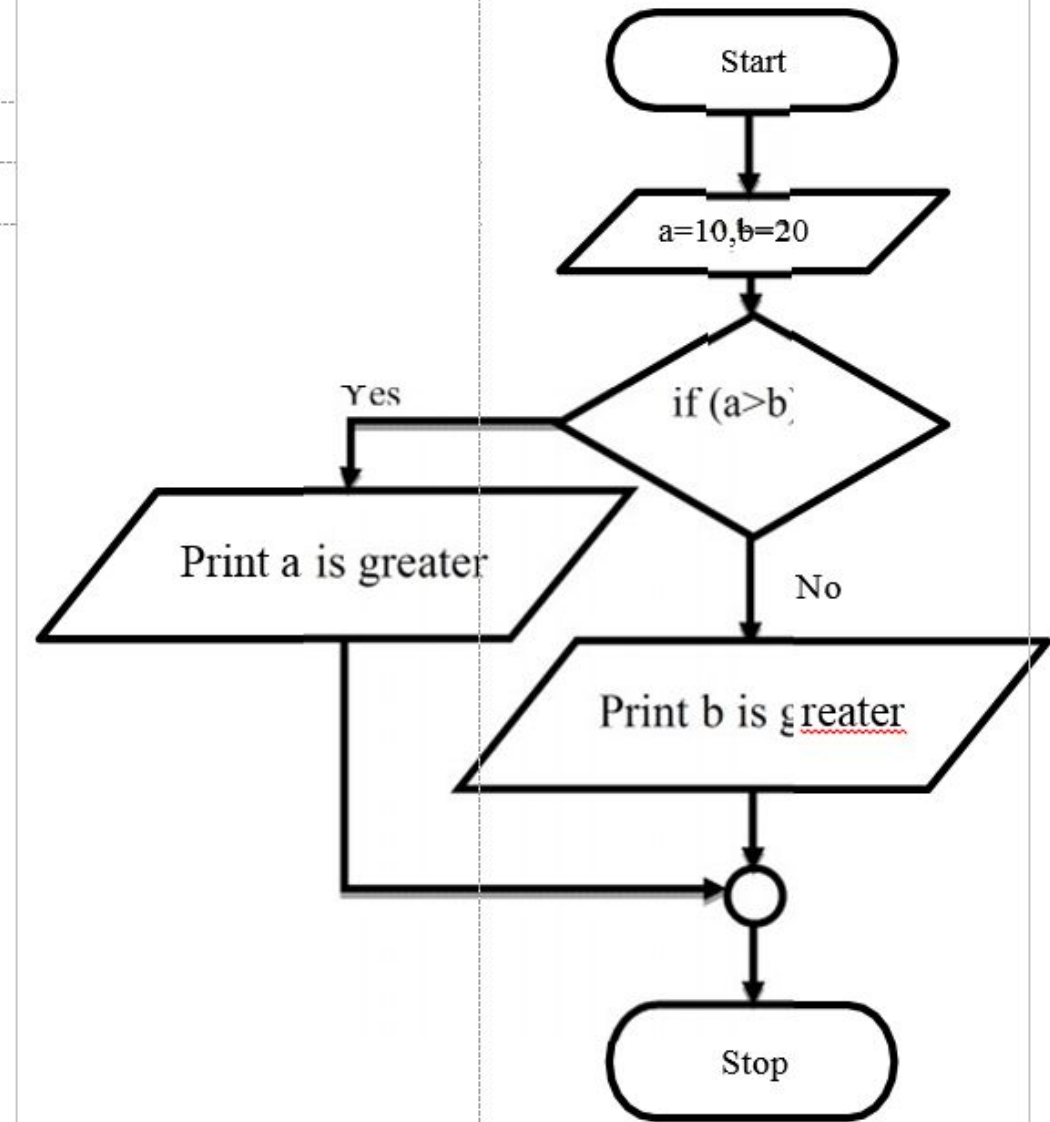
IF... ELSE

IF...THEN...ELSE is the structure used to specify, if the condition is true, then execute Process1, else, that is condition is false then execute Process2

```

READ a
READ b
IF a>b THEN
PRINT a is greater
    
```

Pseudocode	Flow Chart
General Structure	
IF condition THEN Process 1 ELSE Process 2 ENDIF	
Example	





ITERATION OR LOOPING STRUCTURE

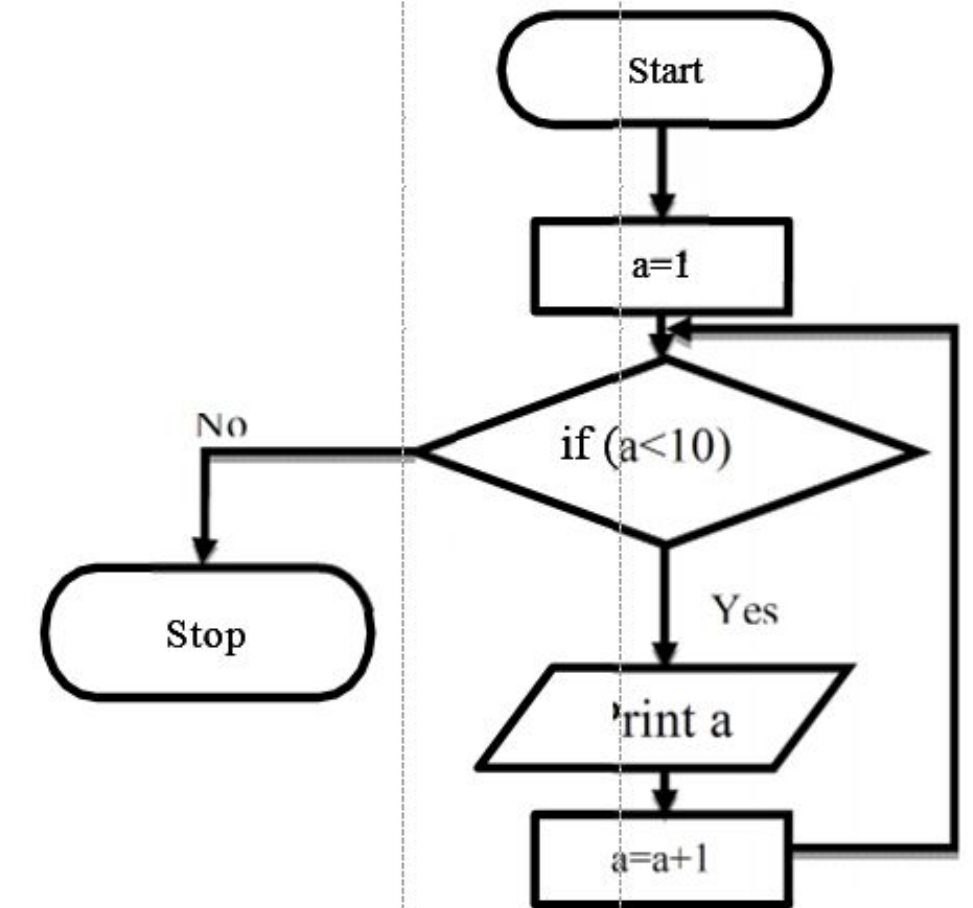


- Looping is generally used with WHILE or DO...WHILE or FOR loop.
- WHILE and FOR is **entry checked loop**.
- DO...WHILE is exit checked loop, so the loop will be executed at least once.

```

INITIALIZE a=1
WHILE a<10 THEN
    PRINT a
    a=a+1
ENDWHILE
    
```

Pseudocode	Flow Chart
General Structure	
WHILE condition Body of the loop ENDWHILE	





ALGORITHM vs. FLOWCHART vs. PSEUDOCODE



Algorithm	Flowchart	Pseudo code
An algorithm is a sequence of instructions used to solve a problem	It is a graphical representation of algorithm	It is a language representation of algorithm.
User needs knowledge to write algorithm.	not need knowledge of program to draw or understand flowchart	Not need knowledge of program language to understand or write a pseudo code.

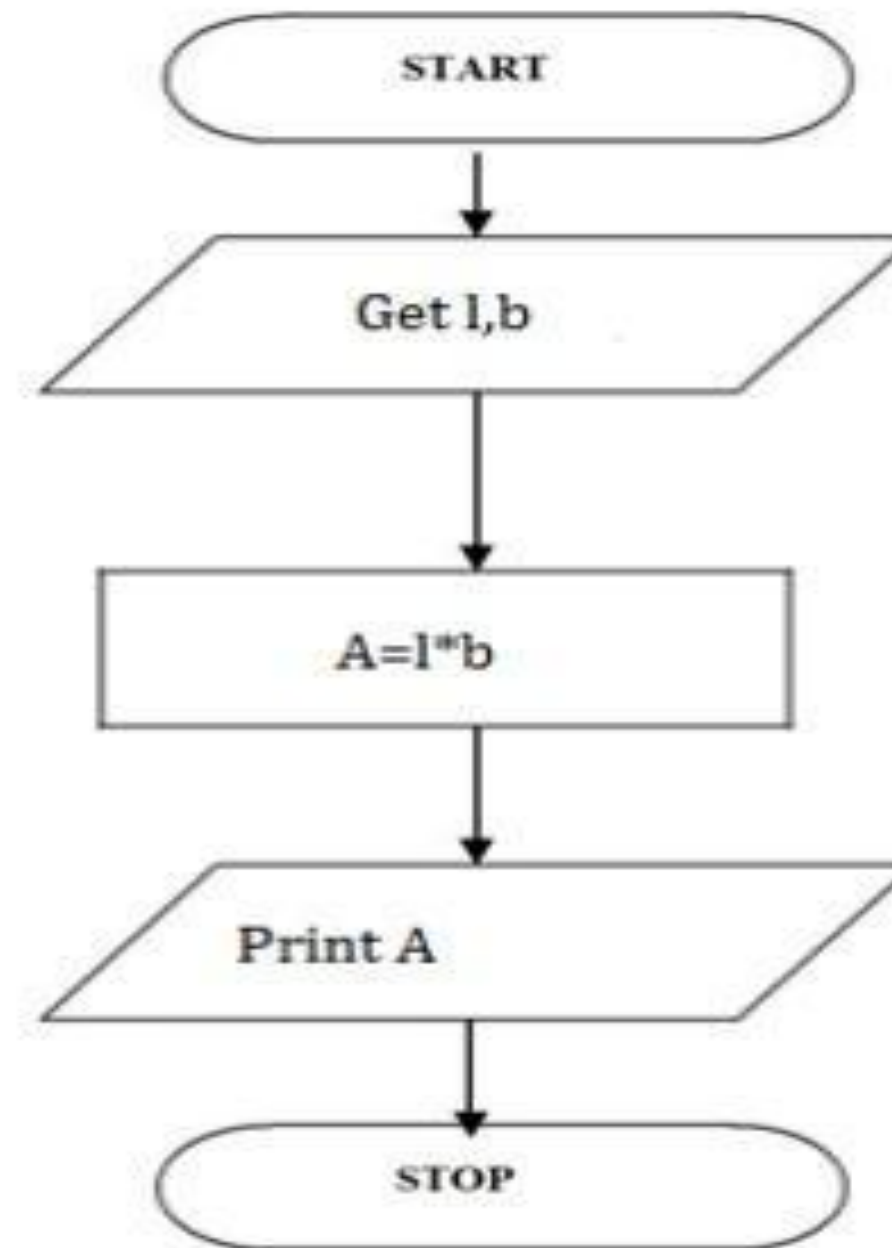


TO FIND AREA OF A RECTANGLE



- Step 1: Start
- Step 2: get l,b values
- Step 3: Calculate $A=l*b$
- Step 4: Display A
- Step 5: Stop

- BEGIN
- READ l,b
- CALCULATE $A=l*b$
- DISPLAY A
- END



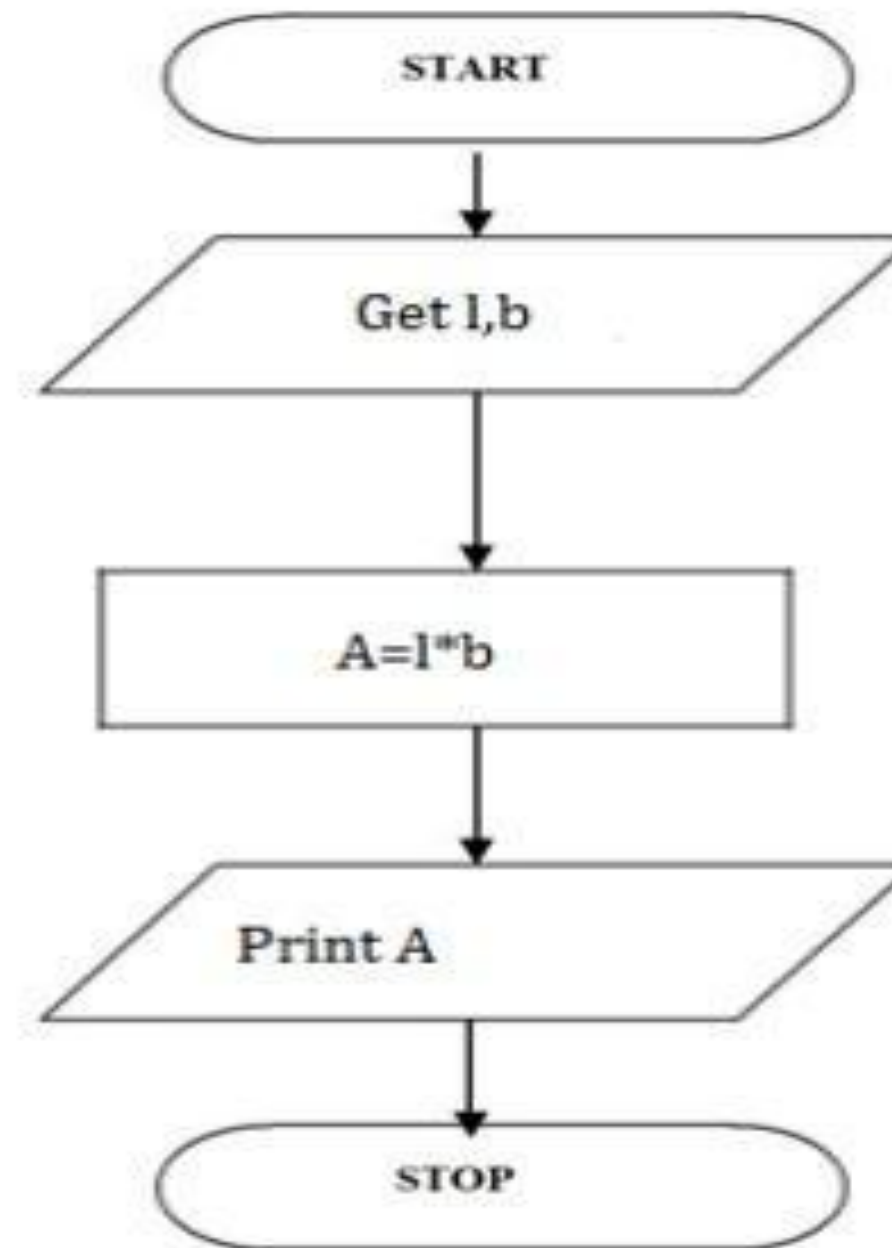


TO FIND AREA OF A RECTANGLE



- Step 1: Start
- Step 2: get l,b values
- Step 3: Calculate $A=l*b$
- Step 4: Display A
- Step 5: Stop

- BEGIN
- READ l,b
- CALCULATE $A=l*b$
- DISPLAY A
- END



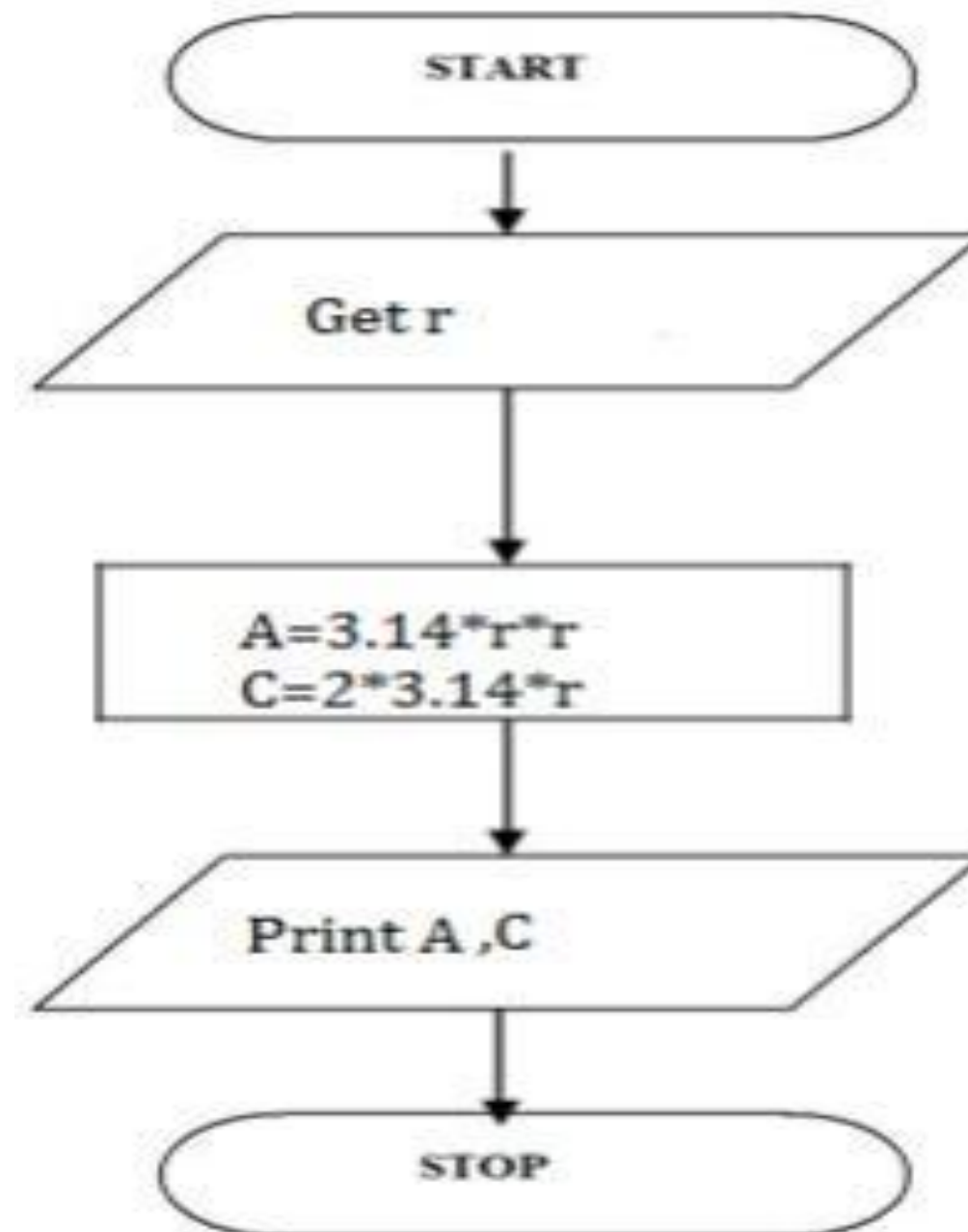


CALCULATING AREA AND CIRCUMFERENCE OF CIRCLE



- Step 1: Start
- Step 2: get r value
- Step 3: Calculate $A=3.14*r*r$
- Step 4: Calculate $C=2*3.14*r$
- Step 5: Display A,C
- Step 6: Stop

- BEGIN
- READ r
- CALCULATE A and C
- $A=3.14*r*r$
- $C=2*3.14*r$
- DISPLAY A
- END



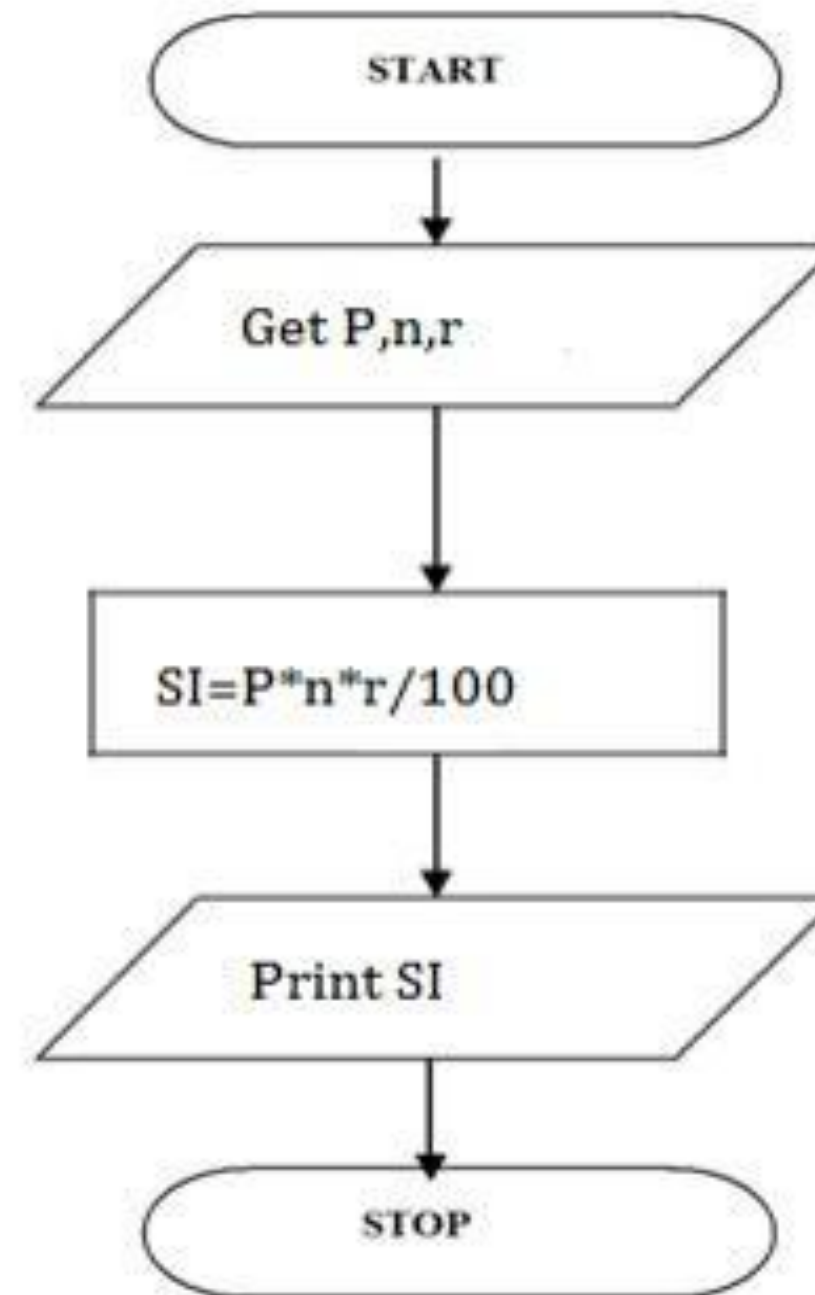


CALCULATING SIMPLE INTEREST



- Step 1: Start
- Step 2: get P, n, r value
- Step 3: Calculate $SI = (p * n * r) / 100$
- Step 4: Display S
- Step 5: Stop

- BEGIN
- READ P, n, r
- CALCULATE S
- $SI = (p * n * r) / 100$
- DISPLAY SI
- END

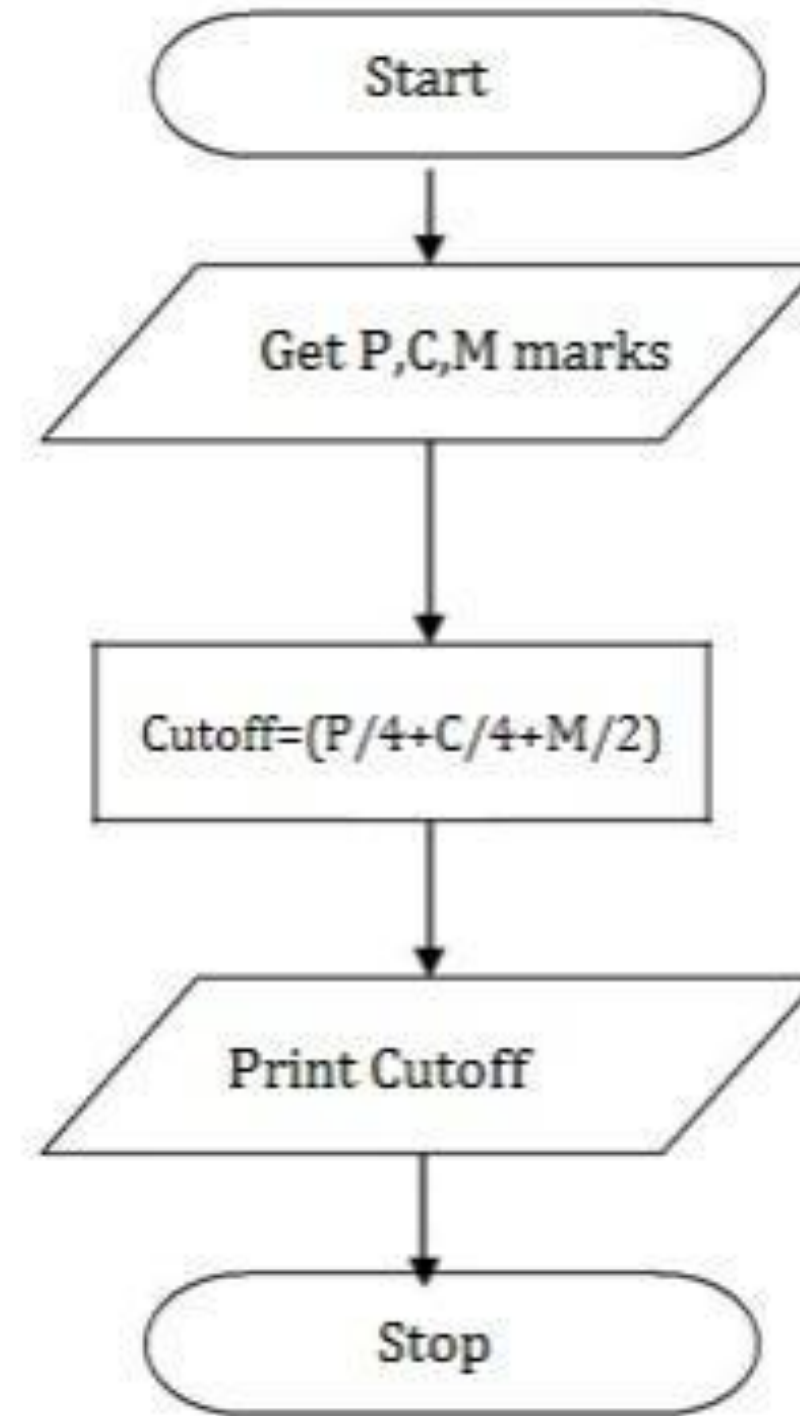




CALCULATING ENGINEERING CUTOFF



- Step 1: Start
 - Step 2: get P,C,M value
 - Step 3: calculate Cutoff= $(P/4+C/4+M/2)$
 - Step 4: Display Cutoff
 - Step 5: Stop
-
- BEGIN
 - READ P,C,M
 - CALCULATE
 - Cutoff= $(P/4+C/4+M/2)$
 - DISPLAY Cutoff
 - END

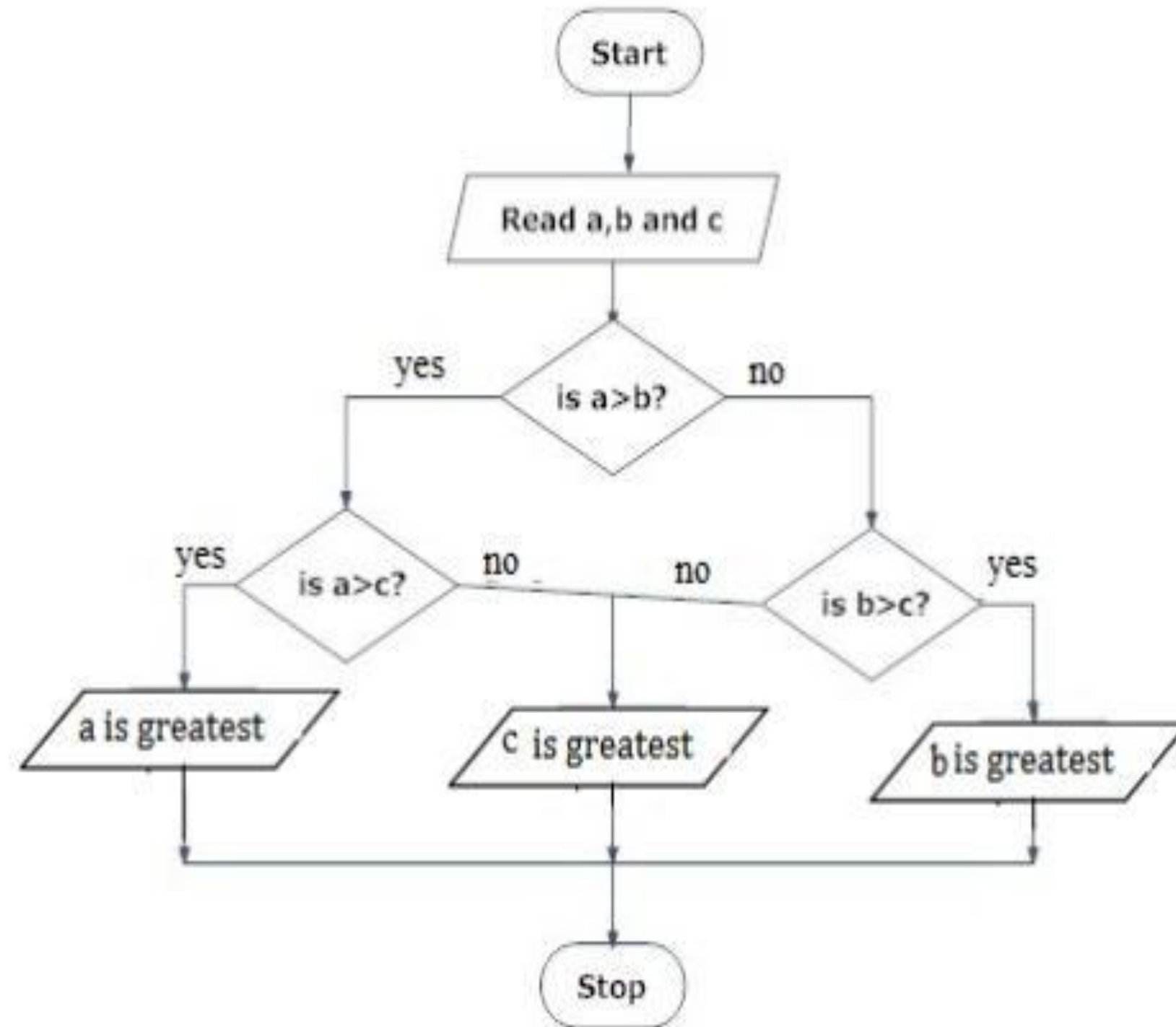




TO CHECK GREATEST OF THREE NUMBERS

- Step 1: Start
- Step 2: Get A, B, C
- Step 3: if(A>B) goto Step4 else goto step5
- Step 4: If(A>C) print A else print C
- Step 5: If(B>C) print B else print C
- Step 6: Stop

- BEGIN
- READ a, b, c
- IF (a>b) THEN
- IF(a>c) THEN
- DISPLAY a is greater
- ELSE
- DISPLAY c is greater
- END IF
- ELSE
- IF(b>c) THEN
- DISPLAY b is greater
- ELSE
- DISPLAY c is greater
- END IF
- END



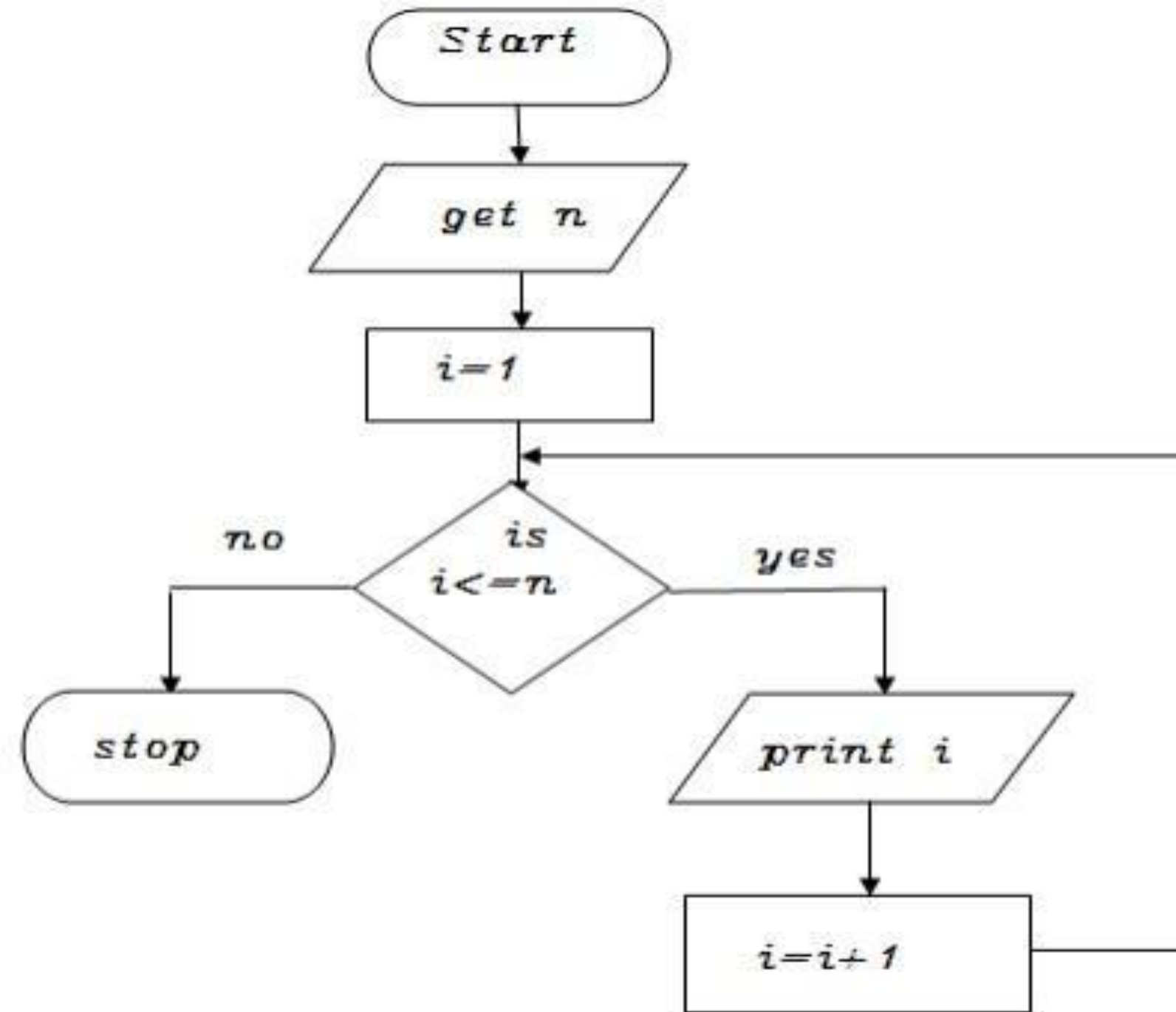


TO PRINT N ODD NUMBERS



- Step 1: start
- step 2: get n value
- step 3: set initial value $i=1$
- step 4: check if($i \leq n$) goto step 5 else goto step 8
- step 5: print i value
- step 6: increment i value by 2
- step 7: goto step 4
- step 8: stop

- BEGIN
- GET n
- INITIALIZE $i=1$
- WHILE($i \leq n$) DO
- PRINT i
- $i=i+2$
- ENDWHILE
- END



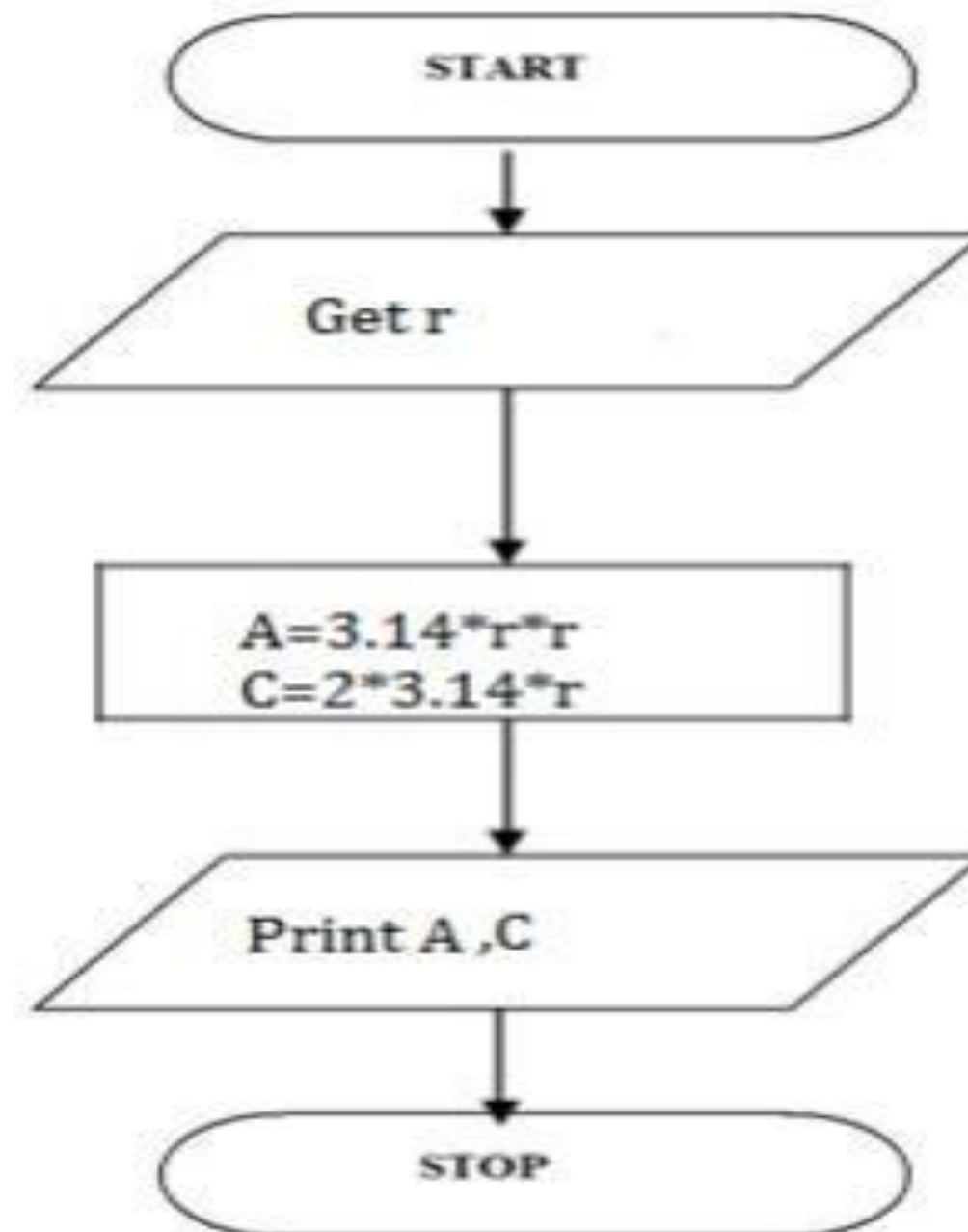


CALCULATING AREA AND CIRCUMFERENCE OF CIRCLE



- Step 1: Start
- Step 2: get r value
- Step 3: Calculate $A=3.14*r*r$
- Step 4: Calculate $C=2*3.14*r$
- Step 5: Display A,C
- Step 6: Stop

- BEGIN
- READ r
- CALCULATE A and C
- $A=3.14*r*r$
- $C=2*3.14*r$
- DISPLAY A
- END



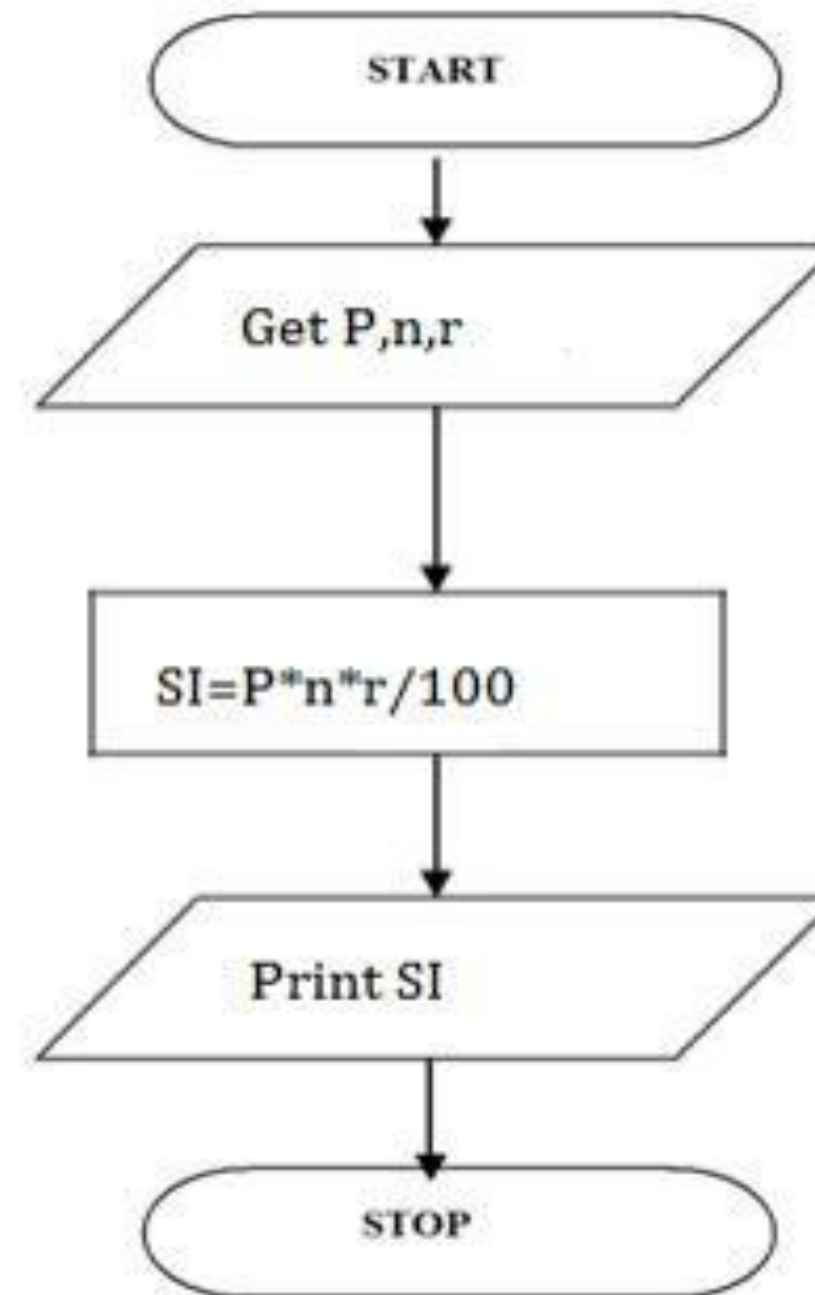


CALCULATING SIMPLE INTEREST



- Step 1: Start
- Step 2: get P, n, r value
- Step 3: Calculate $SI = (p * n * r) / 100$
- Step 4: Display S
- Step 5: Stop

- BEGIN
- READ P, n, r
- CALCULATE S
- $SI = (p * n * r) / 100$
- DISPLAY SI
- END

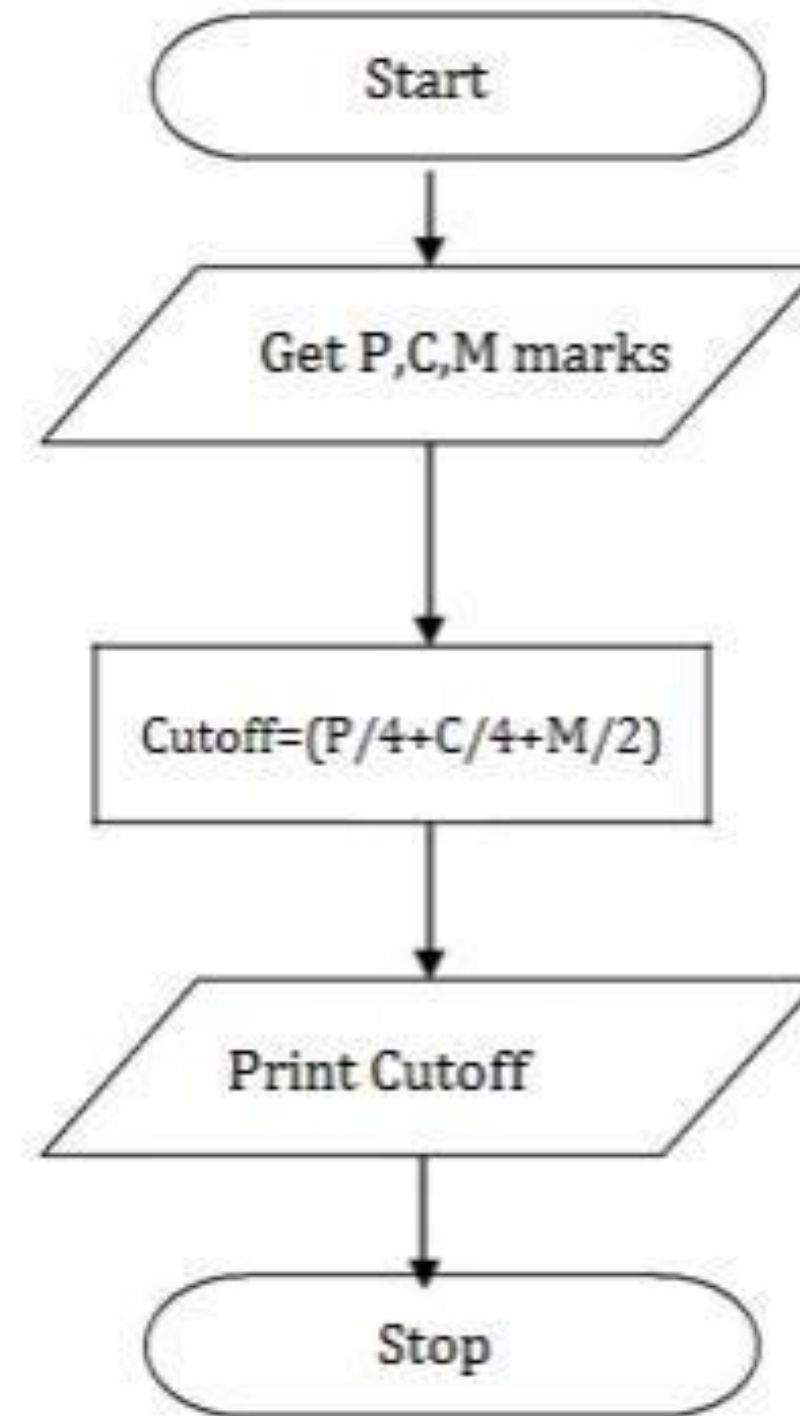




CALCULATING ENGINEERING CUTOFF



- Step 1: Start
 - Step 2: get P,C,M value
 - Step 3: calculate Cutoff= $(P/4+C/4+M/2)$
 - Step 4: Display Cutoff
 - Step 5: Stop
-
- BEGIN
 - READ P,C,M
 - CALCULATE
 - Cutoff= $(P/4+C/4+M/2)$
 - DISPLAY Cutoff
 - END

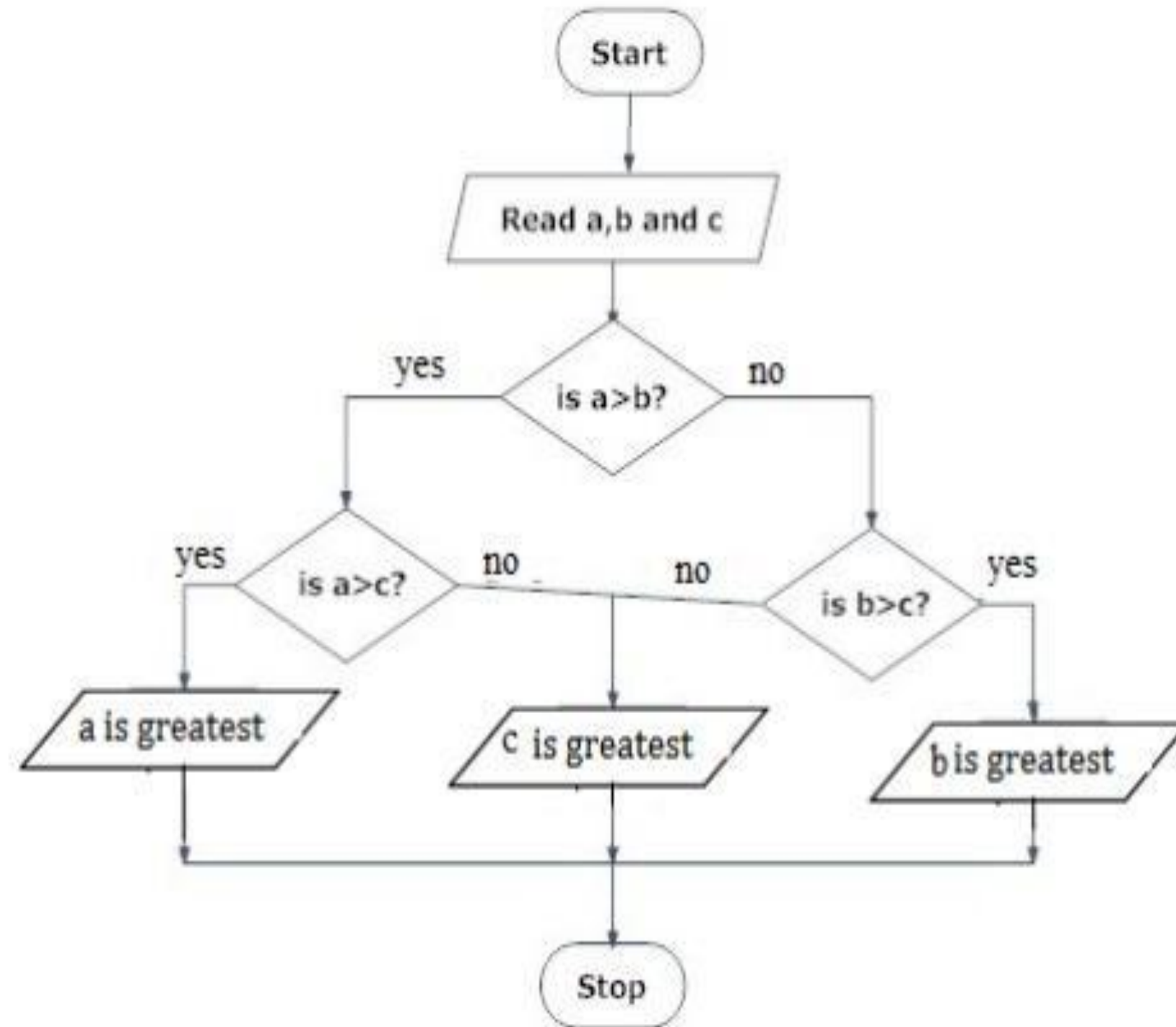




TO CHECK GREATEST OF THREE NUMBERS

- Step 1: Start
- Step 2: Get A, B, C
- Step 3: if(A>B) goto Step4 else goto step5
- Step 4: If(A>C) print A else print C
- Step 5: If(B>C) print B else print C
- Step 6: Stop

- BEGIN
- READ a, b, c
- IF (a>b) THEN
- IF(a>c) THEN
- DISPLAY a is greater
- ELSE
- DISPLAY c is greater
- END IF
- ELSE
- IF(b>c) THEN
- DISPLAY b is greater
- ELSE
- DISPLAY c is greater
- END IF
- END



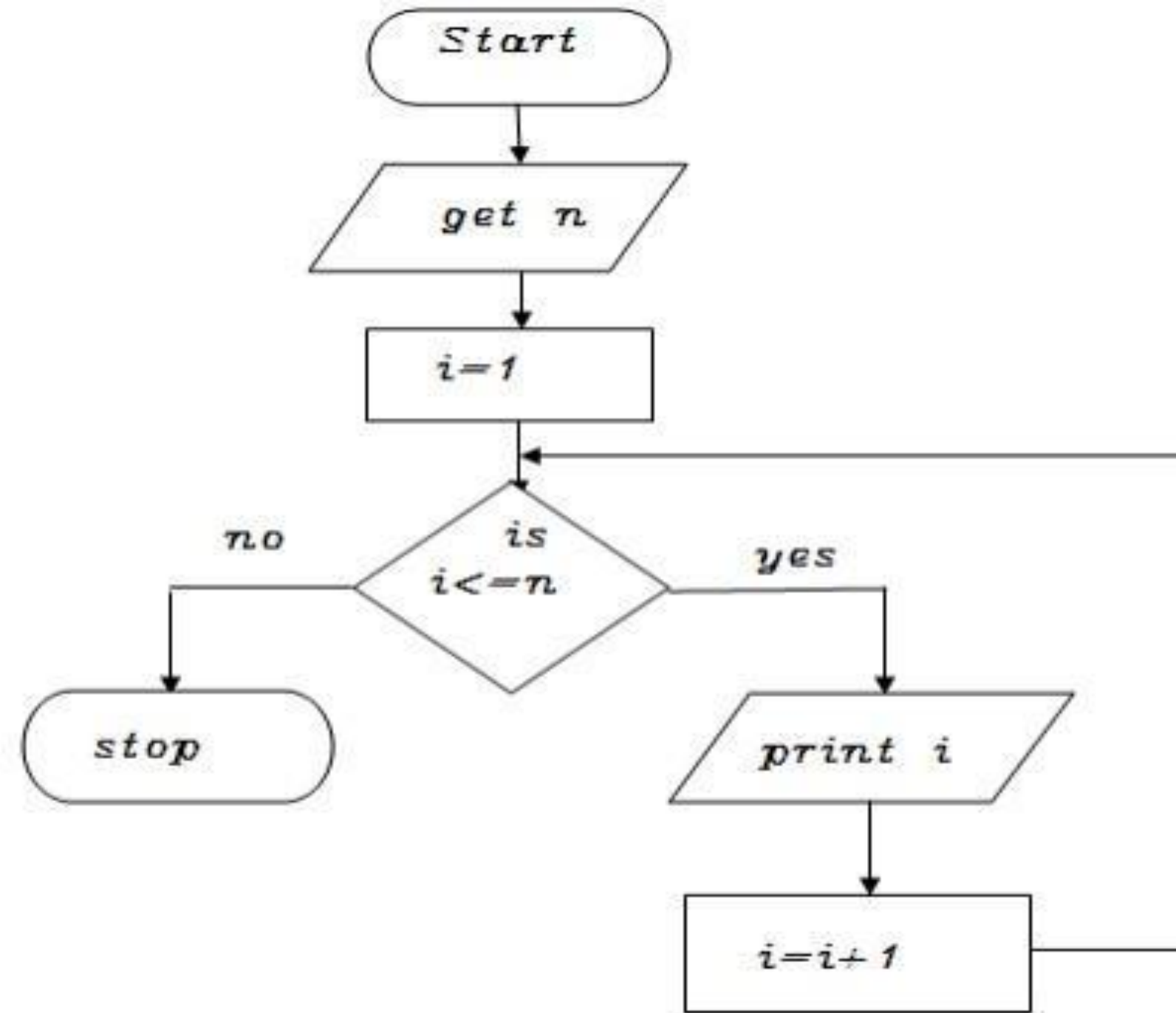


TO PRINT N ODD NUMBERS



- Step 1: start
- step 2: get n value
- step 3: set initial value $i=1$
- step 4: check if($i \leq n$) goto step 5 else goto step 8
- step 5: print i value
- step 6: increment i value by 2
- step 7: goto step 4
- step 8: stop

- BEGIN
- GET n
- INITIALIZE $i=1$
- WHILE($i \leq n$) DO
- PRINT i
- $i=i+2$
- ENDWHILE
- END





PROGRAMMING LANGUAGE



- A programming language is a vocabulary and set of grammatical rules for instructing a computer or computing device to perform specific tasks.
- In other word it is set of instructions for the computer to solve the problem.
- Programming Language is a formal language with set of instruction, to the computer to solve a problem.
- The program will accept the data to perform computation.
- The programmers have to follow all the specified rules before writing program using programming language.
- The user has to communicate with the computer using language which it can understand.

Program= Algorithm + Data



NEED & TYPES OF PROGRAMMING LANGUAGES



□ Need for Programming Languages

- Programming languages are also used to organize the computation.
- Using Programming language we can solve different problems.
- To improve the efficiency of the programs.

□ Types of Programming Language

□ In general Programming languages are classified into three types. They are

- Low – level or Machine Language
- Intermediate or Assembly Language
- High – level Programming language