Problem Solving

SESSION 3

Objectives

- Explain algorithm
- Explain flowcharts
- Explain pseudocodes
- Explain dry run

An Introduction

Use of computers

- Solve problems
- Perform calculations

Program

- *"A program is a precise sequence of steps to solve a particular problem."*

Algorithms

- Logical and concise list of steps required to solve a problem

Algorithm

- Algorithm is an ordered set of instructions.
- The algorithm should have the ability to alter the order of execution of the instructions.
- The three types of statement constructs that an algorithm can have are as follows:
  - Sequential
  - Conditional
  - Iteration

Algorithms (Example)

Steps to find the sum of two numbers:

- Read the two numbers
- Add them up
- Display the sum

Steps to write a program

- Analyze a problem statement, typically expressed as a word problem.
- Express its essence, abstractly and with examples.
- Formulate statements and comments in a precise language.
- Evaluate and revise the activities in light of checks and tests and
- Pay attention to detail.

The Programming Approach

Algorithm

- Method or approach used to solve a problem

Steps involved in solving a problem:

- Studying the problem in detail
- Gathering the relevant information
- Processing the information
- Arriving at the results

The Programming Approach (Example)

Steps for booking a railway ticket:

- Passenger enters details like name, age, the starting point of journey, destination, date of journey in the reservation slip
- Counter assistant checks for the availability of the seats on receiving the slip from the passenger
- If the required number of seats are available the passenger is given a confirmed ticket
- Otherwise, a wait-listed ticket is issued
- A wait-listed ticket is confirmed if another person cancels his ticket
- Passenger is given a refund if he is not given the confirmation

Flowcharts

- A flowchart is a graphical representation of an algorithm.
- It charts the logical flow of instructions or activities in a process.
- Flowcharts help programmers to view how the statements in a program are interrelated.
- Each activity in a flowchart is depicted using symbols.

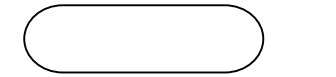
Flowcharts

Diagrammatic representation that illustrates the sequence of operations to be performed to arrive at a solution

Serve the following purpose:

- Easier to understand at a glance than a narrative description
- Programs can be reviewed and debugged easily
- Provide effective program documentation
- Explaining a program or discussing a solution is made easy

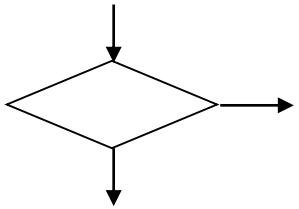
Symbols used in a Flowchart



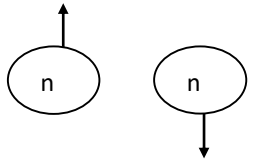
Start or end of the program

Computational steps

Input or output instructions



Decision making and branching

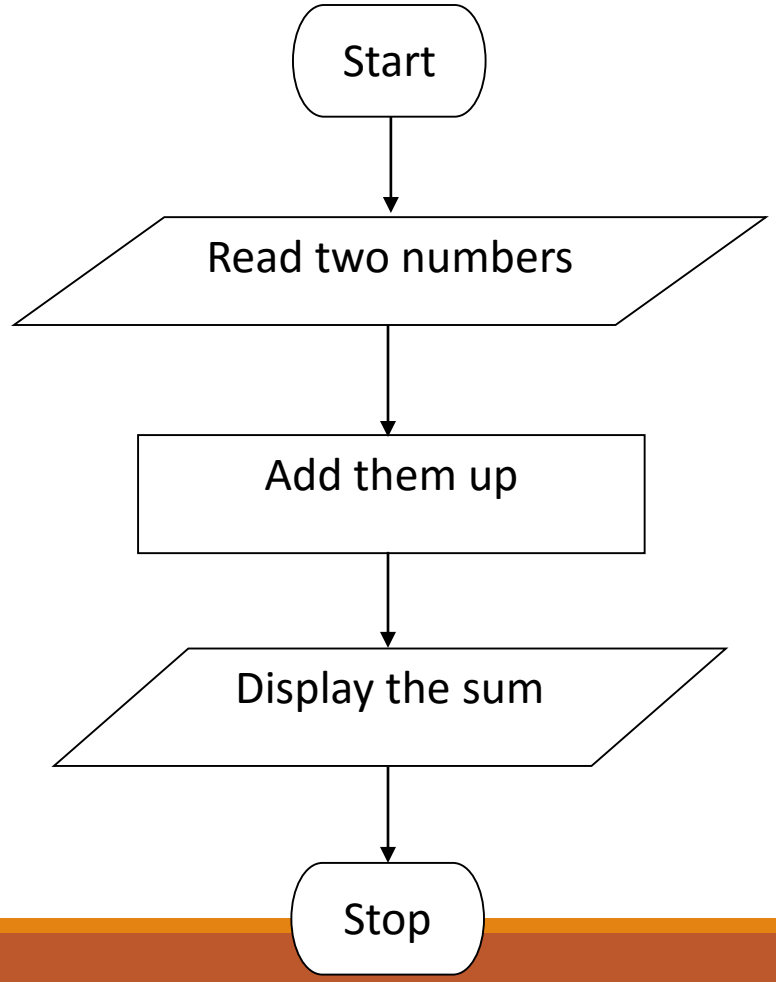


Connectors of two parts of a program

Flow of the program

Flowcharts (Example)

To find the sum of two numbers



Flowcharts

Also involves

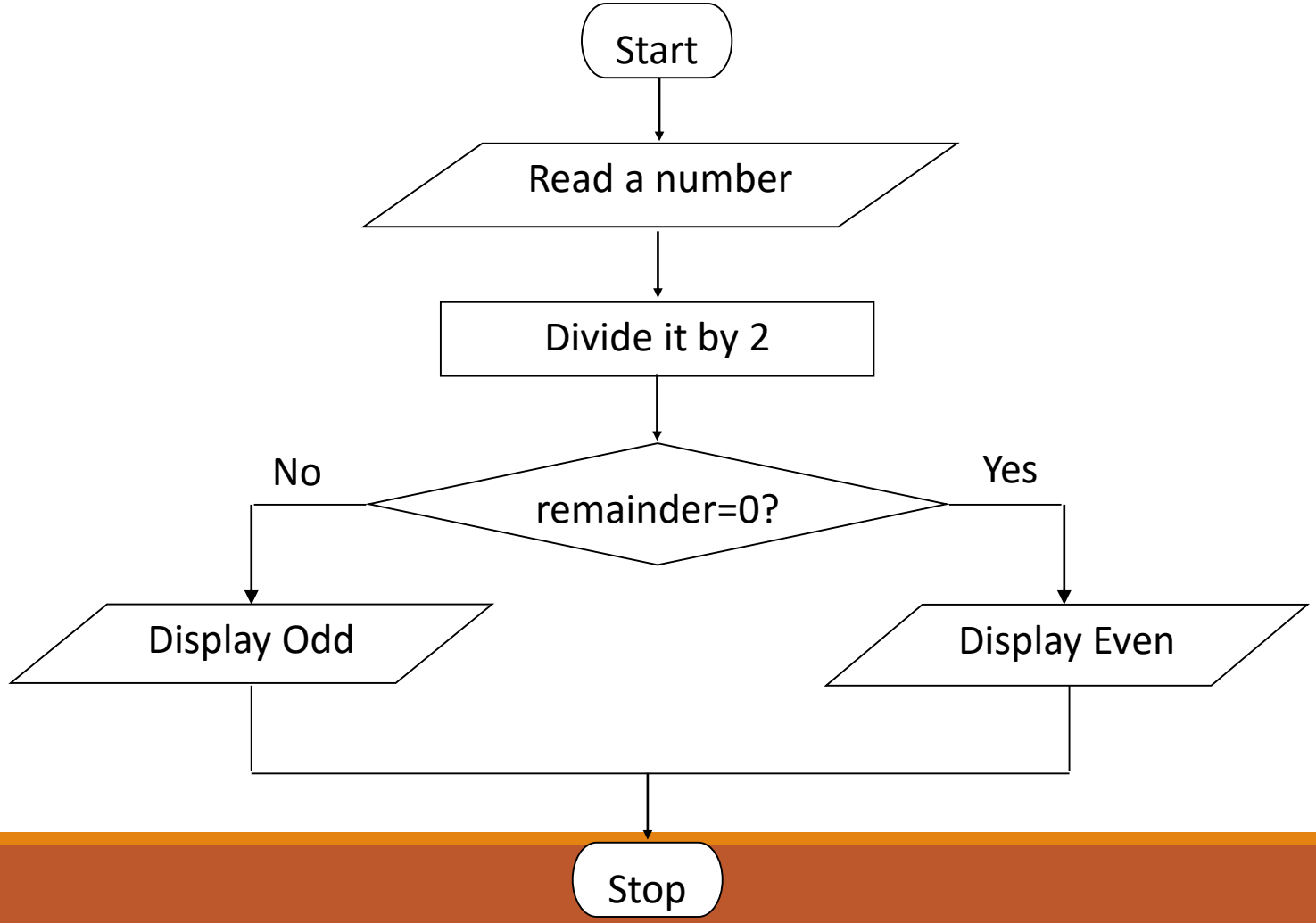
- Repeating certain steps of a program
- Taking alternate sequence steps for some situations

Branching

Process of following one of two or more alternate paths of program logic

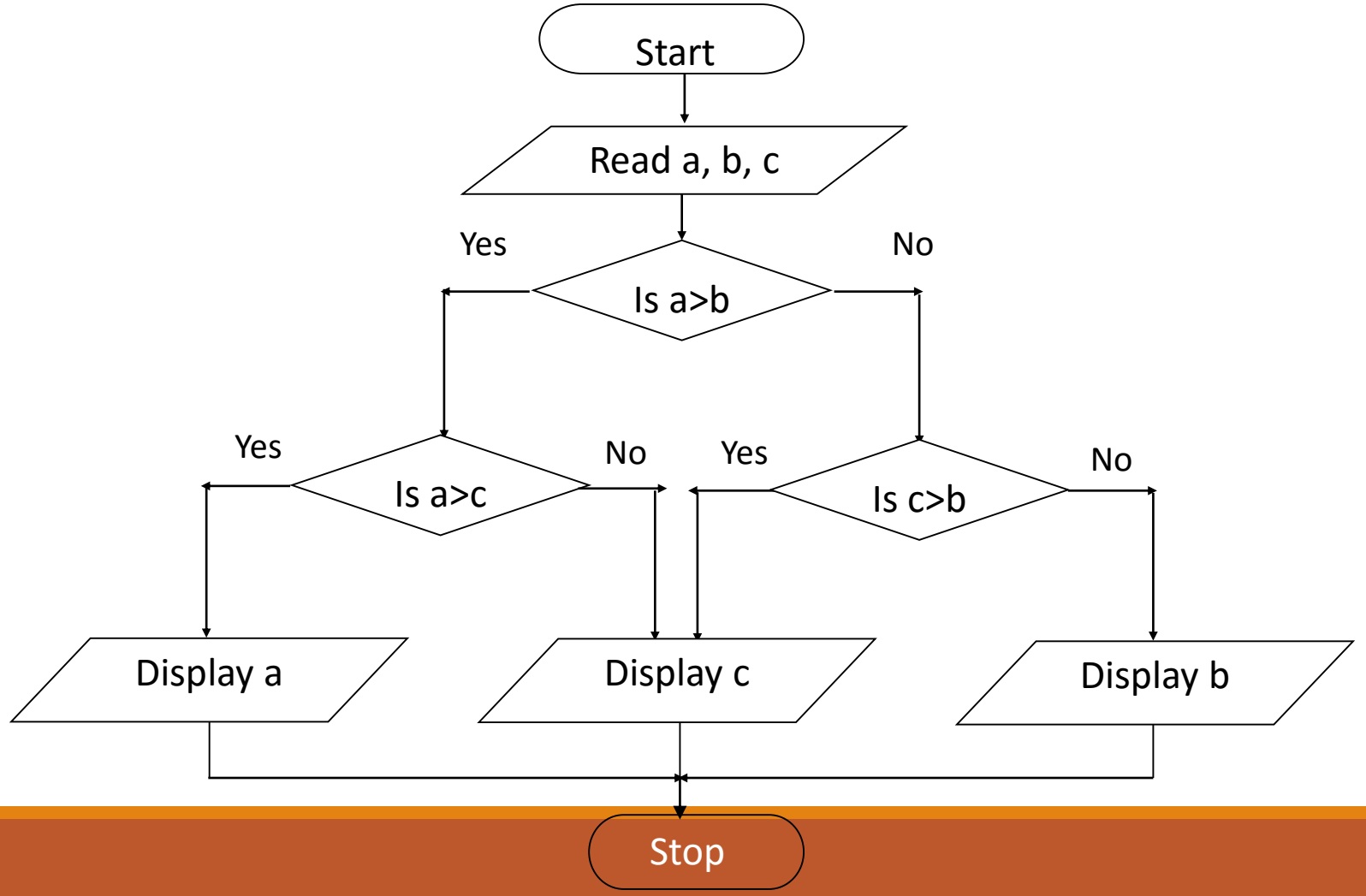
Branching (Example 1)

To find if a number is odd or even

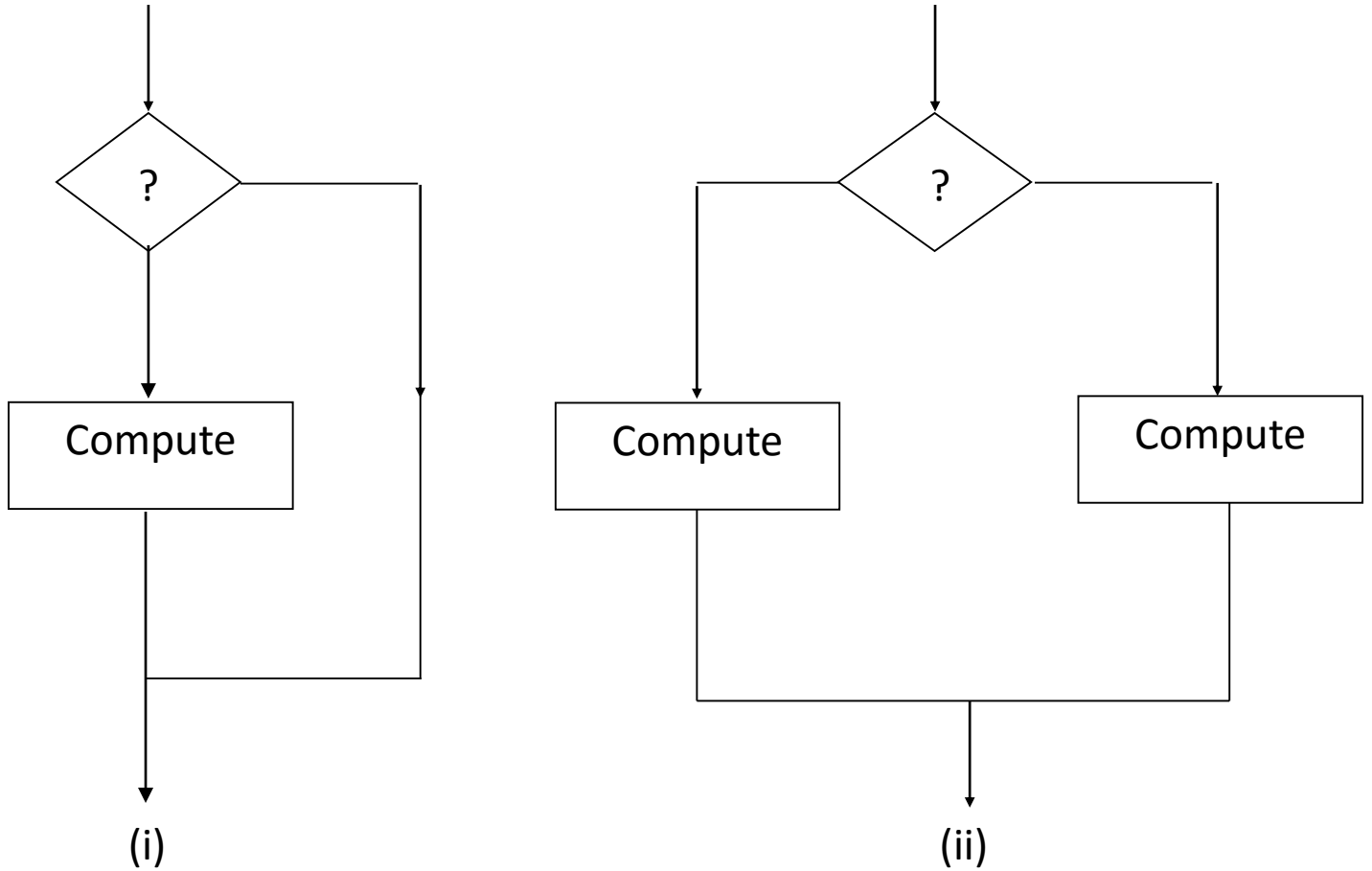


Branching (Example 2)

To find the greatest of the three numbers



General form of flowcharts involved in branching



Looping

Refers to the repeated use of one or more steps

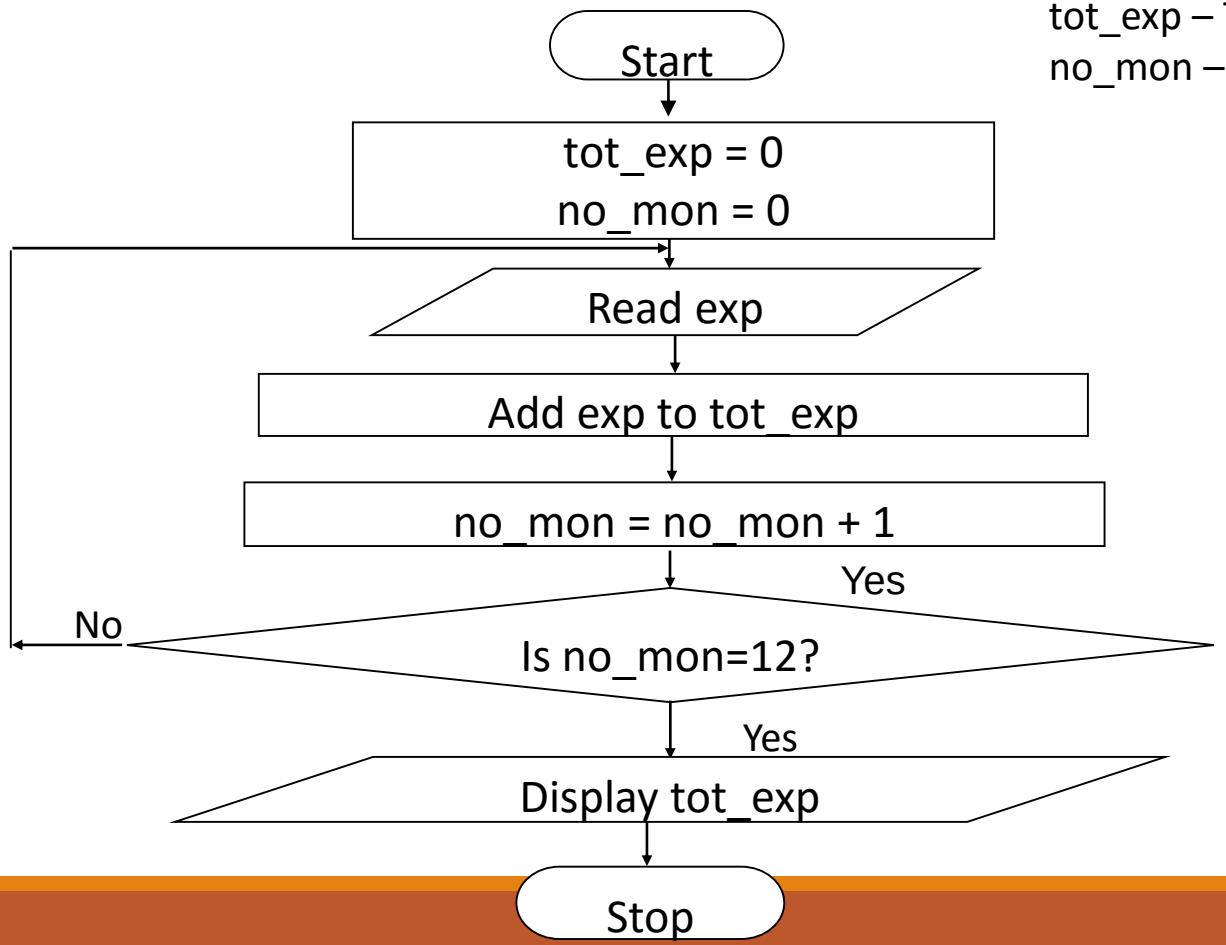
Types of loops

- Fixed
- Variable

Looping (Example 1)

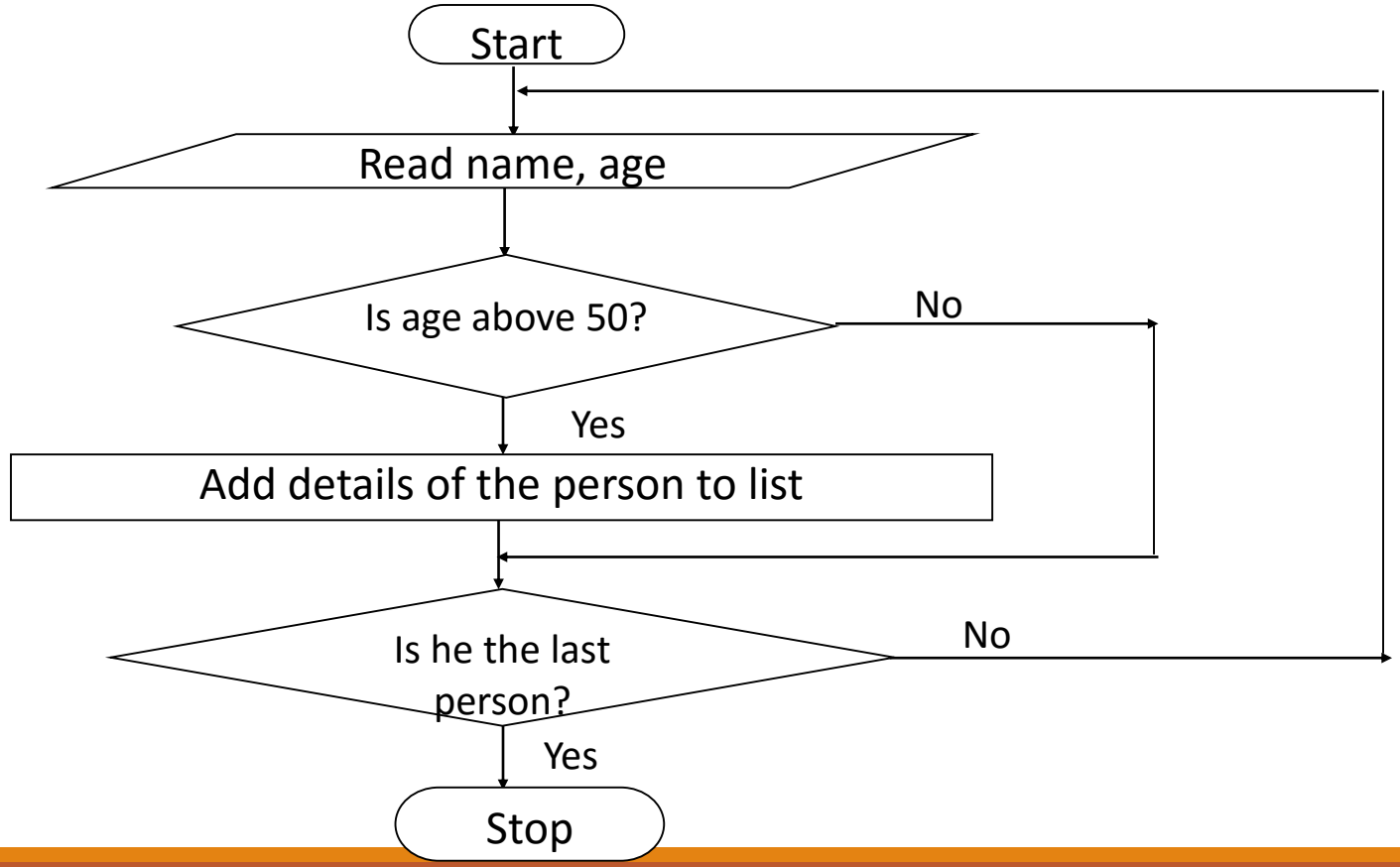
To calculate the sum of monthly expenditure for an entire year

tot_exp – Total expenditure
no_mon – Number of months

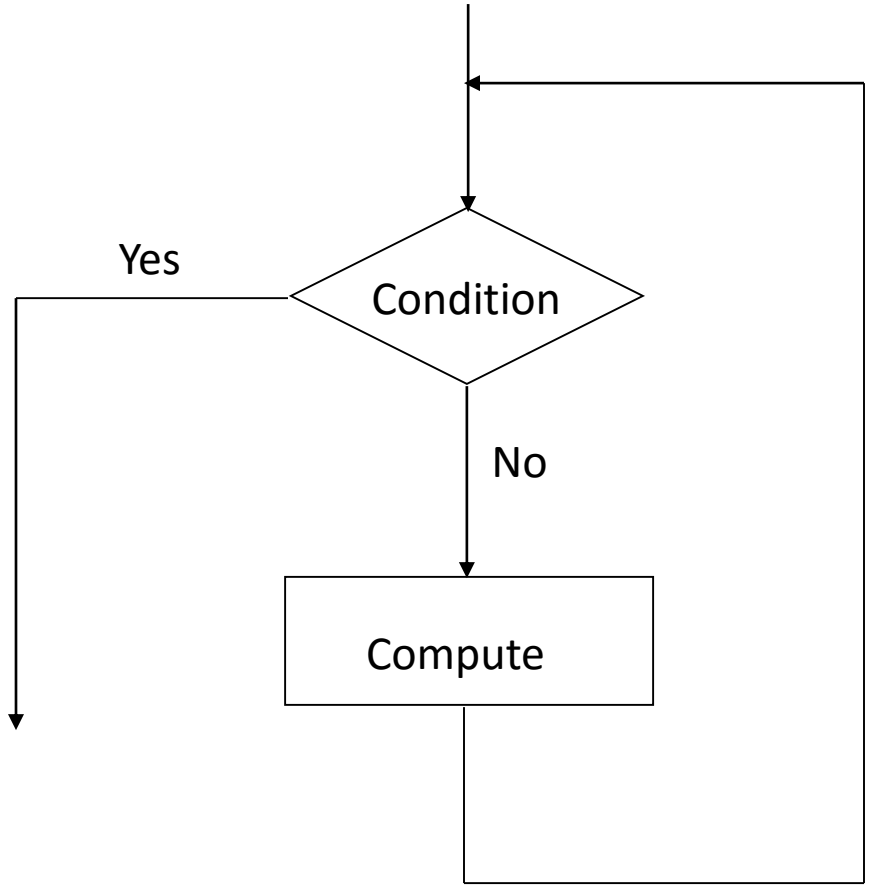


Looping (Example 2)

Given the information such as name and age and to maintain a list of people aged above 50



Basic flowchart for a loop



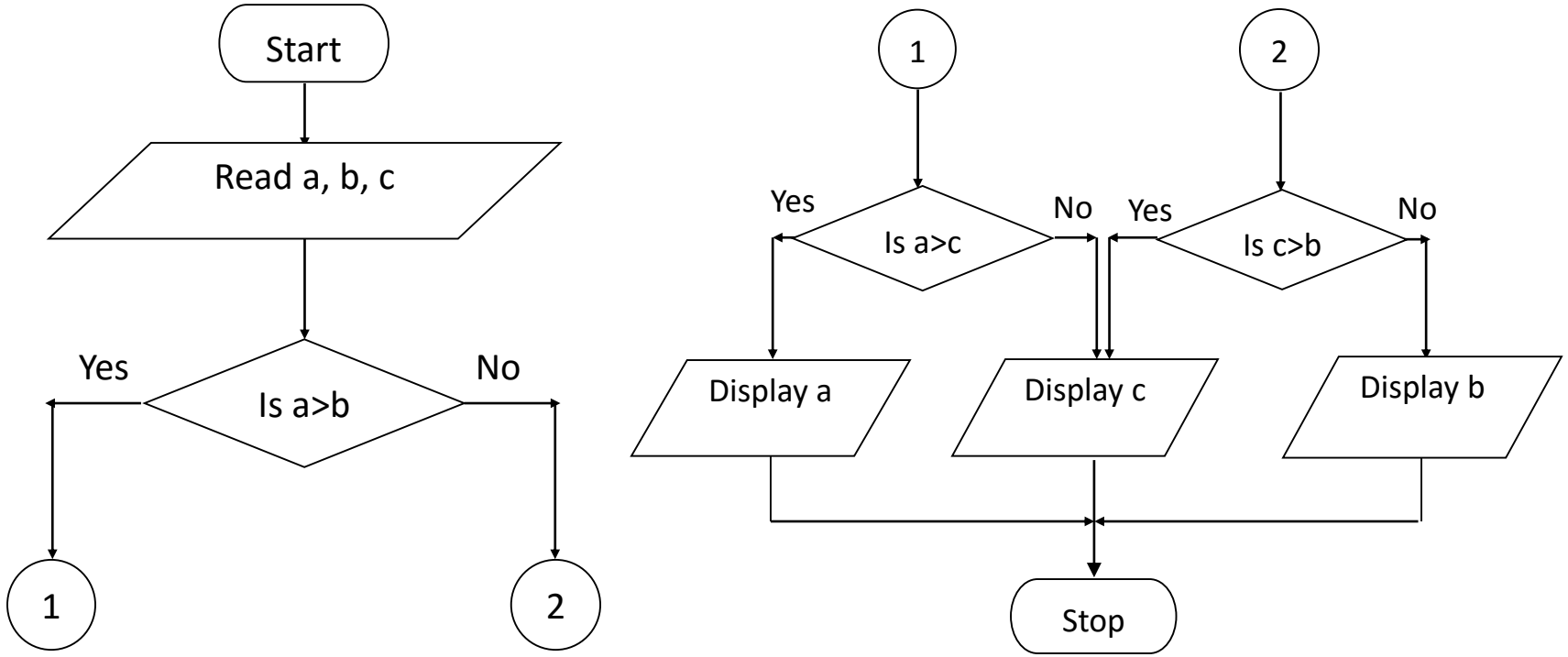
Connectors

In preparing flowcharts for complex problems

- The flowchart may not fit in a single page
- It may be difficult to interconnect all boxes directly

Connectors (Example)

- To find the largest of three numbers a, b and c



Tips for drawing a Flowchart

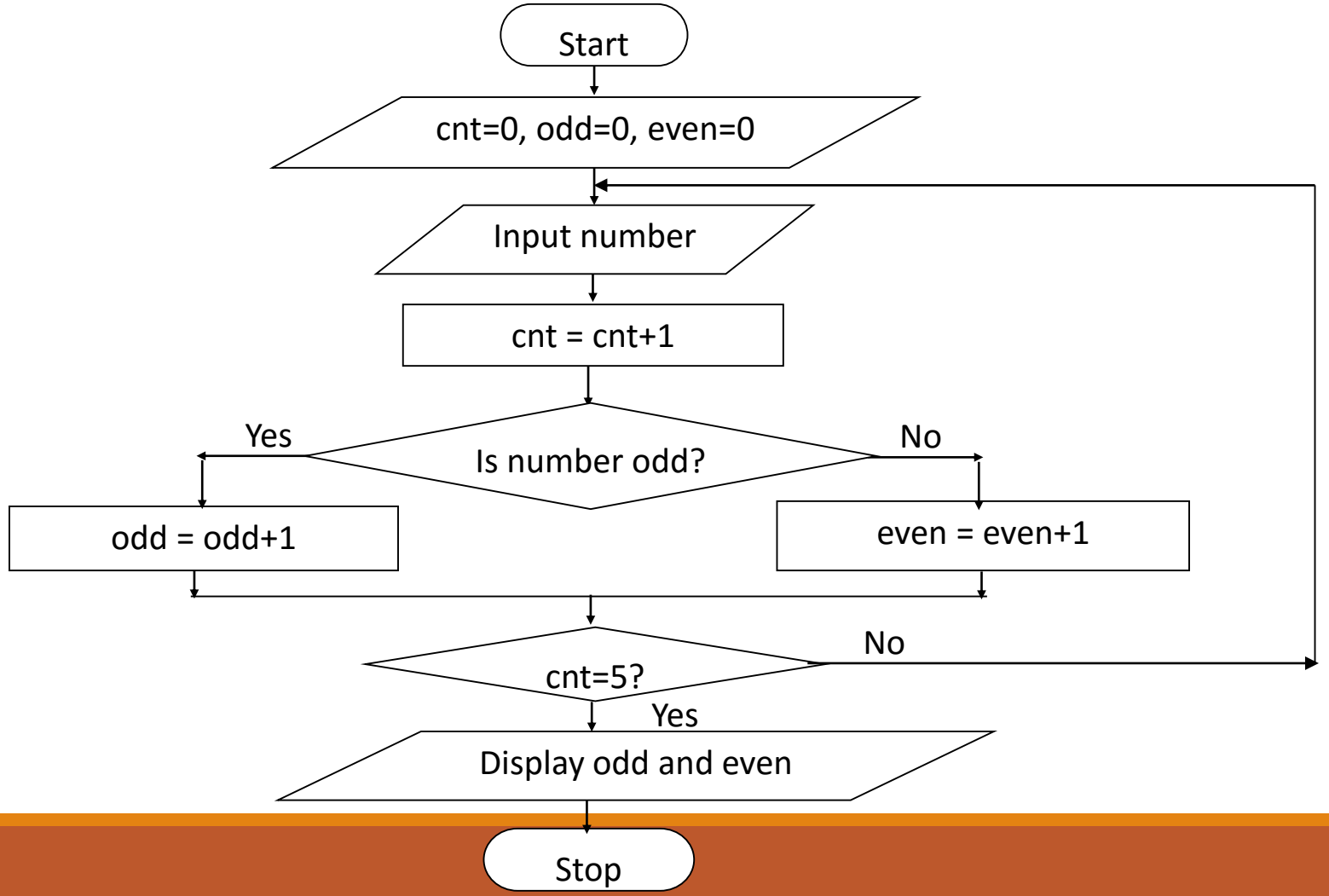
- Initially concentrate on the logic of the problem and draw the main-path of the flowchart
- Then add all the branches and loops
- A flowchart can have only one **Start** point and one **Stop** point
- Every step of a program need not be represented in a flowchart
- Use descriptive terms that aptly represent the logic of a problem
- Remember that another user or programmer should easily understand the flowchart

Counter Variables

- Keeps track of the number of times a loop has been performed
- Initially is assigned a value of zero, which increases as and when a loop is performed
- Usually named as **cnt**

Counter Variables (Example)

To count the number of odd and even numbers accepted



Pseudocodes 1-6

- The word pseudo means false. As the name suggest, pseudocode is not the actual code.

- It is a method of algorithm writing which uses a certain standard set of words which makes it resemble a code.

- However, pseudocode cannot be complied or executed as a code.

Pseudocodes 2-6

- Each pseudocode must start with the word **BEGIN** or **START**, and end with **END** or **STOP**.
- The statements between **START** and **END** are English phrases and indented to make the word **START** and **END** stand out.
- To display some value, the word **DISPLAY**, **WRITE**, or **PRINT** is used.
- To accept a value from the user, the word **INPUT** or **READ** is used.

Pseudocodes 3-6

- The pseudocode shows the process to store the sum of variables in a third variable, and then display the value stored in this third variable as shown in the example.

```
BEGIN
   INPUT A, B
   C = A + B
   DISPLAY C
END
```

- A set of instructions or steps in a pseudocode is collectively called a construct. There are three types of programming constructs namely, sequence, selection, and iteration.

Pseudocodes 4-6

- Some of the rules to be followed while writing pseudocodes are as follows:
  - The pseudocode must be easy to understand by all and not just the programmer. The variables mentioned in the pseudocode must be self-descriptive. Avoid using abbreviations and shortened versions of words in the pseudocode.
  - The pseudocode must not contain actual programming code but should have only logical steps to show how to operate a code.

Pseudocodes 5-6

- Some of the advantages of using pseudocode are as follows:
  - Easy to create
  - No symbols
  - No specific syntax
  - Easy to translate
  - Reduces time

Pseudocodes 6-6

- Besides the advantages, pseudocodes have some disadvantages. They are as follows:
  - Lack of standards
  - Do not focus on big picture

Dry Run

Is a manual way of testing the correctness of an algorithm

Dry Run (Example)

Step 1. Start

Step 2. X=10

Step 3. Y=5

Step 4. M=0

Step 5. M=X+Y+(X*Y)

Step 6. Y=Y+4

Step 7. M=M+Y

Step 8. Display X,Y,M

Step 9. End

DRY RUN TABLE

	X	Y	M
Initial values	10	5	0
After Step 5	10	5	65
After Step 6	10	9	65
After Step 7	10	9	74

A Sample C# Program

```
public static void Main(string[] args)
{
        Declaration of variables;
        Statements;
}
```

Working of a C# Program

- void Main() must be present in all the programs
- { and } specify the beginning and the end of the program
- Initial statements are for the declaration of variables
- Statements that follow the declaration are the processing statements
- All statements should end with a semicolon

Keywords

- Names reserved for specific purpose and cannot be used by the programmer
- All data types are reserved as keywords

Keywords [Cont.....]

abstract	as	base	bool	break
byte	case	catch	char	checked
class	const	continue	decimal	default
delegate	do	double	else	enum
event	explicit	extern	false	finally
fixed	float	for	foreach	goto
if	implicit	in	int	interface
internal	is	lock	long	namespace
new	null	object	operator	out
override	params	private	protected	public
readonly	ref	return	sbyte	sealed
short	sizeof	stackalloc	static	string
struct	switch	this	throw	true
try	typeof	uint	ulong	unchecked
unsafe	ushort	using	virtual	void
volatile	while			