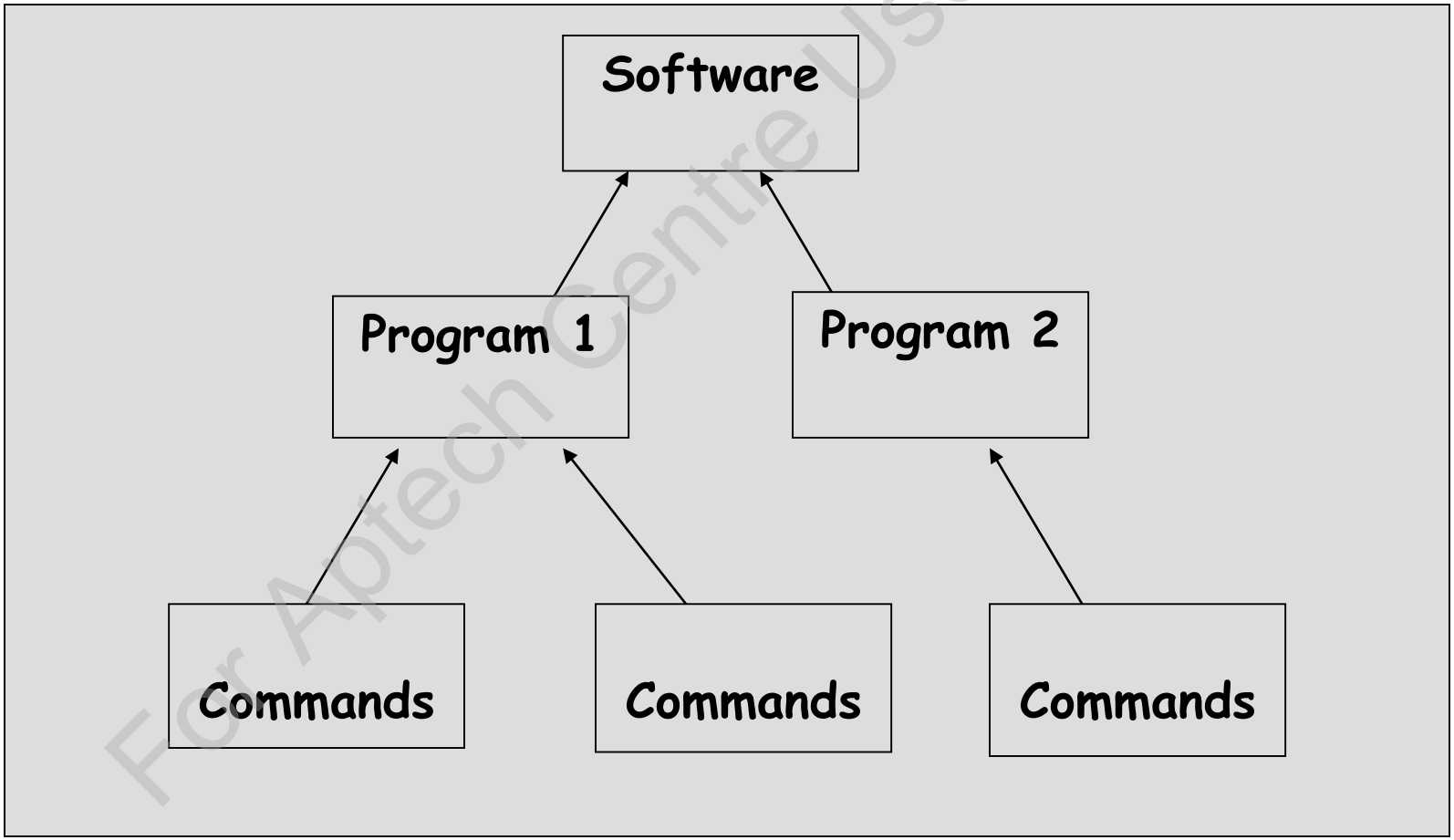
Basics of C

Session 1

Objectives

- Differentiate between Command, Program and Software
- Explain the beginning of C
- Explain when and why is C used
- Discuss the C program structure
- Discuss algorithms
- Draw flowcharts
- List the symbols used in flowcharts

Software, Program and Command



The Beginning of C

BPCL – Martin Richards

B – Ken Thompson

C – Dennis Ritchie

Bell Laboratories, Inc.

Application Areas Of C

- C was initially used for systems programming
- A system program forms a portion of the operating system of the computer or its support utilities
- Operating Systems, Interpreters, Editors, Assembly programs are usually called system programs
- The UNIX operating system was developed using C
- There are C compilers available for almost all types of PC's

Middle Level Language

High Level Language

C

Assembly Language

Structured Language

- C allows compartmentalization of code and data
- It refers to the ability to sectic off and hide all information a instructions, necessary to perform a specific task, from rest of the program

```
do
{
        i = i + 1;
        .
        .
        .
} while (i < 40);
```

- Code can be compartmentalized in C by using functions or code blocks.

About C

- **C has 32 keywords**
- These keywords combined with a formal syntax form a C programming language
- Rules to be followed for all programs written in C:
  - All keywords are lowercased
  - C is case sensitive, do while is different from DO WHILE
  - Keywords cannot be used as a variable or function name

```
main()
{
/* This is a sample Program*/
      int i,j;
      i=100;
      j=200;
        :
   }
```

The C Program Structure-1

main()

C programs are divided into units called functions.

Irrespective of the number of functions in a program, the operating system always passes control to main() when a C program is executed.

The function name is always followed by parentheses.

The parentheses may or may not contain parameters.

The C Program Structure-2

Delimiters { ... }

The function definition is followed by an open curly brace ({).

This curly brace signals the beginning of the function.

Similarly a closing curly brace (}) after the codes, in the function, indicate the end of the function.

The C Program Structure-3

Statement Terminator ...;

A statement in C is terminated with a semicolon

A carriage return, whitespace, or a tab is not understood by the C compiler.

A statement that does not end in a semicolon is treated as an erroneous line of code in C.

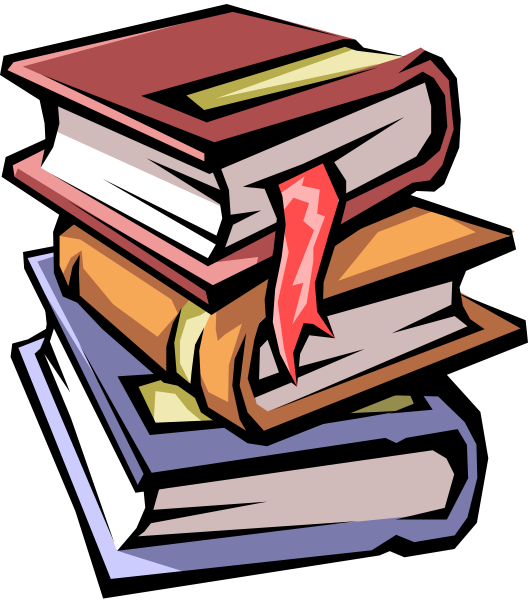
The C Program Structure-4

/* Comment Lines */

Comments are usually written to describe the task of a particular command, function or an entire program.

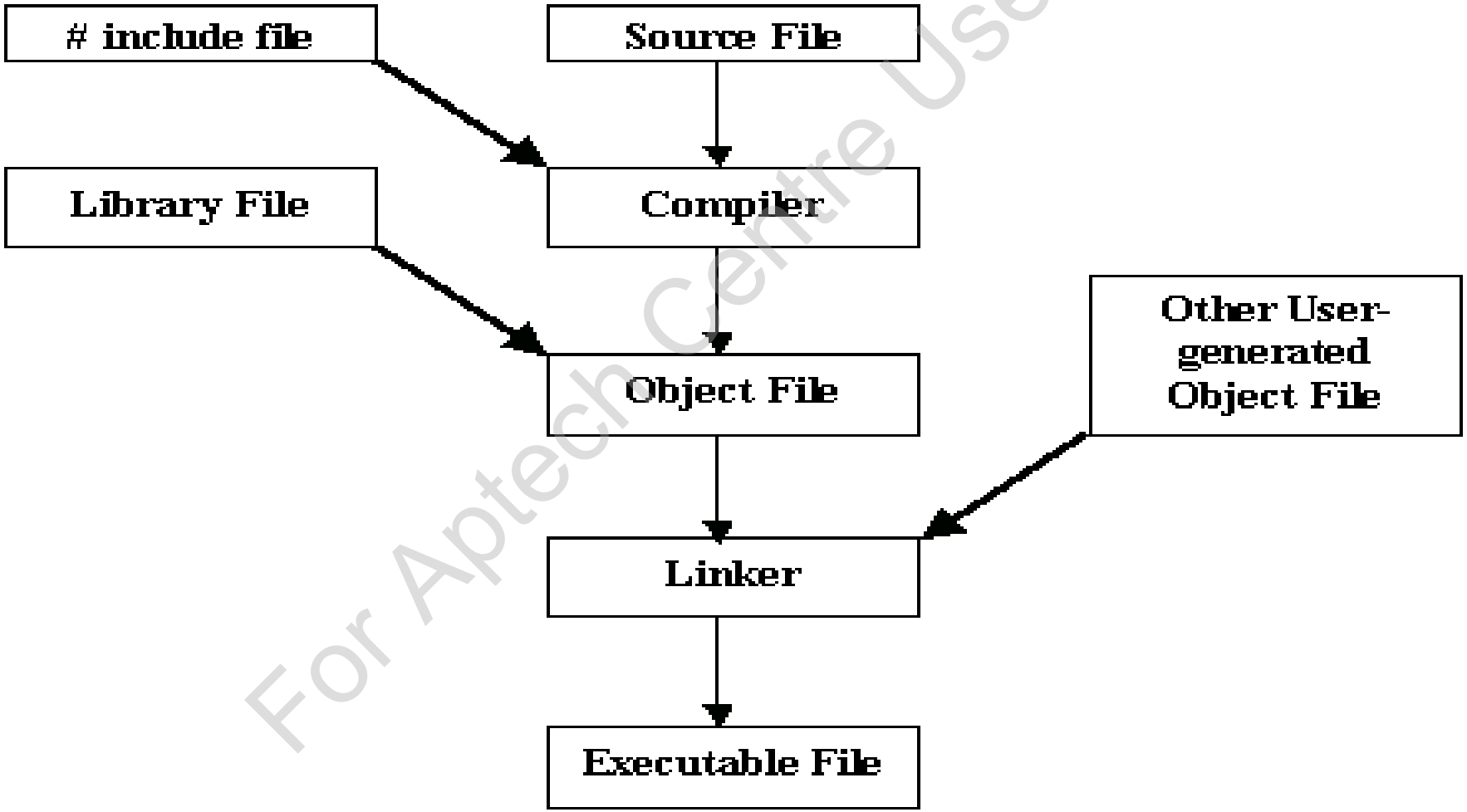
The compiler ignores them. In C, comments begin with /* and are terminated with */, in case the comments contain multiple lines

The C Library



- All C compilers come with a standard library of functions
- A function written by a programmer can be placed in the library and used when required
- Some compilers allow functions to be added in the standard library
- Some compilers require a separate library to be created

Compiling & Running A Program



The Programming Approach to Solving Problems

Algorithm is a set of steps that are performed to solve a problem. The example below describes an algorithm

Classroom

Leaving the classroom

Head towards the staircase

Go to the basement

Head for the cafeteria

Cafeteria

These are the steps followed when a student wants to go to the cafeteria from the classroom

Solving a Problem

In order to solve a problem

- Understand the problem clearly
- Gather the relevant information
- Process the information
- Arrive at the solution

Pseudocode

It is not actual code. A method of algorithm - writing which uses a standard set of words which makes it resemble code

```
BEGIN
DISPLAY 'Hello World !'
END
```

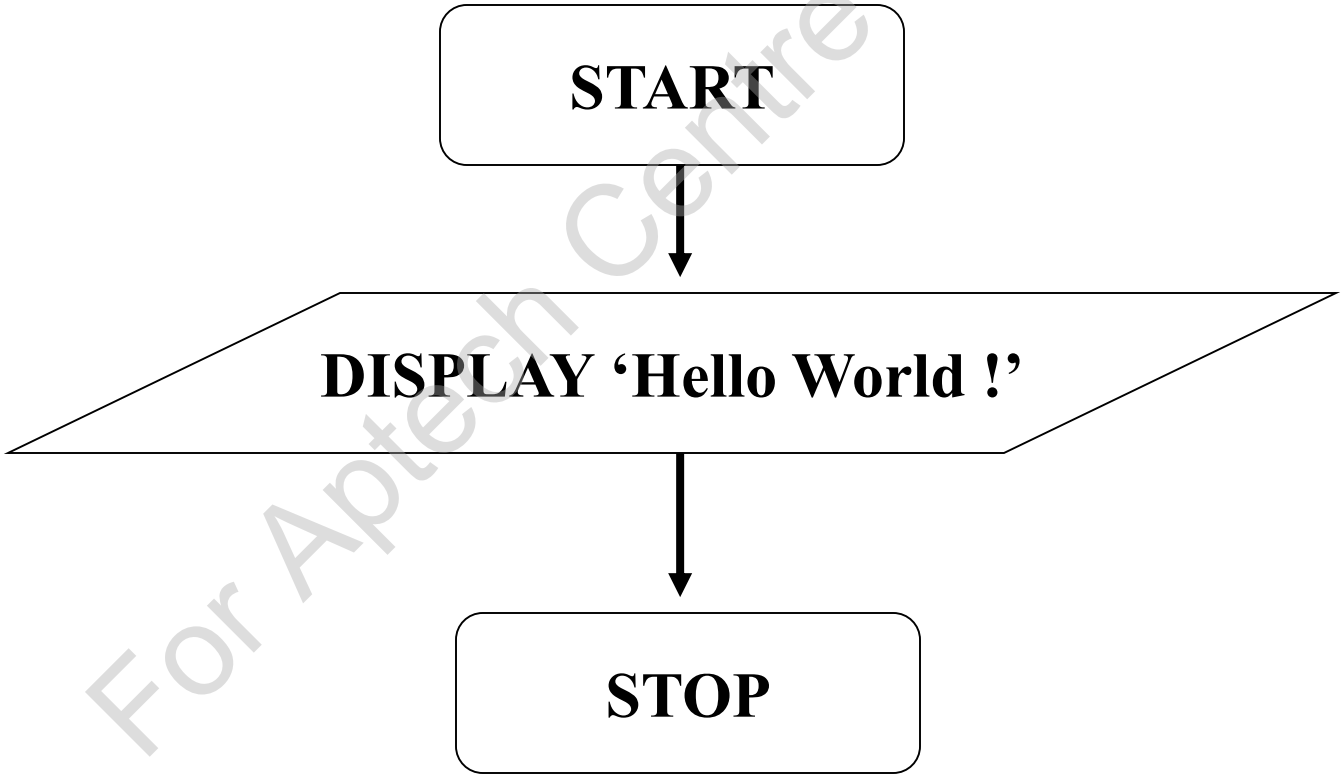
Each pseudocode starts with a BEGIN

To show some value , the word DISPLAY is used

The pseudocode finishes with an END

Flowcharts

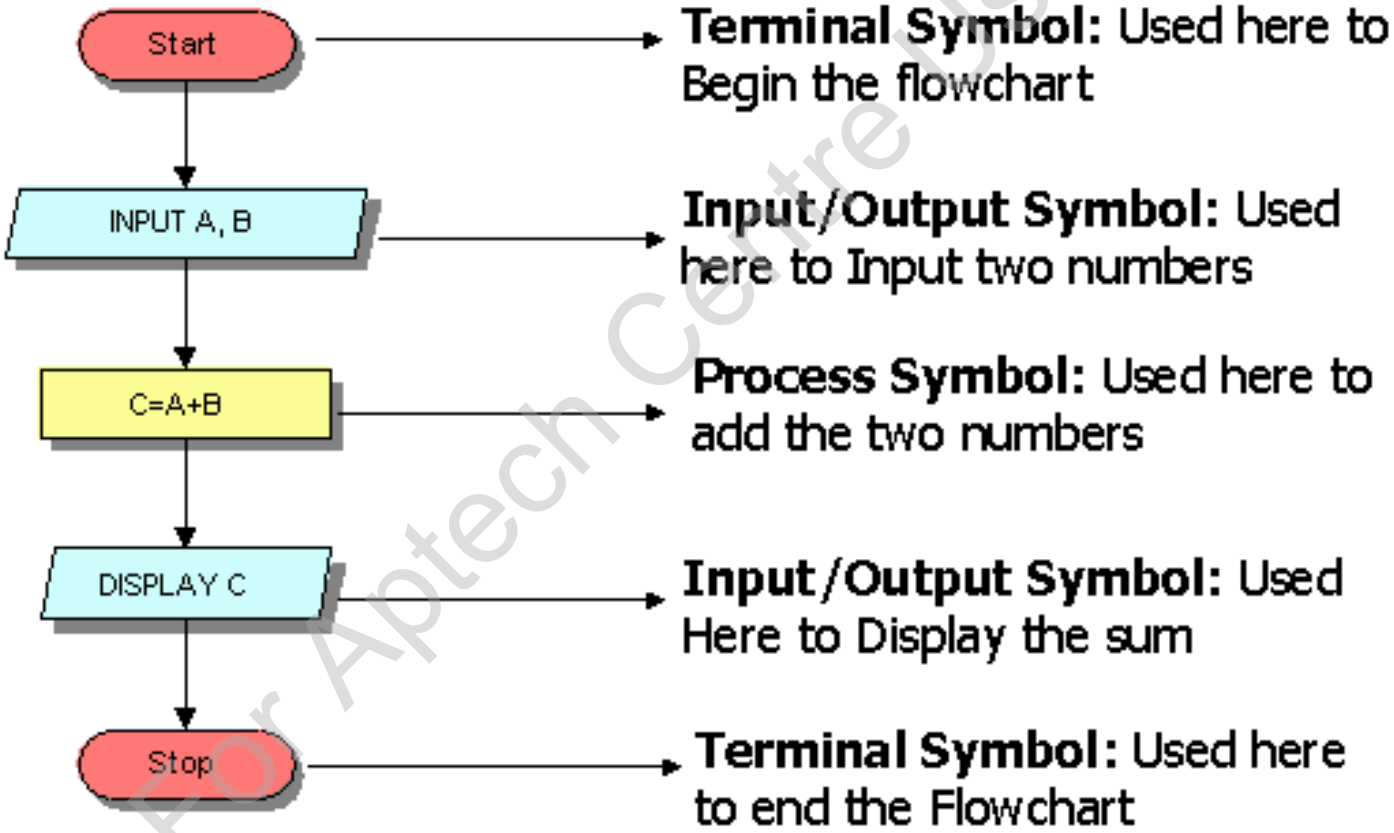
It is a graphical representation of an algorithm



The Flowchart Symbol

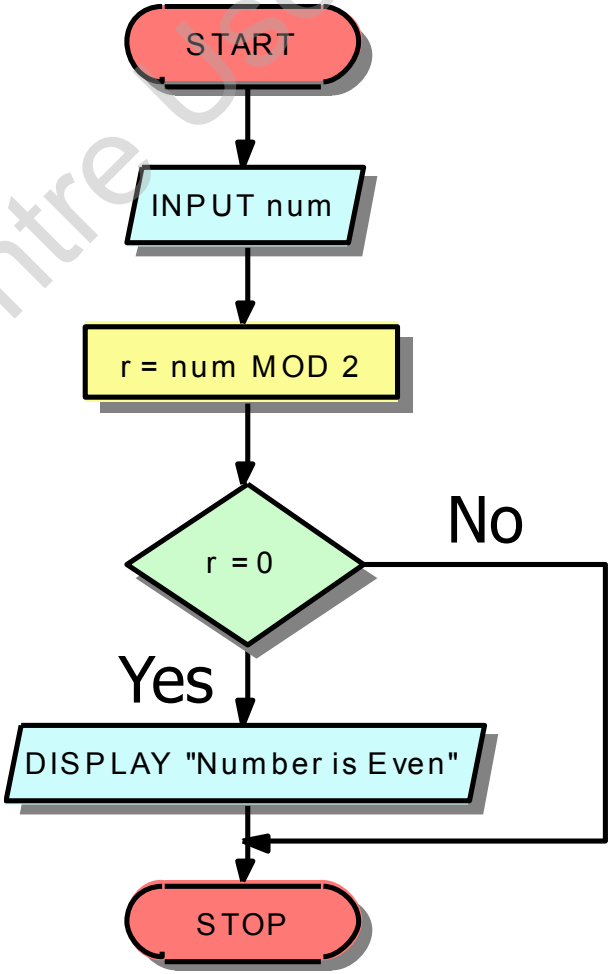
Symbol	Description
	Start or End of the Program
	Computational Steps
	Input / Output instructions
	Decision making & Branching
	Connectors
	Flow Line

Flowchart to add two numbers



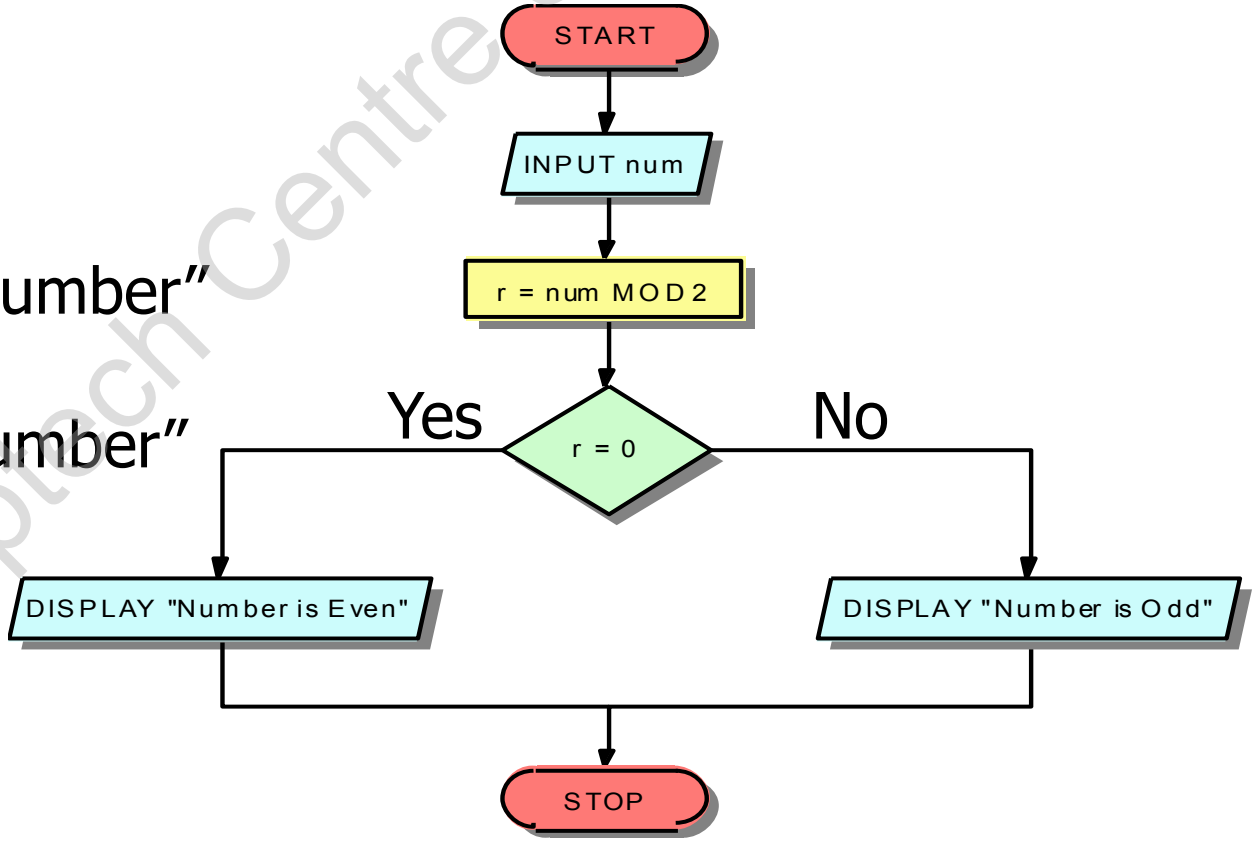
The IF Construct

BEGIN
INPUT num
r = num MOD 2
IF r=0
Display “Number is even”
END IF
END



The IF-ELSE Construct

```
BEGIN
INPUT num
r=num MOD 2
IF r=0
   DISPLAY "Even Number"
ELSE
   DISPLAY "Odd Number"
END IF
END
```



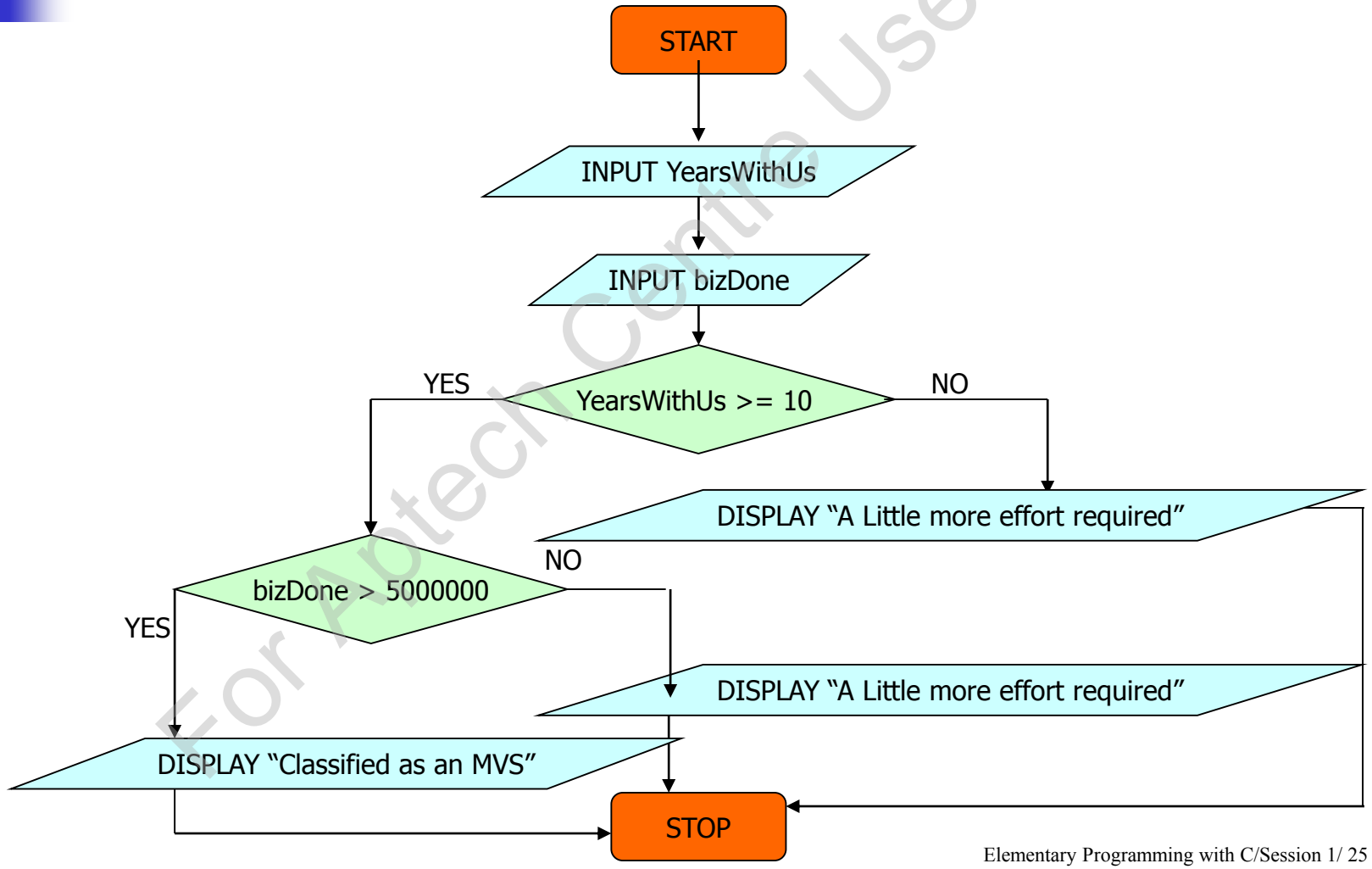
Multiple criteria using AND/OR

```
BEGIN
INPUT yearsWithUs
INPUT bizDone
IF yearsWithUs >= 10 AND bizDone >=5000000
        DISPLAY "Classified as an MVS"
ELSE
        DISPLAY "A little more effort required!"
END IF
END
```

Nested IFs-1

```
BEGIN
INPUT yearsWithUs
INPUT bizDone
IF yearsWithUs >= 10
IF bizDone >=5000000
        DISPLAY "Classified as an MVS"
          ELSE
                DISPLAY "A little more effort required!"
END IF
ELSE
        DISPLAY "A little more effort required!"
END IF
END
```

Nested IFs-2



Loops

```
BEGIN
cnt=0
WHILE (cnt < 1000)
DO
        DISPLAY "Scooby"
        cnt=cnt+1
END DO
END
```