# Condition and Branching

ESC108A Elements of Computer Science and Engineering
B. Tech. 2017

**Course Leaders:**

**Roopa G.**

**Ami Rai E.**

**Chaitra S.**

# Objectives

- At the end of this lecture, student will be able to
  - explain branched program execution
  - Identify flow chart elements and connectors that are associated with branched control flow
  - Identify the constructs in algorithms that are associated with branched control flow
  - Apply  branched control flow to solve a problem
  - Express branched control flow in C programming language

# Contents

- if, if-else and Nested if statements
- switch-case statement

# Conditions?

Why do we need them?

# Types of Control Structures

- All programs could be written in terms of following control structures

1. Selection structure/decision making statements
   - if statement (single selection)
   - if…else statement (double selection)
   - switch statement (multiple selection)

2. Repetition structure/loop statements
   - while statement
   - do…while statement
   - for statement

# If Statement

- We naturally take steps based on conditions
  - ***if*** I get CET seat ***then*** I pay Rs. X ***else*** if I am in Management quota ***then*** I pay Rs. Y

- Programs are also written to use such conditions and ***branch*** to a block for execution
- Logical condition is tested which results in either *true* or *false*

# *if* Statement

- In C programming language, this is done with ***if statement***

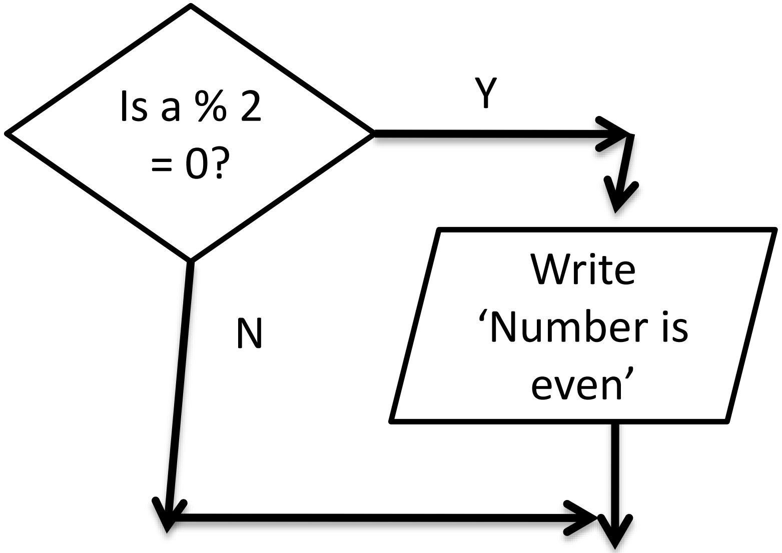
- Single selection statement
  - It selects or ignores a single action

- Algorithm

```
if (<condition>) then
begin
        <statements>
end
```

# *if* Statement - Example

```
if (a%2==0){
    printf(“Number is even”);
}
```



# An Alternative Choice

- What happens if there is an alternative?

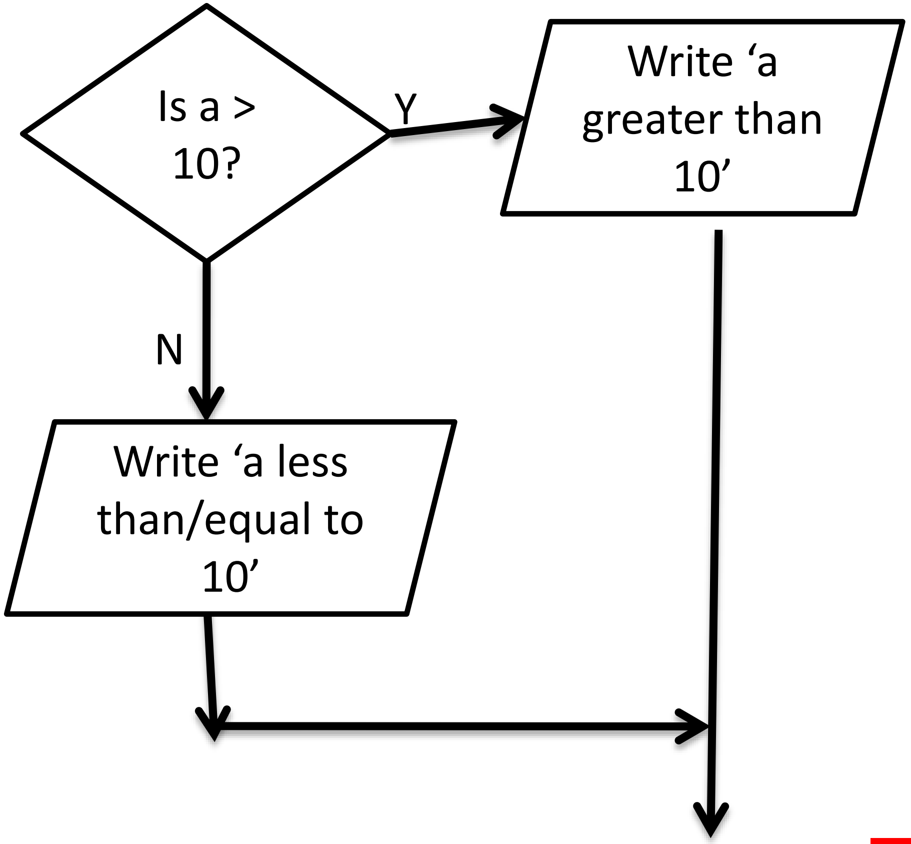
- ***if –else statement***

- double-selection statement
  - selects between two different actions

```
 if (<condition>) then
   begin
        <statements>
   end
else
   begin
        <statements>
   end
```

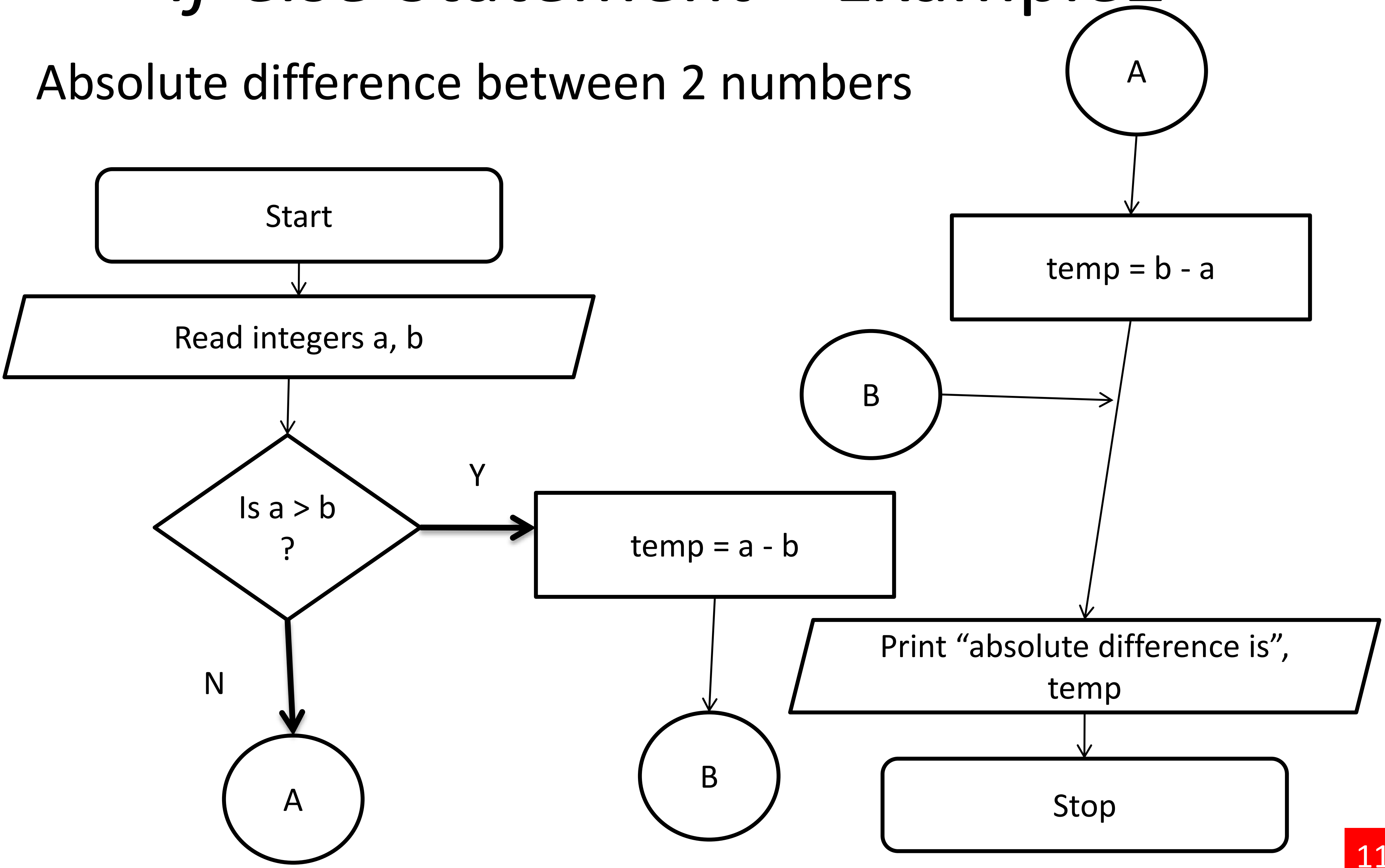
# *if-else* Statement – Example1

```
if (a>10){
     printf(“a is greater than
       10”);
}
else{
   printf(“a is less than or
   equal to 10”);
}
```



# *if-else* Statement – Example2

- Absolute difference between 2 numbers



# *if-else* Statement – Example2 contd.

- Absolute difference between 2 numbers

```
Algorithm absoluteDifference (a, b : Integer)
var temp: Integer;
begin
    if ( a > b ) then
    begin
       temp := a - b;
    end
    else
    begin
       temp := b - a;
    end
    writeln ('The absolute difference is  ', temp);
end
```

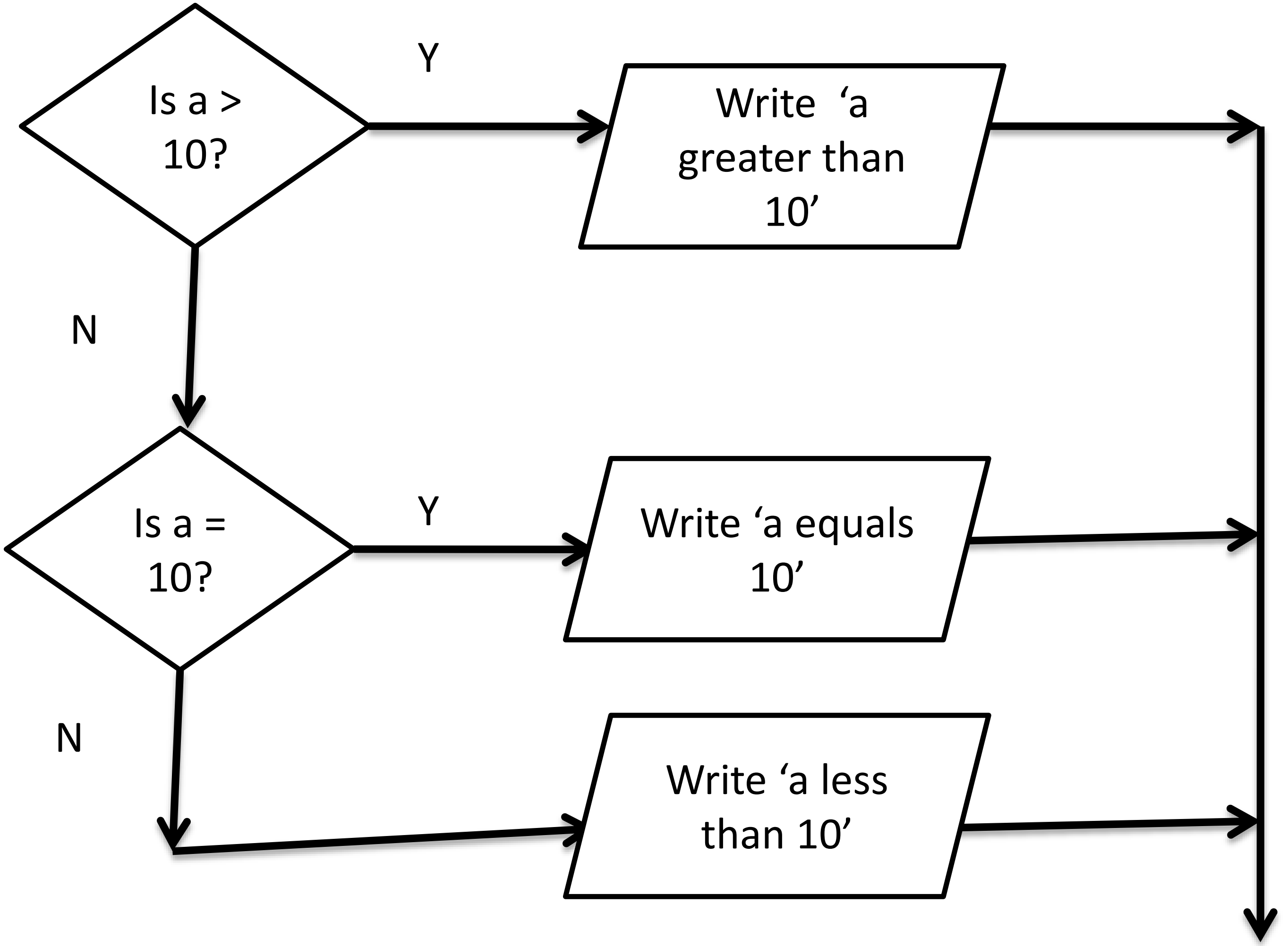
# Multiple Choices – *else if*

- What happens if there are multiple alternatives?
- ***else if ladder***

```
if (a>10){
     printf(“a is greater than 10”);
}
else if (a==10){
   printf(“a is equal to 10”);
}
else{
   printf(“a is less than 10”);
}
```

- Multiple ‘else if’ conditions can be appended to an if statement

# *else if* Ladder - Flow Chart



Is a > 10?

Y

Write 'a greater than 10'

N

Is a = 10?

Y

Write 'a equals 10'

N

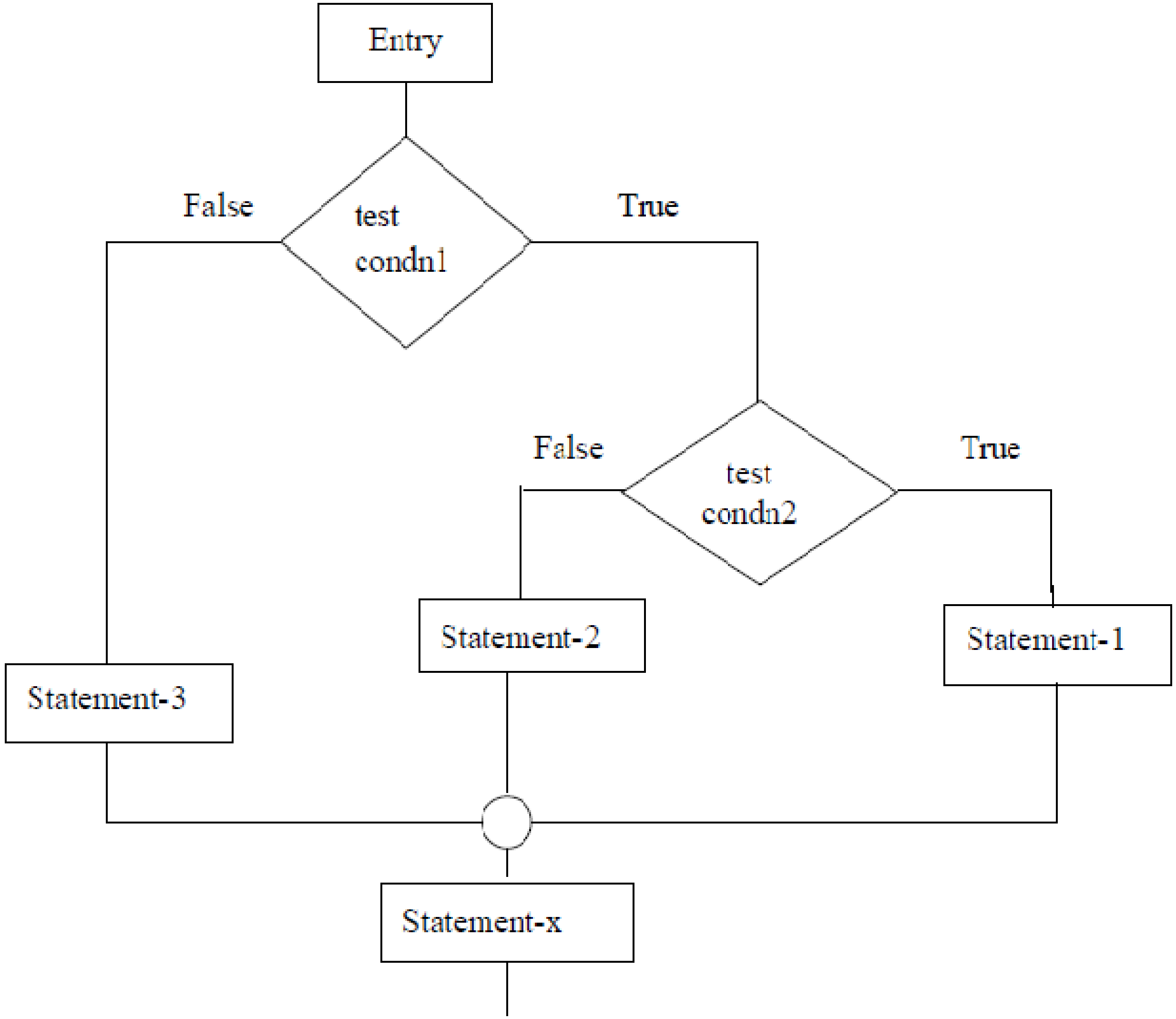
Write 'a less than 10'

# Multiple Choices – *nested if*

- **Nested if** statement

```
if (a>b){
    if(a>c){
        printf(“a is greater than b and c”);
    }
    else{
        printf(“c is greater than a and b”);
    }
}
else{
    printf(“hi..”);
}
```

# *Nested if* - Flowchart



# Many Choices

- What happens if there are too many alternatives?
- ***switch-case statement***

- Multiple-selection statement
  - it selects among many different actions

- Consists of a series of case labels, and an optional default case
  - Can be in any order

# *switch-case* Statement

```
switch (a){
    case 1: printf(“a is 1”);
            break;
    case 2: printf(“a is 2”);
            break;
    case 3: printf(“a is 3”);
            break;
        .....
    default: printf(“a is not matching \n”);
}
```

- Notice ***break***?
  - It stops execution and makes the control flow to move to end of block
- What is ***default***?
  - Break statement is optional for default
- Works only on ***enum, int*** and ***char*** values

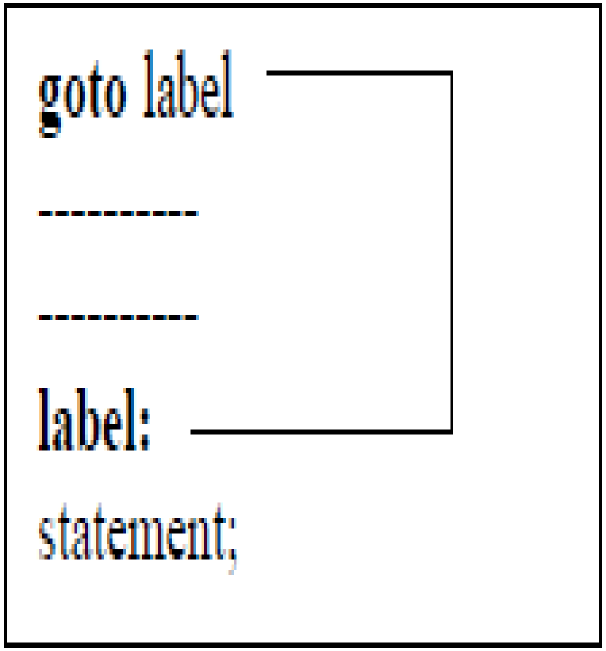
# *goto* Statement

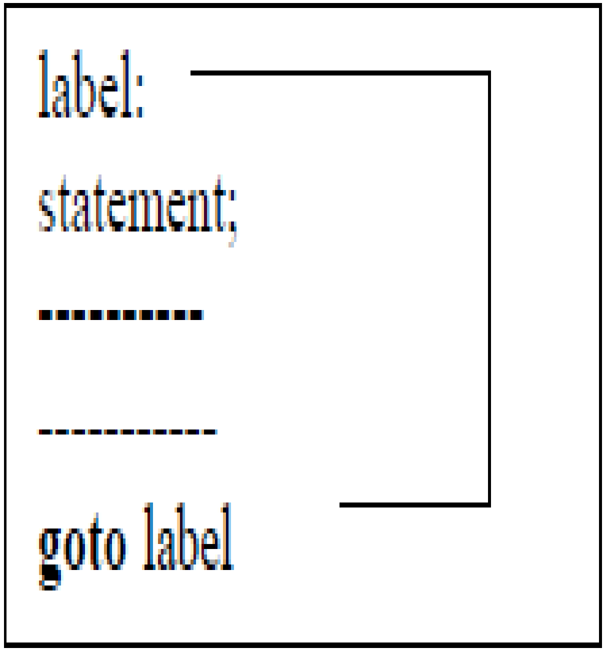
- Unconditional control statement
- Transfer the control from one point to another point in the program
- A branching statement that requires a **label** (valid identifier)
- Syntax

```
goto label;
```

# *goto* Statement contd.

- Label can be placed anywhere in the program





# Summary

- Conditional branching alters the control flow based on a condition
- Control structures that branch based on a condition are
  - If Statements: If, If-Else and Nested If
  - Switch-Case Statement
- All such control structures operate on logical or comparison operators that give true or false values
- Goto statement is an unconditional branch statement

# Further Reading

Dromey, R. (1982) *How To Solve it By Computer.* Noida: Pearson Education Inc.

Kernighan, B. W. and Richie, D. (1992) *The C Programming Language.* 2nd ed., New Delhi:PHI.