if-else Statements

- Also known as branching structures
- Allow program to decide on course of action
  - Based on whether a certain condition is true or false.
- Form:

```
if condition:
    indented block of statements
else:
    indented block of statements
```

- This form is also known as a two-way if statement

Two-way if-else Statements

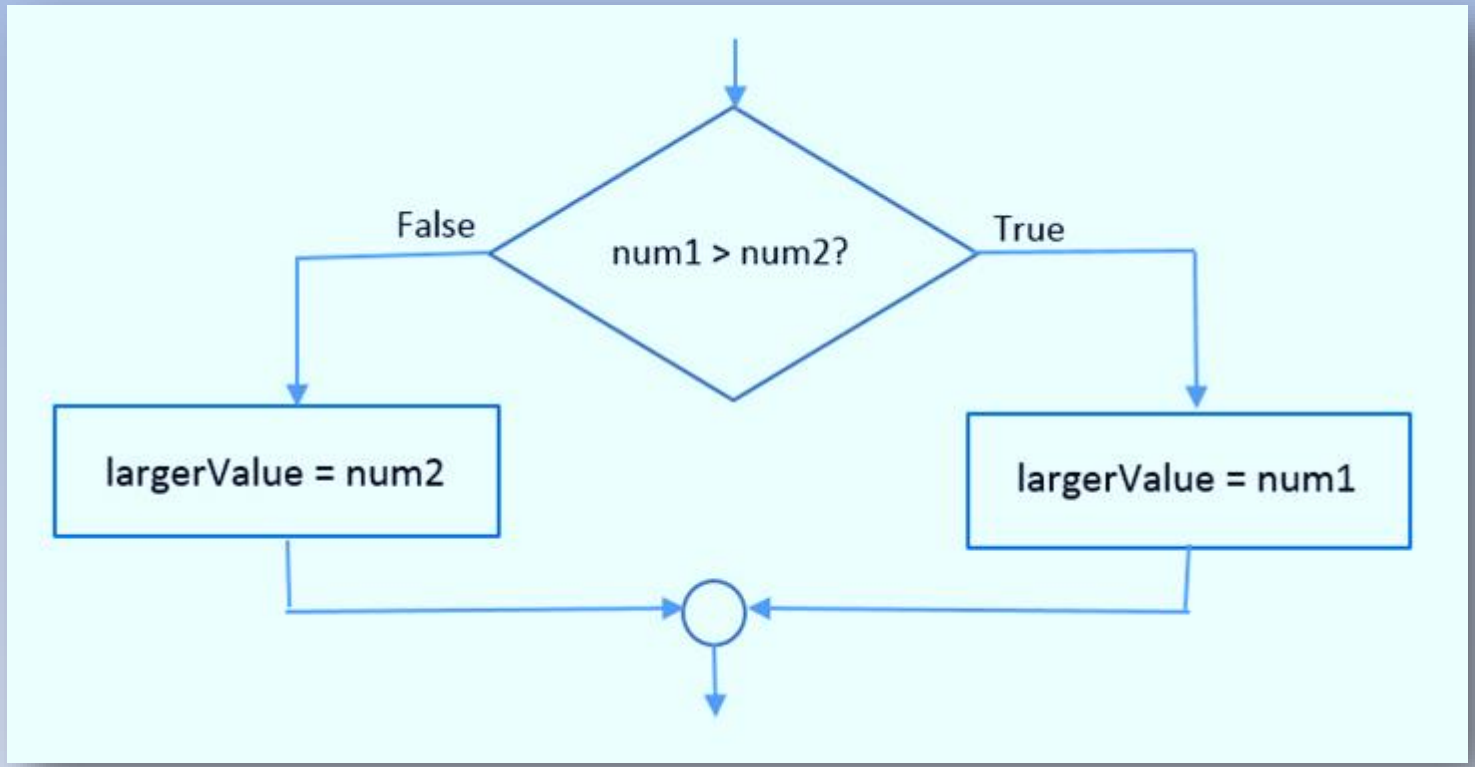
- Example: Program finds larger of two numbers input by user.

```
## Determine the larger of two numbers.
# Obtain the two numbers from the user.
num1 = eval(input("Enter the first number: "))
num2 = eval(input("Enter the second number: "))
# Determine and display the larger value.
if num1 > num2:
    largerValue = num1  # execute this statement if the condition is true
else:
    largerValue = num2  # execute this statement if the condition is false
print("The larger value is", str(largerValue) + ".")
```

[Run]

```
Enter the first number: 3
Enter the second number: 7
The larger value is 7.
```

Two-way if-else Statements



Flowchart for the if-else statement

One-way if Statements

- The *else* part of an *if-else* statement can be omitted
- When the condition is false
  - Execution continues with line after the if statement block

One-way if Statements

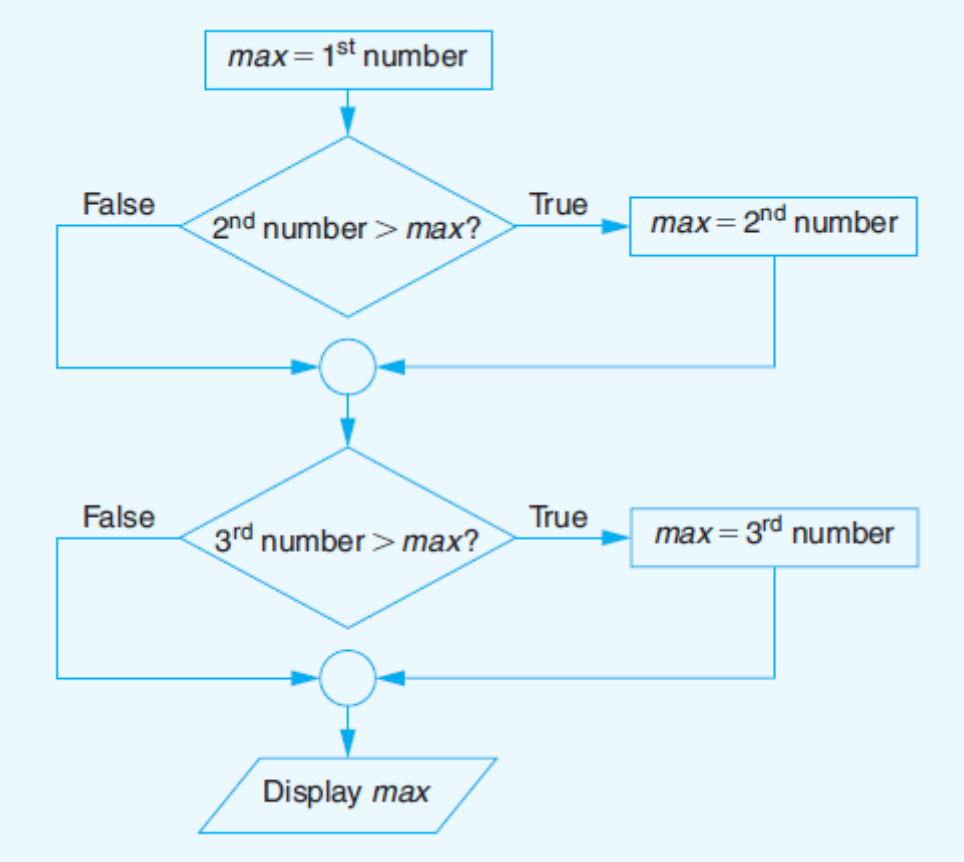
- Example: Program contains two *if* statements

```
## Find the largest of three numbers.
# Input the three numbers.
firstNumber = eval(input("Enter first number: "))
secondNumber = eval(input("Enter second number: "))
thirdNumber = eval(input("Enter third number: "))
# Determine and display the largest value.
max = firstNumber
if secondNumber > max:
    max = secondNumber
if thirdNumber > max:
    max = thirdNumber
print("The largest number is", str(max) + ".")
```

[Run]

```
Enter first number: 3
Enter second number: 7
Enter third number: 4
The largest number is 7.
```

One-way if Statements



Flowchart for multiple if statements

Multi-way `if` Statements

- A program may be faced with testing conditions that entail more than two alternative courses of action

LETTER GRADE	RANGE OF NUMERIC GRADES
A	All grades above 89
B	All grades above 79 and below 90
C	All grades above 69 and below 80
F	All grades below 70

- Can be described in code by a **multi-way selection statement**

Multi-way *if* Statements

```
number = int(input("Enter the numeric grade: "))
if number > 89:
    letter = 'A'
elif number > 79:
    letter = 'B'
elif number > 69:
    letter = 'C'
else:
    letter = 'F'
print("The letter grade is", letter)
```

- Syntax:

```
if condition1:
    indented block of statements to execute if condition1 is true
elif condition2:
    indented block of statements to execute if condition2 is true
    AND condition1 is not true
elif condition3:
    indented block of statements to execute if condition3 is true
    AND both previous conditions are not true
else:
    indented block of statements to execute if none of the above
    conditions are true
```

Nested *if-else* Statements

- Indented blocks of *if-else* and *if* statements can contain other *if-else* and *if* statements
  - The *if-else* statements are said to be nested
- Consider the task of interpreting a beacon
  - The color of the beacon light atop Boston's old John Hancock building forecasts the weather

Steady blue, clear view.
Flashing blue, clouds due.
Steady red, rain ahead.
Flashing red, snow instead.

Nested *if-else* Statements

```
## Interpret weather beacon.
# Get color and mode.
color = input("Enter the color (BLUE or RED): ")
mode = input("Enter the mode (STEADY or FLASHING): ")
color = color.upper()
mode = mode.upper()
result = ""
# Analyze responses and display weather forecast.
if color == "BLUE":
    if mode == "STEADY":
        result = "Clear View."
    else:  # mode is FLASHING
        result = "Clouds Due."
else:      # color is RED
    if mode == "STEADY":
        result = "Rain Ahead."
    else:  # mode is FLASHING
        result = "Snow Ahead."
print("The weather forecast is", result)
```

Example: Warning Signs on Highway

- Display warning signs based on weather conditions
- "temp" variable contains temperature
  - Possible values: 32, Less than 32, Greater than 32
- "dry" variable indicates whether the weather is dry or wet
  - Possible values: 1 for dry, 0 for wet
- Possible messages:
  - One of *Icy Roads* or *Thunderstorms* or *Snow*
  - One of *Wet or Slippery Conditions* or *Do not speed*
  - Always display *Drive with care*