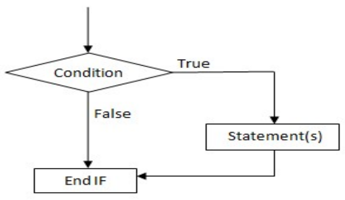
Conditional statements

- ✓ conditional statements are used to control the flow of a program based on certain conditions
- ✓ Conditional statements are also known as decision-making statements.
- ✓ The basic conditional statements are **if, else**, and **elif (short form "else if").**

If statement:

- ✓ The if statement is used to test a condition. If the condition is true, the code inside the if block is executed.
- ✓ **Syntax**

```
if condition:
  # Code to be executed if the condition is true
```



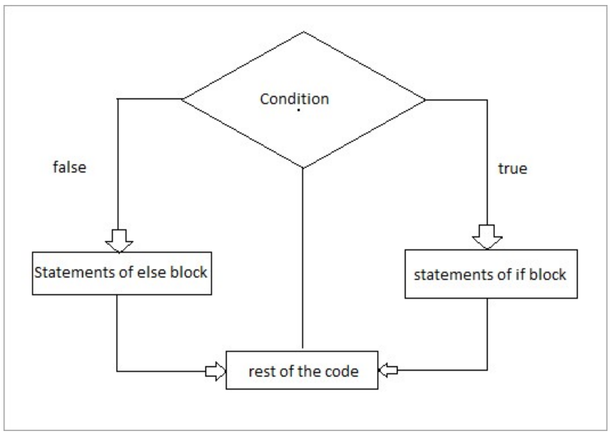
Example:

```
x = 10
if x > 5:
    print("x is greater than 5")
```

If-else statement:

- ✓ The if-else statement allows you to specify two blocks of code: one to be executed if the condition is true, and another if the condition is false.
- ✓ **Syntax:**

```
if condition:
  # Code to be executed if the condition is true
else:
  # Code to be executed if the condition is true
```



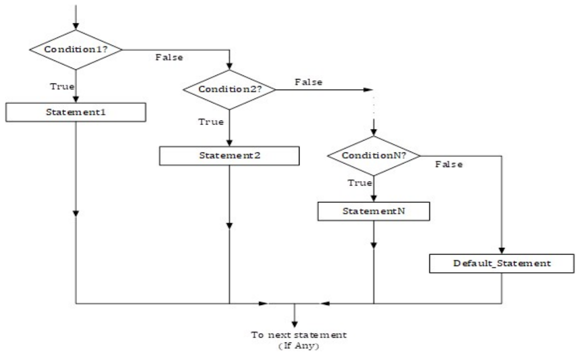
- ✓ **Example**:

```
x = 3
if x > 5:
   print("x is greater than 5")
else:
   print("x is not greater than 5")
```

If-elif-else statement:

- ✓ The if-elif-else statement allows you to test multiple conditions in sequence. The code inside the block associated with the first true condition is executed.
- ✓ **Syntax :**

```
if condition1:
  # Code to be executed if condition1 is true
elif condition2:
  # Code to be executed if condition1 is false and condition2 is true
else:
  # Code to be executed if both condition1 and condition2 are false
```



Example

```
# Example using if-elif-else
score = 85

if score >= 90:
   grade = 'A'
elif score >= 80:
   grade = 'B'
elif score >= 70:
   grade = 'C'
else:
   grade = 'F'
print(f"Your grade is {grade}")
```

Nested-if statements

- ✓ Nested-if statements are nested inside other if statements. That is, a nested-if statement is the body of another if statement.
- ✓ **Syntax :**

```
if condition1:
  # Code to be executed if condition1 is true

  if condition2:
    # Code to be executed if condition2 is true (nested within condition1)

  # More code for condition1 (outside the inner if block)

# Code after the if statement(s) (outside the outer if block))
```

Example

```
x = 10
if x > 5:
    print("x is greater than 5")
    if x > 8:
        print("x is also greater than 8")
    print("This is still inside the outer if block")
print("This is outside the outer if block")
```

Q. Wrie a program to find whether a number iseven or odd.

```
number = int(input("Enter a number: "))

if number % 2 == 0:
   print("even")
else :
   print("odd")
```

Q.Write a program to check the largest among the given three numbers.

```
 # Taking user input for three numbers
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))
```

Q. Checking for the largest number using if statements

```
if num1 >= num2 and num1 >= num3:
   largest = num1
elif num2 >= num1 and num2 >= num3:
   largest = num2
else:
   largest = num3
# Displaying the result
print("The largest number ",largest)
```

Q. Write a Python program to check if the input year is a leap year or not.

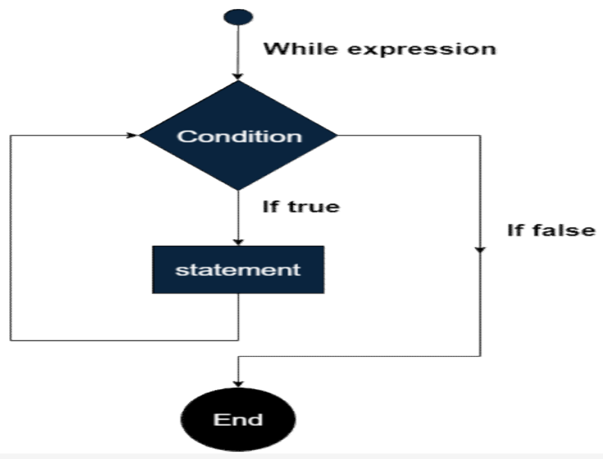
```
 # Taking user input for the year
year = int(input("Enter a year: "))

# Checking if it's a leap year
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
   print(f"{year} is a leap year.")
else:
   print(f"{year} is not a leap year.")
```

Loop

- ✓ A loop is a programming structure that repeats a sequence of instructions until a specific condition is met.
- ✓ A loop statement allows us to execute a statement or group of statements multiple times.
- ✓ Python programming language provides following types of loops to handle looping requirements:
- ✓ **a.While loop b. For loop c. Nested loop**

While loop

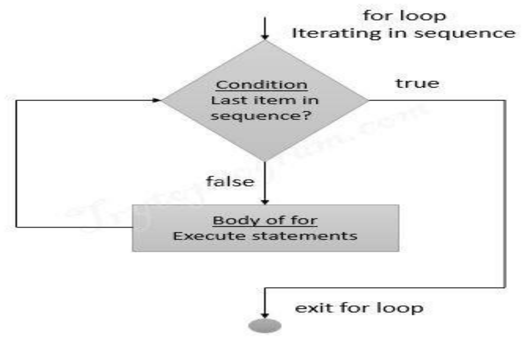
- ✓ The while loop checks the condition.
- ✓ If the condition is True, the loop enters, and the The while loop checks the condition.
- ✓ If the condition is True, the loop enters, and the code in the loop's body is executed.
- ✓ The loop continues to execute the body as long as the condition remains True.
- ✓ When the condition becomes False, the program exits the loop, and any code after the loop will be executed.



```
For example :
        count = 0
        while (count < 9) :
           print 'The count is :', count
           count = count +
```

For loop

- ✓ The for loop iterates over each element in the iterable.
- ✓ For each iteration, the loop executes the code in its body.
- ✓ The loop continues until all elements in the iterable have been processed.
- ✓ After the loop completes, any code after the loop will be executed.



```
For example :
        i = 1
        for i in range(1, 8) :
          print 2*i
```

Nested loop

- ✓ Loop defined within another loop is known as nested loops.
- ✓ 2. Nested loops are the loops that are nested inside an existing loop, that is, nested loops are the body of another loop.
- ✓ **Syntax :**

```
        for condition1 :
           for condition2 :
              #Body of for loop
```

```
For example :
for i in range(1,9,2) :
    for j in range(i) :
     print( i, end = ' ')
    print()
```

Break statement:

✓ The break statement is used to exit a loop prematurely.

```
Example:
for i in range(1, 10):
    if i == 5:
        break
    print(i)
```

Continue statement:

✓ The continue statement is used to skip the rest of the code inside the loop for the current iteration and move on to the next iteration.

Example:

```
for i in range(1, 6):
    if i == 3:
        continue
    print(i)
```

Q. WAP for these patterns :

```
* * * *      *          * * * *        +
* * * *      * *        * * *        + +
* * * *      * * *      * *        + + +
* * * *      * * * *    *        + + + +
```

```
1. for i in range(0, 4):
    # Print asterisks
    for j in range(0,4):
      print("*", end=" ")
    print()
```

```
2.for i in range(0, 4):
    # Print asterisks
    for j in range(0,i+1):
      print("*", end=" ")
    print()
```

```
3.for i in range(0, 4):
    # Print asterisks

    for j in range(i,4 ):
      print("*", end=" ")
    print()
```

```
4.for i in range(0, 4):
    # Print spaces
    for j in range(i, 3):
      print(" ", end=" ")
    # Print plus
    for j in range(0,i+1 ):
      print("+", end=" ")
    print()
```

Q . Write a Python program to print multiplication table and input take from the user

```
# Input from the user
num = int(input("Enter the number for multiplication table: "))

# Print the multiplication table
print(f"Multiplication Table for {num}:")
for i in range(1, 11):
  print(f"{num} x {i} = {num * i}")
```

Q . Write a Python program to print fibonocci series and input take from the user

```
# Input from the user
n = int(input("Enter number of terms for Fibonacci series: "))

prevNum = 0
currNum = 1
print(prevNum, currNum)
for i in range(2, n):
  nextNum = prevNum + currNum
  print(nextNum, end=", ")
  prevNum, currNum = currNum, nextNum
```

Q .Write a Python program to check whether number is armstrong or not .

```
# Input from the user
n = int(input("Enter a number to check for
Armstrong: "))

order = len(str(n))
temp = n
sum = 0
while temp > 0:
    digit = temp % 10
    sum += digit ** order
    temp //= 10

# Check and print the result
if (sum==num):
    print("number is an Armstrong number.")
else:
    print( "number  is not an Armstrong
number.")
```

Q . What will be the output after the following statements ?

```
x = ['P', 'y', 't', 'h', 'o', 'n']
for i in x:
  print(i, end=' ')
```