Python Unit II

if... Statement

The if…elif…else statement is used in Python for decision making.

Syntax

if test expression:

    statement(s)

Here, the program evaluates the test expression and will execute statement(s) only if the text expression is True.If the text expression is False, the statement(s) is not executed.

Python if Statement Flowchart

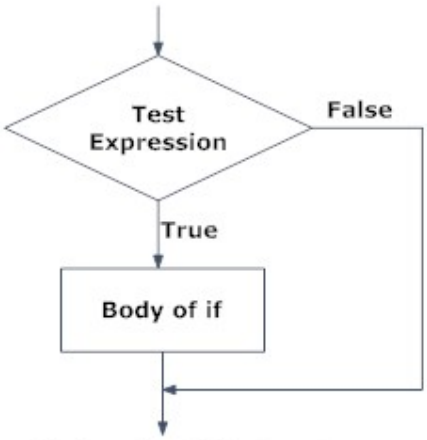

Fig: Operation of if statement

Example

```
# If the number is positive, we print an appropriate message

num = 3

if num > 0:

    print num, "is a positive number."

num = -1

if num > 0:

    print num, "is a positive number."
```

Output

3 is a positive number

Python if else statement

The if..else statement evaluates test expression and will execute body of if only when test condition is True.If the condition is False, body of else is executed. Indentation is used to separate the blocks.

Syntax

```
if test expression:
        Body of if
else:
        Body of else
```

Python if..else Flowchart

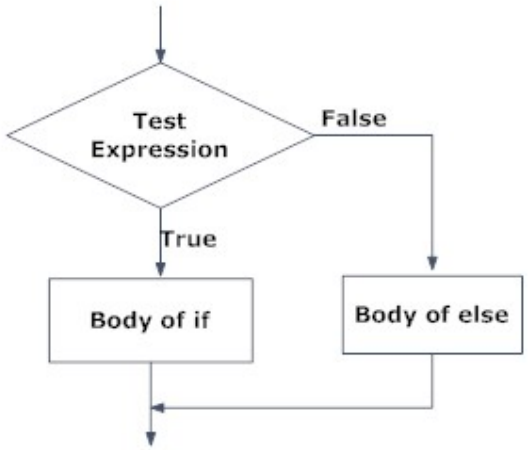

Fig: Operation of if...else statement

Example

```
#Python Program to Check if a Number is Odd or Even

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

if (num % 2) == 0:
        print("{0} is Even".format(num))
else:
        print("{0} is Odd".format(num))
```

Python if...elif...else

The elif is short for else if. It allows us to check for multiple expressions. If the condition for if is False, it checks the condition of the next elif block and so on.

If all the conditions are False, body of else is executed. Only one block among the several if...elif...else blocks is executed according to the condition. The if block can have only one else block. But it can have multiple elif blocks.

Syntax

if test expression:

    Body of if

elif test expression:

    Body of elif

else:

    Body of else

Flowchart of if...elif...else

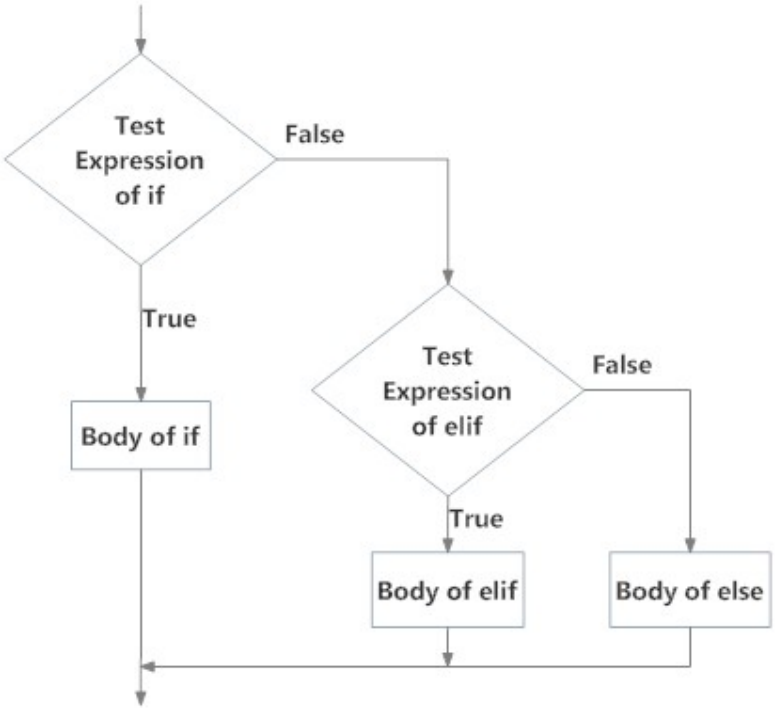

Fig: Operation of if...elif...else statement

Example

#Write a program to display calculator operations(addition, subtraction, #multiplication and division) for taking any two numbers thorough the #keyboard.

print("1: ADDITION")

print("2: SUBTRACTION")

```
print("3: MULTIPLICATION")

print("4: DIVISION")


CHOICE = raw_input("Enter the Numbers:")


if CHOICE == "1":

        a = (input("Enter the value of a:"))

        b = (input("Enter the value of b:"))

        c = a + b

        print c


elif CHOICE == "2":

        a = (input("Enter the value of a:"))

        b = (input("Enter the value of b:"))

        c = a - b

        print c


elif CHOICE == "3":

        a = (input("Enter the value of a:"))

        b = (input("Enter the value of b:"))

        c = a * b

        print c


elif CHOICE == "4":

        a = float(input("Enter the value of a:"))

        b = float(input("Enter the value of b:"))

        c = float(a / b)

        print c


else:

        print "Invalid Number"
```

```
print "\n"
```

Example

```
# Python program to check if the input year is a leap year or not

year = input("Enter a year: ")


if (year % 4) == 0:
    if (year % 100) == 0:
        if (year % 400) == 0:
            print("{0} is a leap year".format(year))
        else:
            print("{0} is not a leap year".format(year))
    else:
    print("{0} is a leap year".format(year))
else:
print("{0} is not a leap year".format(year))
```

Exercise

1. Write a program to display for your name?(take input from keyboard)
2. Test if a number is a multiple of 3,5 or 7.
3. Write a program to check which grade the student got marks. If student A marks=300, student B Marks=450 and Student C Marks=550. (Grades 550 marks==A grade, 450 marks==B grade, 300 marks==C grade)
4. Write a Python program that requests an integer value from the user. If the value is between 1 and 100 clusive, print ”OK;” otherwise, do not print anything.
5. Write an if statement that checks if the variable a is equal to 1. If it is equal to 1, print a message saying ‘a equals 1’, else print ‘a is not equal to 1’.
6. Write a program that asks the user for a number in the range of 1 through 7. The program should display the corresponding day of the week, where 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday, and 7 = Sunday. The program should display an error message if the user enters a number that is outside the range of 1 through 7.

7. Write a Python program to get the difference between a given number and 17, if the number is greater than 17 return double the absolute difference.

8. Write a Python program to calculate the sum of three given numbers, if the values are equal then return thrice of their sum.

9. Write a Python program to sum of three given integers. However, if two values are equal sum will be zero.

10. Write a Python program to sum of two given integers. However, if the sum is between 15 to 20 it will print 20.

while Loop

The while loop in Python is used to iterate over a block of code as long as the test expression (condition) is true. Or A **while** loop statement in Python programming language repeatedly executes a target statement as long as a given condition is true.

Syntax

while expression:

    Body of while

In while loop, test expression is checked first. The body of the loop is entered only if the test_expression evaluates to True. After one iteration, the test expression is checked again. This process continues until the test_expression evaluates to False.

In Python, the body of the while loop is determined through indentation. Body starts with indentation and the first unindented line marks the end. Python interprets any non-zero value as True. None and 0 are interpreted as False.

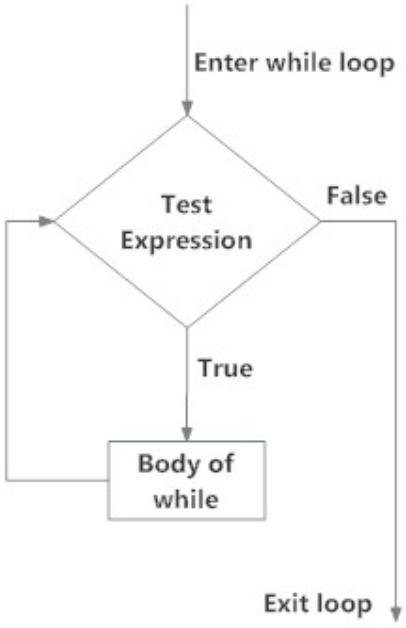

Fig: operation of while loop

Example 1

```
# Program to display the Fibonacci sequence up to n-th term where n is provided by the user

# change this value for a different result

nterms = 10


# uncomment to take input from the user

#nterms = int(input("How many terms? "))


# first two terms

n1 = 0

n2 = 1

count = 0


# check if the number of terms is valid

if nterms <= 0:

        print("Please enter a positive integer")

elif nterms == 1:

         print("Fibonacci sequence upto",nterms,":")
```

```
  print(n1)
else:
  print("Fibonacci sequence upto",nterms,":")
  while count < nterms:
     print(n1)
     nth = n1 + n2
     # update values
     n1 = n2
     n2 = nth
     count += 1
```

Example 2

```
# Python program to check if the number provided by the user is an Armstrong number or not

# take input from the user
num = int(input("Enter a number: "))

# initialize sum
addition = 0

# find the sum of the cube of each digit
temp = num
while temp > 0:
        digit = temp % 10
        addition += digit ** 3
        temp /= 10

# display the result
if num == addition:
        print(num,"is an Armstrong number")
```

```
else:
        print(num,"is not an Armstrong number")
```

Exercise

1. Write a program to print your name 10 times.

```
name="rajani"
i=0
while i<10:
        print name
        i=i+1
```

2. Write a program to print numbers from 10 to 0.

```
num=int(raw_input("Enter a valid integer: "))
#num=10
while(num>=0):
        print num
        num=num-1
```

3. Write a program to print all numbers from x to y using while loop.

```
num1=int(raw_input("Enter first Number: "))
num2=int(raw_input("Enter second Number: "))
if(num1>num2):
        while(num1>=num2):
        print num1
        num1=num1-1
elif(num1<num2):
        while(num1<=num2):
        print num1
        num1=num1+1
```

4. Write a program to print all even numbers below 20 like 2,4,6,.....18,20.(Use if statement before print statement)

```
num=1
while(num<=20):
        if(num%2==0):
```

```
        print num
    num=num+1
```

5. Write a program to print all odd numbers below x.

```
num=int(raw_input("Enter a num to get odd num:  "))
x=1
while(x<=num):
        if(x%2!=0):
        print x
    x=x+1
```

6. Write a program to count all even numbers in between x and y.(read x and y from user).

```
count=0
num1=int(raw_input("Enter first Number: "))
num2=int(raw_input("Enter second Number: "))
if(num1>num2):
        while(num1>=num2):
                if(num1%2==0):
                        count=count+1
                num1=num1-1
        print count
elif(num2>num1):
while(num2>=num1):
if(num2%2==0):
        count=count+1
        num2=num2-1
print count
```

Exercises:

1. Write a python program to reverse a given number from the user input.
2. Write a python program to print following format.
   1
   22
   333
   4444

55555
666666
7777777
88888888
9999999999

3. Write a Python program to create the multiplication table (from 1 to 10) of a number.
4. Write a python program to check given number is palindrome number or not.
5. Write a python program to print following format.

```
*
**
***
****
*****
****
***
**
*
```

For Loop

The for loop in Python is used to iterate over a sequence (list, tuple, string) or other iterable objects. Iterating over a sequence is called traversal.

Syntax

```
for val in sequence:
    Body of for
```

Here, val is the variable that takes the value of the item inside the sequence on each iteration. Loop continues until we reach the last item in the sequence. The body of for loop is separated from the rest of the code using indentation.

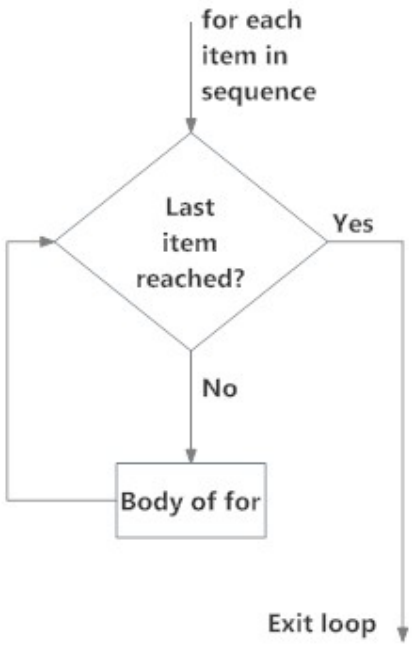

Fig: operation of for loop

Example

Python Program to Find the Factorial of a Number

```
num = 7

# uncomment to take input from the user

#num = int(input("Enter a number: "))

factorial = 1

# check if the number is negative, positive or zero

if num < 0:

print("Sorry, factorial does not exist for negative numbers")

elif num == 0:

print("The factorial of 0 is 1")

else:

        for i in range(1,num + 1):

                factorial = factorial*i

        print("The factorial of",num,"is",factorial)
```

Exercise

1. write a program to print all numbers below the given number 'x'.

```
num=int(raw_input("Enter a number: "))
```

```
for x in range(0,num):
        print (x)
```

2. Write a program to print all the multiples 3, 5 and 7 below 100.

```
for i in range(1,10):
if(i%3==0 and i%5==0 and i%7==0):
            print i
```

3. Write a python program read the elements in sequence.

```
for counter in 1,2,"RGUKT", 3,"IIIT",4:
        print counter
```

The range() function

We can generate a sequence of numbers using range() function. range(10) will generate numbers from 0 to 9 (10 numbers).

We can also define the start, stop and step size as range(start,stop,step size). step size defaults to 1 if not provided.

This function does not store all the values in memory, it would be inefficient. So it remembers the start, stop, step size and generates the next number on the go.To force this function to output all the items, we can use the function list().

Example

```
# Output: range(0, 10)
print(range(10))
# Output: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
print(list(range(10)))
# Output: [2, 3, 4, 5, 6, 7]
print(list(range(2, 8)))
# Output: [2, 5, 8, 11, 14, 17]
print(list(range(2, 20, 3)))
```

Example2

```
# Program to iterate through a list using indexing
genre = ['pop', 'rock', 'jazz']
# iterate over the list using index
for i in range(len(genre)):
        print("I like", genre[i])
```

Exercises:

1. Write a Python program that accepts a word from the user and reverse it.

2. Write a Python program to count the number of even and odd numbers from a series of numbers.
   numbers= (1, 2, 3, 4, 5, 6, 7, 8, 9)
   output:
   Number of even numbers : 5
   Number of odd numbers : 4

3. Write a Python program to count the number 4 in a given list.

4. Write a Python program to calculate the sum of the digits in an integer.(using while)

break statement

The break statement terminates the loop containing it. Control of the program flows to the statement immediately after the body of the loop.

If break statement is inside a nested loop (loop inside another loop), break will terminate the innermost loop.

Syntax

Break

Example

Python Program to Check Prime Number

```
# Python program to check if the input number is prime or not


num = 407


# take input from the user

# num = int(input("Enter a number: "))


# prime numbers are greater than 1

if num > 1:

  # check for factors

  for i in range(2,num):

        if (num % i) == 0:

              print(num,"is not a prime number")

               print(i,"times",num//i,"is",num)

              break
```

```
else:
    print(num,"is a prime number")


# if input number is less than
# or equal to 1, it is not prime
else:
  print(num,"is not a prime number")
```

Python continue statement

The continue statement is used to skip the rest of the code inside a loop for the current iteration only. Loop does not terminate but continues on with the next iteration.

Syntax

```
Continue
```

Example

```
# Program to show the use of continue statement inside loops
for val in "string":
if val == "i":
continue
print(val)
print("The end")
```

Exercise

1. Write a program to print n numbers and break the program when n==10.

```
n=input("Enter the value")
for n in range(0,n):
if(n==10):
break
print n
```

2. Write a program to print the n numbers and use the continue statement to when n=10.(Note n should be more than 10)

```
n=input("Enter the value:  ")
```

```
if(n>=10):
        for n in range(0,n):
        print n
elif(n<10):
        for n in range(0,n):
                continue
        print "Continue is used"
```

pass statement

In Python programming, pass is a null statement. The difference between a comment and pass statement in Python is that, while the interpreter ignores a comment entirely, pass is not ignored.

However, nothing happens when pass is executed. It results into no operation (NOP).We generally use it as a placeholder.

Suppose we have a loop or a function that is not implemented yet, but we want to implement it in the future. They cannot have an empty body. The interpreter would complain. So, we use the pass statement to construct a body that does nothing.

Example

```
# pass is just a placeholder for
# functionality to be added later.
sequence = {'p', 'a', 's', 's'}
for val in sequence:
pass
```