Chapter 5 "Conditional and Iterative Statements"

Introduction:-

- **Generally a program executes its statements from beginning to end . But not many programs execute all their statements in strict order from beginning to end.**
- **Programs depending upon the need can choose to execute one of the available alternatives or even repeat a set of statements .**
- **To perform manipulative miracles programs need tools for performing repetitive actions and for making decisions.**
- **In python tools are available called program control statements.**
- **Selection statement if and iteration statements for and while.**

Statement

- **Statements are the instructions given to the computer to perform any kind of action(include data movements , making decisions or repeating actions)**
- Type of Statement
- Empty Statement
- Simple Statement
- Compound Statement

Empty Statement

Empty statement of python is a do nothing statement i.e empty statement or null operation statement.

Pass statement useful in those instances where the syntax of program requires the presence of a statement but where the logic of the program does not.

```
for letter in 'Python' :
    if letter=='h':
        pass
        print'use of pass'
    print'current  letter',
```

```
current letter P
current letter y
current letter t
this is pass block
current letter h
current letter o
```

- **Simple Statement**

  ✓ Any single executable statement is a simple statement in Python.
  Eg simple Statement in Python
  >>> name=input("Your name")
  >>> print(name)
  ✓ Simple statement are single line statements.

- **Compound Statement**
  ✓ **A compound statement represents a group of statements executed as a unit.**
  ✓ **A compound statement in Python has a header ending with a colon(:) and a body con-taining a sequence of statements at the same level of indentation.**
  ✓ **Eg**

<compound statement header >:

<indented body containing multiple simple and /or compound statements>

Compound statement has
- **A header line which begins with a keyword and ends with a colon.**
- **A body consisting of one or more Python statements, each indented inside the header line.**
  All statements in the body are at the same level of indentation.

PROGRAMMING CONSTRUCTS

- In a program, statements may be executed sequentially, selectively or iteratively.
- Every programming language provides constructs to support sequence, selection or iteration.

Sequence- The sequence construct means the statements are being executed sequentially.

- ✓ This represents the default flow of statement.

Selection-

- ✓ The selection construct means the execution of statements depending upon a condition-test.
- ✓ If a condition evaluates to true, a course-of-action(a set of statements) is followed otherwise another course-of-action(a different set of statement) is followed.
- ✓ It is also called a decision construct.
- ✓ Java provides two types of selection statements : **if** and **switch.**
- ✓ **if** statement tests a particular condition ; if the condition evaluates to true, a course-of-action is followed i.e , a statement or set-of-statements is executed. Otherwise (if the condition evaluates to false), the course-of-action is ignored.
- ✓ **Syntax** if (expression)
  Statement;
- ✓ **If ...else Statement –**In an if-else statement, only the code associated with if(i.e, statement-1) or the code associated with else (i.e, statement-2) executes, never both.
- ✓ **Switch Statement** –It is a multiple – branch selection statement .
- ✓ This selection statement .successively tests the value of an expression against a list of integer or character constants.. When a match is found, the statements associated with that constant are executed.
- ✓ The data type of expression in a switch must be **byte, char, short or int**

Iteration Statements

- ✓ The iteration statements allow a set of instructions to be performed repeatedly until a certain condition is fulfilled.
- ✓ The iteration statements are also called loops or looping statements.
- ✓ Java provides three kinds of loops: **for** loop, **while** loop, and **do-while loop.**
- ✓ For all three loop statements, a true condition is the one that returns Boolean true value and the false condition is the one that returns Boolean false value.
- ✓ Element of control loop are Initialization Expression(s), Test Expression, Update Expression(s), the body –of-the loop.

Logic development tools:--

- **Before developing the solution of a problem in terms of a program, we should read and analyze the given problem and decide about basic sub-tasks needed to solve a problem and the order of these subtasks.**

- **Algorithm:- "An algorithm is a step –by-step procedure (well-defined instructions) to solve a given problem.**

Eg The algorithm to find the remainder of given two numbers is :

- ✓ **Input first number**
- ✓ **Input second number**
- ✓ **Divide first number with second number and store the remainder as third number.**
- ✓ **Display**

- **An algorithm is a set of ordered and finite steps (the subtasks) to solve a given problem.**
  - ✓ **Eg 2 (using the same logic determine if the first number is divided by second number or not)**
    - ➢ **Input** first number
    - ➢ **Input** second number
    - ➢ **Divide** first number with second number and store the **remainder** in third number.
    - ➢ **Check** if the **third number is 0.**

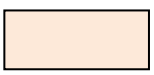
      (a) If Yes , Display 'the first number IS divisible by second number'.
      (b) If No, Display' the first number IS NOT divisible by second number'.

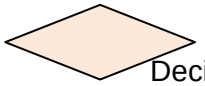
- **Algorithms Tools are :---(i) pseudocode , flow charts , or decision trees and tables.**

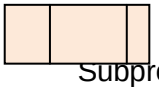
Note:-----In syllabus only flowcharts we have to know

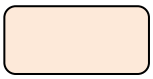
Flowcharts

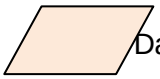
- A flowchart is a graphical representation of steps an algorithm to solve a given problem.
- **Flowchart symbol**

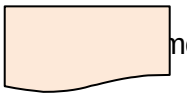

Process


Decision


Subprocess


Start/End


Data


ment

- Use Data symbol for Input/Output (I/O) operation(taking input and showing output).
- Use Process symbol for any type of computation and internal operations like initialization, calculation etc.
- Use Subprocess symbol to invoke a procedure written already.

The if STATEMENTS OF PYTHON

Simple IF

- The if statements are the conditional statements in Python and these implement selection constructs (decision constructs).
- Eg

```
ch=input('Enter a single character:')
if ch==' ' :
   print("You entered a space")
if ch>='0' and ch<='9':
   print("you entered a digit.")
```

The if –else statement

- In this the block below **if** gets executed **if** the condition evaluates to true and the block below **else** gets executed if the condition evaluates to false.
- Eg

```
num=int (input("Enter an integer:"))
if num%2==0:
   print(num,"is EVEN number.")

else:

  print(num,"is ODD number.")
```

```
sum1= sum=2=0

num1=int(input("Enter number 1:"))

num2=int(input("Enter number 2:"))

num3=int(input("Enter number 3:"))

sum1=num1+num2+num3

if num1!=num2 and num1!=num3:

sum2+=num1

if num1!=num1 and num2!=num3:
```

```
sum2+=num2

if num3!=num1 and num3!=num2:

sum2+=num3

print("Numbers are",num1,num2,num3)

print("Sum of three given number is ",sum1)

print("Sum of  non-duplicate number is ",sum2)
```

The if –elif STATEMENTS OF PYTHON

The if –elif statement

- **if , elif and else** all are block or compound statements.
- **Sometimes, we need to check another condition in case the test –condition of if evaluates to false i.e we want to check a condition when control reaches else part , i.e condition test in the form of else if .**
- Eg

```
If  runs are more than 100

        then it is a century

else if runs are more than 50

        then it is a fifty
```

```
num1= float(input("Enter first number :"))
num2=float(input("Enter second number:"))
op=input("Enter operator[+-*/%]:")
result=0
if op=='+'
    result=num1+num2
elif op=='_':
  result=num1-num2
elif op=='*':
      result=num1* num2
elif op=='/':
   result=num1/num2

elif op=='%':

    result=num1% num2

else:

    print("Invalid operator!!")

print(num1, op, num2,'=',result)
```

```
Enter first  number:5

Enter second number:2

Enter operator[+  -  * /  %]:*

5.0 * 2.0=10.0

==========RESTART==============

Enter first number       : 5
```

```
x=int(input("Enter first number:"))
y=int(input("Enter second number:"))
z=int( input("Enter third number:"))
min=mid=max=None
if x<y and x< z:
   if y<z:
      min, mid, max=x,y,z
   else:
     min, mid, max=x, z, y

elif y<x and y<z:

   if x<z:

      min, mid, max=y,x,z

   else:

      min ,mid, max=y,z,x

   else:

      if x< y:

      min, mid, max=z,x, y

      else:

      min, mid, max=z,y,x

print("Numbers in ascending order:", min, mid, max)
```

```
Enter first number       :5

Enter second number  :9

Enter third number       :2

Numbers in ascending order:2    5    9

==========RESTART===============

Enter first number       :9
```

Statement	Values generated
range(10)	**0,1,2,3,4,5,6,7,8,9**
range(5,10)	**5,6,7,8,9**
range(3,7)	**3,4,5,6**
range(5,15,3)	**5,8,11,14**
range(9,3,-1)	**9,8,7,6,5,4**
range(10,1,-2)	**10,8,6,4,2**

Operators **in** and **not in**

- The **in** operator tests if a given value is contained in a sequence or not and returns True or False accordingly.
- In operator used with range( ) in for loops.
  Eg

  3 in[1,2,3,4] ---this expression will test if value 3 is contained in the given sequence
  5 in [1,2,3,4] will return False as value 5 is not contained in sequence[1,2,3,4]
  5 not in [1,2,3,4] will return True as value 5 is not contained in sequence[1,2,3,4]

✓ Operator in and not in are also called membership operators.

- Iteration /Looping Statements(for loop)
  - The iteration statement or repetition statements allow a set of instructions to be performed repeatedly until a certain condition is fulfilled. The iteration statements are also called loops or looping statements.
  - Python provides two kinds of loops: for loop and while loop to represent two categories of loops
    - **Counting loops: The loops that repeat a certain number of items; Python's for loop is a counting loop.**
    - **Conditional loops:-The loops that repeat until a certain thing happens i.e they keep repeating as long as some condition is true ; Python's while loop is conditional loop.**

The For Loop

- **The for loop of Python is designed to process the items of any sequence , such as a list or a string one by one.**
- **The general form of for loop**

This is the body of the for loop. All statements in the body of the loop will be executed for each value of loop variable a , i.e firstly for a=1; then for a =4 and then for a=7

Eg for a in [1,4,7]:

print(a)

print(a*a)

The loop variable a . Variable a will be assigned each value of list one by one, i.e for the first time a will be 1, then 4 and then

Iteration :--- Each time , when the loop –body is executed is called an iteration.

A for loop in Python is processed as :

- ✓ **The loop-variable is assigned the first value in the sequence .**
- ✓ **All the statements in the body of for loop are executed with assigned value of loop variable(step 2)**
- ✓ **Once step 2 is over , the loop –variable is assigned the next value in the sequence and the loop-body is executed (i.e step 2 repeated) with the new value of loop –variable.**
- ✓ **This continues until all values in the sequences are processed.**

```
Out put

1

1

4

16
```

Program to print table of a number (5)

```
Num=5

for a in range(1,11):

    print(num,'x',a,'=',num*a)
```

The above code will print the output as shown here:

```
5 x1=5

5x2=10

5 x3=5

5x4=10

5 x5=5

5x6=10

5 x7=5
```

```
sum=0

for n in range(1,8):

    sum+=n

    print("Sum of natural numbers <=",n,"is',sum)
```

```
Output

Sum of natural numbers<=1 is 1

Sum of natural numbers<=2 is 3

Sum of natural numbers<=3 is 6

Sum of natural numbers<=4 is 10

Sum of natural numbers<=5 is 15

Sum of natural numbers<=6 is 21

Sum of natural numbers<=7 is 28
```

The while Loop

- ✓ A while loop is a conditional loop that will repeat the instructions within itself as long as a conditional remains true(Boolean True or truth value true)
- ✓ While<logical expression>:
  loop-body
  - ❖ where the loop –body may contain a single statement or multiple statement or an empty statement (i.e pass statement).
  - ❖ The loop iterates while the logical expression evaluates to true. When the expression becomes false, the program control to the line after the loop –body.

```
a=5

while a>0:

  print("hello",a)
```

```
Output:

Hello 5

Hello 2
```

Jump Statements(break and continue statement)" Jump Statements(break and continue statement)"

- ❖ Python offers two jump statements to be used within loops to jump out of loop –iterations .
- ❖ These are break and continue statements.

Break statement

- ❖ The break statement enables a program to skip over a part of the code .
- ❖ A break statement terminates the very loop it lies within . Execution resumes at the statement immediately following the body of the terminated statement.

Continue Statement

- ▪ The continue statement is another jump statement like the break statement as both the statements skip over a part of the code. But the continue statement is somewhat different from break .
- ▪ Instead of forcing termination , the continue statement forces the next iteration of the loop to take place, skipping any code in between.