CHAPTER -4 CONDITIONAL AND ITERATIVE STATEMENTS IN PYTHON
PART- 1

Bimlendu Kumar

PGT Computer Sc.

Kendriya Vidyalaya Garhara

STATEMENTS IN PYTHON

In Python statements are of following types:

1. Empty Statement
2. Simple Statement
3. Compound Statement

1. Empty Statement: Empty statement is the simplest statement. This is a statement which does nothing. In Python an empty statement is pass statement.

Pass statement of Python is a do nothing statement i.e. empty statement or null operation statement.

Whenever python encounters a pass statement Python does nothing and simply moves to the next statement. It is useful where the syntax of the language is requires presence of a statement but there is no any specific logic. Example delay loop.

STATEMENTS IN PYTHON

2. **Simple Statement:** Any single executable statement is a simple statement in Python.

   Example

   >>>name=input("Enter your name please:- ")

   >>>print("My name is ",name)

3. **Compound Statement:** A compound statement in Python has a header ending with a colon(:) and a body containing a sequence of statements at the same level of indentation.

   <compound statement header>:

   <Indented body containing multiple simple / and or compound statement>

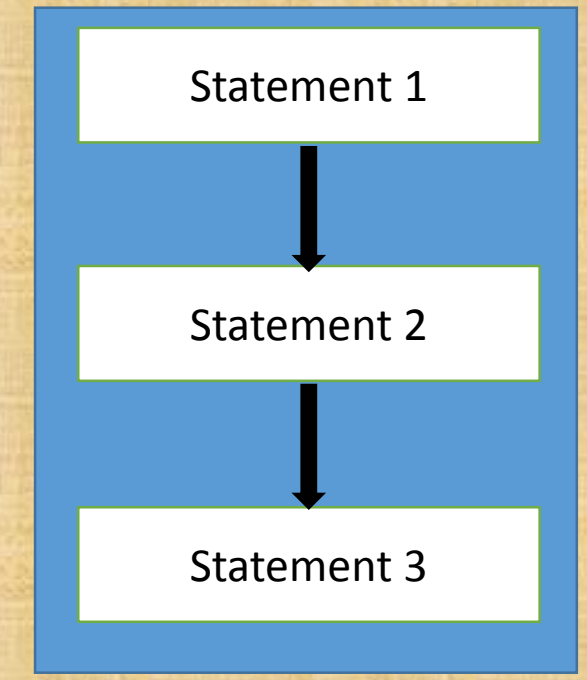
   Header Line: It begins with a keyword and ends with a colon

   Body: Consists of one or more Python statement each indented inside the header line. All statements of the body are at same level of indentation.

STATEMENTS IN PYTHON

4. FLOW CONTROL STATEMENT:- statements may execute sequentially, selectively or iteratively.

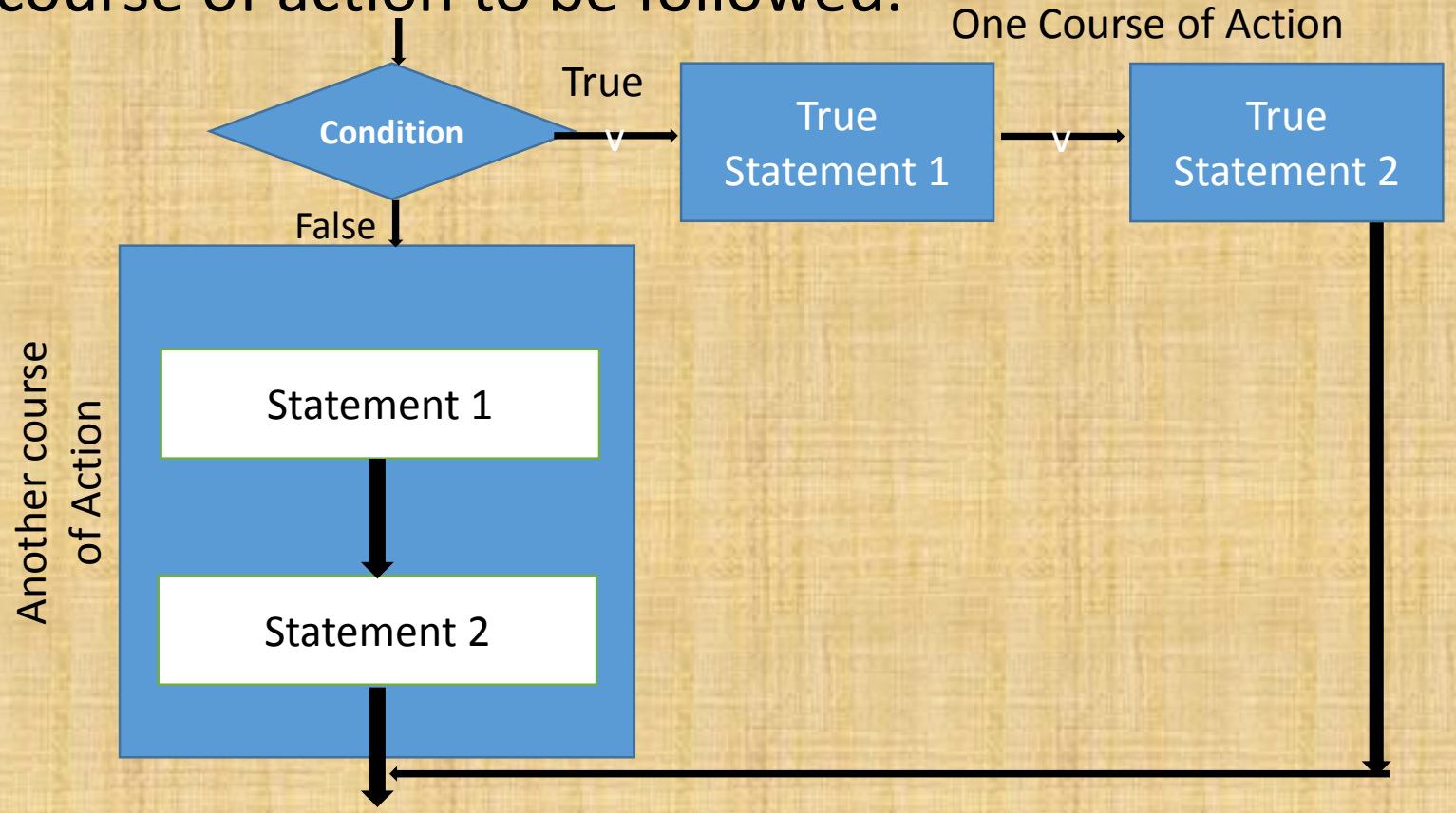
(A) Sequence: The sequence means statements are being executed one by one in the order in which it appears in the source code. This is the default flow of control.



Sequence represent normal flow of control and is the simplest one.

STATEMENTS IN PYTHON

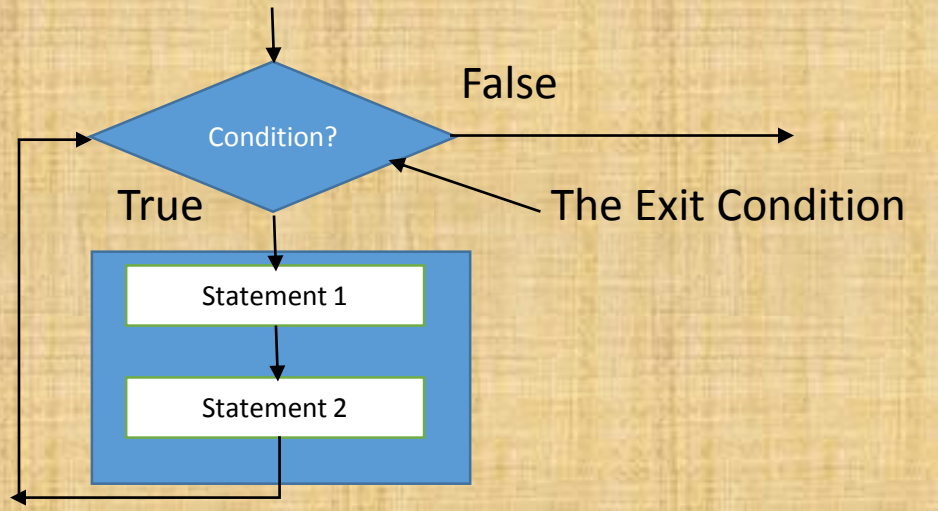
(B) Selection: The selection construct means the execution of statement(s) depending upon a condition testing. If a condition evaluates to True, a course-of-action is followed otherwise another course of action is followed. The construct selection is also called decision construct because it helps in making decision about which course of action to be followed.



STATEMENTS IN PYTHON

(C) **Iteration(Looping):** Iteration constructs are meant for repetition of one or more statements depending upon a condition. Till the time a condition is True a set of statements are executed again and again. As soon as the condition becomes false the repetition stops.

The iteration construct is also called looping construct.



PROGRAM LOGIC DEVELOPMENT TOOLS

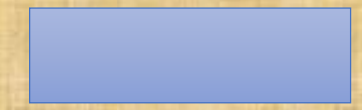
Before developing a Program for a given problem we need to do some basic tasks. Following tasks are involved in the given sequence to solve the problem

1. Algorithm: *Step by step procedure to solve a specific problem is called algorithm. An algorithm must have following properties*
   (a) Finiteness: Must be finished in countable number of statement
   (b) Preciseness: A statement is mandatorily required then only it should be used in algorithm.
   (c) Unambiguous: Meaning of every statement must be clear and it must perform one of the basic functions a computer can perform.
   (d) Input: There must be some input data and may or may not input conditions or constraints.
   (e) Output: The algorithm must produce some result.

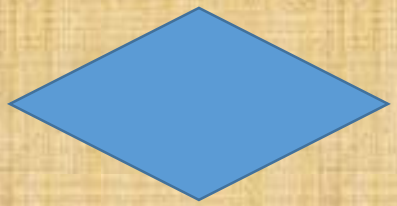
PROGRAM LOGIC DEVELOPMENT TOOLS

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

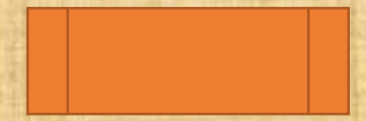
Following graphical objects are being used by flowchart to represent different tasks. These are



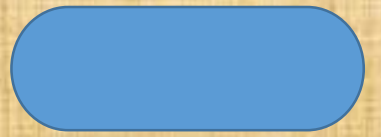
Process



Decision

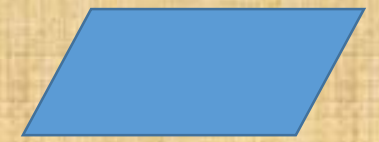


Sub Process



Start/End

Document



Data

Arrow showing flow of control

PROGRAM LOGIC DEVELOPMENT TOOLS

Pseudocode: Pseudocode is an informal way of describing the steps of a program's solution without using any strict programming language syntax or underlying technology considered.

Decision Trees: Decision tree is a tool to represent a hierarchical structure of outcomes based on certain rules.

Example: Multiple condition testing in hierarchical manner for awarding grades to a students base on marks range.

THE if STATEMENTS of PYTHON

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

An if statements tests a particular condition. If the condition evaluates **true** a course-of-action is followed otherwise another course-of-action is performed.

If statement has various forms which is presented here

```
if <conditional expression>:
    statement
    [statement]
```

Indentation of all statements of the true block must be same

Example

```
if ch==‘ ‘:
   no_of_space+=1
   no_of_chars+=1
```

THE if STATEMENTS of PYTHON

Another Form

```
if <conditional expression>:
    statement
    [statement]          } True Block
else:
    statement
    [statement]          } False Block
num=int(input("Enter a number:- "))
if num%2==0:
    print(num," is an even number")
else:
    print(num, " is an odd number")
```

THE if STATEMENTS of PYTHON

Next Form

```
if <conditional expression>:
    if <condition expression2>:      # Nested if
        statement
        [statement]                  True Block of nested if
    else:
        statement
        [statement]                  False Block of nested if
else:
    statement
    [statement]                      #False Block of outer if
```

THE if STATEMENTS of PYTHON

Example

```
year=int("input year value in 4 digits:- year"))
if year % 4 == 0:
    if year % 100 == 0:
        if year % 400 == 0:
            print(year, " is a leap year and century year")
        else:
            print(year, " is a century year but not leap year")
    else:
        print(year, " is a leap year but not century")
else:
    print(year, "is neither leap year nor a century year")
```

THE if STATEMENTS of PYTHON

Next Form (Nested if, else if ladder)

```
if <conditional expression>:
    if <condition expression2>:          # Nested if
        statement
        [statement]
    else:
        statement
        [statement]
else:
    if <condition expression3>:          #else if ladder
        statement
        [statement]
    else:
        statement
        [statement]
```

THE if STATEMENTS of PYTHON

Example

```
num1=int(input("Enter first number:- "))
num2=int(input("Enter second number:- "))
num3=int(input("Enter third number:- "))
if num1>num2:
    if num1>num3 :
        print(num1, "is largest number")
    else:
        print(num3, " is the largest number")
 else:
     if num2>num3:
         print(num2, "is the largest number")
     else:
         print(num3, "is the largest number")
```

THE if STATEMENTS of PYTHON

Next Form (Nested if, else if ladder)

```
if <conditional expression>:
    statement
    [statement]
elif <condition expression>:
    statement
    [statement]
 else:
    statement
    [statement]
```

THE if STATEMENTS of PYTHON

Form1 of If Statement

```
if <conditional statement>:
    if <conditional statement>:
        staements(s)
    else:
        statement(s)
elif <conditional statement>:
    statement
    [statements]
else:
    statement
    [statements]
```

THE if STATEMENTS of PYTHON

Form2 of If Statement

```
if <conditional statement>:
    statement
    [staements]
elif <conditional statement>:
    if <conditional statement>:
        statement
        [statements]
    else:
        statement
        [statements]
else:
    statement
    [statements]
```

THE if STATEMENTS of PYTHON

Form 3 of If Statement

```
if <condition expression>:
    statement
    [statements]
elif <condition expression>:
    statement
    [statements]
else:
    if <condition expression>:
        statement(s)
    else:
        statement(s)
```

THE if STATEMENTS of PYTHON

Form 4 of If Statement

```
if <condition expression>:
    if <condition expression>:
        statement(S)
    else:
        statement(s)
    statement    [statements]
elif <condition expression>:
    if <condition expression>:
        statement(S)
    else:
        statement(s)
else:
    if <condition expression>:
        statement(s)
    else:
        statement(s)
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-1 (If without else structure demonstration)

#w.a.p. in python to read the qty purchased and unit price. Person will get discount of 5% if the purchasevalue exceeds 1000 rupees. Print qty purchased, unit price, discount percent, discount amount, net payment

```
qty=int(input("Enter quantity purchased:- "))
price=float(input("Enter unit price:- "))
discount=0
discountamount=0
purchasevalue=qty*price;
if purchasevalue >1000:
    discount=5
    discountamount=purchasevalue*discount/100
netpayment=purchasevalue-discountamount
print("Qtypurchased+ ",qty)
print("Unit Price = ",price)
print("discount percent = ",discount)
print("discount amount = ",discountamount)
print("net amount = ",netpayment)
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-2 (If with else structure demonstration)

```
#write a program in Python that will test whether a number is an
#even number or an odd number
num=int(input("enter an integer :- "))
if num%2 == 0:
    print(num, " is an even number")
else:
    print(num, " is an odd number")
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-3 (If within if i.e. nested if structure demonstration)

#Write a program to test whether a year is a leap year and #century year or not

```
yearval=int(input("Enter year :- "))
if yearval%4==0:
    if yearval%100==0:
        if yearval%400==0:
            print(yearval, " is a leap year and century year")
        else:
            print(yearval," is a century year but not leap year")
    else:
        print(yearval, " is a leap year but not century year")
else:
    print(yearval, "is neither leap year nor century year")
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-4 (If elif demonstration)

```
#Write a python program to find the largest of the three integers
x=eval(input("enter first number  "))
y=eval(input("Enter Second number "))
z=eval(input("Enter third number  "))
if x > y and x > z:
        print(x , " is the largest value")
elif y> x and y > z:
        print(y , " is the largest value")
elif z> x and z>y:
        print(z , " is the largest value")
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-4 (If within if and if within demonstration)

```
#Write a python program to find minimum of the three integers
x=int(input("Enter first integer:- "))
y=int(input("Enter Second Integer:- "))
z=int(input("Enter Third Integer :- "))
if x < y:
    if x < z:
        min = x
    else:
        min = z
else:
    if y < z:
        min = y
    else:
        min = z;
print("smallest among ",x,y,z," = ",min)
```

FEW PROGRAM EXAMPLES RELATED TO SELECTION CONTROL STATEMENT

Program-4 Find Roots of a quadratic equation

```
import math
print("for quadratic equation, ax**2+bx+c=0 enter coefficient value:- ")
a=eval(input("enter value for a :-"))
b=eval(input("Enter value for b:- "))
c=eval(input("enter value for c:- "))
if a == 0:
    print("Value of a should not be zero")
    print("aborting program")
else:
    delta=b*b-4*a*c
    if delta > 0:
        root1=(-b+math.sqrt(delta))/(2*a)
        root2=(-b-math.sqrt(delta))/(2*a)
        print("Roots are Real and Unequal")
        print("Root1 = ",root1, "Root2 ",root2)
    elif delta == 0:
        root1= -b/(2*a)
        print("Roots are Real and Equal")
        print("Root1 = ",root1, "and root2 = ",root1)
    else:
        print("Roots are complex and imaginary")
```

ITERATIVE STATEMENTs IN PYTHON

Some problem's solution demands execution of one or more statements more than once. For such problem's solution Python provides iterative statement or loop control statement.

Python has two kinds of loop to represent two categories of loops

1. counting loops : Loops that repeat a certain number of times. Pyhton for loop is a counting loop.
2. Conditional Loops: Loops that repeat until a certain things happens. They keep repeating as long as some condition is true. Python while loop is conditional loop.

ITERATIVE STATEMENTs IN PYTHON

Before we move the structure of loop control statement let us see the function range

range() function:- range() function of Python generates a list which is special sequence type. A sequence in Python is a sequence of values bound together by a single name. Some python sequences are string, tuples, lists etc.

This function has following form

range(lowerlimit, upperlimit, stepvalue)

Example

range(5)	will give a list[0,1,2,3,4]
range(0,5)	will give a list[0,1,2,3,4]
range(1,11,2)	will give a list[1,3,5,7,9]
range(10,1,-2)	will give a list[10,8,6,4,2]
range(5,0)	will give a empty list [ ]

ITERATIVE STATEMENTs IN PYTHON

The for loop

Syntax of for loop is as below

```
for <variablename> in <sequence>:
        statements to repeat
```

Example

```
>>> for a in [1,4,7]:      # here as a sequence a list with three numbers 1,4 and 7 are given
        print(a)
        print(a*a)
```

Output

```
1
1
4
16
7
49
```

ITERATIVE STATEMENTs IN PYTHON

2nd Example

```
>>> for ch in 'computer':     #here as a sequence a string is given
        print(ch)
```

Output

```
c
o
m
p
u
t
e
r
```

ITERATIVE STATEMENTs IN PYTHON

3rd Example

```
>>> for val in range(5):    # Here a sequence is created by range() function
        print(val)
```

Output

```
0
1
2
3
4
```

Note: if no initial value and step value is given by default initial value is 0 and the step value is +1.

ITERATIVE STATEMENTS IN PYTHON

4th Example

```
>>> for a in range(5,0):
        print(a)
```

Output

Nothing has been displayed as the starting value is given 10 , end value is given 0 but by default step value which is not given is +1. hence an empty list has been created by the range function

5th Example

```
>>> for a in range(5,0,-1):
        print(a)
```

Output

5
4
3
2
1

ITERATIVE STATEMENTs IN PYTHON

While loop

initialization of loop control variable
while(logical expression):
    loop body statements
    updation of loop control variable

Example:

```
i=1
while(i<=5):
   j=1;
   while(j<=5):
      print("*",end=" ")
      j=j+1
   i=i+1
   print( )
```

output

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

ITERATIVE STATEMENTs IN PYTHON

break:

This is jumping statement in Python. This statement when executed takes the control out of current loop.

Continue:

This is also a jumping statement in python. This statement on execution forces the interpreter for next iteration leaving statements lying below continue statement unexecuted.

Program Demonstration to show the difference between break and continue

```
print("The loop with break produces output as below")
for i in range(1,5):
    if i%3==0:
        break;
    else:
        print(i)
```

Output

1

2

```
print("The loop with continue")
for i in range(1,5):
    if i%3==0:
        continue
    else:
        print(i)
```

Output

1

2

4

ITERATIVE STATEMENTs IN PYTHON

Loop else statement: loop else statement is executed only when the loop is terminated normally. It is not executed when the control comes out of loop due to break statement.

syntax

```
loop control variable initialization
while(logical expression):
    body of the loop
    updation of loop control variable
else:
    statement(s)
```

```
for variable in <sequence>:
        loop body statements
else:
        statement(s)
```

ITERATIVE STATEMENTs IN PYTHON

Demonstration of loop else statement

```
count=sum=0
ans='y'
while ans=='y':
    num=int(input("Enter number :"))
    if num<0:
        print("Number entered is below zero. aborting!")
        break
    sum=sum+num
    count=count+1
    ans=input("Want to enter more numbers? (y/n).. ")
else:
    print("You entered ",count," Numbers so far.")
print("Sum of numbers entered is ",sum)
```

```
Enter number : 5
Want to enter more numbers?(y/n).. y
Enter number : 10
Want to enter more numbers?(y/n).. y
Enter number : 15
Want to enter more numbers?(y/n).. n
You entered 3 numbers so far
Sum of numbers entered is 30
```

Thanks for Watching This Presentation