Python Programming

LECTURES BY

DR. AVINASH GULVE

Lecture -1

- IF Statement
- If- else and elif statements

Decision Making

Decision making is choosing among alternates (branches).

Why is it needed?

- When alternative courses of action are possible and each action may produce a different result.

In terms of a computer program the choices are stated in the form of a question that only yield a binary answer (is it true or false that the user made a particular selection).

Decisions are questions with answers that are either true or false.

The program may branch one way or another depending upon the answer to the question.

Decision making/branching constructs

- `If` (reacts differently only for true case)
- `If-else` (reacts differently for the true or false cases)
- `If-elif-else` (multiple cases possible but only one case can apply, if one case is true then it’s false that the other cases apply)

If Statement

Single alternative decision structure:

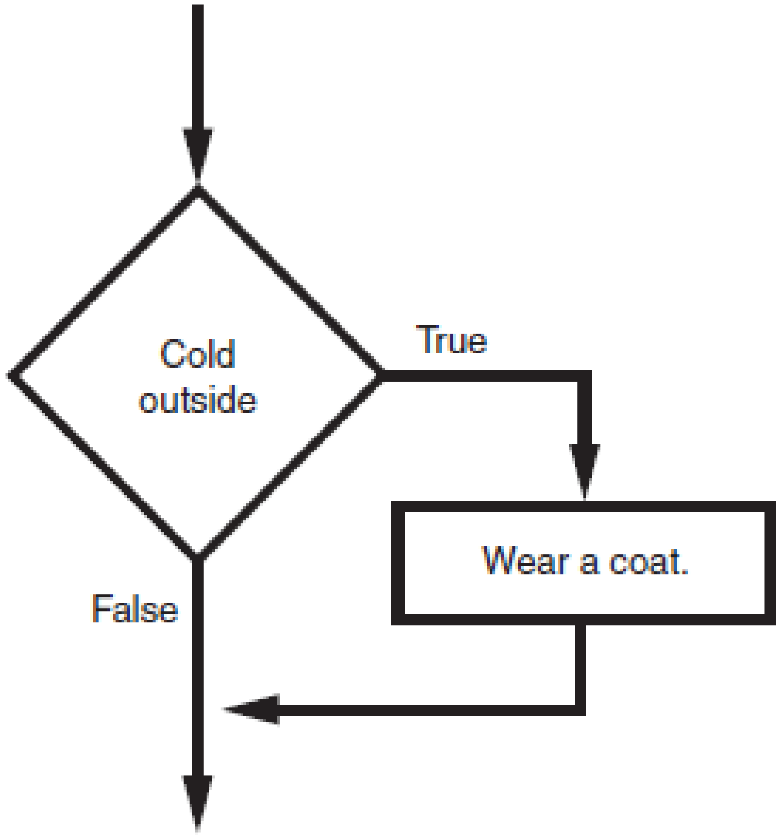
- provides only one alternative path of execution
- If condition is not true, exit the structure

Syntax:

```
if condition:
    do_something
```

Condition must be statement that evaluates to a boolean value

If Statement



If Statement

Python syntax:

```
if condition:
    Statement
    Statement
```

First line known as the `if` clause

- Includes the keyword `if` followed by condition
  - The condition can be true or false
  - When the `if` statement executes, the condition is tested, and if it is true the block statements are executed. otherwise, block statements are skipped

If Statement

Python Program

```
a=int(input("Enter first  number\t"))
b=int(input("Enter Second number\t"))
if(a > b):
   print("First number ",a," is greater than ",b)
```

```
from random import randint
import sys
num = randint(1,10)
guess = eval(input('Enter your guess: '))
if guess==num:
   print('You got it!')
   #quit() #exit()
   sys.exit("Correct guess!!!")
print("Incorrect guess!!!!")
```

```
#Short hand if statement
a=int(input("Enter First  number\t"))
b=int(input("Enter Second number\t"))
if(a > b): print("First number is greater than Second number")
```

If-else Statement

Dual alternative decision structure:

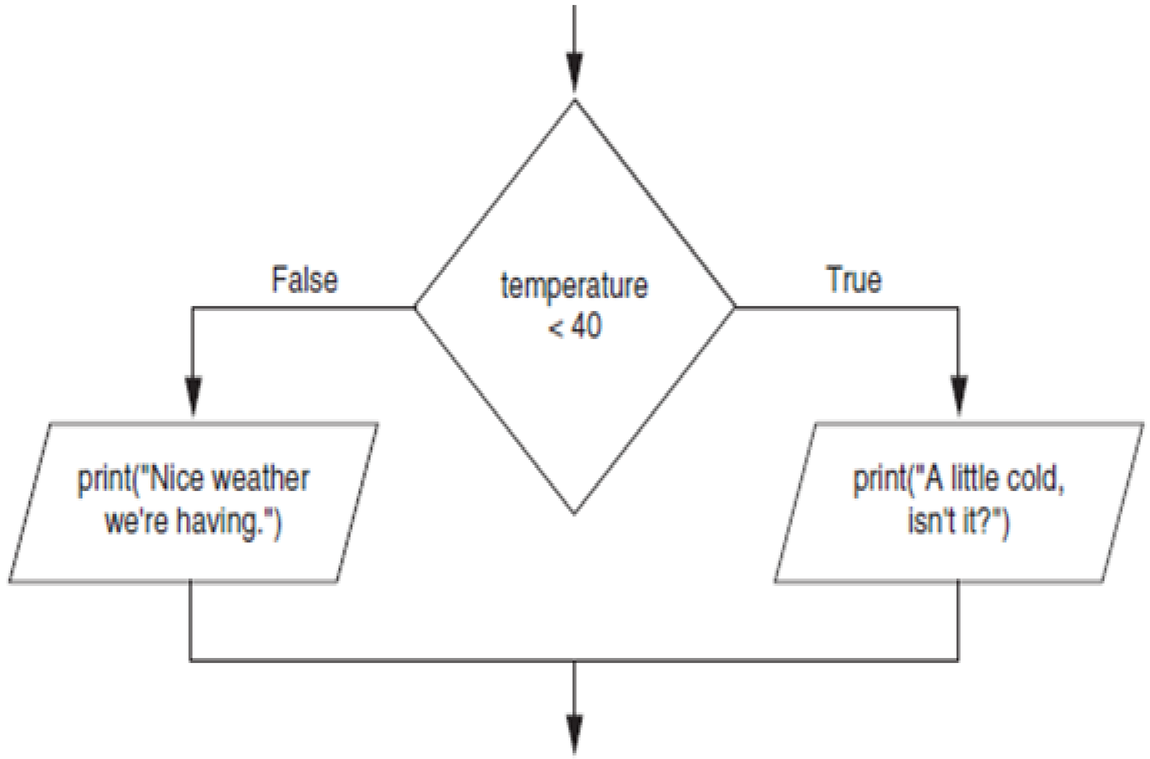
- Two possible paths of execution
- One is taken if the condition is true, and the other if the condition is false

Syntax:

```
if condition:
        statements
else:
        other statements
```

- `if` clause and `else` clause must be aligned
- Statements must be consistently indented

If-else Statement


False
temperature < 40
True
print("Nice weather we're having.")
print("A little cold, isn't it?")

If-else Statement

```
#test whether division by 0
x = int(input("x? "))
y = int(input("y? "))
if y != 0:
   print( x / y )
else:
   print ("Attempted division by zero")
```

```
#find largest of two number
a=int(input("Enter first  number\t"))
b=int(input("Enter Second number\t"))
if(a > b):
   print("Number ",a," is greater than ",b)
else:
   print("Number ",b," is greater than ",a)
```

```
#Short hand  if-else statement
a=int(input("Enter a  number\t"))
print("Positive") if(a > 0) else print("Either 0 or Negative")
```

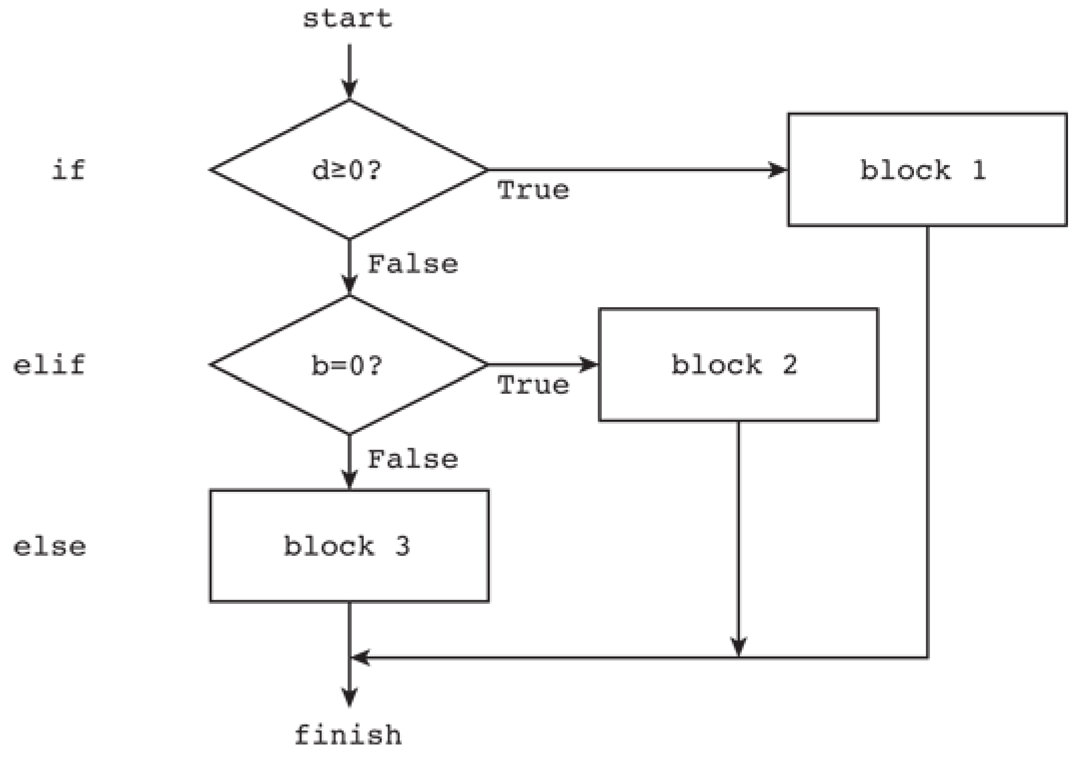
The `if-elif-else` Statement

Syntax

```
if <condition-1>:
    <sequence of statements-1>

elif <condition-n>:
    <sequence of statements-n>
else:
    <default sequence of statements>
```

The `if-elif-else` Statement



The `if-elif-else` Statement

```
a=int(input("Enter First  number\t"))
b=int(input("Enter Second number\t"))
if(a > b):
    print("First number is greater than Second number")
elif(a==b):
    print("Both numbers are same")
else:
    print("Second number is greater than First number")
```

The `if-elif-else` Statement

```
num = int(input("Enter a positive integer number "))
if num < 10:
    print("One digit number")
elif 9 < num < 99:
    print("Two digit number")
elif 99 < num < 999:
    print("Three digit number")
elif 999 < num < 9999:
    print("Four digit number")
else:
    print("Five or more digit number")
```

If-else Statement

```
#short hande if-elif else statement
a=int(input("Enter a  number\t"))
print("Positive") if(a > 0) else print("Zero") if(a==0) else print("Negative")
```

Nested if statement

```
num = 15
if num >= 0:
    if num == 0:
        print("Zero")
    else:
        print("Positive number")
else:
    print("Negative number")
```

Loops

For

While

For Loops 1

- A for-loop steps through each of the items in a collection type, or any other type of object which is “iterable”

```
for <item> in <collection>:
  <statements>
```

- If <collection> is a list or a tuple, then the loop steps through each element of the sequence
- If <collection> is a string, then the loop steps through each character of the string

```
for someChar in “Hello World”:
     print someChar
```

For Loops 2

```
for <item> in <collection>:
    <statements>
```

- <item> can be more than a single variable name
- When the <collection> elements are themselves sequences, then <item> can match the structure of the elements.

For loops & the *range()* function

- Since a variable often ranges over some sequence of numbers, the *range()* function returns a list of numbers from 0 up to but not including the number we pass to it.
- range(5) returns [0,1,2,3,4]
- So we could say:

```
for x in range(5):
    print x
```

- (There are more complex forms of *range()* that provide richer functionality…)

Range Function

range(stop)

range(start, stop[, step])

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(10,5,-1)	10,9,8,7,6
RANGE(10,1,-2)	10,8,6,4,2

For Loop Example

```
r=range(100)
for i in r:
    print("%3d"%i,end="")

for i in range(1,100):
 print("%3d"%i,end='')

for i in range(1,100,3):
 print("%3d"%i,end='')
```

For Loop Example

```
#multiple of 5 and 7
r=range(100)
sum=0
for i in r:
    if(i%5==0 and i%7==0):
        sum+=i
print("Sum = ",sum)
```

For Loop Example

```
#Program to print prime numbers from 1 to 50
for num in range(1,50):
    cnt=0
    for i in range(1, num+1):
        if (num % i == 0):
            cnt=cnt+1
    if(cnt==2):
            print(num, "is a prime number")
```

For Loop using string

```
#Program to count vowels in a string
count=0
str=input("Enter a string\t")
str=str.casefold()      #converts to lowercase
for x in str:
    if(x=='a' or x=='e' or x=='i' or x=='o'or x=='u'):
      count+=1
print("Number of vowels ",count)
```

For loop using list

```
#add list element

lst=[10,20,30,40,50,60,70,80,90,100]

sum=0

for i in lst:

   sum=sum+i

print("Sum of list items is ",sum)
```

```
r=range(0,101,10)
lst=list(r)
print(lst)
sum=0
for i in lst:
   sum=sum+i
print("Sum of list items is ",sum)
```

For loop using list

```
r=range(1,50)
Lst=list(r)
Lprime=[]
for i in Lst:
    flag=1;
    for j in range(2,i//2):
        if(i%j==0):
            flag=0;
            break
    if(flag==1):
        Lprime.append(i)
print("Prime Numbers are ",Lprime)
```

Problem Statement-
Find prime numbers from the range 1-50

For loop using dictionary

```
dic1={1:50, 2:20}
dic2={3:95, 4:60}
dic3={5:80,6:60}
dic4 = {}
for d in (dic1, dic2, dic3):
    dic4.update(d)
print(dic4)
t=dic4[1]
indx=1
for key,val in dic4.items():
    if(t<val):
        t=val
        indx=key
print("Index ",indx," has the largest value ",t)
```

Problem Statement-
Find element with largest value stored in dictionary

For loop using dictionary

Problem Statement-
Process result of a student

```
exam={"name":"Rohan","Math":78,"Physics":89,"Chemistry":86}
print("Student Details")
for key,value in exam.items():
   print(key,"\t",value)
sum=0
for marks in exam.values():
    temp=str(marks)
    if(temp.isnumeric()==True):
       sum=sum+int(temp)
per=sum/3
print(exam['name'],"has secured ",per," marks")
```

For loop using dictionary

```
dicts = {}
keys = range(5)
for i in keys:
        x=int(input("Enter a 
number "))
        dicts[i] = x
print(dicts)
prod=1

for key in dicts:
    prod=prod*dicts[key]
print("Product ",prod)
```

```
prod=1
for val in dicts.values():
    prod=prod*val
print("Product ",prod)

prod=1
for key in dicts.keys():
    prod=prod*dicts[key]
print("Product ",prod)
```

Loop manipulation using pass

```
#program to print sum of first 10 odd numbers
r=range(20)
sum=0
for i in r:
    if(i%2==0):
        pass
    else:
        sum += i
print("Sum of odd numbers is ", sum)
```

Loop manipulation using break

```
#List prime numbers upto 50
for i in range(3,50):
    j=0
    for j in range(2,i):
        if (i % j == 0):
            break
    if  (i%j):
            print("{0:2d}*, ".format(i), end = '')
    else:
            print("{0:2d} ,".format(i), end = '')
```

The break statement ends the current loop and jumps to the statement immediately following the loop

It is like a loop test that can happen anywhere in the body of the loop

Loop manipulation using continue

```
r=range(20)
sum=0
for i in r:
    if(i%2==0):
        continue
    else:
        sum += i
print("Sum of odd numbers is ", sum)
```

The continue statement ends the current iteration and jumps to the top of the loop and starts the next iteration

Loop manipulation using else

```
#Print Armstrong numbers
for num in range(1,500):
    n=num
    sum=0
    L=len(str(n))
    for i in range(L):
        rem=n%10
        sum=sum+rem*rem*rem
        n//=10
    if(sum==num):
        print(num," is a Armstrong number")
else:
    print("Finished printing the Armstrong numbers...")
```

While Loop

Whether or not a part of a program repeats is determined by a loop control (typically the control is just a variable).

- Initialize the control to the starting value
- Testing the control against a stopping condition (Boolean expression)
- Executing the body of the loop (the part to be repeated)
- Update the value of the control

While Loop

(Simple condition)

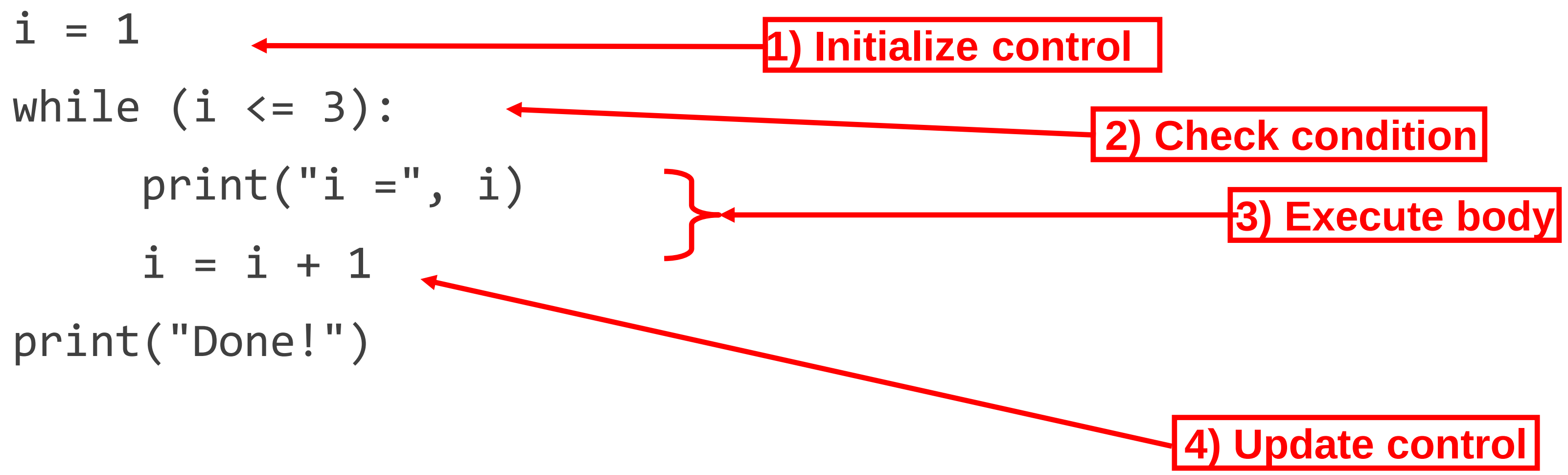
```
    while (Boolean expression):
        body
```

(Compound condition)

```
    while (Boolean expr) Boolean op (Boolean expr):
        body
```

While Loop

Program name: `while1.py`



```
i = 1
while (i <= 3):
    print("i =", i)
    i = i + 1
print("Done!")
```

While Loop

```
#sum of digits
num=onum=2149
sum=0;
while(num>0):
  sum=sum+num%10;
  num=num//10;
print("sum of digits of ",onum," is ",sum)
```

While Loop

```
#Check Palindrome
prompt="Enter a number "
onum=num=int(input(prompt))
rev=0
while(num!=0):
    rev=rev*10+num%10
    num=num//10
if(onum==rev):
    print("The number ",onum," is palindrome");
else:
    print("The number ",onum," is not palindrome");
```

While Loop

```
#Decimal to Binary
num=int(input("Enter a number \t"))
binary=""
temp=0
while(num>0):
    binary+=str(num%2)
    num=num//2
binary=binary[::-1]
print(binary)
```