unacademy

Strings

Course on Data Structure and Algorithms Using Python

Ankush Saklecha • Lesson 3 • Dec 10, 2023

Python Programming

Operators

Operators

- Operators are used to perform several operations on variables values and constants.
- Operators are defined between operands.

Examples:
a=b
a+b
a>=b
a and b

Types of Operator

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Arithmetic operators

+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

Guess Output?

```
x=30.5
y=4
print(x/y)
print(x%y)
print(x//y)

x=0
y=4
print(x/y)
print(x%y)
print(x//y)

x=30
y=0
print(x/y)
print(x%y)
```

Let's Try…

Guess Output?

```
x=30.5
y=4
print(x/y)
print(x%y)
print(x//y)
```

7.625

2.5

7

```
x=0
y=4
print(x/y)
print(x%y)
print(x//y)
```

0.0

0

0

```
x=30
y=0
print(x/y)
print(x%y)
```

Error

Error

Assignment Operators

=	x = 5	x = 5
+=	x += 3	x = x + 3
-=	x -= 3	x = x - 3
*=	x *= 3	x = x * 3
/=	x /= 3	x = x / 3
%=	x %= 3	x = x % 3
//=	x //= 3	x = x // 3
**=	x **= 3	x = x ** 3
&=	x &= 3	x = x & 3
|=	x |= 3	x = x | 3
^=	x ^= 3	x = x ^ 3

Guess Output?

```
x=6
x+=-2
print(x)

x=6
x/=-2.5
print(x)

x=6
x//=-2.5
print(x)

print (-5//2)

print (5.0//2)

print (-5.0//2)
```

Let's Try…

Guess Output?

```
x=6
x+=-2
print(x)
```

4

```
x=6
x/=-2.5
print(x)
```

-2.4

```
x=6
x//=-2.5
print(x)
```

-3.0

```
print (-5//2)
```

-3

```
print (5.0//2)
```

2.0

```
print (-5.0//2)
```

-3.0

Comparison Operators

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Logical Operators

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

Bitwise Operators

&	AND	Sets each bit to 1 if both bits are 1
|	OR	Sets each bit to 1 if one of two bits is 1
^	XOR	Sets each bit to 1 if only one of two bits is 1
~	NOT	Inverts all the bits
<<	Zero fill left shift	Shift left by pushing zeros in from the right and let the leftmost bits fall off
>>	Signed right shift	Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

unacademy

Guess Output? .

```
x=6
x|=2
print(x)

x=7
x&=5
print(x)

x=11
x>>=1
print(x)

x=28
x<<=2
print(x)

x = 5
x ^= 3
print(x)
```

Let's Try…

Guess Output? .
x=6 x|=2 print(x) **6**
x=7 x&=5 print(x) **5**
x=11 x>>=1 print(x) **5**
x=28 x<<=2 print(x) **112**
x = 5 x ^= 3 print(x) **6**

Identity Operators

Operator	Description	Example
Operator	Description	Example
is	Returns True if both variables are the same object	x is y
is not	Returns True if both variables are not the same object	x is not y

Membership Operators

Operator	Description	Example
in	Returns True if a sequence with the specified value is present in the object	x in y
not in	Returns True if a sequence with the specified value is not present in the object	x not in y

Guess Output?

```
Letter=['a','b','c','e','g']
print('A' in Letter)

Letter=['a','b','c','e','g']
print('hi' in Letter)

Letter=['a','b','c','e','g']
print('HELLO' not in Letter)

Letter=['A','b','c','e','g']
print('%c' %65 in Letter)

a=[2,4,6,8,10]
b=[2,4,6,8,10]
c=a
print(a is c)
print(a is b)
print(a==b)
```

Let's Try…

Guess Output?

```
Letter=['a','b','c','e','g']
print('A' in Letter)
```

False

```
Letter=['a','b','c','e','g']
print('hi' in Letter)
```

False

```
Letter=['a','b','c','e','g']
print('HELLO' not in Letter)
```

True

```
Letter=['A','b','c','e','g']
print('%c' %65 in Letter)
```

True

```
a=[2,4,6,8,10]
b=[2,4,6,8,10]
c=a
print(a is c)
print(a is b)
print(a==b)
```

True

False

True

Operators Precedence

Operator	Description	Associativity
()	Parentheses	Left to Right
**	Exponentiation	Right to Left
+x -x ~x	Unary plus, unary minus, and bitwise NOT	
* / // %	Multiplication, division, floor division, and modulus	Left to Right
+ -	Addition and subtraction	Left to Right

Operators Precedence

Operator	Description	Associativity
<< >>	Bitwise left and right shifts	Left to Right
&	Bitwise AND	Left to Right
^	Bitwise XOR	Left to Right
|	Bitwise OR	Left to Right
== != > >= < <= is is not in not in	Comparisons, identity, and membership operators	Left to Right
not	Logical NOT	Left to Right
and	AND	Left to Right
or	OR	Left to Right

Guess Output?

```
X=((4+12)*(15-10)*(12//3))
print(X)

A=24+12*2-3**3
print(A)

print(24//2//4)

print(36*2%7/2)

print(20<<2>>2)

print(80>>1&10)
```

Let's Try…

Guess Output?	
X=((4+12)*(15-10)*(12//3)) print(X)	320
A=24+12*2-3**3 print(A)	21
print(24//2//4)	3
print(36*2%7/2)	1.0
print(20<<2>>2)	20
print(80>>1&10)	8

```
# Left Shift
#print(12<<1)
#print(11<<1)
#print(-12<<1)
#print(-11<<1)
#print(-12<<2)
#print(-12.5<<1)
#Right Shift
#print(12>>1)
#print(11>>1)
#print(-12>>1)
#print(-11>>1)#print(-12.5>>1), #print(-11>>2)x=6x//=-2.5print(x)print(-5.0//2)
```

Python Programming

Control Statements

Content

- Control Statements
- Types of Control Statements
- Conditional Statement
- Looping Statement
- Jumping Statement
- Sample Problems

Control Statements

- Control statements define the order of statement executions for the python program.

Sequential Statement Examples:

```
a=20
b=60
print(a+b)
```

Conditional Statement Examples:

```
If a>b:
   print('a is greater')
Else:
   print('b is greater')
```

Looping Statement Examples:

```
While True:
    print('Loop');
```

Jumping Statement Examples:

```
Name='Jhon'
for ch in Name:
    print(ch);
    if ch=='o':
        break;
```

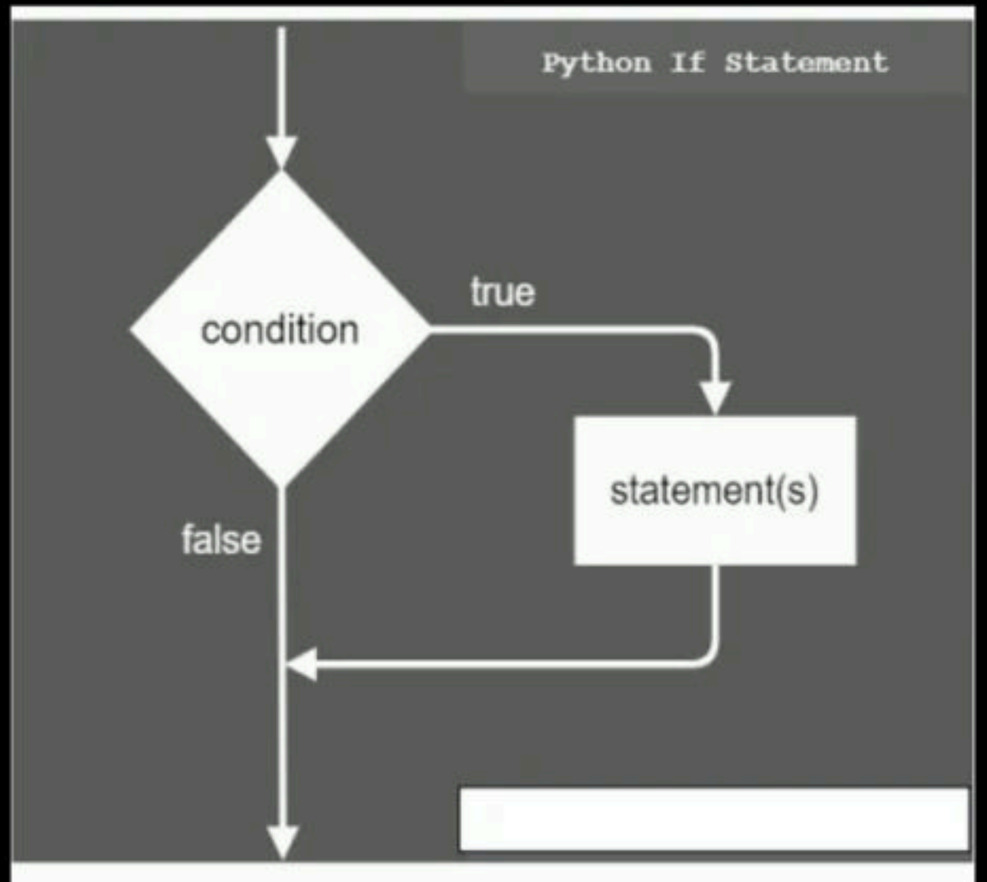
Conditional Statements

In python, the execution flow of codes may switch on the basis of certain decisions. There are four types of conditional statements:

- Simple If Statement
- If-Else Statement
- Nested If Statement
- Elif Ladders

Simple If Statement

- This statement allows execution of certain line of codes when condition is true or satisfied.



Simple If

Syntax:

```
if <condition>:
    statement1
    statement2
```

Example:

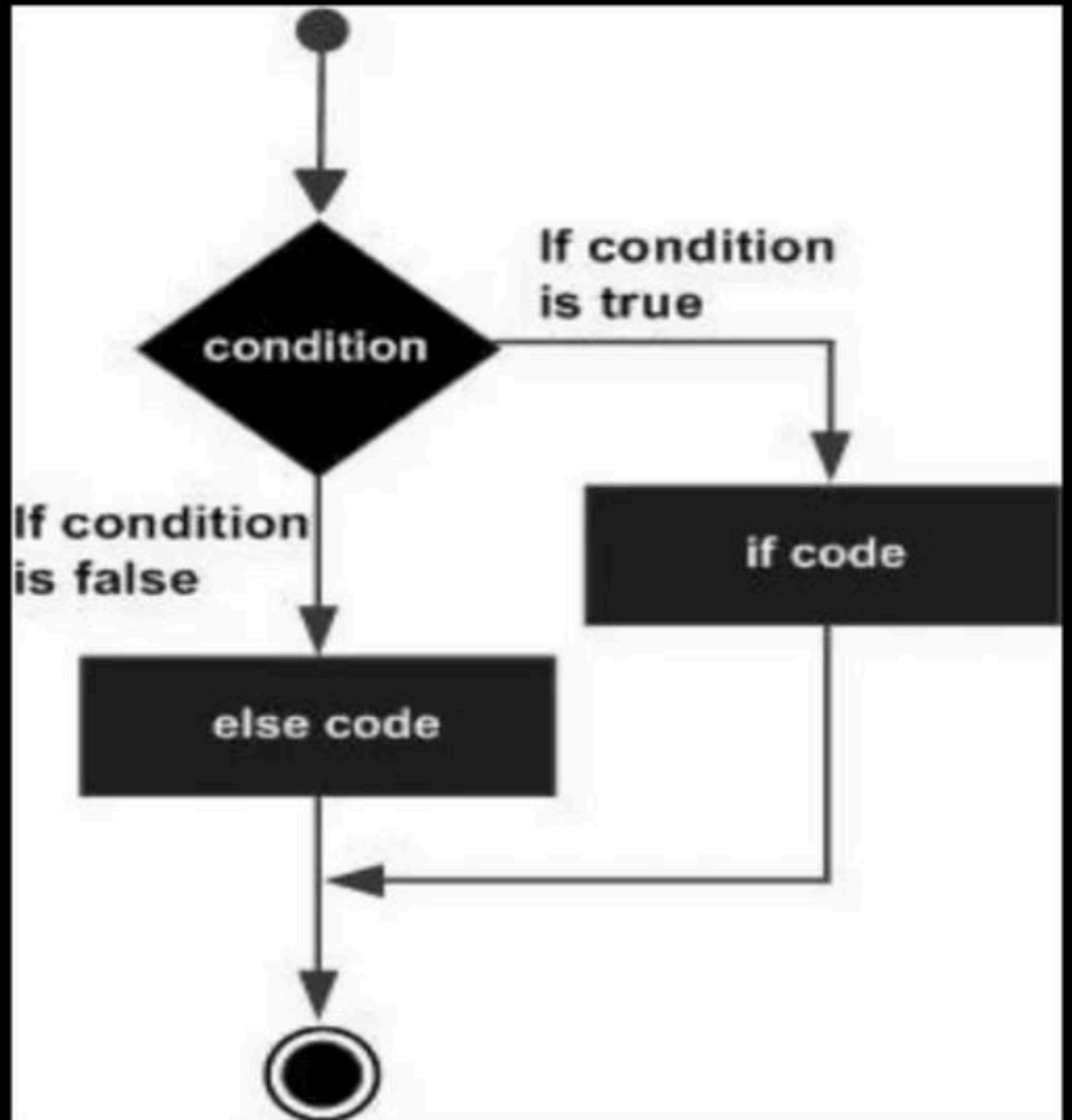
```
a=20
if a>18:
    print("You are eligible to vote")
    print("You are eligible for driving licenses ")
```

Let's Try…

Guess Output?

```
if NULL:
  print('NULL')
print('Exit')

if None:
  print('NULL')
print('Exit')

if -5:
  print('Negative number')

if 0:
  print('Zero')

if 2**4:
  print('Zero')

if 'abc':
  print('String')
```

If Else Statement



If-Else

Syntax:

```
if <condition>:
    statement1
else:
    statement2
```

Example:

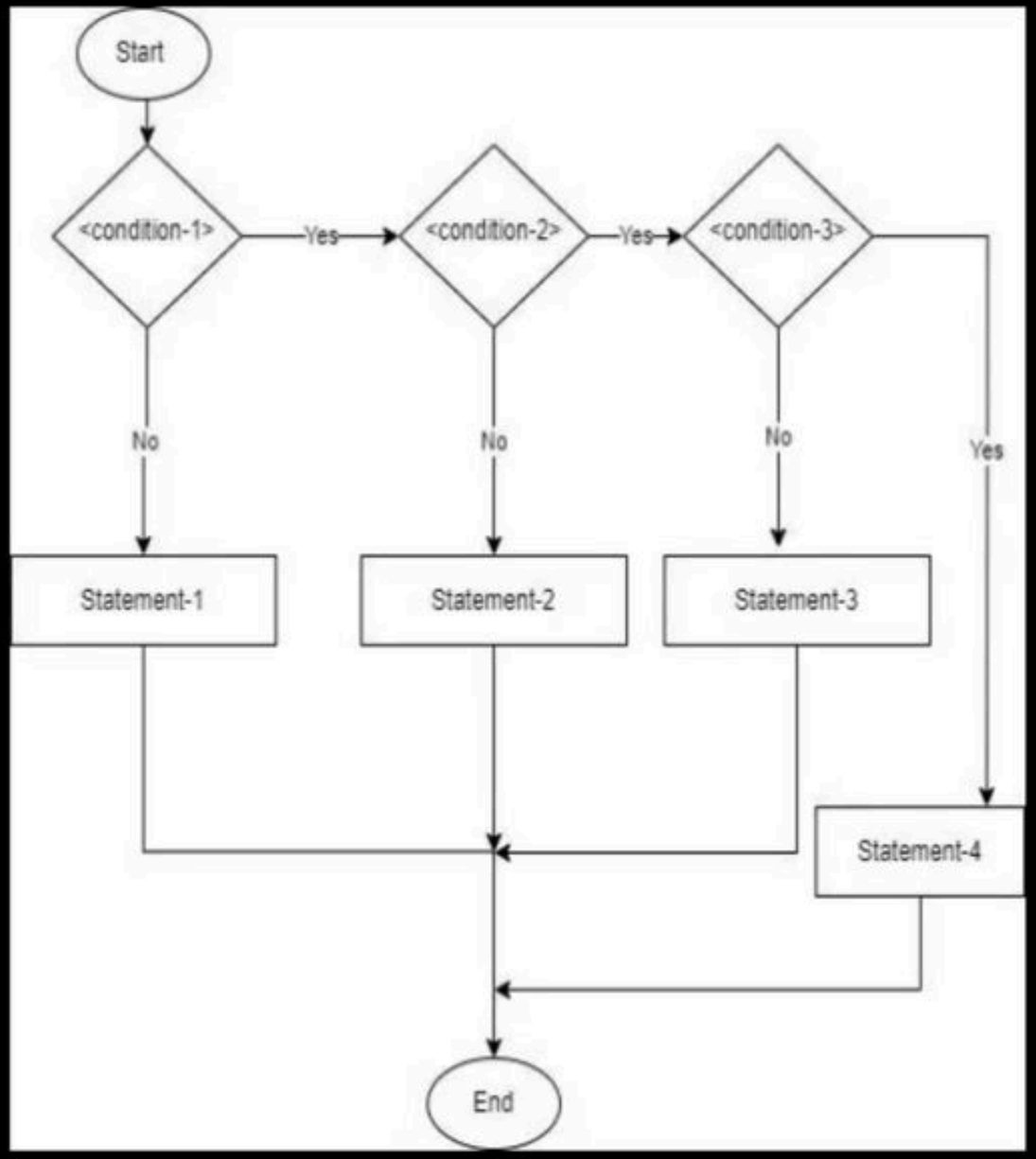
```
a=20
if a>=18:
    print("You are eligible to vote")
else:
    print("You are required to wait up to 18")
```

Let's Try…

Guess Output?

```
if True:
  print('True')
else:
  print('False')

if 6+6>=10:
  print('True')
else:
  print('False')

A=5
if a>5 or a<=5:
  print('It works')
else:
  print('It does not work')

if(a//b==2):
        print ("Yes")
else :
        print("No")
```

Nested If-Else Statement



Nested If-Else

Syntax:

```
if <condition>:
    statement1
    if<condition>:
      statement2
    else:
      statement3
  else:
    statement4
```

Example:

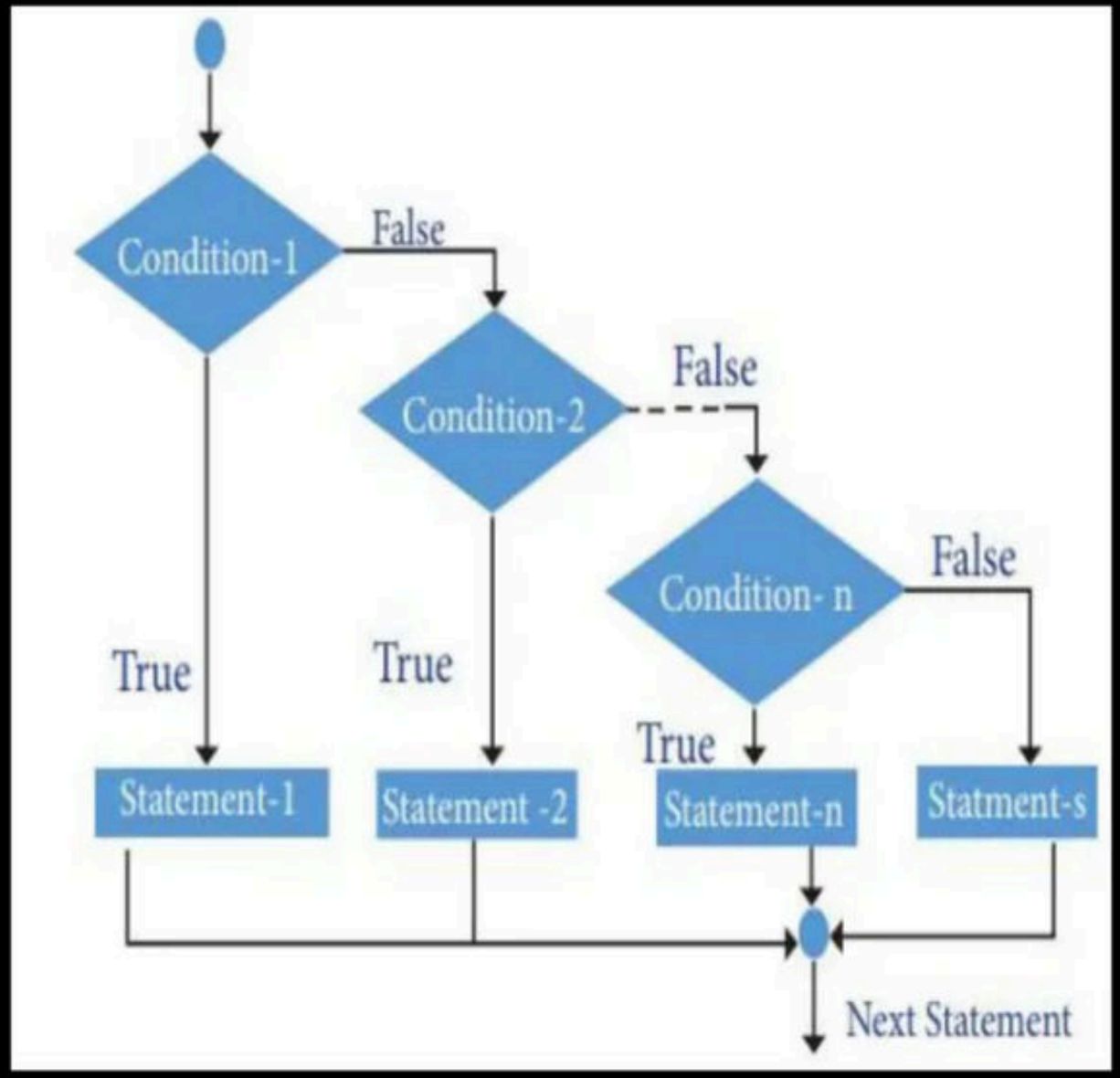
```
a=20
if a>=18:
  print("You are eligible to vote")
  if a>60:
      print("You are senior citizen")
  else:
      print("You are not senior citizen")
else:
  print("You are child")
```

Let's Try…

Guess Output?

```
n=50
if n >20:
    if n>35:
        print("OK",)
        if n>45:
            print("GOOD")
        else:
            print("NO OUTPUT")
   else:
      print("NO OUTPUT")
else:
  print("NO OUTPUT")

a=10
b=5
if (a%b==0):
   print ( "Greater")
   if (a//b==0):
     print ( "Example")
   else:
     print ("Easy to learn with Learnpython4cbse")
else:
  print ("No Output")
```

Elif Ladder



Elif Ladder

Syntax:

```
if <condition>:
    statement1
elif<condition>:
    statement2
elif<condition>:
    statement3
else:
    statement4
```

Example:

```
age=20
if age>=18:
  print("You are eligible to vote")
elif age>15:
  print("You are require to three year more for vote")
elif age>12:
  print("You are require to six year more for vote")
else:
  print("You are child")
```

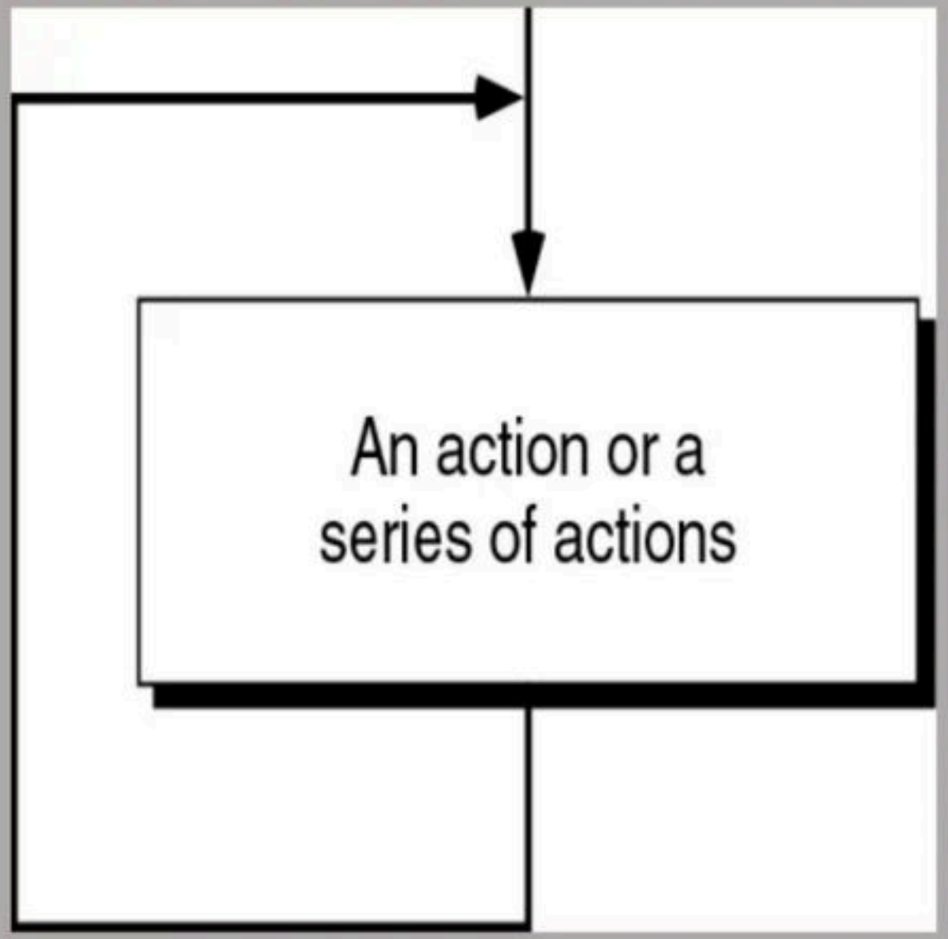
Let's Try…

Guess Output?

```
x = 3
if x == 0:
    print ("Am I learning python?", end = ' ')
elif x == 3:
    print("Or learning python?", end = ' ')
else :
    pass
print ("Or learning python 4 cbse?")

name = "maya"
if name == "saya":
    print("delhi")
elif name == "mana":
    print("mumbai")
else:
    print("india")
```

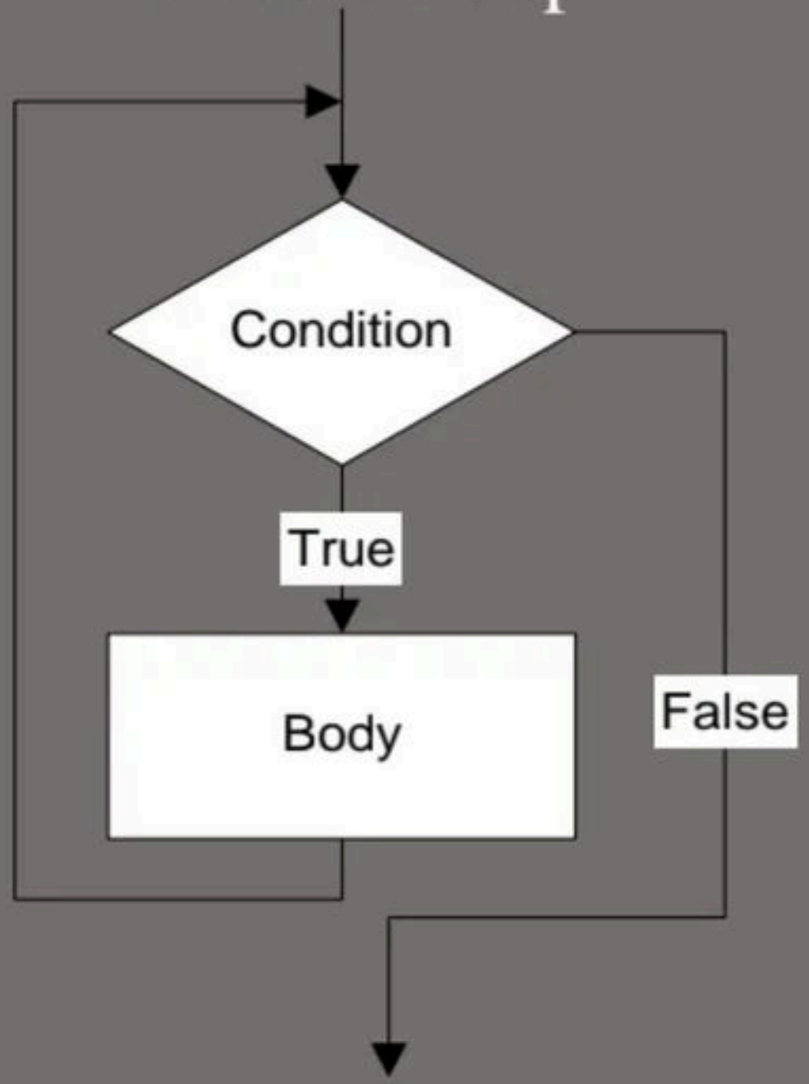
Looping Statement

- The main idea of a loop is to repeat an action or a series of actions.

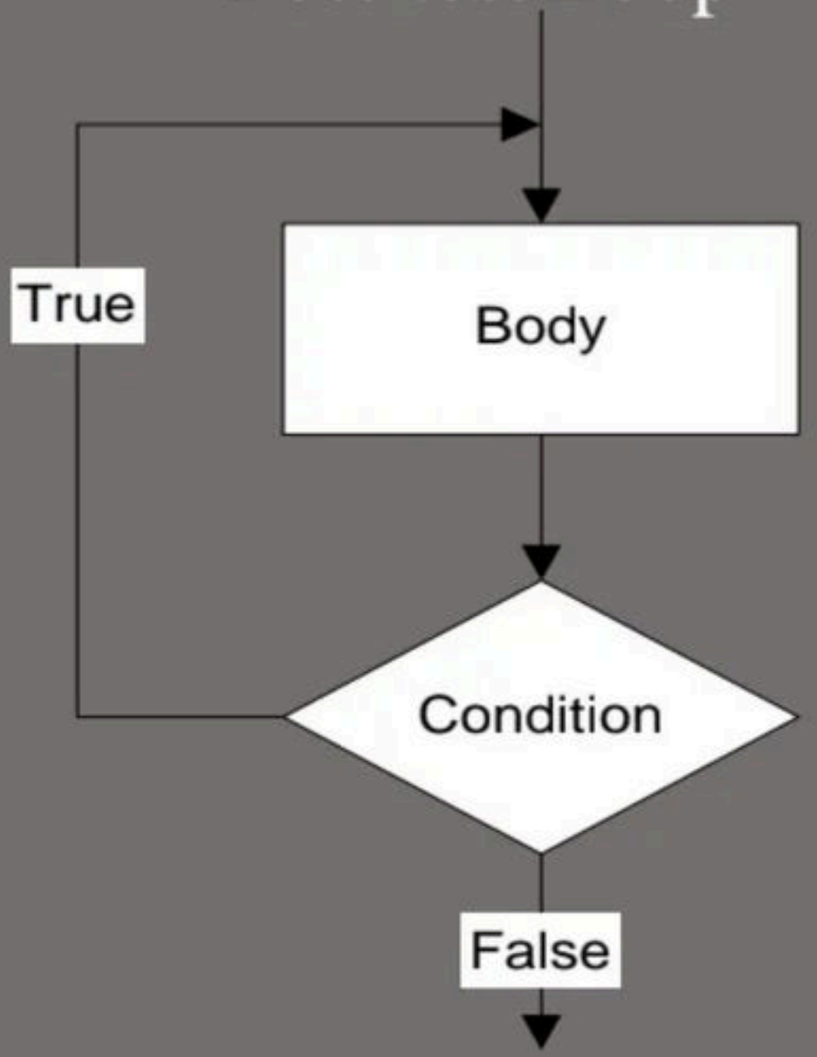


Types of Loop

Pre-test Loop



Post-test Loop



Switch-Case Statement

Switch Case

Syntax:

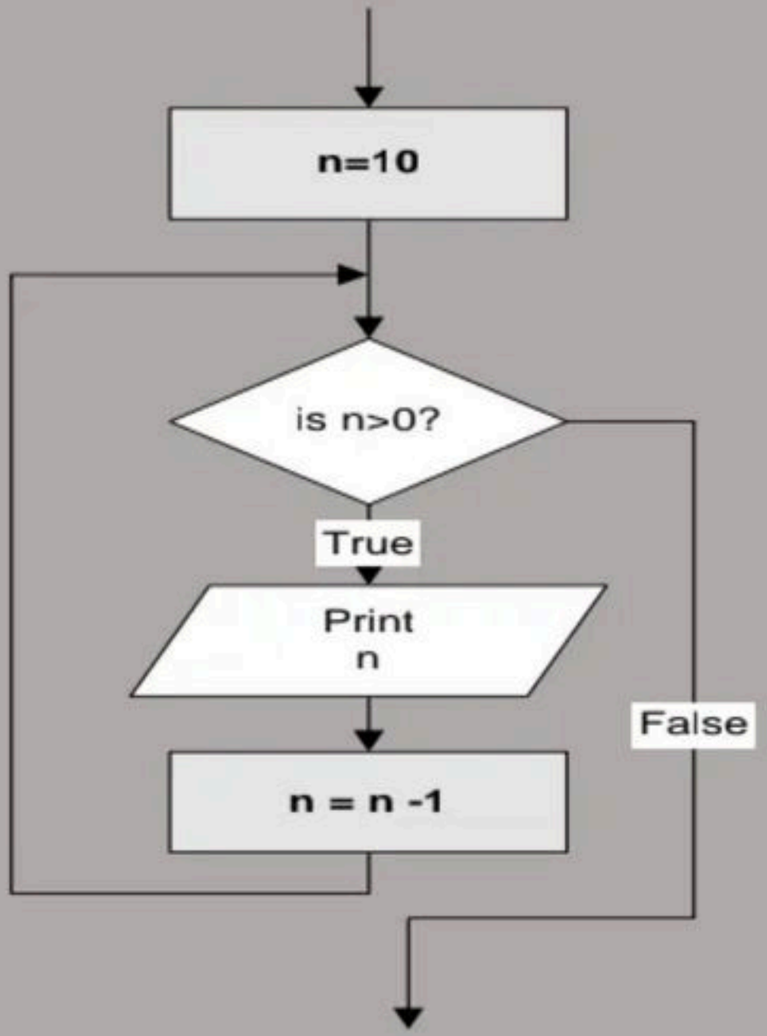
```
match <argument>:
    case 1:
        statement1
    case default:
        statement2
```

Example:

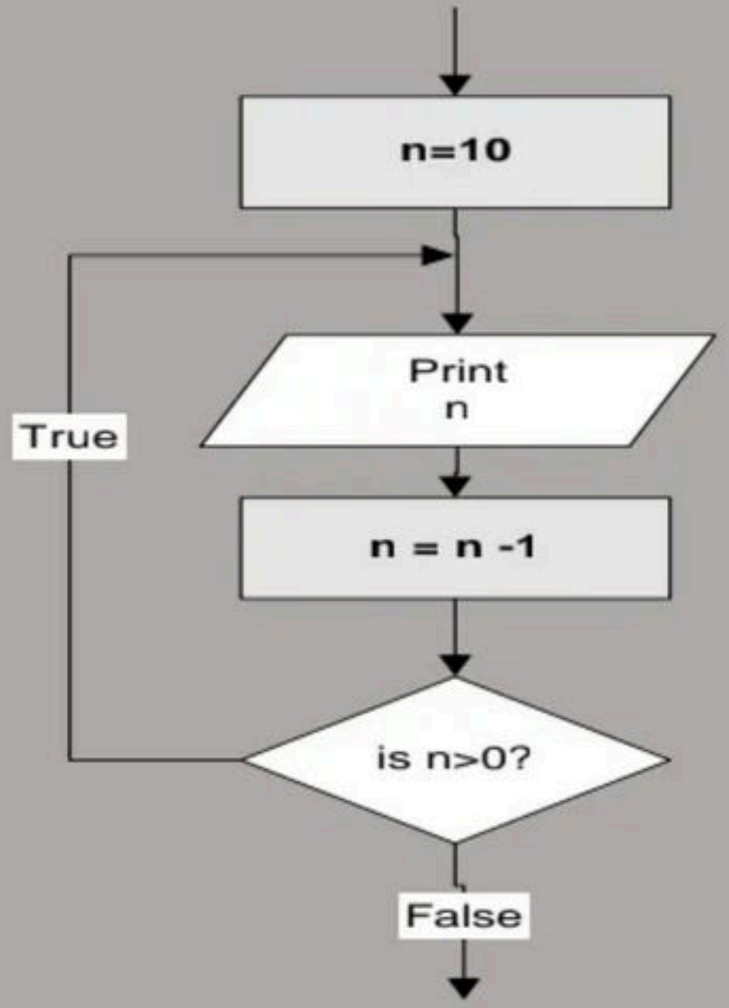
```
number=int(input('Enter a number: '))
match number:
    case 0:
        print("zero")
    case 1:
        print("one")
    case 2:
        print("two")
    case default:
        print("something")
```

Examples: Types of Loop

Pre-test Loop Example

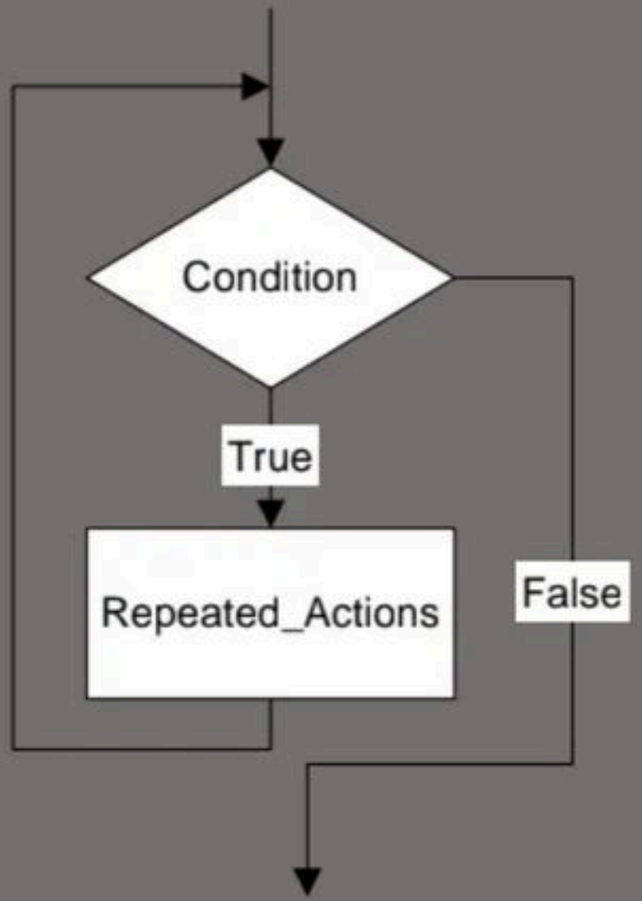


Post-test Loop Example



Pre-test Loop: While Loop

While Loop Flow Chart



While Loop

Syntax:

```
while  <condition>:
        statement1
        statement2
        statement3
```

Example:

```
num=int(input('Enter a number'))
while i<=10:
  print(i*num)
  i=i+1
```

Let's Try…

Guess Output?

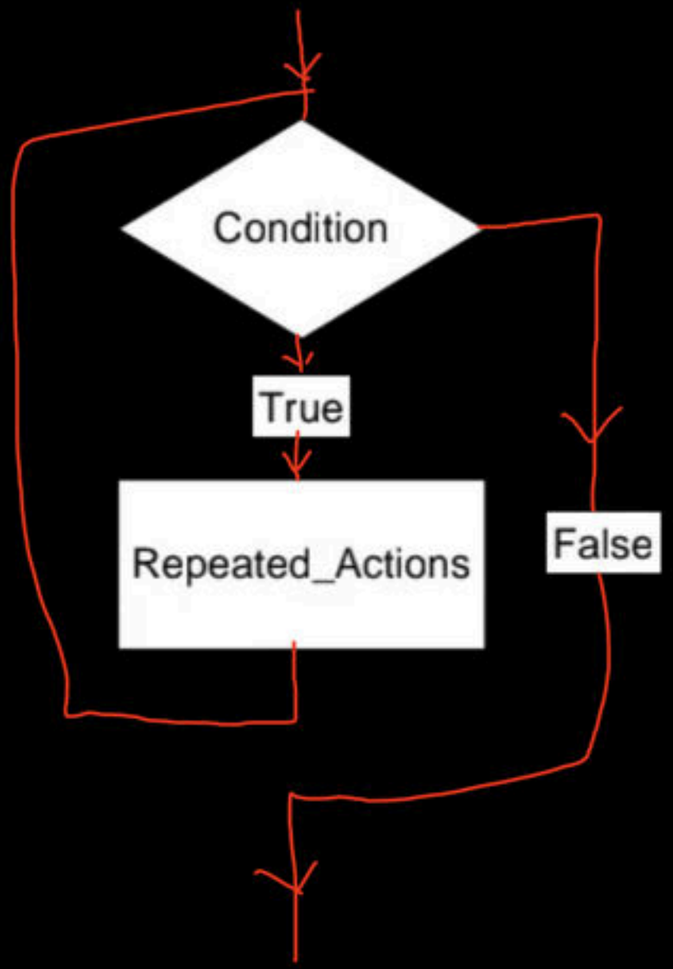
```
a=1
while a<=10:
  print(a)
  a=a+1
else:
  print('while is terminated')

while True:
   print('True')
else:
  print('False')

number = 5
while number <= 5:
    if number < 5:
        number = number + 1
    print(number)
a = "123789"
while x in a:
    print(x, end=" ")
```

Python Programming

Control Statements Part II

Pre-test Loop: For Loop

For Loop Flow Chart



For Loop

Syntax:

```
for<condition>:
    statement1
    statement2
    statement3
```

Example:

```
Msg='Welcome to Bharat'
for i in Msg:
    print(i)
```

Let's Try…

Guess Output?

```
for i in range(0,5):
    print(i)
```

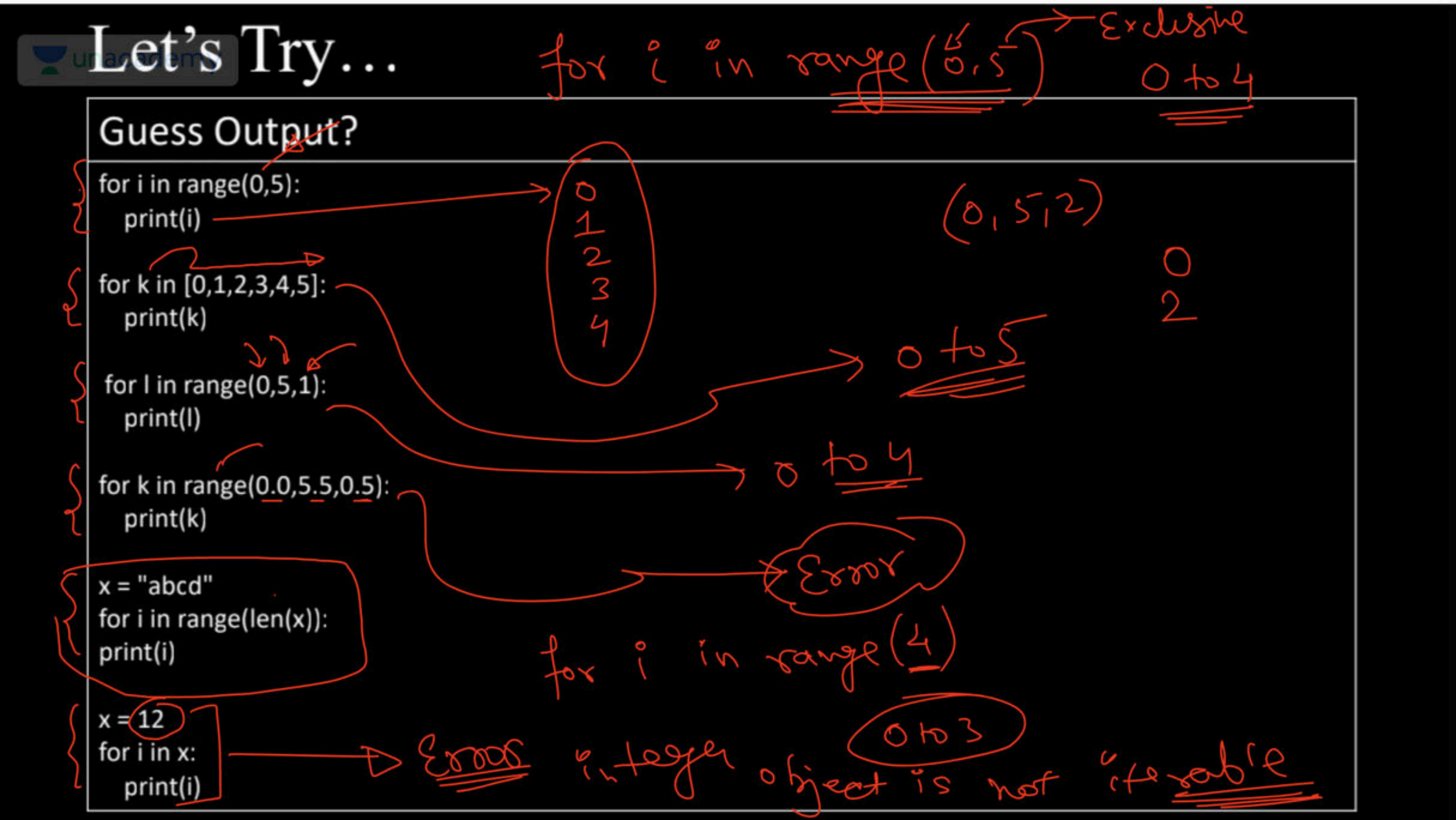
```
for k in [0,1,2,3,4,5]:
    print(k)
```

```
for l in range(0,5,1):
    print(l)
```

```
for k in range(0.0,5.5,0.5):
    print(k)
```

```
x = "abcd"
for i in range(len(x)):
print(i)
```

```
x = 12
for i in x:
    print(i)
```



```
x = 12
for i in x:
    Print(i)
```

⇒ Error

```
x = "12"
for i in x:
    Print(i)
```

1
2

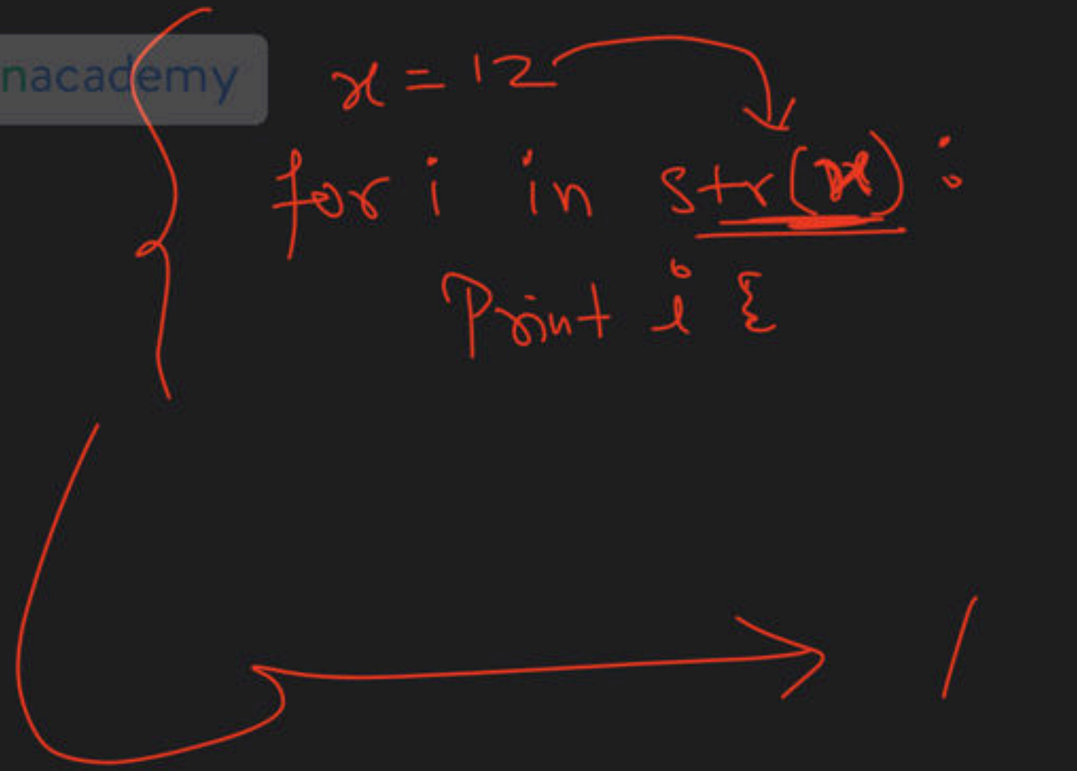
{1, 2}

```
Print(i, end=' ')
```

$x = 1.5$

$y = int(x)$

$Print(y)$



range (0, 5, 1) ⇒

range (0, 5, 0.5)

↳ Error

x = "12"

x = 12

str(x)

1. range(0,5) ⇒ 0 to 4
2. range(5) ⇒ 0 to 4
3. range(1,5) ⇒ 1 to 4

---

range(0,5,2)

Jumping Statement: break

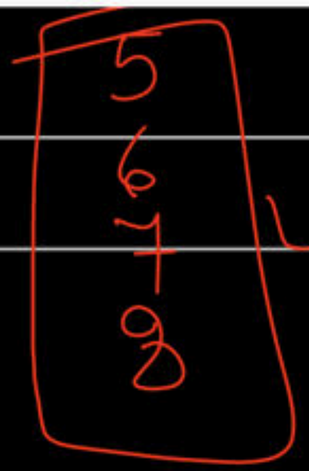
break statement

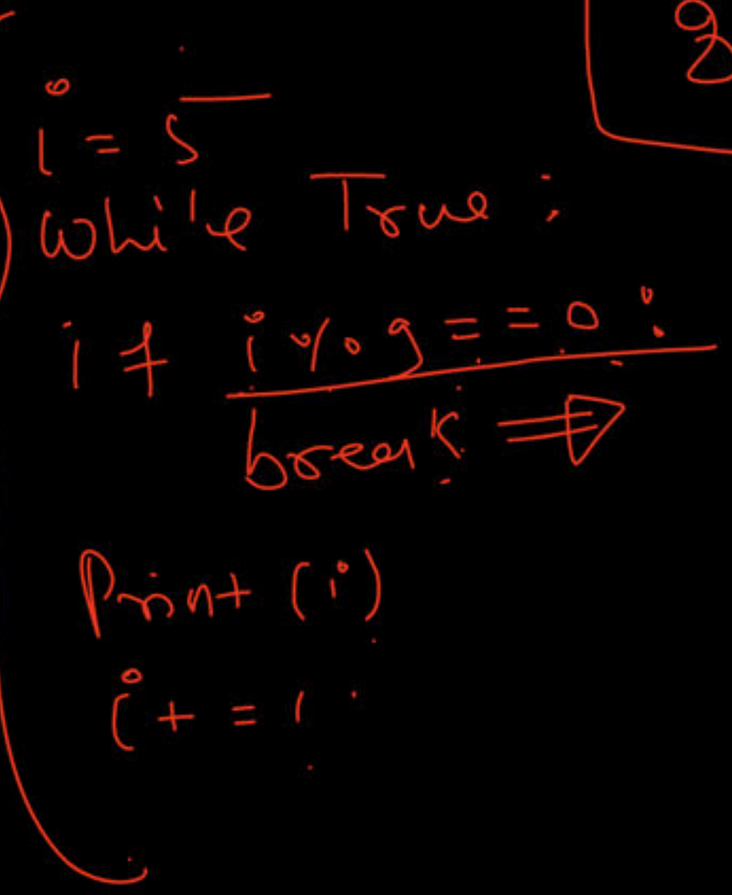
Syntax:
for<condition>:
statement1
if <condition>:
break
Example:
Msg='Welcome to Bharat'
for i in Msg:
print(i)
if(i=='e'):
break

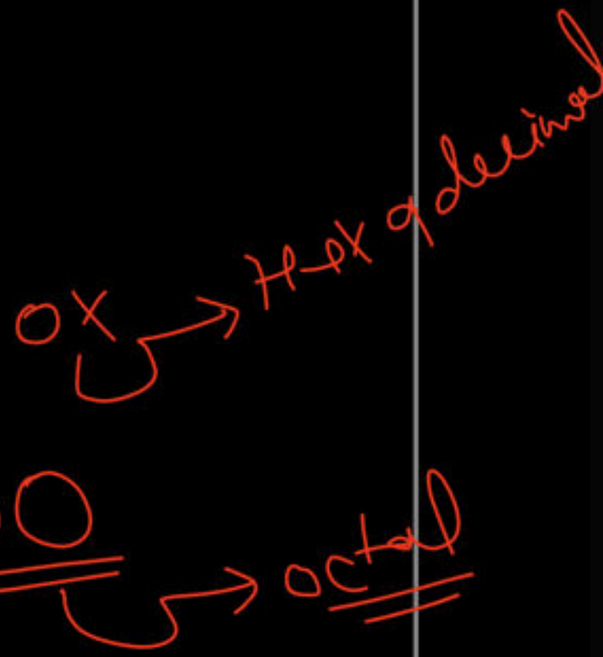
Let's Try...

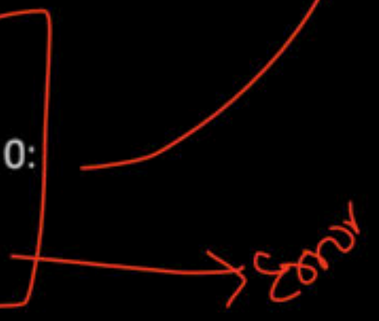
Guess Output?

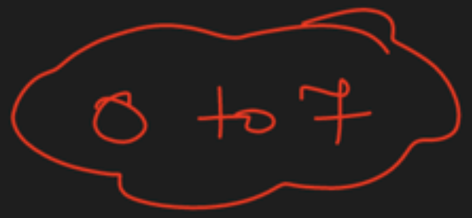
```
n=7
c=0
while(n):
   if(n>5):
      c=c+n-1
      n=n-1
   else:
      break
print(n)
print(c)

i = 1
while True:
    if i%3 == 0:
       break
    print(i)
    i + = 2
i = 5
while True:
   if i%009 == 0:
      break
   print(i)
   i += 1
```











Jumping Statement: continue

continue statement

Syntax:

```
for<condition>:
    if <condition>:
        continue
    statement1
```

Example:

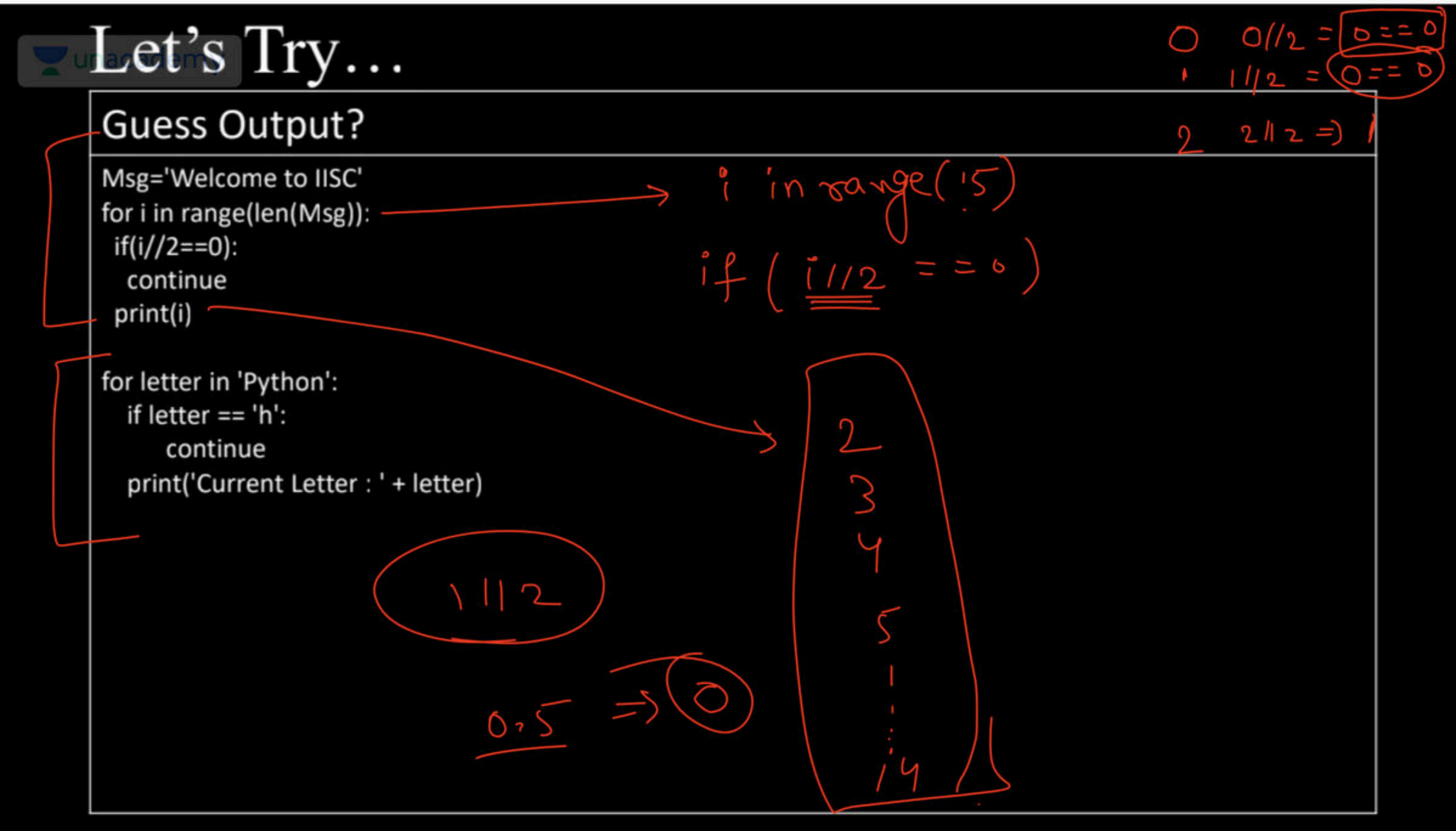
```
Msg='Welcome to Bharat'
for i in range (len(Msg)):
    if(i%2==0):
        continue
    print(i)
```

Let's Try…

Guess Output?

```
Msg='Welcome to IISC'
for i in range(len(Msg)):
 if(i//2==0):
   continue
 print(i)
```

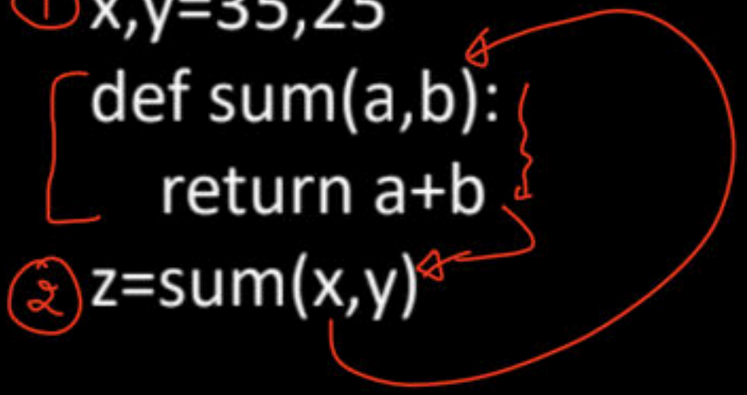
```
for letter in 'Python':
   if letter == 'h':
         continue
   print('Current Letter : ' + letter)
```



Jumping Statement: return

return statement

Syntax:
def function_name(argumentlist):
return statement
var=function_name(argumentlist)
Example:
① x,y=35,25
def sum(a,b):
return a+b
② z=sum(x,y)



Sample Problems

Problem-1:

Write python code to read an integer N from the user and try to print the following:
1234...N

Example

= N=4

Print the string: 1234

0 to N

range(N+1)

```
N = int(input("Enter the number:"))
for i in range(1, N+1)
    Print(i, end="")
```

Sample Problems

Problem-1:

Write python code to read an integer N from the user and try to print the following:
1234…N
Example
N=4
Print the string: 1234

1 2 3 4

```
# Read an integer N from the user
N = int(input("Enter an integer N: "))

# Iterate through numbers from 1 to N and print them without a newline
for i in range(1, N + 1):
    print(i, end="")

# Print a newline character to move to the next line
print()
```

1
2
3
4

print(i)

" "

Sample Problems

Problem-2:

Write a program that asks the user for a number and prints out that many stars on a line. Continue asking the user for new numbers until they decide not to.

Example:

Please enter a number: 4
****
Do you want to try again? (y/n) y
Please enter a number: 7
****
Do you want to try again? (y/n) n
Goodbye!

```
While True:
    N = int(input("Please Enter a number:"))
    Star = '*' x N
    Print(star)
    K = input("Do you want to try again(y/n)?")
    if K != y:
        Print("Good bye!")
        break
```

Sample Problems

Problem-2:

Write a program that asks the user for a number and prints out that many stars on a line. Continue asking the user for new numbers until they decide not to.

Example:

Please enter a number: 4
****

Do you want to try again? (y/n) y
Please enter a number: 7
****

Do you want to try again? (y/n) n
Goodbye!

```
while True:
    try:
        num = int(input("Please enter a number: "))
    except ValueError:
        print("Invalid input. Please enter a valid number.")
        continue

    stars = '*' * num
    print(stars)

    choice = input("Do you want to try again? (y/n): ").lower()
    if choice != 'y':
        print("Goodbye!")
        break
```

Sample Problems

Problem-3:

Write a program that asks the user for a number, and then outputs all the factors of the number.
Example:
Sample Input: Please enter a number: 18
Sample Output: Factors are 1 2 3 6 9 18

```
N = int(input("Please Enter a number:"))
for i in range(1, N+1)
    if(N % i == 0)
        Print(i, End=" ")
```

End

Sample Problems

Problem-3:

Write a program that asks the user for a number, and then outputs all the factors of the number.
Example:
Sample Input: Please enter a number: 18
Sample Output: Factors are 1 2 3 6 9 18

```python
# Get user input
try:
    num = int(input("Please enter a number: "))
except ValueError:
    print("Invalid input. Please enter a valid number.")
    exit(1)

# Initialize a list to store factors
factors = []

# Find factors of the number
for i in range(1, num + 1):
    if num % i == 0:
        factors.append(i)

# Output the factors
print("Factors are:", *factors)
```

Sample Problems

Problem-4:

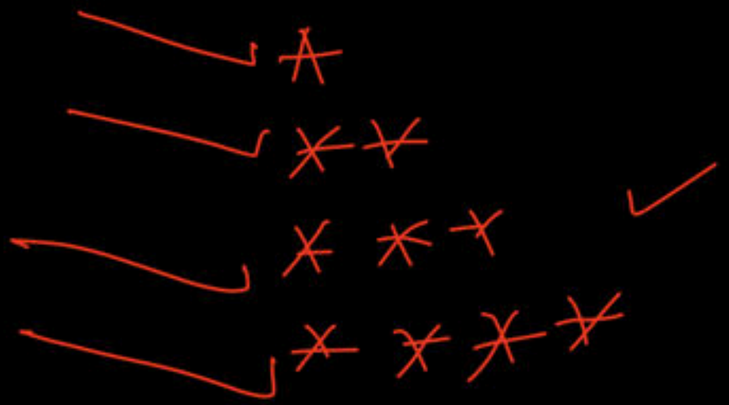
Write a program that asks the user for a number, and then outputs a pyramid of '*'s whose largest layer is the entered number.

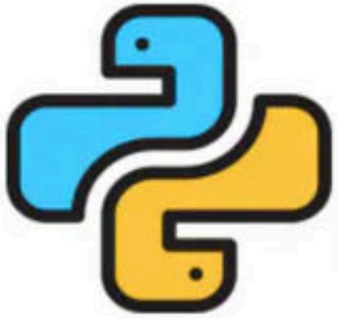
Example:

Sample Input: Please enter a number: 4

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

```
N = int(input("Enter a number: "))
for i in range(1, N+1):
    Print("*" x i)
```





Python Programming

String Manipulations

Content

- String
- String Indexing & Slicing
- String Methods
- Sample Problems

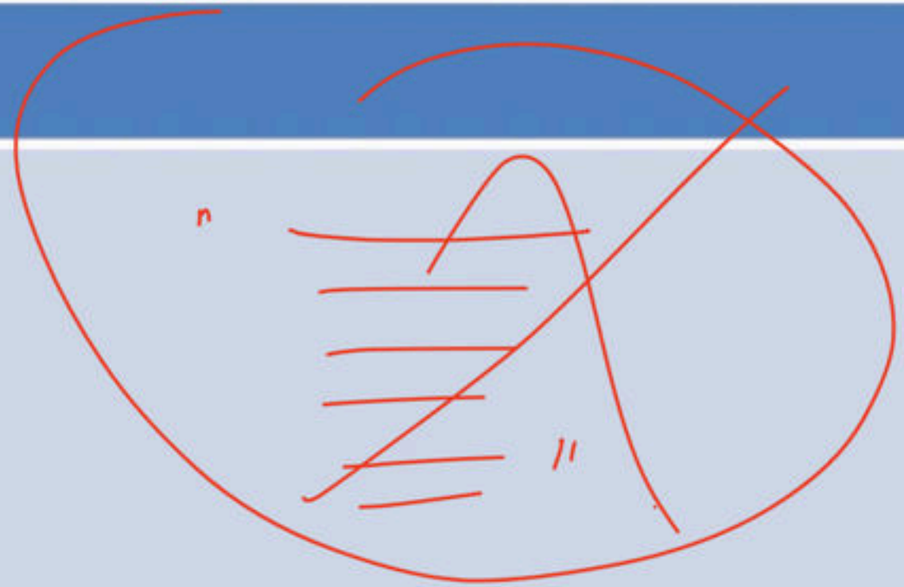
String

- In python, a string is defined as single character or group of characters.

  It is denoted as single quote ‘’ or double quote “”.

String Examples:

```
Name=‘Jhon’
Qualification=“Master of Technology”
print(Name)
print(Qualification)
print(‘Welcome to python string’)
print(“Hello STUDENTS…”)
```

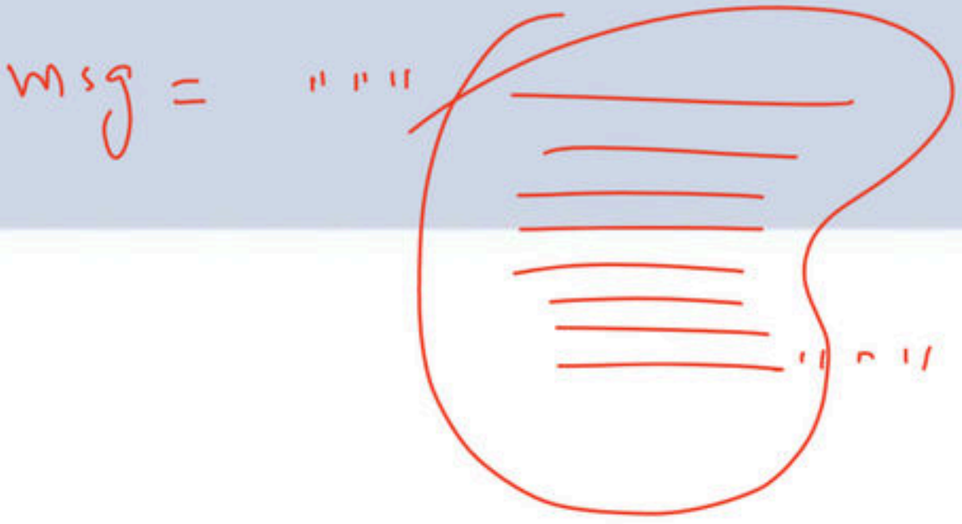


Multiline String

- String may be defined in multi-lines.

Multiline String Examples:

```
MSG = """Python is high level, object oriented programming. It  has many
advanced features such as library for machine and deep learning, IoT, Data
Science and many more"""
print(MSG)
```



unacademy

▲ 1 • Asked by Ankit

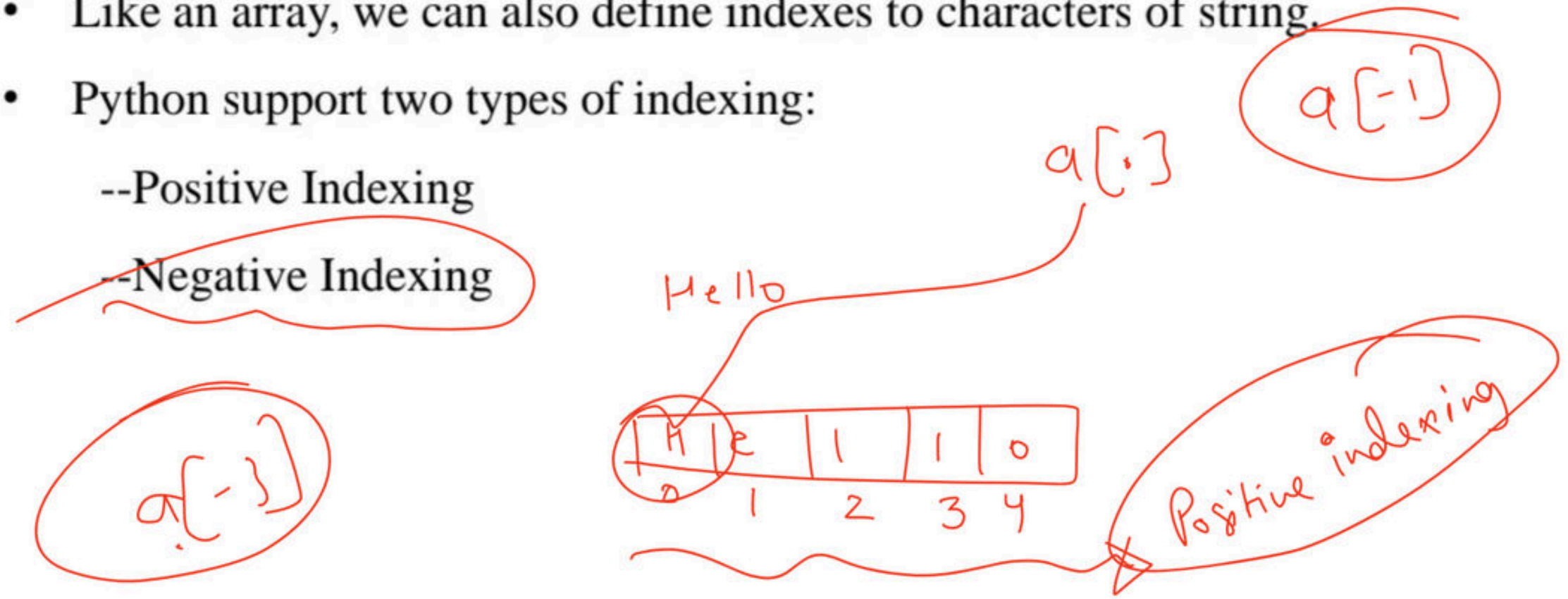
Sir yeh shi h?

String Indexing & Slicing

- Like an array, we can also define indexes to characters of string.
- Python support two types of indexing:

  --Positive Indexing

  --Negative Indexing

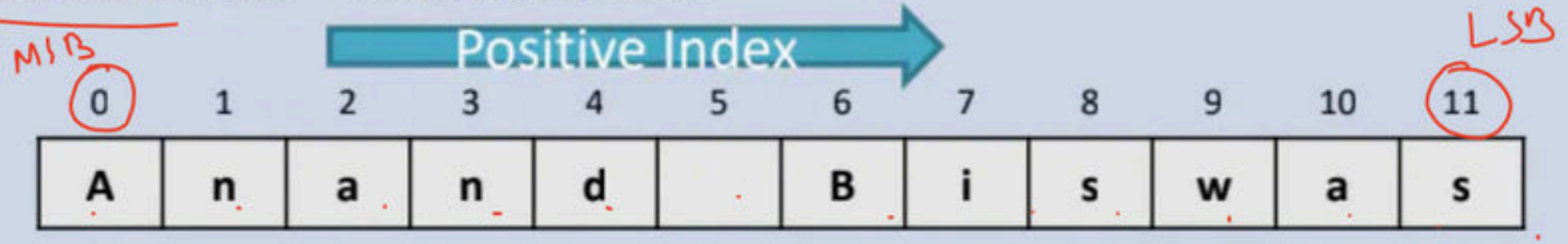


Positive Indexing

- It begin with zero and end with string length-1 from left hand side.

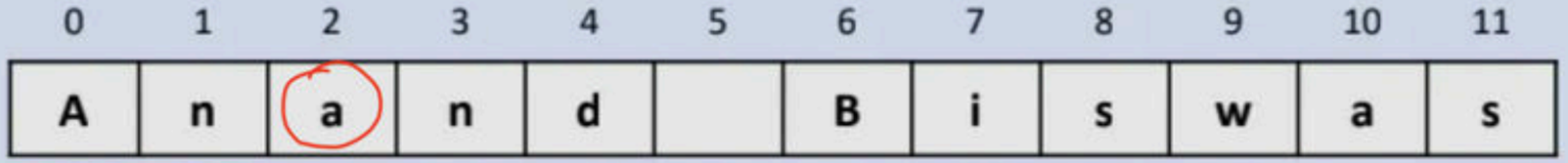
String Positive Indexing Examples:

StudentName="Anand Biswas"



Access String Character via Index:

StudentName="Anand Biswas"

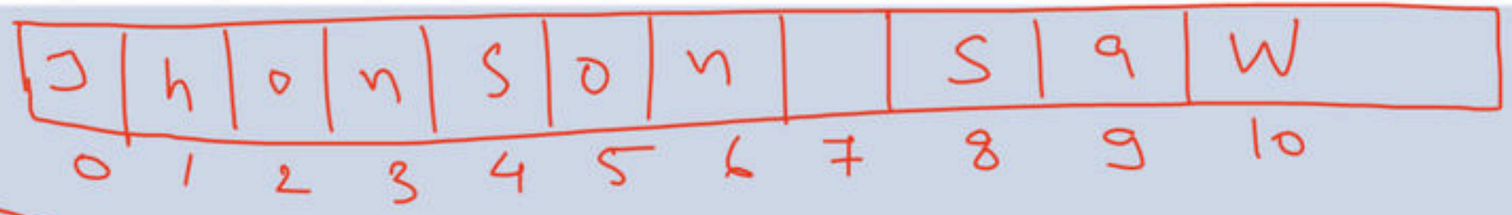


print(StudentName[2])

Let's Try…

Guess Output?

EmpName="Jhonson Saw"



1 print(EmpName[2+3]) → o

2 print(EmpName[10/3]) → Error

3 print(EmpName[10//3]) → n

4 print(EmpName[2**2]) → S

5 print(EmpName[2>>1]) → h

6 print(EmpName[2&4]) → J

7 print(EmpName[6^3]) → o

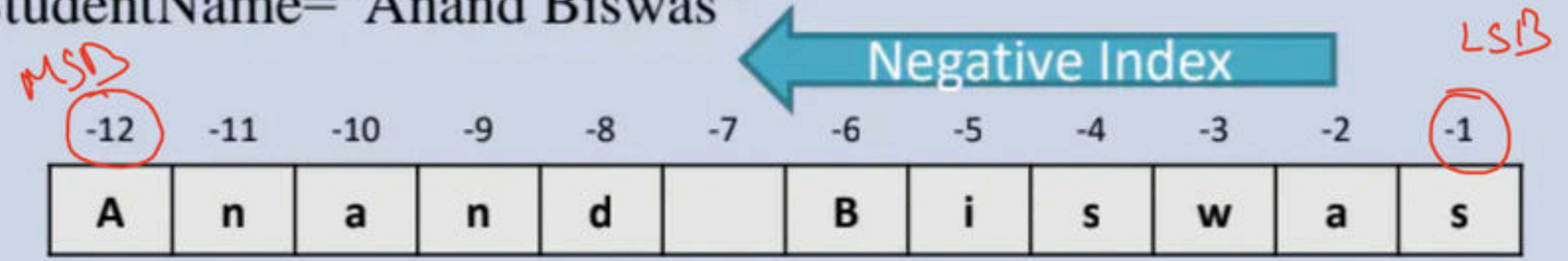
print(EmpName[int(10/3)]) → n

Negative Indexing

- It begin with -1 and end with string length in negative from right hand

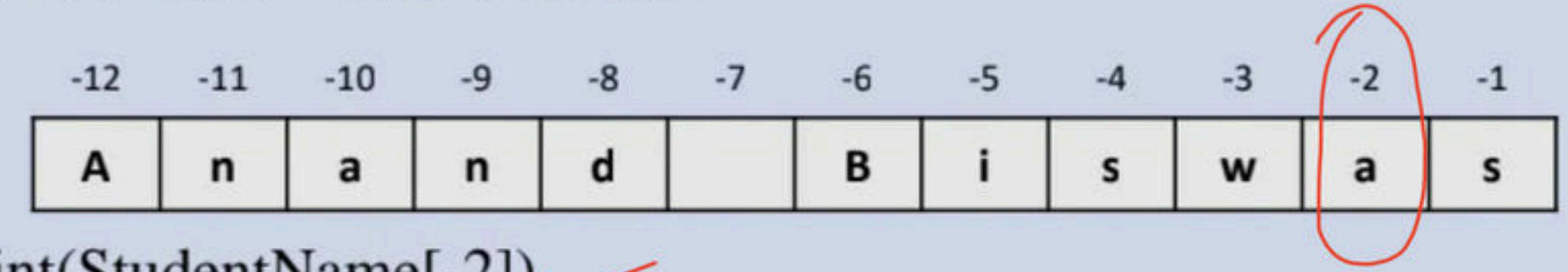
String Negative Indexing Examples:

StudentName="Anand Biswas"



Access String Character via Negative Index:

StudentName="Anand Biswas"



print(StudentName[-2])

Let's Try…

Guess Output?

College="Indian Institute of Technology"

1. print(College[-4]) l
2. print(College[-6-4]) T
3. print(College[-9+10]) n
4. print(College[14-9]) n
5. print(College[-8-3]) blank space
6. print(College[-4+4]) I
7. print(College[-12+6]) n

String Slicing

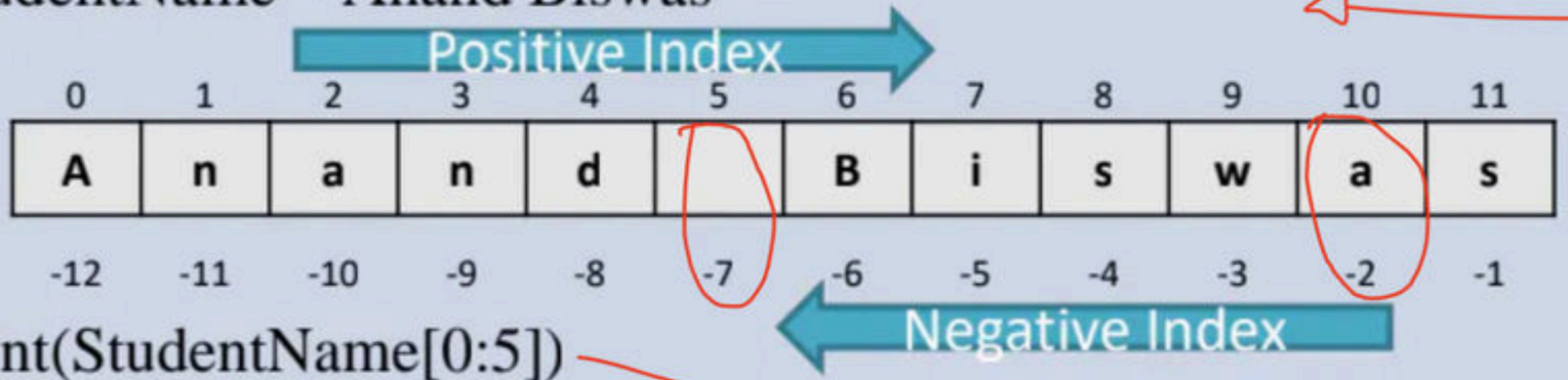
- It is way to access set of characters or substring from the given string. It is denotes as:

string_name[startindex : endindex]

- It slice string from startindex and end with endindex-1 i.e. it excludes endindex value. Slice index may be positive or negative range.

String Slicing Examples:

StudentName="Anand Biswas"



print(StudentName[0:5])

Anand

print(StudentName[-7:-2])

Biswa

Let's Try…

Guess Output?

College="Indian Institute of Technology"

print(College[1+1:4+0])

print(College[0:2**3])

print(College[-12:-7+3])

print(College[-1:-7])

print(College[:10])

print(College[:-6])

print(College[2:-2])

print(College[-8:14])

Concatenate String

- In python, we can merge or combine two or more string using + operator.

String Concatenate Examples:

```
Fname="Anand"
Lname="Biswas"
print(Fname+Lname)
print(Fname+" "+ Lname)
```

Let's Try…

Guess Output?

Fname="Anand"

Lname="Biswas"

1 print(Fname+ 5+ Lname) → Error

2 print(Fname+ '9'+ Lname) → Anand9Biswas

3 print(Fname+ 7.8+ Lname) → Error

4 print(Fname+ + Lname) → Error

5 print(Fname++ Lname) → Error

Print ("Anand" + "Biswas")

String Format

- In python, with the help of format() method we can concatenate any number with the string.

String Format Examples:

```
FName="Anand"
Lname="Biswas"
Age=36
print(Fname+Lname+36)
Name=Fname+Lname+" is {} year old"
print(Name.format(Age))
```

Print(Age)

String Format

String Format Examples:

Fname="Anand"

Lname="Biswas"

Year=36

Month=5

Name=Fname+Lname+" is {} year and {} month old"

print(Name.format(Age,Month))

String Methods

Name.endswith("a")

In python, several built-in methods are available.

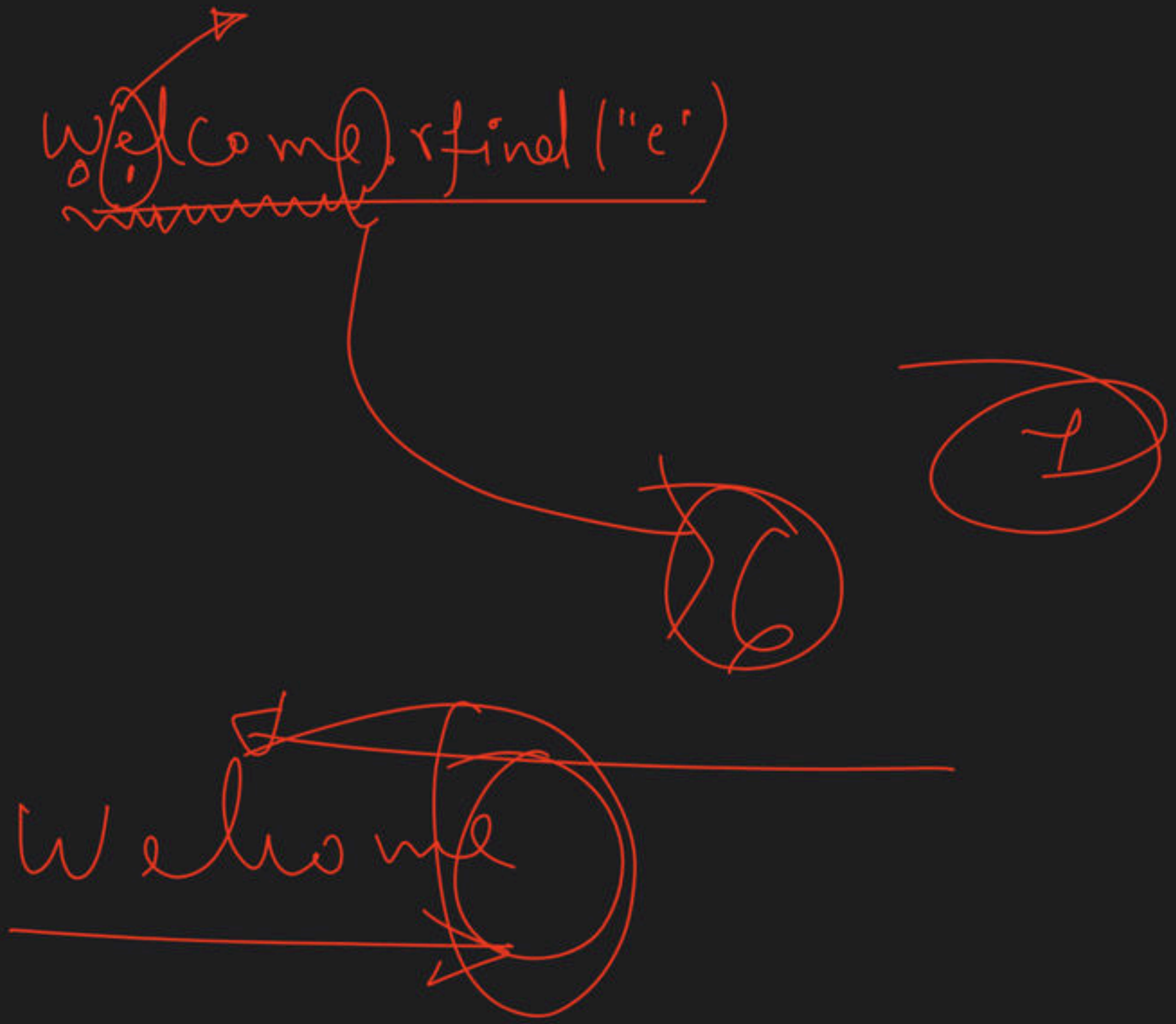
Method Name	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found

String Methods

Method Name	Description
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title

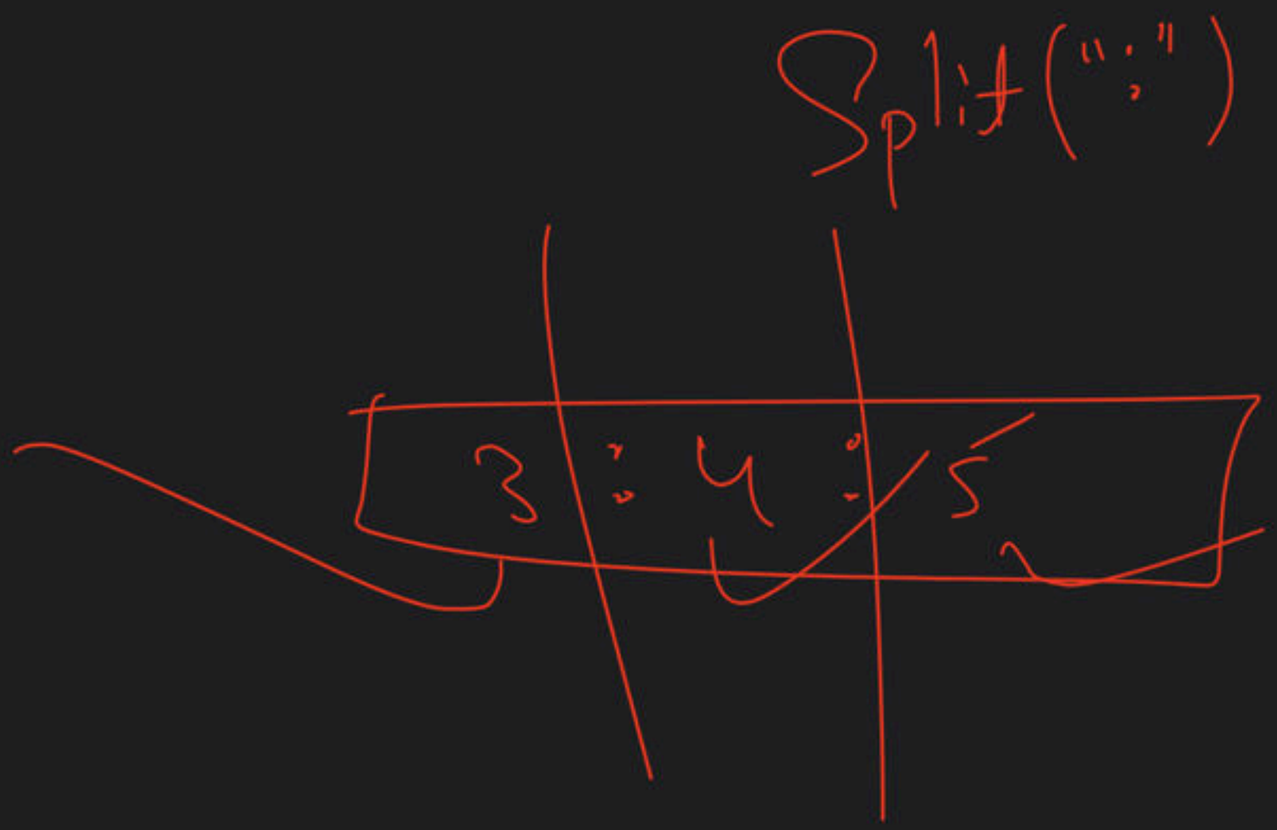
String Methods

Method Name	Description
isupper()	Returns True if all characters in the string are upper case
join()	Joins the elements of an iterable to the end of the string
ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string
maketrans()	Returns a translation table to be used in translations
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value
rfind()	Searches the string for a specified value and returns the last position of where it was found
rindex()	Searches the string for a specified value and returns the last position of where it was found



String Methods

Method Name	Description
rjust()	Returns a right justified version of the string
rpartition()	Returns a tuple where the string is parted into three parts
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string
split()	Splits the string at the specified separator, and returns a list
splitlines()	Splits the string at line breaks and returns a list
startswith()	Returns true if the string starts with the specified value
strip()	Returns a trimmed version of the string
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case



String Methods

capitalize( ):

```
Course="btech"
print(Course.capitalize( ))
```

Course = " btech hi"

→ Btech hi

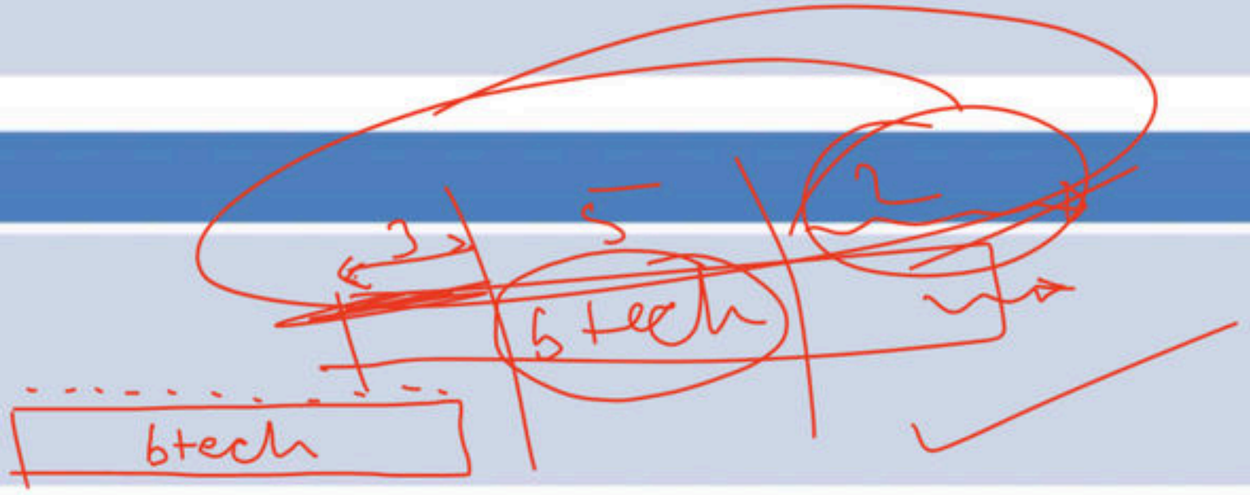
casefold( ):

```
Course="BTECH"
print(Course.casefold( ))
```

→ btech

center( ):

```
Course="btech"
print(Course.center(10))
```



String Methods

count( ):

```
Course="btech in cse"
print(Course.count('e'))
print(Course.count('e',4,10))
```

encode( ):

```
Course="BTECH"
print(Course.encode( ))
```

endswith( ):

```
Course="btech"
print(Course.endswith("."))
print(Course.endswith(".",2,7 ))
```

Let's Try…

Guess Output?

1.
```
str="hELLO"
print(str.capitalize())
```
→ Hello

2.
```
str="hELLo"
print(str.casefold())
```
→ hello

3.
```
str="hELLo how are you"
print(str.center(-4))
```
→ hELLo how are you

4.
```
str="hELLo how are you"
print(str.count('LL',1,12))
```
→ 1

5.
```
str="hELLo how are you"
print(str.count('o',-11,-2))
```
→ 1

6.
```
str="hELLo how are you!"
print(str.endswith('.'))
```
→ False

String Methods

expandtabs( ):

```
Course="btech\tin\tcse"
print(Course.expandtabs())
print(Course.expandtabs(2))
```

find( ):

```
Course="btech in cse"
print(Course.find(e))
```

index( ):

```
Course="btech in cse"
print(Course.index(e))
print(Course.index(a))
```

String Methods

isalnum( ):

```
Course="btechincse"
print(Course.isalnum())
Course="btech in cse"
print(Course.isalnum())
```

isalpha( ):

```
Course="btechincse"
print(Course.isalpha())
```

isascii( ):

```
Course="btech in cse"
print(Course.isascii())
```

Let's Try…

Guess Output?

```
str="hELLo\tHOW\tare\tyou!"
print(str.expandtabs(10.5))

str="hELLo\tHOW\tare\tyou!“
print(str.find('o' and 'O'))

str="hELLo\tHOW\tare\tyou!“
print(str.index('T'))

str="hELLo HOW are you 9"
print(str.isalnum())

str="hELLoHOWareyou910"
print(str.isalpha())

str="hELLo HOW are you 9"
print(str.isascii())
```

String Methods

isdecimal( ):

```
number="1234"
print(number.isdecimal())
number="a1234"
print(number.isdecimal())
```

isdigit( ):

```
number="102346"
print(number.isdigit())
```

isdentifier( ):

```
identifier="a b c"
print(identifier.isidentifier())
```

String Methods

islower( ):

```
str="hello"
print(str.islower())
str="Hello"
print(str.islower())
```

isnumeric( ):

```
number="10"
print(number.isnumeric())
```

isprintable( ):

```
str="a b c"
print(str.isprintable())
```

Let's Try…

Guess Output?

```
num='6.6'
print(num.isdecimal())

num='1 2 3 4 5 6'
print(num.isdigit())

ident='_abcx123'
print(ident.isidentifier())

str='helLo'
print(str.islower())

num='45667'
print(num.isnumeric())

str="Hello\tHi"
print(str.isprintable())
```

Sample Problems

Problem:

Write a python code to check and display valid and invalid identifier names given from the user.

Example

Sample Input- var1 _xvalue abc%xy PI CircleRaduis 67abx

Sample Output-

Valid Identifier- var1 _xvalue PI CircleRaduis

Invalid Identifier- abc%xy 67abx

Solution- https://colab.research.google.com/drive/1swpewE109qQdJW7zxn56HvfDOIxxWd_6

String Methods

isspace( ):

str="hello hru"

print(str.isspace())

str="HelloHRU"

print(str.isspace())

istitle( ):

str="Hello Hru"

print(str.istitle())

isupper( ):

str="HELLO"

print(str.isupper())

String Methods

join( ):

```
str="hello"
str1=" ".join(str)
print(str1)
```

ljust( ):

```
str="Hello"
print(str.ljust(6), "HRU")
```

lower( ):

```
str="HELLO"
print(str.lower())
```

Let's Try…

Guess Output?

```
str="  "
print(str.isspace())

str="Welcome to You"
print(str.istitle())

str="WELCOME_101"
print(str.isupper())

str="WELCOME_101"
str1="ABC".join(str)
print(str1)

str="WELCOME_101"
print(str.lower())
```

String Methods

lstrip( ):

```
str="      hello"
print(str.lstrip)
str = ",,,,,ssaaww.....hello"
print(str.lstrip(",.asw"))
```

maketrans( ):

```
str="Hello Hi"
s= str.maketrans("e","a")
print(str.translate(s))
```

partition( ):

```
str="Hello How Are You"
print(str.partition("How"))
```

String Methods

replace( ):

```
str=“ hello”
print(str.replace(“hello”,”Welcome”)
```

rfind( ):

```
str=“Hello Hi”
print(str.rfind(“H”))
```

rindex( ):

```
str=“Hello How Are You”
print(str.rindex(“l”))
```

Let's Try…

Guess Output?

```
str="    Hello     Hi"
print(str.lstrip())

str="Hello Hi"
print(str.partition(' '))

str="Hello hi"
print(str.replace("Hi","Jhon"))

str="Hello hi"
print(str.rfind("L"))

str="Hello hi"
print(str.rindex("h"))
```

String Methods

rjust( ):

```
str=“ hello”
print(str.rjust(30,”0”))
```

rpartition( ):

```
str=“Hello How Are You”
print(str.rpartition(“H”))
```

rsplit( ):

```
str=“Hello How Are You”
print(str.rsplit(“ ”))
```

String Methods

rstrip( ):

```
str="hello          "
print(str.rstrip())
```

split( ):

```
str="Hello How Are You"
print(str.split(" "))
```

splitlines( ):

```
str="Hello How Are You\n Welcome to Python Training"
print(str.splitlines())
```

Let's Try…

Guess Output?

```
str="Hello How Are You"
print(str.rpartition('H'))

str="Hello How Are You"
print(str.split(' '))

str="Hello How Are You"
print(str.rsplit('o'))

str="Hello How Are You\nWelcome Here"
print(str.splitlines())

str="    Hello   Hi   "
print(len(str))
print(str.rstrip())
print(len(str.rstrip()))
```

String Methods

startswith( ):

```
str="hello how are you"
print(str.startswith('h'))
```

swapcase( ):

```
str="Hello How Are You"
print(str.swapcase())
```

strip( ):

```
str="      Hello How Are You      "
print(str.strip())
```

String Methods

title( ):

```
str="hello how are you"
print(str.title())
```

upper( ):

```
str="Hello How Are You"
print(str.upper())
```

zfill( ):

```
str="Hello How Are You"
print(str.zfill(15))
```

Let's Try…

Guess Output?

```
str="Hello How Are You"
print(str.startswith('h'))

str="hELlO aRe yOu"
print(str.swapcase())

str="hELlO aRe yOu"
print(str.title())

str="hELlO aRe yOu"
print(str.upper())

str="hELlO aRe yOu"
print(str.zfill(10))
```

```
str="hello"
print(str.replace("hell0","Welcome"))
#str="Hello Hi"
#print(str.rfind("H"))
#str="Hello Hi"
#print(str.rindex("H"))
#str="hello"
#print(str.rjust(30,"0"))
```

```
#str="Hello How Are You"
#print(str.rpartition("H"))
#str="Hello How Are You"
#print(str.rsplit(" "))
#str="hello          "
#print(str.rstrip())
#str="Hello How Are You"
#print(str.split(" "))
#str="How Are You\n Welcome to Python Training"
#print(str.splitlines())
#str="hello how are you"
#print(str.startswith('e'))
```

```
#str="Hello How Are You"
#print(str.swapcase())
#str="       Hello How Are You       "
#print(str.strip())
#str="hello how are you"
#print(str.title())
#str="Hello How Are You"
#print(str.upper())
str="HELlO HOW ARE YOU"
print(str.zfill(10))
```

Thank You…

Python Programming

Function and It's Types

Content

- Function
- Function Types
- User Defined Functions
- Recursive Functions
- Sample Problems

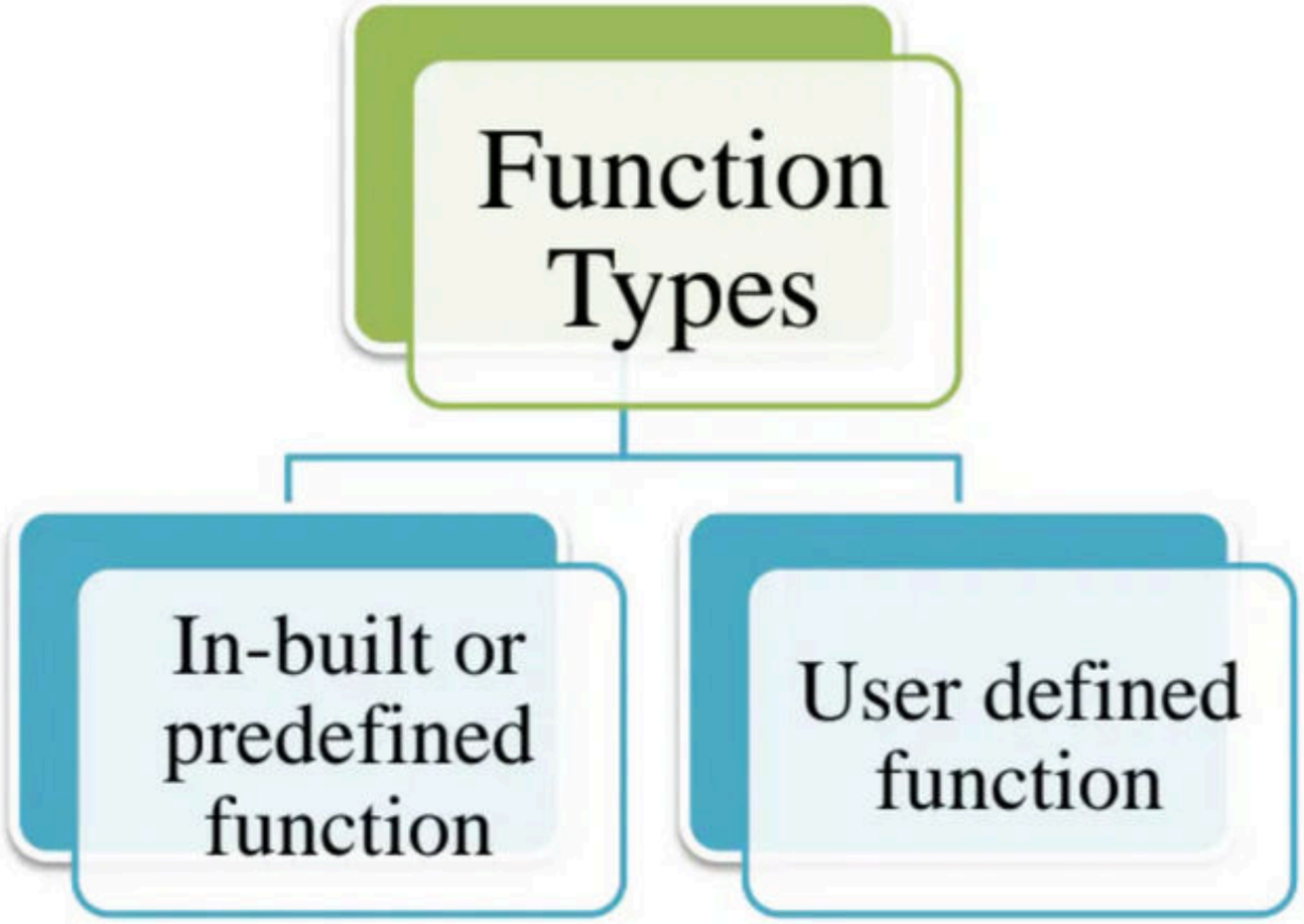
Function

A function is group of statements which are executed when it is called.

In-built Function Examples:	User Defined Function Example:
Name='Jhon' Qualification="Master of Technology" print(len(Name)) print(len(Qualification))	Def power(x,y): print(x**y) x,y=input('Enter two numbers: ').split(' ') x=int(x) y=int(y) power(x,y)

Function Types

In python, two types of function are exist.



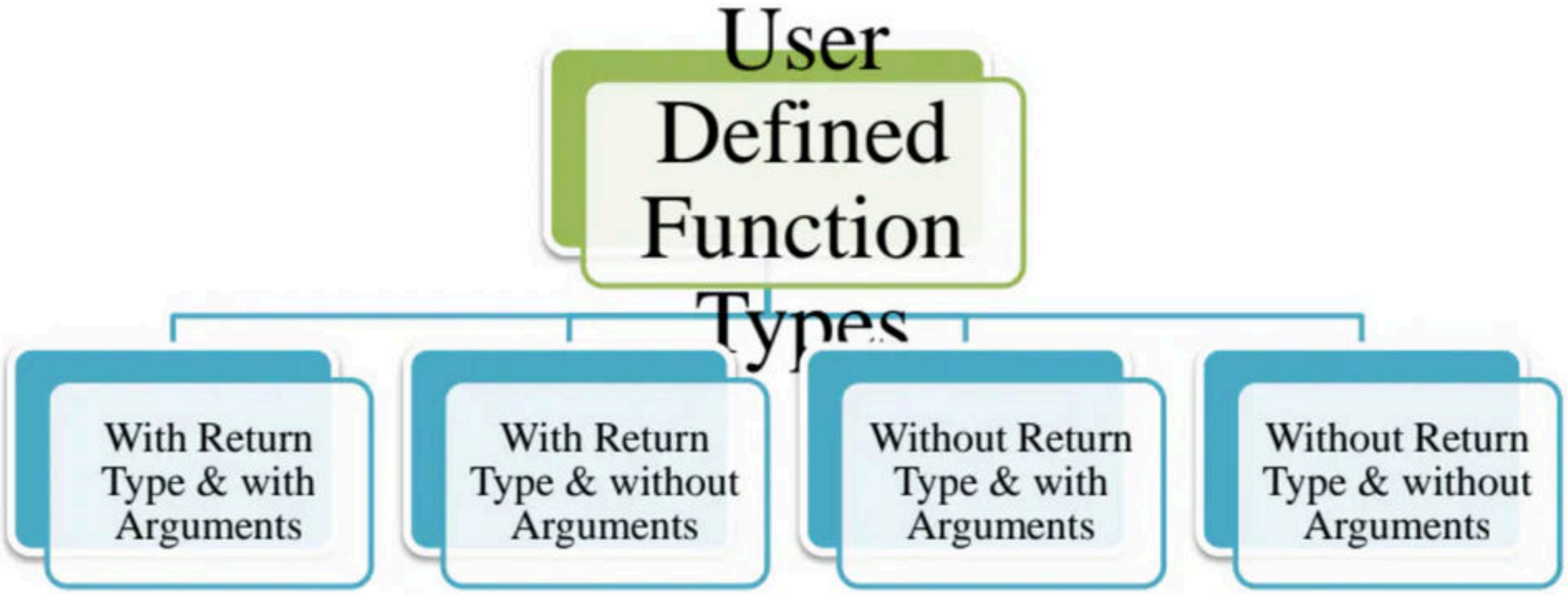
In-built Functions

There are many pre-defined or in-built functions.

General	String	List/Set/Tuples	Dictionary
Print()	Find()	Count()	Clear()
Len()	Index()	Remove()	Copy()
	Capitalize ()	Union()	Items()
	Lower()	Pop()	Keys()
	Strip()	Index()	Get()
	Isalnum ()		Pop()
	Isalpha()		Update()
	Center()		
	Isdigit()		

User Defined Functions

In python, we can also write our own function for specific needs i.e. known as user defined function.



With Return Type & with Arguments

Example

```
# Python Code for With Return types and with arguments
def Exp(x, y):
  return (x**y)
x=int(input('Enter first number:'))
y=int(input('Enter second number:'))
z=Exp(x,y)
print('Exponential is: ', z)
```

Output:

```
Enter first number:10
 Enter second number:4
 Exponential is: 10000
```

With Return Type & without Arguments

Example

```
# Python Code for With Return types and without arguments
def Exp():
  x=int(input('Enter first number:'))
  y=int(input('Enter second number:'))
  return (x**y)
z=Exp()
print('Exponential is: ', z)
```

Output:

```
Enter first number:15
Enter second number:3
Exponential is: 3375
```

Without Return Type & with Arguments

Example

```
# Python Code for Without Return types and with arguments
def Exp(x, y):
  print('Exponential is: ',x**y)
x=int(input('Enter first number:'))
y=int(input('Enter second number:'))
Exp(x,y)
```

Output:

```
Enter first number:15
Enter second number:4
Exponential is: 50625
```

Without Return Type & without Arguments

Example

```
# Python Code for Without Return types and without arguments
def Exp():
  x=int(input('Enter first number:'))
  y=int(input('Enter second number:'))
  print('Exponential is: ',x**y)
Exp()
```

Output:

```
Enter first number:8
Enter second number:2
Exponential is: 64
```

Recursive Function

When a function is called itself iteratively then it is known as recursive function.

Recursive Function Example:

```
# Python Code for recursive functions
def Fact(x):
  if x==1:
    return(x)
  else:
    return(Fact(x-1)*x)
x=int(input('Enter  number:'))
print(Fact(x))
```

Let's Try…

Guess Output?

College="Institute of Engineering and Technology"

print(College[1+1:4+0])

print(College[0:2**3])

print(College[-12:-7+3])

print(College[-1:-7)

print(College[:10])

print(College[:-6])

print(College[2:-2])

print(College[-8:14])

Thank You…

Sample Problems

Problem-4:

Write a program that asks the user for a number, and then outputs a pyramid of '*'s whose largest layer is the entered number.

Example:

Sample Input: Please enter a number: 4

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

```python
# Get user input
try:
    num = int(input("Please enter a number: "))
except ValueError:
    print("Invalid input. Please enter a valid number.")
    exit(1)

# Print the pyramid
for i in range(1, num + 1):
    print('*' * i)
```