Codes

1. Creating List
empty listmy_list = []
list of integersmy_list = [1, 2, 3]
list with mixed datatypesmy_list = [1, "Hello", 3.4]

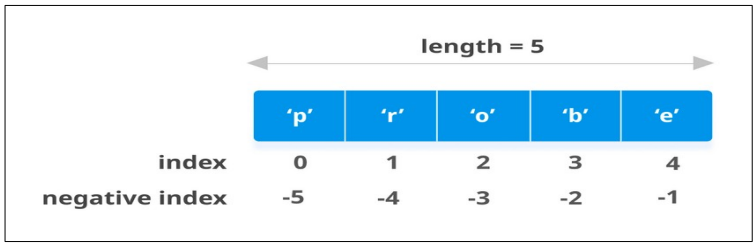
2. Fetching List Elements Using Index Number

```
my_list = ['p','r','o','b','e']

print(my_list[0]) # Output: p

print(my_list[2]) # Output: o

print(my_list[-1]) # Output: e
```



3. Slicing List

```
my_list = ['p','r','o','g','r','a','m','i','z']

print(my_list[2:5]) # elements 3rd to 5th

print(my_list[:-5]) # elements beginning to 4th

print(my_list[5:]) # elements 6th to end

# elements beginning to end

print(my_list[:])  #output p r o g r a m i z
```

4. Creating Tuple
empty tuple = ()
tuple of integersmy = (1, 2, 3)
tuple with mixed datatypesmy_list = (1, "Hello", 3.4)

5. Fetching Tuple Elements Using Index Number

```
my_tuple = ('p','r','o','b','e')

print(my_tuple[0]) # Output: p
```

```
print(my_tuple[2]) # Output: o
```

```
print(my_tuple[-1]) # Output:e
```

Slicing List

```
my_tuple = ['p','r','o','g','r','a','m','i','z']

print(my_tuple[2:5]) # elements 3rd to 5th

print(my_tuple[:-5]) # elements beginning to 4th

print(my_typle[5:]) # elements 6th to end

# elements beginning to end
```

```
print(my_tuple[:]) #output p r o g r a m i z
```

7. String Slicing Operations

```
str = 'GLS Univeristy'
s = '''This is a multiline
string'''
'''
hbvdsjhvsdgfhdsf
fhvdshfsdhfsdhf
dhfbdshfbdshf
'''
a=10
print (s)
print (str)
print (str[0])
print (str[2:5])
print (str[2:])
print (str * 2)
print (str + "TEST" )
```

7. #!usr/bin/python

```
str = 'GLS Univeristy'
s = '''this is a multiline string'''
a=10
print (s)
print (str)          # Prints complete string
print (str[0])       # Prints first character of the string
print (str[2:5])     # Prints characters starting from 3rd to 5th
print (str[2:])      # Prints string starting from 3rd character
print (str * 2)      # Prints string two times
print (str + "TEST" )# Prints concatenated string
# update strings
```

```
print "New String\nis",str[:4]+'Society'
'''
print "Following table is a (\a) list of \bescape or non-printable characters that \tcan be represented\nstring1_1.py with backslash notation""".
'''
print " following table is a \t list of \vescape \rcharacters"
print "the number is %E"% (a)
print r'usr//bin//python'
print u'usr//bin//python'
```

8. String In-built functions

```
#usr/bin/python
str='gls'
print str.capitalize()

str = "This is multiline text containing some characters sequence for testing of function";
print "str.center(100, 'a') : ", str.center(150, 'a')

print "str.count : ", str.count('i',1,100 )
str1 = "Encoded String is"+str.encode('base64','strict');
print str1

#str2 = "Decoded String is"+str1.decode('base64','strict');
#print str2
print str.upper()
print str.lower()
print str.find('a',1,100)
print len(str)
print min(str)
print max(str)
num=010201
```

10. Range Function

```
start = 2
stop = 14
step = 1

print(list(range(start, stop, step)))
```

11. Range Function

```
# empty range
print(list(range(0)))
```

```
# using range(stop)
print(list(range(10)))

# using range(start, stop)
print(list(range(1, 10)))
```

12. Range Function

```
start = 2
stop = -14
step = -2

print(list(range(start, stop, step)))

# value constraint not met
print(list(range(start, 14, step)))
```

13. String Alignment

```
'''
Number Formatting with alignment
<       Left aligned to the remaining space
^       Center aligned to the remaining space
>       Right aligned to the remaining space
=       Forces the signed (+) (-) to the leftmost position
'''

# integer numbers with right alignment
print("{:10d}".format(12))
# float numbers with center alignment
print("{:^10.3f}".format(12.2346))
# integer left alignment filled with zeros
print("{:<05d}".format(12))
# float numbers with center alignment
print("{:=8.3f}".format(-12.2346))

# string padding with left alignment
print("{:5}".format("cat"))
# string padding with right alignment
print("{:>5}".format("cat"))
# string padding with center alignment
print("{:^5}".format("cat"))
# string padding with center alignment
# and '*' padding character
print("{:*^5}".format("cat"))

# truncating strings to 3 letters
```

```
print("{:.5}".format("caterpillar"))
# truncating strings to 3 letters
# and padding
print("{:>5.3}".format("caterpillar"))
# truncating strings to 3 letters,
# padding and center alignment
print("{:^5.3}".format("caterpillar"))
```

13. String Function

```
s1 = "hello to python"
print(s1.capitalize())

s2 = "o"
print(s1.count(s2,3,15))
s3 = "l"
print(s1.count(s3))

s4 = "python"
print(s1.endswith(s4))
s5 = "to"
print(s1.endswith(s5,2,13))

print(s1.find(s5))
print(s1.find(s3,1,10))
print(s1.find("lo",1))

print(s1.isalnum())
s6="aaaaaa"
print(s6.isalnum())

t = "table112"
print(t.isalpha())
print(s1.isalpha())

s7="23456"
print(s6.isdigit())
print(s7.isdigit())

s1 = "hello to python"
s8="Python"
print(s1.islower())
print(s8.islower())

s9="PYTHON"
print(s8.islower())
print(s9.isupper())
```

```
s10="           "
print(s10.isspace())
print(s1.isspace())

print(len(s10))
print(len(s9))

print(s9.lower())

print(s1.upper())

print(s1.startswith("hello"))
print(s1.startswith("to",6,10))

s11="Welcome to Python"
print(s11.swapcase())

s12="    Hello World    "
print(s12.lstrip())
s13="@@@@@@@Helllloooo"
print(s13.lstrip('@'))

print(s12.rstrip())
```

14. Conversion

```
# int() conversion
print(int('1589'))
print(type(int('1589')))
print(int(3.1411552))

#float() conversion
print(float('5.22113'))
print(float(3))

#str() conversion
print(str(8.456891))
print(str([1,2,3,4]))
s1=str(8.456891)
s2=str([1,2,3,4])
print(s1+s2)


#list() conversion
print(list('Hello'))
print(list((1,2,3,4))) #tuple to list
```

```
#tuple() conversion
print(tuple('Hello'))
print(tuple([1,2,3,4]))  #list to tuple
```

15. {} as a palceholder

```
a = 10; b = 20
print('The value of a is {} and b is {}'.format(a,b))
print('I love {0} and {1}'.format('chocolates','donuts'))
print('Hi {student}, {greeting}'.format(greeting = 'Goodmorning', student = 'Rohan'))
#the curly braces {} are used as placeholders.

# default arguments
print("Hello {}, your balance is {}.".format("Rahul", 458.1254))
# positional arguments
print("Hello {0}, your balance is {1}.".format("Rahul", 458.1254))
# keyword arguments
print("Hello {name}, your balance is {blc}.".format(name="Rahul", blc=458.1254))
# mixed arguments
print("Hello {0}, your balance is {blc}.".format("Rahul", blc=458.1254))

# integer arguments
print("The number is:{0:d}".format(123))
# float arguments
print("The float number is:{0:f}".format(123.4567898))
# octal, binary and hexadecimal format
print("bin: {0:b}, oct: {0:o}, hex: {0:x}".format(12))


# integer numbers with minimum width
print("{:5d}".format(12))
# width doesn't work for numbers longer than padding
print("{:2d}".format(1234))
# padding for float numbers
print("{:8.2f}".format(12.2346))
# integer numbers with minimum width filled with zeros
print("{:05d}".format(12))
# padding for float numbers filled with zeros
print("{:08.3f}".format(12.2346))
```

16. Taking value from the user

```
xString = input("Enter a number: ")
x = int(xString)
yString = input("Enter a second number: ")
y = int(yString)
```

```
print('The sum of ', x, ' and ', y, ' is ', x+y, '.')
a=input("Enter the no");
print(a)
print(type(a))
b=raw_input("enter the no");
print(b)
print(type(b))
a1=input("enter the no")
print(type(a))
a1=eval(input("enter the no"))
print(type(a))

x = int(input("Enter an integer: "))
y = int(input("Enter another integer: "))
sum = x+y
print('The sum of ', x, ' and ', y, ' is ', sum, '.')
```

17. For Loop Example

```
#usr/bin/python

numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]
sum = 0
for val in numbers:
        sum = sum+val
print("The sum is", sum)

'''
'''
num=[1,2,3]
a=1;
for val in num:

        a=a*val
print a

num=[0,2,8,9,4,5]
for val in range(len(num)):
        print ("The no is",val)
print("The no",val)

name=['gls','law','society']
for i in range(len(name)):
        print("I like", name[i])

'''
```

```
'''
digits = [0, 1, 5]

for i in digits:
    print(i)
else:
    print("No items left.")

# Python program to find the factorial of a number provided by the user.
num = int(input("Enter a number: "))

factorial = 1

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)
```

18. Conditional Statement

```
print ("   M A I N - M E N U")
print ("1. Backup")
print ("2. User management")
print ("3. Reboot the server")

choice = input('Enter your choice [1-3] : ')

choice = int(choice)

if choice == 1:
      print ("Starting backup...")
elif choice == 2:
      print ("Starting user management...")
elif choice == 3:
      print ("Rebooting the server...")
else:
      print ("Invalid number. Try again...")
```

19. If Condition from list

```
    a=10
    b=20
    list=[10,20,30,40,50];
    if (a in list):
        print "a is in given list"
    else:
        print "a is not in given list"
    if(b not in list):
        print "b is not given in list"
    else:
        print "b is given in list"
```

19. Creating Dictionary

```
# creating a dictionary
country_capitals = {
  "Germany": "Berlin",
  "Canada": "Ottawa",
  "England": "London"
}

# printing the dictionary
print(country_capitals)
```

20. Accessing Dictionary Items

```
country_capitals = {
  "Germany": "Berlin",
  "Canada": "Ottawa",
  "England": "London"
}

# access the value of keys
print(country_capitals["Germany"])    # Output: Berlin
print(country_capitals["England"])    # Output: London
```

21. Add Items to Dictionary

```
country_capitals = {
  "Germany": "Berlin",
  "Canada": "Ottawa",
}

# add an item with "Italy" as key and "Rome" as its value
country_capitals["Italy"] = "Rome"

print(country_capitals)
```

22. List using loop

```
thislist = ["apple","banana","cherry"]
```

```
forx in thislist:
   print(x)
```

23. Loop Using List index number

```
thislist = ["apple", "banana", "cherry"]
for i in range(len(thislist)):
  print(thislist[i])
```

24. List using while loop

```
thislist = ["apple", "banana", "cherry"]
i = 0
while i < len(thislist):
  print(thislist[i])
  i = i + 1
```

25. Python function and calling

```
def greet():
    print('Hello World!')

# call the function
greet()

print('Outside function')
```

26. Function with argument list

```
def greet(name):
    print("Hello", name)

# pass argument
greet("John")
# function with two arguments
def add_numbers(num1, num2):
    sum = num1 + num2
    print("Sum: ", sum)
# function call with two values
add_numbers(5, 4)
```

27. Function with return statment

```
# function definition
def find_square(num):
    result = num * num
    return result
# function call
square = find_square(3)

print('Square:', square)
```

28. Function & Lambda Function

```
def cube(y):
    return y*y*y;

g = lambda x: x*x*x
print(g(7))

print(cube(5))
```

29. Variable Length Argument

```
def sum_all(*args):
    result = 0
    for num in args:
        result += num
    return result

print(sum_all(1, 2, 3, 4, 5))
```

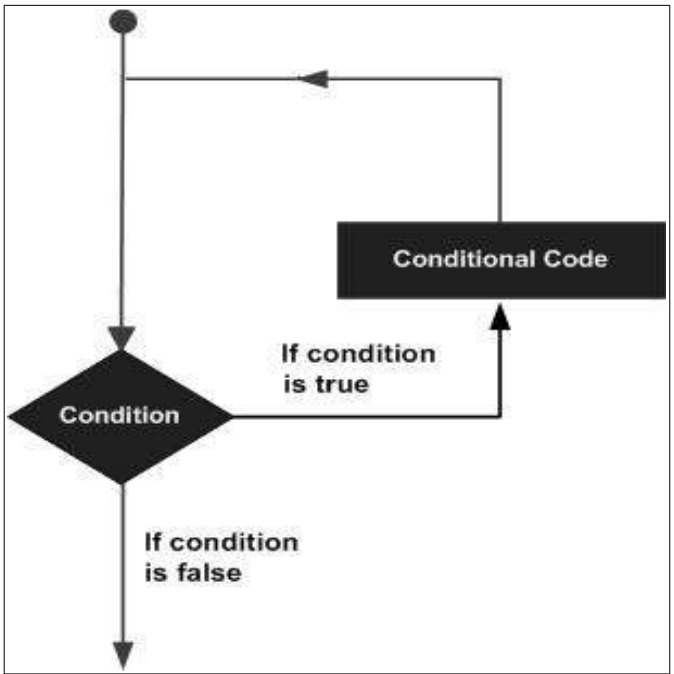
30. Variable Length argument for Dictionary

```
def print_args_and_kwargs(*args, **kwargs):
    print("Positional arguments:")
    for arg in args:
        print(arg)

    print("Keyword arguments:")
    for key, value in kwargs.items():
        print(f"{key}: {value}")

print_args_and_kwargs(1, 2, 3, name="Alice", age=30)
```

WHILE LOOP



Syntax

```
while expression:
    statement(s)
```

Example

```
count = 0
while (count < 3):
     count = count + 1
     print("Hello pyhon")
```

output

```
Hello pyhon
Hello pyhon
Hello pyhon
```

Syntax

```
for iterator_var in sequence:
   statements(s)
```

Example:1

```
print("List Iteration")
l = ["python", "object", "oriented"]
for i in l:
     print(i)
```

output

```
List Iteration
python
object
oriented
```

Example:2

```
print("\nTuple Iteration")
t = ("python", "object", "oriented")
for i in t:
     print(i)
```

output

```
Tuple Iteration
python
object
oriented
```

Example:3

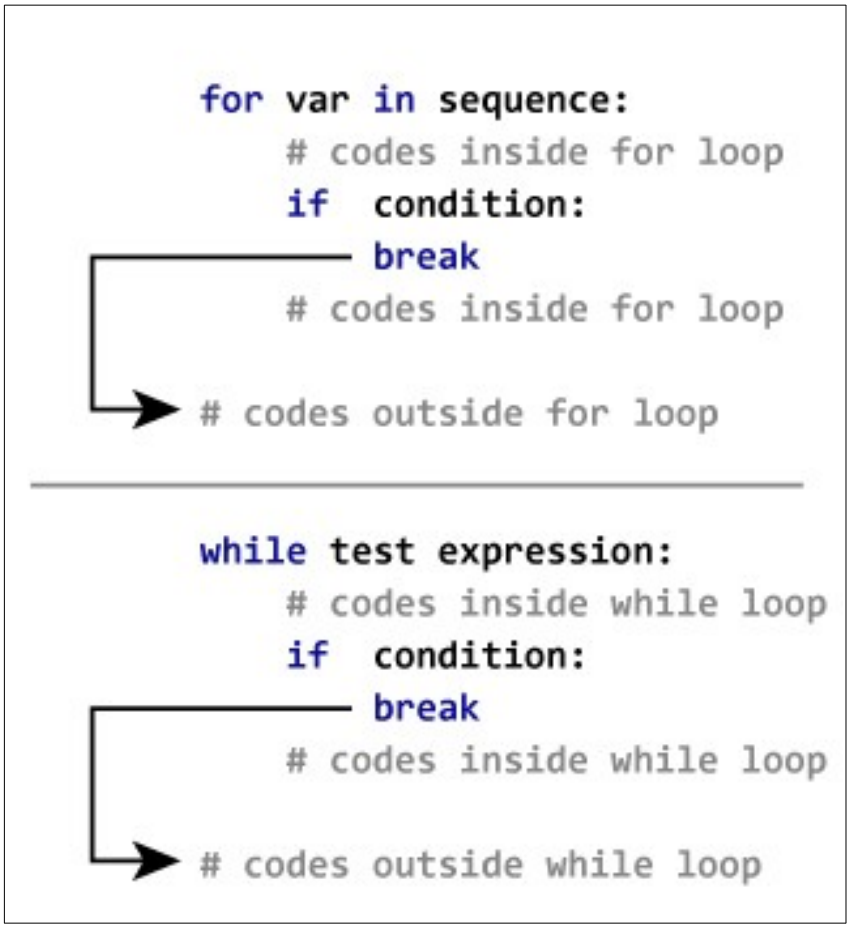
```
print("\nDictionary Iteration")
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
for key in dict:
     print (key, 'corresponds to', dict[key])
```

output
Dictionary Iteration
Name corresponds to Zara
Class corresponds to First
Age corresponds to 7

Example:4

```
print("\nString Iteration")
s = "python"
for i in s :
    print(i)
```

String Iteration
p
y
t
h
o
n




```
for val in "string":
    if val == "i":
        break
    print(val)
```

```
print("The end")
```