```
#to find the length of the string will use len() function
a="python is good programing language"
print(len(a))
```

```
34
```

```
#string slicing there will be two index starting index included ending
index excluded in python index start from zero
a="python is good programing language"
print(a[15:25])
```

```
programing
```

```
#strip() function will remove the white spaces from starting and
ending of the string and also rstrip() will remove white space from
from right end of the string
#lstrip() will remove white space from left end of the string
a=" python is good programing language "
print(a.strip())
b=" python is good programing language"
print(b.lstrip())
c="python is good programing language "
print(c.rstrip())
```

```
python is good programing language
python is good programing language
python is good programing language
```

```
#replace function will replace the specified string charaxcter with
new character
a="python is good programing language"
print(a.replace(" ",""))
```

```
pythonisgoodprograminglanguage
```

```
#count function will count the number of times occurance of the
specified character inside the function as argument
a="python is good programing language"
print(a.count("p"))
```

```
2
```

```
#index() function will return the starting index of the specified
character inside the function as argument
a="python is good programing language"
print(a.index("good"))
#index function gives an value error, if the substring is not present
```

```
to avoid this we use find() function
print(a.find("panda"))
```

```
10
-1
```

```
# To check whether only alphabets are present in the string use
isalpha() function
# To check whether only digits are present in the string use isdigit()
function
# To check whether both alphabets and numbers are present in the
string use isalnum() function
# To check whether only decimal are present in the string use
isdecimal() function

a="python is good programing language123"
a=a.replace(" ","")
print(a)
print(a.isalpha())
print(a.isdigit())
print(a.isalnum())
print(a.isdecimal())
```

```
pythonisgoodprograminglanguage123
False
False
True
False
```

```
#To convert all the character present inside the string to upper case
use the function upper()
#To convert all the character present inside the string to lower case
use the function lower()
#To convert all the word first letter uppercse we will use the functio
title()


a="Python is good programing language123"
a=a.upper()
print(a)
a=a.lower()
print(a)
a=a.title()
print(a)
```

```
PYTHON IS GOOD PROGRAMING LANGUAGE123
python is good programing language123
Python Is Good Programing Language123
```

```python
#islower() is a boolean function this will return true if the
character is lower case or else false
#isupper() is a boolean function this will return true if the
character is upper case or else false

a="A"
print(a.isupper())
print(a.islower())
```

```
True
False
```

```python
#split() function will convert the string into list by adding each
word with white spaces as an element inside the list
#join() function will convert the list into string by adding each word
with white spaces as an element inside the string
#If we are passing the string inside the list() function as an
argument it will add each character of the string as an element inside
the list
a="Python is good programing language123"
b=a.split()
print(b)
c=" ".join(b)
print(c)
```

```
['Python', 'is', 'good', 'programing', 'language123']
Python is good programing language123
```

```python
#append() function will add element inside the list to the end of the
list index
#insert() function will add element in the specified position ,to the
list,position and element are the two parameters of the insert()
function
#pop() function will remove the last element from the list
#remove() function will remove the specified element from the list
#extend() function will join the specified list to the another list
from end
#clear() function will remove all the elements which are existing in
the list and make the list empty
list_1=["benfgaluru","mangaluru","beluru"]
list_1.append("mysuru")
print(list_1)
list_1.insert(2,"chitradurga")
print(list_1)
list_1.pop()
print(list_1)
list_1.remove("mangaluru")
print(list_1)
```

```
list_2=["tumkur","hasan"]
list_1.extend(list_2)
print(list_1)
list_1.clear()
print(list_1)
```

```
['benfgaluru', 'mangaluru', 'beluru', 'mysuru']
['benfgaluru', 'mangaluru', 'chitradurga', 'beluru', 'mysuru']
['benfgaluru', 'mangaluru', 'chitradurga', 'beluru']
['benfgaluru', 'chitradurga', 'beluru']
['benfgaluru', 'chitradurga', 'beluru', 'tumkur', 'hasan']
[]
```

```
#add() function will come in set this function will add the element to
thne existing set
#update() function will add the complete new set to the existing set
and make it set by removing duplicates
#intersection() function will give the common elements in two sets
#union() function will give all the elements from two sets
#issubset function is boolean function it will return true if one set
is subset of another set
#issuperset function is boolean function it will return true if one
set is superset of another set
#pop() function will remove one element from the set randomly
a={1,2,3,3,4,5,6}
b={4,5,6,7,8}
a.add(8)
print(a)
a.update(b)
print(a)
a.intersection(b)
print(a)
a.union(b)
print(a)
print(b.issubset(a))
print(a.issuperset(b))
a.pop()
print(a)
```

```
{1, 2, 3, 4, 5, 6, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
True
True
{2, 3, 4, 5, 6, 7, 8}
```

```
#get() function will return the value of the specified key in the
```

```
dictionary
#update() function will update the existed key value to the new
value,we should pass the dictionary as argument i.e,{key:value}\
#pop() function will remove the specified key value from the
dictionary
#in the dictionary key should be unique
#clear function will remove all the key value pairs from the
dictionary and make it empty dictionary
a={"name":"shashank","age":"23","gender":"male"}
print(a.get("age"))
a.update({"age":24})
print(a)
a.pop("gender")
print(a)
a.clear()
print(a)
```

```
23
{'name': 'shashank', 'age': 24, 'gender': 'male'}
{'name': 'shashank', 'age': 24}
{}
```

```
#list comprehension
list_1=[1,2,3,4,5,6]
a=[x**2 if x%2==0 else x**3 for x in list_1]
print(a)
list_2=[["shashank","rashmi"],["subash","shivkumar"],
["prathap","prajwal"]]
list_3=[j for i in list_2 for j in i]
print(list_3)
```

```
[1, 4, 27, 16, 125, 36]
['shashank', 'rashmi', 'subash', 'shivkumar', 'prathap', 'prajwal']
```

```
#list comprehension
list_4=[(x,y) for x in range(3) for y in range(3) if x!=y]
print(list_4)
list_4=[x for x in range(10) if x>5 or (x<3 and x%2==0) ]
print(list_4)
```

```
[(0, 1), (0, 2), (1, 0), (1, 2), (2, 0), (2, 1)]
[0, 2, 6, 7, 8, 9]
```

```
#lambda functions
# add two numbers using lambda function
a=lambda x,y :x+y
print(a(10,20))
# get the square of a number using lambda function
```

```python
b= lambda x:x**2
print(b(20))
# check if the number is even or odd using lambda function
c=lambda x:"number is even" if x%2==0 else "number is odd"
print(c(21))
# check if the string is a palindrome using lambda function.
d=lambda x: "is palindrome" if x==x[::-1] else "not a palindrome"
print(d("12345"))
```

```
30
400
number is odd
not a palindrome
```

```python
#using lambda in filter(),map(),sorted() function
#filter() function
a=[1,2,3,4,5,6,7]
b=list(filter(lambda x:x%2==0,a)) #we should use boolean expression in
function which are used in filter function
print(b)
#map() function
d=list(map(lambda x:x**2,a))
print(d)
#sorted() function
list_n=[("shashank",1),("rashmi",2),("prathap",3)]
c=list(sorted(list_n,key=lambda x:x[1] ))
print(c)
dict_1={"shashank":1,"rashmi":2,"prathap":3}
e=sorted(dict_1.items(),key=lambda x:x[0])
print(dict(e))
```

```
[2, 4, 6]
[1, 4, 9, 16, 25, 36, 49]
[('shashank', 1), ('rashmi', 2), ('prathap', 3)]
{'prathap': 3, 'rashmi': 2, 'shashank': 1}
```

```python
#loops,for loop and while loop
#count the number of vowels in the string
count=0
a="hi bengaluru"
for i in a:
    if i in "aeiou":
        count=count+1
print(f"there are {count} vowels in the string")

#get the factorial of the number using while loop
n=int(input("enter the number"))
factorial=1
```

```
while(n>0):
    factorial=factorial*n
    n=n-1
print(f"the factorial of given number  is {factorial}")
```

```
there are 5 vowels in the string
enter the number 10
the factorial of given number  is 3628800
```

```
#file operations
# read the file,using read() function we can read all the lines of the
file and store it into variable as a string
#with readline() function we can read the first line from the file as
a string
#using readlines() function we can read all the lines of the file and
store it into variable as a list
with open("something.txt",'w') as f2:
    f2.write("Hello World!\n")
    f2.write("Hello World!\n")

with open("something.txt",'r') as f1:
    a=f1.read()
    b=f1.readline()
    c=f1.readlines()
print(a)
print(b)
print(c)
#to write/append to a file,
#write() function is used to write a string to a file.
#if file is opened in "w" (write) mode, the contents of the file are
overwritten
#if file is opened in "a" (append) mode, the string is appended to the
end of the file
with open("something2.txt",'w') as f2:
    f2.write("Hello World!\n")

with open("something2.txt","a") as f2:
    f2.write("Python is a programming language\n")
```

```
Hello World!
Hello World!


[]
```

```
### OS functions
import os
```

```
#to rename filename, use os.rename() function where first parameter is
original file name and second parameter is new file name
os.rename("something.txt","something_is_nothing.txt")
#to remove file, use os.remove() function where parameter is filename
os.remove("something_is_nothing.txt")
#to obtain the current working directory, use os.getcwd() function
current = os.getcwd()
print(current)
#to list all the files in the directory, use os.listdir() function. By
default, it takes current directory
print(os.listdir(current))
#to create a folder, use os.mkdir() function
os.mkdir("example")
#to check if a file exists in the current working directory, use
os.path.exists()
print(os.path.exists("something2.txt"))
```

```
C:\Users\aishwarya\shashi
['.ipynb_checkpoints', 'BangaloreTopCompanies.csv',
'cleaned_loan_data_set.csv', 'Data-Science-from-Scratch-First-
Principles-with-Python-by-Joel-Grus-z-lib.org_.epub_.pdf', 'data-
science-lifecycle-ebook.pdf', 'dictionary.py', 'fileoperation.ipynb',
'functions.ipynb', 'Hands-On-Machine-Learning-with-Scikit-Learn-and-
TensorFlow.pdf', 'Introduction to Machine Learning with Python
( PDFDrive.com )-min.pdf', 'loan_data_set.csv', 'loops.py',
'MachineLearningTomMitchell.pdf', 'nettapp_interview.txt',
'new_file.json', 'numpy.ipynb', 'onlinedeliverydata.csv',
'pandas.ipynb', 'pandas.pdf', 'pandas_new.csv', 'pandas_new.xlsx',
'practice.py', 'product_file.json', 'programs_practce.ipynb',
'project', 'python_practice_notes.txt', 'requirements.txt',
'revision.ipynb', 'RR.ipynb', 'sample _xl.xls', 'something.npy',
'something2.txt', 'subhash..show.ipynb', 'todaysNews.txt']
True
```

```
# functions
# function to find the factorial of a number
# def is the keyword used to create the function. Function will have
parameters and it should return. By default it returns None
def factorial(a):
    b=a
    factorial=1
    while(a>0):
        factorial=factorial*a
        a=a-1
    print(f"the factorial of given number {b}  is {factorial}")
factorial(int(input("enter the number")))
#function to check if a number is prime or not
def prime_number(a):
    for i in range(2,int(a**0.5)+1):
        if a%i==0:
            return "number is not a prime"
    return f"the number {a} is prime number"
print(prime_number(2))
```

```
enter the number 10

the factorial of given number 10  is 3628800
the number 2 is prime number
```

```
#classes
# class keyword is used to define classes in python.
# __init__() is used to initialise the variables in the class
# methods inside the class should have a self parameter to be able to
use the init variables.
class BankAccount:
    def __init__(self,account_number,name,city,balance=500):
        self.account_number=account_number
        self.name=name
        self.city=city
        self.balance=balance
    def deposit_amount(self,amount):
        self.amount=amount
        self.balance=self.balance+self.amount
        print(f"PARENT CLASS! amount of rupees {amount} updated to the
account {self.name},current balance is {self.balance} ")

#To inherit a class, pass the parent class as a parameter while
defining the child class
class SavingAccount(BankAccount):

    def __init__(self,account_number,name,city,balance=500):
        # super().__init__() has to be passed in the init function to
initialise,parent class variables
        # same parameters as parent class has to be given in
super().__init__() function
        super().__init__(account_number,name,city,balance)

    def withdraw(self,w_amount):
        if(self.balance>500 and w_amount<self.balance+500):
            self.balance=self.balance-w_amount
            print(f"amount of Rs {w_amount} succesfull ,your current
balance is {self.balance}")
        else:
            print("ERROR")

    ## method overriding - if same name is defined as function in both
parent and child class,
    ## if the object is created with child class, the child class
method is used
    def deposit_amount(self,amount):
        self.amount=amount
        self.balance=self.balance+self.amount
        print(f"CHILD CLASS! amount of rupees {amount} updated to the
account {self.name}")
        print(f"current balance is {self.balance}")

#shashank=BankAccount("0001","shashank","chitradurga")
rashmi=BankAccount("0002","rashmi","bengaluru")
#shashank.deposit_amount(2000)
rashmi.deposit_amount(1)
shashank=SavingAccount("0001","shashank","chitradurga")
shashank.deposit_amount(2000)
# shashank.withdraw(500)
```

```
PARENT CLASS! amount of rupees 1 updated to the account rashmi,current
balance is 501
CHILD CLASS! amount of rupees 2000 updated to the account shashank
current balance is 2500
```

```
## method overloading is not supported in python, the function with
most number of parameters is considered
class NumberOperation:
    def __init__(self):
        pass
    def sum_num(self,a,b):
        return a+b
    def sum_num(self,a,b,c):
        return a+b+c

a = NumberOperation()
# print(a.sum_num(1,2))
print(a.sum_num(1,2,3))
```

```
6
```

```
## decorators
## It is adding additional functionality to existing function
## The decorator function is the outer function and functionality is
added to inner function
## we should return function object for decorator function
import time
def calculate_time(func):
    def inner_function(*args):
        start_time=time.time()
        result=func(*args)
        end_time=time.time()
        time_taken=end_time-start_time
        print(f"time taken to execute the function is :{time_taken}
s")
        return result
    return inner_function

def multiply_num(a,b):
    time.sleep(3)
    return a*b

## either use this command or use @calculate_time before
multiply_num() function definition.
multiply_num=calculate_time(multiply_num)

print(multiply_num(10,20))
```

```
time taken to execute the function is :3.0035712718963623 s
200
```