In [8]:
```
dict={"Name":"Sudu","Age":22}
print(type(dict))
print(dict.items())
```

```
<class 'dict'>
dict_items([('Name', 'Sudu'), ('Age', 22)])
```

In [14]:
```
dict={"Name":"Sudu","Age":22}
dict["Place"]="Shivamogga"
print(dict)
dict.update({"Place":"Mysuru"})
print(dict)
```

```
{'Name': 'Sudu', 'Age': 22, 'Place': 'Shivamogga'}
{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
```

In [20]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
dict2=dict.copy()
print(dict)
print(dict2)
```

```
{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
```

In [26]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
dict.clear()
print(dict)
```

```
{}
```

In [3]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
print(dict.keys())
```

```
dict_keys(['Name', 'Age', 'Place'])
```

In [32]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
print(dict.values())
```

```
dict_values(['Sudu', 22, 'Mysuru'])
```

In [12]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
for items in dict.values():
    print(items)
```

```
Sudu
22
Mysuru
```

In [16]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
for items in dict.keys():
    print(items)
```

```
Name
Age
Place
```

In [2]:
```
dict={'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}
del dict["Place"]
print(dict)
```

```
{'Name': 'Sudu', 'Age': 22}
```

In [18]:
```
dict={'Name': 'Sudu', 'Age': 22}
a=dict.get('Name')
print(a)
```

```
Sudu
```

In [4]:
```
dict={'Name': 'Sudu', 'Age': 22,'College': 'NIE'}
dict.pop("College")
print(dict)
```

```
{'Name': 'Sudu', 'Age': 22}
```

2. Dictonary in List

In [35]:
```
x=[{'Name': 'Sudu', 'Age': 22}]
y=[{'Place': 'Mysuru'}]
x.extend(y)
print(x)
```

```
[{'Name': 'Sudu', 'Age': 22}, {'Place': 'Mysuru'}]
```

In [17]:
```
x=[{'Name': 'Sudu', 'Age': 22}]
print(type(x))
```

```
<class 'list'>
```

In [21]:
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
x.append({'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'})
print(x)
```

```
[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}, {'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'}]
```

In [23]:
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
y=x.copy()
print(y)
```

```
[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
```

In [128…
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru', 'College': 'NIE'},
   {'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga', 'College': 'NIE'}]
for i in x:
    print(i["Name"],i["Age"])
```

```
Sudu 22
Ram 23
```

In [25]:
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'},
   {'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'}]
print(len(x))
```

```
2
```

In [27]:
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'},
   {'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'}]
x.reverse()
print(x)
```

```
[{'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'}, {'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
```

In [37]:
```
x=[{'Name': 'Ram', 'Age': 23, 'Place': 'Shivamogga'},
   {'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
```

```
x.pop(0)
print(x)
```

```
[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
```

In [39]:
```
x=[{'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'}]
x.remove({'Name': 'Sudu', 'Age': 22, 'Place': 'Mysuru'})
print(x)
```

```
[]
```

In [43]:
```
word=input("Enter a word :")
dict={}
for i in word:
    if i in dict:
        dict[i]+=1
    else:
        dict[i]=1
print(str(dict))
#for i in dict:
 #   print({ i : dict[i]})
```

```
Enter a word : Hello
{'H': 1, 'e': 1, 'l': 2, 'o': 1}
```

3. Sets

In [217…
```
a={1,2,3,4}
print(type(a))
```

```
<class 'set'>
```

In [199…
```
a={1,2,3,4}
a.add(5)
print(a)
```

```
{1, 2, 3, 4, 5}
```

In [201…
```
a={1,2,3,4}
a.update([6,7])
print(a)
```

```
{1, 2, 3, 4, 6, 7}
```

In [55]:
```
a={1, 2, 3, 4, 5, 6, 7}
a.remove(7)
print(a)
```

```
{1, 2, 3, 4, 5, 6}
```

In [63]:
```
a={1, 2, 3, 4, 5, 6}
a.discard(4)
print(a)
```

```
{1, 2, 3, 5, 6}
```

In [215…
```
a={1, 2, 3, 4, 6, 7}
a.clear()
print(a)
```

```
set()
```

In [221…
```
a={1,2,3,4}
b={5,6,7,8}
c=a.union(b)
print(c)
```

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

In [223…
```
a={1,2,3,4}
b={2,6,4,8}
c=a.intersection(b)
print(c)
```

```
{2, 4}
```

In [225…
```
a={1,2,3,4}
b={2,6,4,8}
c=a.difference(b)
print(c)
```

```
{1, 3}
```

In [5]:
```
a={1,2,3,4}
b={2,6,4,8}
```

```
c=b.difference(a)
print(c)
```

```
{8, 6}
```

In [6]:
```
a={1,2,3,4}
b={2,4}
c=b.issubset(a)
print(c)
```

```
True
```

In [245…
```
a={1,2,3,4}
b={5,6,7,8}
c=a.isdisjoint(b)
print(c)
```

```
True
```

4. Tuples

In [248…
```
t=(1,2,3,4,5,6)
print(type(t))
```

```
<class 'tuple'>
```

In [250…
```
t=(1,2,3,4,5,6)
print(t.index(3))
```

```
2
```

In [256…
```
t=(1,2,3,4,3,5,2,2)
print(t.count(2))
```

```
3
```

In [258…
```
t=(1,2,3,4,5,6)
print(len(t))
```

```
6
```

In [264…
```
t=(1,2,3,4,5,6)
print(t[1])
```

```
2
```

5. Class

In [273…
```
class faculty:
    def getdata(a):
        a.id=input("Enter Faculty ID :")
        a.name=input("Enter Faculty name :")
        a.salary=input("Enter Faculty Salary:")
    def putdata(a):
        print("\nFaculty ID is ",a.id)
        print("Faculty Name is ",a.name)
        print("Faculty Salary is ",a.salary)
a=faculty()
a.getdata()
a.putdata()
```

```
Enter Faculty ID : 100
Enter Faculty name : Sudu
Enter Faculty Salary: 1922992929

Faculty ID is  100
Faculty Name is  Sudu
Faculty Salary is  1922992929
```

In [321…
```
class person:
    def __init__(a,id,name,salary):
        a.id=id
        a.name=name
        a.salary=salary
    def putdata(a):
        print("\nFaculty ID is",a.id)
        print("Faculty Name is",a.name)
        print("Faculty Salary is",a.salary)
p1=person(100,"Sudu",100000)
p1.putdata()
```

```
Faculty ID is 100
Faculty Name is Sudu
Faculty Salary is 100000
```

In [4]:
```python
class student:
    def __init__(a,id,name,marks):
        a.id=id
        a.name=name
        a.marks=marks
    def avg(a):
        return sum(a.marks)/len(a.marks)
    def putdata(a):
        print("Student ID is",a.id)
        print("Student Name is",a.name)
        print("Student marks is",a.marks)

stu=student(1,"Sudu",[50,48,49,45,44,47])
stu.putdata()
print("Student Average marks is",stu.avg())
```

```
Student ID is 1
Student Name is Sudu
Student marks is [50, 48, 49, 45, 44, 47]
Student Average marks is 47.166666666666664
```

In [48]:
```python

```

```
Enter the number of students 2

Enter the details of student 1
Enter the name of Student sudu
Enter the marks of subject 34
Enter the marks of subject 45
Enter the marks of subject 56

Details of Student  1 is
ram got  [34, 45, 56]
ram got  [34, 45, 56]
45.0

Enter the details of student 2
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
Cell In[48], line 20
     18 s = students(name,[{}])
     19 print("\nEnter the details of student",i+1)
---> 20 name = input("Enter the name of Student")
     21 s.enterMarks()
     22 print("\nDetails of Student ",i+1,"is")

File ~\AppData\Roaming\Python\Python312\site-packages\ipykernel\kernelbase.py:1282, in Kernel.raw_input(self, prompt)
   1280     msg = "raw_input was called, but this frontend does not support input requests."
   1281     raise StdinNotImplementedError(msg)
-> 1282 return self._input_request(
   1283     str(prompt),
   1284     self._parent_ident["shell"],
   1285     self.get_parent("shell"),
   1286     password=False,
   1287 )

File ~\AppData\Roaming\Python\Python312\site-packages\ipykernel\kernelbase.py:1325, in Kernel._input_request(self, prompt, iden
t, parent, password)
   1322 except KeyboardInterrupt:
   1323     # re-raise KeyboardInterrupt, to truncate traceback
   1324     msg = "Interrupted by user"
-> 1325     raise KeyboardInterrupt(msg) from None
   1326 except Exception:
   1327     self.log.warning("Invalid Message:", exc_info=True)

KeyboardInterrupt: Interrupted by user
```

In [12]:

```
class Student:
    def __init__(self, name):
        self.name = name
        self.marks = []

    def enterMarks(self):
        for i in range(3):
            m = int(input(f"Enter subject {i+1} marks: "))
            self.marks.append(m)
```

```
    def display(self):
        print(f"{self.name} got {self.marks}")

    def average(self):
        return sum(self.marks) / len(self.marks)

n = int(input("Enter the number of students: "))

for i in range(n):
    print(f"\nEnter the details of student {i+1}")
    name = input("Enter the name of Student: ")
    s = Student(name)
    s.enterMarks()
    print(f"\nDetails of Student {i+1} is:")
    s.display()
    print(f"Average marks: {s.average()}\n")
```

```
Enter the number of students:  2

Enter the details of student 1
Enter the name of Student:  Subbu
Enter subject 1 marks:  50
Enter subject 2 marks:  49
Enter subject 3 marks:  48

Details of Student 1 is:
Subbu got [50, 49, 48]
Average marks: 49.0


Enter the details of student 2
Enter the name of Student:  Ram
Enter subject 1 marks:  49
Enter subject 2 marks:  48
Enter subject 3 marks:  47

Details of Student 2 is:
Ram got [49, 48, 47]
Average marks: 48.0
```

In [90]:
```
#Using Functions
class complex:
    def __init__(self,real,img):
        self.real=real
        self.img=img

    def add(self,number):
        self.real=self.real+number.real
        self.img=self.img+number.img
        result=complex(self.real,self.img)
        return result

    def sub(self,number):
        self.real=self.real-number.real
        self.img=self.img-number.img
        result=complex(self.real,self.img)
        return result

    def mul(self, number):
        result_real = self.real * number.real - self.img * number.img
        result_img = self.real * number.img + self.img * number.real
        return complex(result_real, result_img)

    def div(self, number):
        denominator = number.real ** 2 + number.img ** 2
        result_real = (self.real * number.real + self.img * number.img) / denominator
        result_img = (self.img * number.real - self.real * number.img) / denominator
        return complex(result_real, result_img)

    def display(self):
        print(f"{self.real} + {self.img}i")

n1=complex(5,6)
n2=complex(-6,7)

print("Addition is")
result=n1.add(n2)
result.display()

print("Subtraction is")
```

```
result=n1.sub(n2)
result.display()

print("Multiplication is")
result=n1.mul(n2)
result.display()
#(a+bi)×(c+di)=(ac-bd)+(ad+bc)i

print("Division is")
result=n1.div(n2)
result.display()
#(a+bi)/(c+di)=((ac+bd)+(bc-ad)i)/(c^2+d^2)
```

```
Addition is
-1 + 13i
Subtraction is
5 + 6i
Multiplication is
-72 + -1i
Division is
0.1411764705882353 + -0.8352941176470589i
```

In [16]:
```
#Using init Function
class complex:
    def __init__(s,real,img):
        s.real=real
        s.img=img

    def __add__(s,num):
        s.real=s.real+num.real
        s.img=s.img+num.img
        result=complex(s.real,s.img)
        return result

    def __sub__(s,num):
        s.real=s.real-num.real
        s.img=s.img-num.img
        result=complex(s.real,s.img)
        return result
```

```
    def __mul__(s, num):
        result_real = s.real * num.real - s.img * num.img
        result_img = s.real * num.img + s.img * num.real
        return complex(result_real, result_img)

    def __truediv__(s, num):
        denominator = num.real ** 2 + num.img ** 2
        result_real = (s.real * num.real + s.img * num.img) / denominator
        result_img = (s.img * num.real - s.real * num.img) / denominator
        return complex(result_real, result_img)

    def __str__(s):
        return (f"{s.real} + {s.img}i")

n1=complex(5,6)
n2=complex(-6,7)

print("Addition is",n1+n2)

print("Subtraction is",n1-n2)

print("Multiplication is",n1*n2)

print("Division is",n1/n2)
```

```
Addition is -1 + 13i
Subtraction is 5 + 6i
Multiplication is -72 + -1i
Division is 0.1411764705882353 + -0.8352941176470589i
```

In [ ]: