```
# 29

def my_func(var_x):
  mylist=[]

  for x in range (0,var_x):
    mylist.append(input("Enter the argument values: "))

  for x in range(0,var_x,2):
    print(mylist[x])

a=int(input("Enter number of arguments: "))
my_func(a);
```

```
Enter number of arguments: 5
Enter the argument values: Sneha
Enter the argument values: Shubham
Enter the argument values: Shaggy
Enter the argument values: Abhi
Enter the argument values: Karan
Sneha
Shaggy
Karan
```

```
# 30

a_dict={}
def func_dict(e_id,e_name,e_dob,e_sal,e_comm,e_dept):
  a_dict[e_id]=[e_name,e_dob,e_sal,e_comm,e_dept]

func_dict(1,"Sneha","26-02-2000","2000","5000","CBI")
func_dict(2,"Karan","18-01-2001","5000","7000","CBI")
print(a_dict)
```

```
{1: ['Sneha', '26-02-2000', '2000', '5000', 'CBI'], 2: ['Karan', '18-01-2001', '5000', '7000', 'CBI']}
```

```
# 31

list_a = [1,2,3,4]

def my_list_a(b_list):
  list_c=[]
  for x in b_list:
    list_c.append(x*10)
  print(list_c)

my_list_a(list_a)
```

```
[10, 20, 30, 40]
```

```
# 32

def error():
  try:
    list2=[1,2,3]
    # print(list2[4])
    # var_a=12/0
    # print(varX)
  except ZeroDivisionError:
    print("Denominator cannot be 0.")
  except IndexError:
    print("Index Out of Bound.")
  except:
    print("Variable not found.")
error()
```

```
# 33

def readaFile():
  f = open("./testcsv.csv","r")
  content = f.read()
  print(content)
  f.close()
```

```
# 34
import shutil

def FN_file(Fname):
  shutil.copyfile(Fname,'test.csv')
  print('file Copied!')

FN_file('/Projects.csv')
```

```
     file Copied!
```

```
# 35

import shutil

def checkFile(name):
  try:
    path = "./" + name
    f = open(path,"r")

  except FileNotFoundError:
    print("File does not exists")
  else:
    f1 = open("./copiedfile.txt","w")
    copycontent = f.readlines()
    f1.writelines(copycontent[:4])
    print("File has been copied")

checkFile("test.txt")
```

```
     File does not exists
```

```
# 36


class employee:
  def __init__(self,eid,ename,edept) :
    self.eid = eid
    self.ename = ename
    self.edept = edept

empobj = employee(1,"Sneha","Data Analyst")
print(empobj.eid,empobj.ename,empobj.edept)
```

```
     1 Sneha Data Analyst
```

```
# 37

import datetime
class Sales:
  def __init__(self,qty,date,currency,unit,amount):
    self.qty=qty
    self.date=datetime.datetime.strptime(date,"%d-%m-%Y")
    self.currency=currency
    self.unit=unit
    if qty==100:
      self.amount=amount-(0.1*amount)
    elif qty==200:
      self.amount=amount-(0.2*amount)
    else:
      self.amount=amount
```

38

All classes have a function called **init**(), which is always executed when the class is being initiated.

Use the **init**() function to assign values to object properties, or other operations that are necessary to do when the object is being created

The **str** method in Python represents the class objects as a string – it can be used for classes. The **str** method should be defined in a way that is easy to read and outputs all the members of the class. This method is also used as a debugging tool when the members of a class need to be checked.

The **str**() method returns a human-readable, or informal, string representation of an object. This method is called by the built-in print(), str(), and format() functions.

If we have not defined the **str**, then it will call the **repr** method. The **repr** method returns a string that describes the pointer of the object by default (if the programmer does not define it).

39

In Python, a class is similar to a blueprint from which objects are constructed. Let’s use an automobile as an example to better understand the word.

An automobile is a collection of various elements, such as an engine, wheels, and so on, rather than a single entity. The term “vehicle” refers to a “class,” while the terms “engine” and “wheels” refer to objects.

A class instance is sometimes known as an object, and the process of producing this object is known as instantiation. Each class has objects with numerous arguments that can be used to conduct actions on data.

```
## 40

A Class is like an object constructor, or a "blueprint" for creating
objects.

A class instance is sometimes known as an object, and the process of
producing this object is known as instantiation. Each class has obje
with numerous arguments that can be used to conduct actions on data.

Attributes are variables that belong to an object and contain inform
about its properties and characteristics. They can be used to repres
details or facts related to the object.

 Methods are functions belonging to an object and are designed to pe
 actions or operations involving the object's attributes. Methods ar
 defined as part of the class the object belongs to and are executed
 instances of that particular class.
```

40

A Class is like an object constructor, or a "blueprint" for creating objects.

A class instance is sometimes known as an object, and the process of producing this object is known as instantiation. Each class has objects with numerous arguments that can be used to conduct actions on data.

Attributes are variables that belong to an object and contain information about its properties and characteristics. They can be used to represent details or facts related to the object.

Methods are functions belonging to an object and are designed to perform actions or operations involving the object's attributes. Methods are defined as part of the class the object belongs to and are executed using instances of that particular class.