```
# Task 1

class ComputerA:

    def __init__(self,name,price):
        #initialising instance/member variable
        self.name = name
        self.price = price

    #Setters
    def setName(self,name):#assigns values to the constructor after being called
        self.name=name

    def setPrice(self,price):
        self.price=price

    #getters
    def getName(self):
        return self.name

    def getPrice(self):
        return self.price

    #An instance/member method
    def display(self):
        print("Computer's name is: " +str(self.getName()).capitalize()+ " It's
price is:"+str(self.getPrice()))

#User provides a valid computer name
isValidName = False

while not isValidName: #when true
    computerName= str(input("Enter computer's name: "))

    if 2<=len(computerName)<=10:
        isValidName= True

    else:
        print("You must enter a valid name between 2 and 10 chars length")

#User provides a valid computer price
isValidPrice = False

while not isValidPrice: #Price=True
    computerPrice=float(input("Enter computer's price: "))
```

```
    if 99.99<=computerPrice<=999.99:
        isValidPrice=True
    else:
        print("You must enter a valid price between 99.99 and 999.99")

comp =ComputerA(computerName,computerPrice)
comp.display()

#Setting a different computer name using the setName method
comp.setName("Dell")
#Setting a new price using the setPrice method
comp.setPrice("890.90")
comp.display()
#Setting a new name
comp.setName("Lenovo")
comp.display()
```

Output:

```
mputerClass.py"
Enter computer's name: HP
Enter computer's price: 199.99
Computer's name is: Hp It's price is:199.99
Computer's name is: Dell It's price is:890.90
Computer's name is: Lenovo It's price is:890.90
PS C:\TBC\Sem2\POP_Project>
```

```
# Task 1

class ComputerA:

    def __init__(self,name,price):
        #initialising instance/member variable
        self.name = name
        self.price = price

    #Setters
    def setName(self,name):#assigns values to the constructor after being called
        self.name=name

    def setPrice(self,price):
        self.price=price

    #getters
    def getName(self):
        return self.name

    def getPrice(self):
        return self.price

    #An instance/member method
    def display(self):
        print("\nComputer's name is: " +str(self.getName()).capitalize()+ " It's
price is: "+ str(self.getPrice()))

#User provides a valid computer name
isValidName = False

while not isValidName: #when true
    computerName= str(input("Enter computer's name: "))

    if 2<=len(computerName)<=10:
        isValidName= True

    else:
        print("You must enter a valid name between 2 and 10 chars length")

#User provides a valid computer price
isValidPrice = False

while not isValidPrice: #Price=True
    computerPrice=float(input("\nEnter computer's price: "))
```

```
    if 99.99<=computerPrice<=999.99:
        isValidPrice=True
    else:
        print("You must enter a valid price between 99.99 and 999.99")

#Creating the comp object based on user's input
comp =ComputerA(computerName,computerPrice)
comp.display()

#Setting a different computer name using the setName method
#comp.setName("Dell")
#Setting a new price using the setPrice method
#comp.setPrice("890.90")
#comp.display()
#Setting a new name
#comp.setName("Lenovo")

#Setting a different computer name by accessing the name variable which is public
#Direct method
comp.name = "Dell"
comp.price = 890.90
comp.display()
```

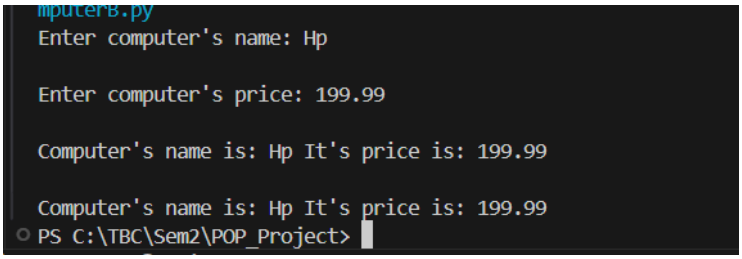
Output:

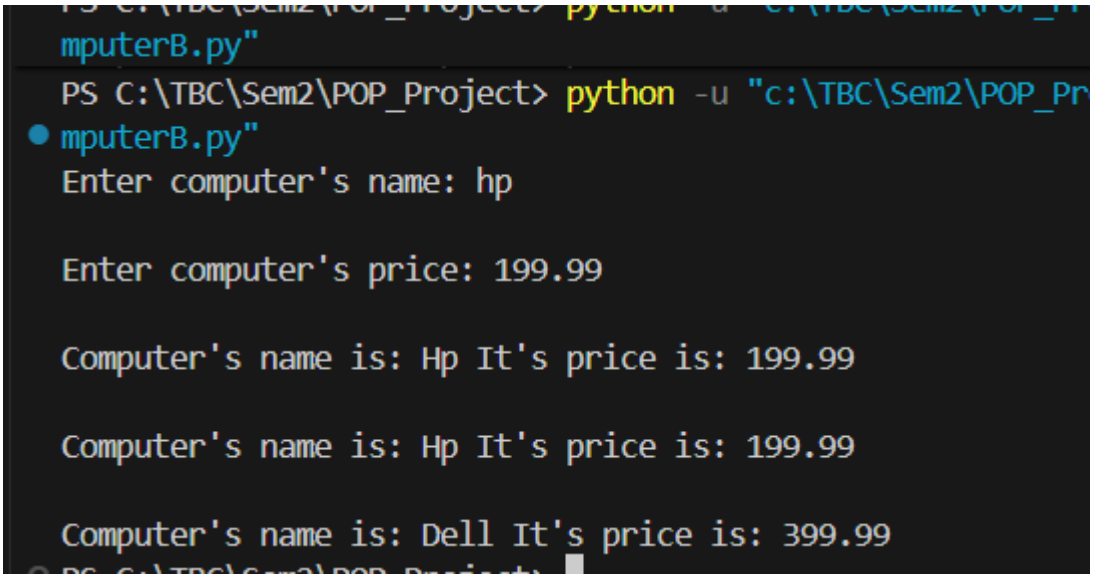
```
PS C:\TBC\Sem2\POP_Project> python -u "c:\TBC\Sem2\
mputerClass.py"
Enter computer's name: HP

Enter computer's price: 199.99

Computer's name is: Hp It's price is: 199.99

Computer's name is: Dell It's price is: 890.9
PS C:\TBC\Sem2\POP_Project>
```

```python
# Task 1

class ComputerB:

    def __init__(self,name,price):
        #initialising instance/member variable
        #Private variable
        self.__name = name
        self.__price = price

    #Setters
    def setName(self,name):#assigns values to the constructor after being called
        self.__name=name

    def setPrice(self,price):
        self.__price=price

    #getters
    def getName(self):
        return self.__name

    def getPrice(self):
        return self.__price

    #An instance/member method
    def display(self):
        print("\nComputer's name is: " +str(self.getName()).capitalize()+ " It's
price is: "+ str(self.getPrice()))

#User provides a valid computer name
isValidName = False

while not isValidName: #when true
    computerName= str(input("Enter computer's name: "))

    if 2<=len(computerName)<=10:
        isValidName= True

    else:
        print("You must enter a valid name between 2 and 10 chars length")

#User provides a valid computer price
isValidPrice = False

while not isValidPrice: #Price=True
```

```
    computerPrice=float(input("\nEnter computer's price: "))

    if 99.99<=computerPrice<=999.99:
        isValidPrice=True
    else:
        print("You must enter a valid price between 99.99 and 999.99")

#Creating the comp object based on user's input
comp =ComputerB(computerName,computerPrice)
comp.display()

#Setting a different computer name using the setName method
#comp.setName("Dell")
#Setting a new price using the setPrice method
#comp.setPrice("890.90")
#comp.display()
#Setting a new name
#comp.setName("Lenovo")

#Setting a different computer name by accessing the name variable which is public
#Direct method
comp.__name = "Dell"
comp.__price = 890.90
comp.display()

# TASK 1.4

#The private variable can only be acessed by the class (mathod in that class)
#that's why we cant acess and change the variable value directly using object
#Using public methods we can get the private names but if we try to print it
directly it'll give us error.

#Task 1.5
#Public method to access private variables
comp.setName("Dell")
comp.setPrice("399.99")
comp.display()
```

Output:


mputerB.py
Enter computer's name: Hp
Enter computer's price: 199.99
Computer's name is: Hp It's price is: 199.99
Computer's name is: Hp It's price is: 199.99
PS C:\TBC\Sem2\POP_Project>

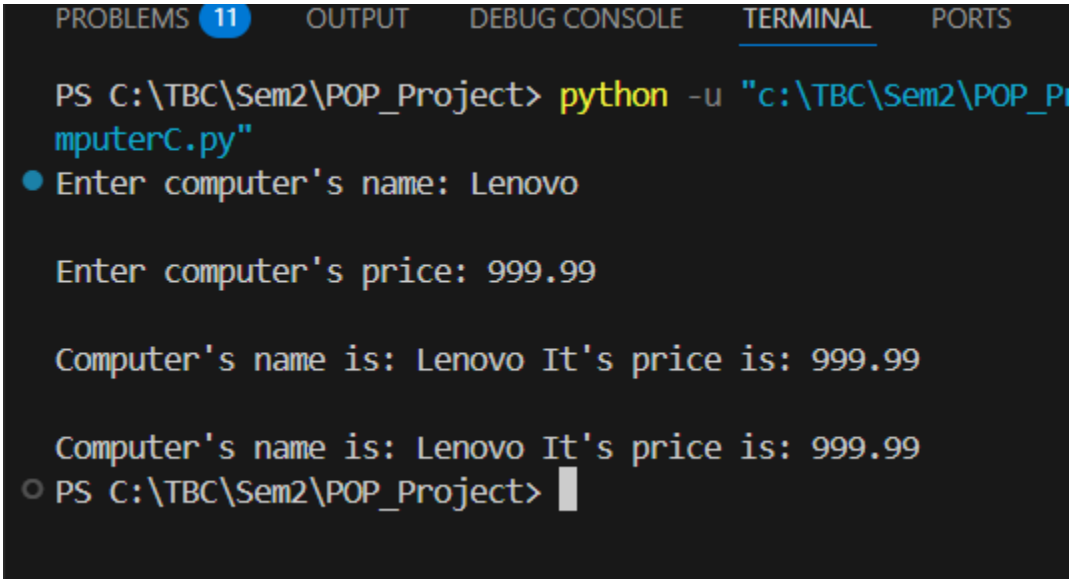

mputerB.py"
PS C:\TBC\Sem2\POP_Project> python -u "c:\TBC\Sem2\POP_Pr
mputerB.py"
Enter computer's name: hp
Enter computer's price: 199.99
Computer's name is: Hp It's price is: 199.99
Computer's name is: Hp It's price is: 199.99
Computer's name is: Dell It's price is: 399.99

```
# Task 1


class ComputerC:

    def __init__(self,name,price):
        #initialising instance/member variable
        #Private variable
        self.__name = name
        self.__price = price

    #Setters
    #def setName(self,name):#assigns values to the constructor after being called
    #    self.__name=name

    #def setPrice(self,price):
    #    self.__price=price

    #getters
    def getName(self):
        return self.__name

    def getPrice(self):
        return self.__price

    #An instance/member method
    def display(self):
        print("\nComputer's name is: " +str(self.getName()).capitalize()+ " It's
price is: "+ str(self.getPrice()))

#User provides a valid computer name
isValidName = False

while not isValidName: #when true
    computerName= str(input("Enter computer's name: "))

    if 2<=len(computerName)<=10:
        isValidName= True

    else:
        print("You must enter a valid name between 2 and 10 chars length")

#User provides a valid computer price
isValidPrice = False
```

```
while not isValidPrice: #Price=True
    computerPrice=float(input("\nEnter computer's price: "))

    if 99.99<=computerPrice<=999.99:
        isValidPrice=True
    else:
        print("You must enter a valid price between 99.99 and 999.99")

#Creating the comp object based on user's input
comp =ComputerC(computerName,computerPrice)
comp.display()

#Setting a different computer name using the setName method
#comp.setName("Dell")
#Setting a new price using the setPrice method
#comp.setPrice("890.90")
#comp.display()
#Setting a new name
#comp.setName("Lenovo")

#Setting a different computer name by accessing the name variable which is public
#Direct method
comp.__name = "Dell"
comp.__price = 890.90
comp.display()

# TASK 1.4
#The private variable can only be acessed by the class (mathod in that class)
#that's why we cant acess and change the variable value directly using object
#Using public methods we can get the private names but if we try to print it
directly it'll give us error.

#Task 1.5
#Public method to access private variables
#comp.setName("Dell")
#comp.setPrice("399.99")
#comp.display()
```

```
# TASK 1.6
""".
We cant change the values of those member variable because they are private
variable.
It cannot be directly access nor can be directly changed by creating the object.
Private variables can be access only within the Class not beyond(outside) it's
scope.
The only way to access and change it is by using public methods.
"""
```

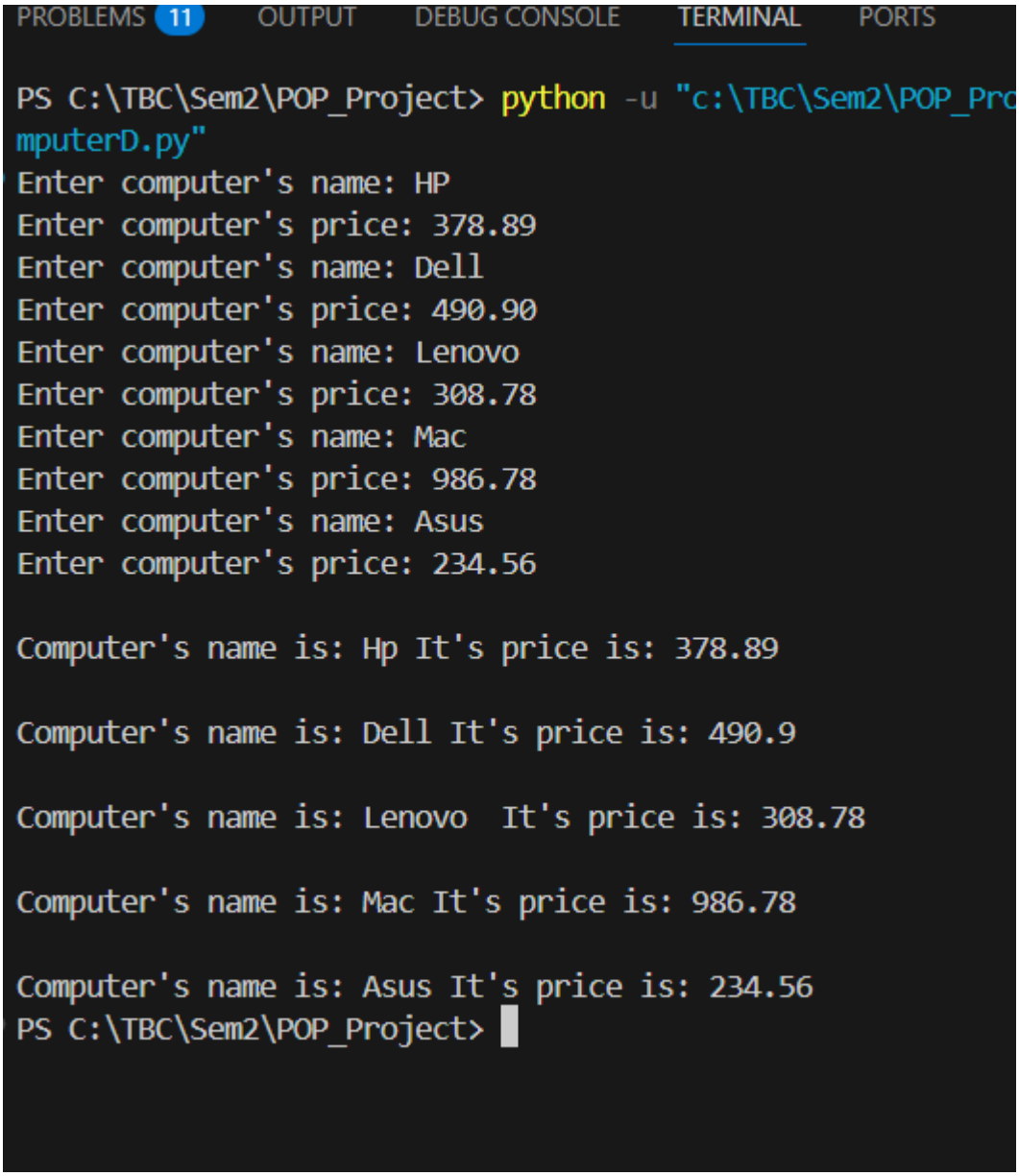
Output:



```
# Task 1.7
class ComputerD:

    def __init__(self,name,price):
        #initialising instance/member variable
        #Private variable
        self.__name = name
        self.__price = price

    #Setters
    def setName(self,name):#assigns values to the constructor after being called
        self.__name=name

    def setPrice(self,price):
        self.__price=price

    #getters
    def getName(self):
        return self.__name

    def getPrice(self):
        return self.__price

    #An instance/member method
    def display(self):
        print("\nComputer's name is: " +str(self.getName()).capitalize()+ " It's
price is: "+ str(self.getPrice()))



#Creating the comp object based on user's input
#comp =ComputerB(computerName,computerPrice)
#comp.display()

#Setting a different computer name using the setName method
#comp.setName("Dell")
#Setting a new price using the setPrice method
#comp.setPrice("890.90")
#comp.display()
#Setting a new name
#comp.setName("Lenovo")

#Setting a different computer name by accessing the name variable which is public
#Direct method
#comp.__name = "Dell"
```

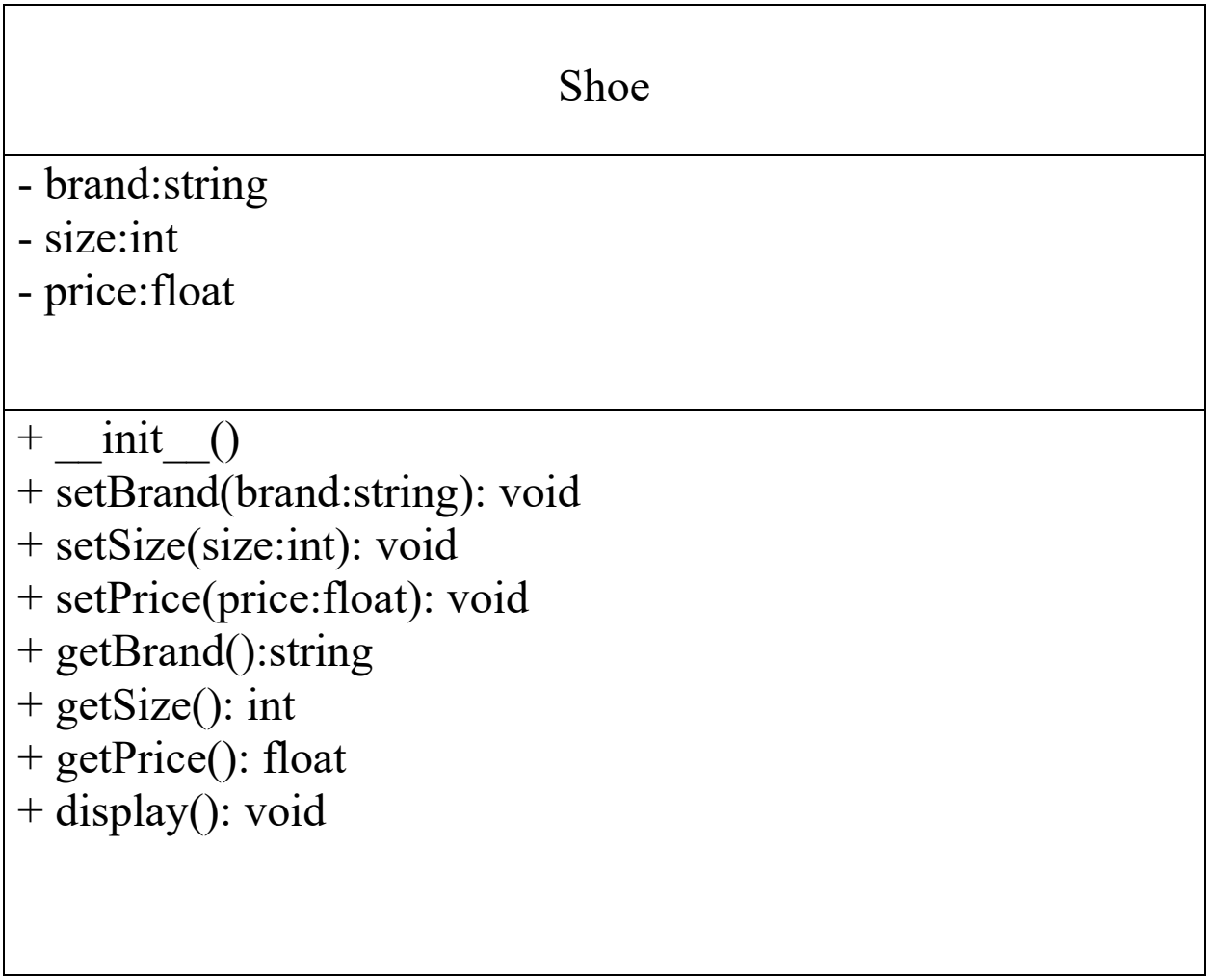
```
#comp.__price = 890.90
#comp.display()

# TASK 1.4
#The private variable can only be acessed by the class (mathod in that class)
#that's why we cant acess and change the variable value directly using object
#Using public methods we can get the private names but if we try to print it
directly it'll give us error.

#Task 1.5
#Public method to access private variables
#comp.setName("Dell")
#comp.setPrice("399.99")
#comp.display()

#EMpty list
objectList=[]

#appending claass instances to list
for i in range (5):
    #User provides a valid computer name
    isValidName = False
    while not isValidName: #when true
        computerName= str(input("Enter computer's name: "))

        if 2<=len(computerName)<=10:
            isValidName= True

        else:
            print("You must enter a valid name between 2 and 10 chars length")

    #User provides a valid computer price
    isValidPrice = False
    while not isValidPrice: #Price=True
        computerPrice=float(input("Enter computer's price: "))

        if 99.99<=computerPrice<=999.99:
            isValidPrice=True
        else:
            print("You must enter a valid price between 99.99 and 999.99")
    objectList.append(ComputerB(computerName,computerPrice))
#displaying computer's information
for obj in objectList:
    obj.display()
```

Output:


PROBLEMS 11
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\TBC\Sem2\POP_Project> python -u "c:\TBC\Sem2\POP_Pro
mputerD.py"
Enter computer's name: HP
Enter computer's price: 378.89
Enter computer's name: Dell
Enter computer's price: 490.90
Enter computer's name: Lenovo
Enter computer's price: 308.78
Enter computer's name: Mac
Enter computer's price: 986.78
Enter computer's name: Asus
Enter computer's price: 234.56
Computer's name is: Hp It's price is: 378.89
Computer's name is: Dell It's price is: 490.9
Computer's name is: Lenovo It's price is: 308.78
Computer's name is: Mac It's price is: 986.78
Computer's name is: Asus It's price is: 234.56
PS C:\TBC\Sem2\POP_Project>

TASK 2:

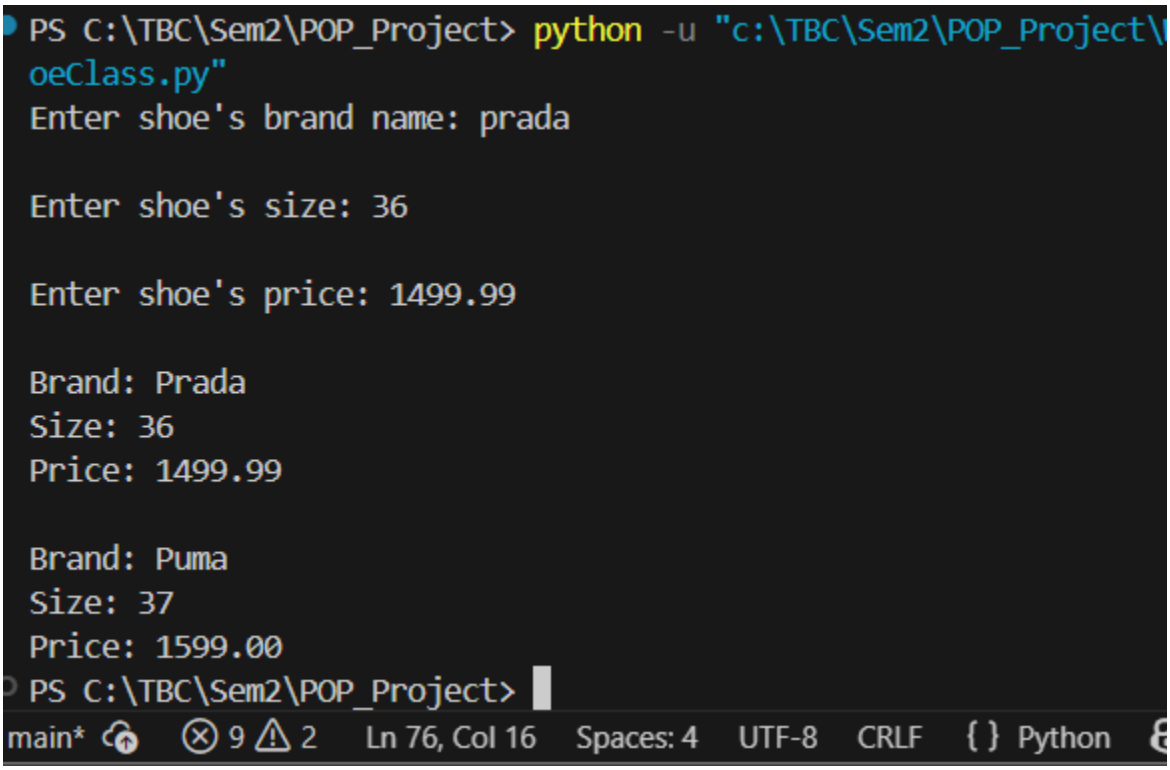
1. Class Shoe UML diagram:



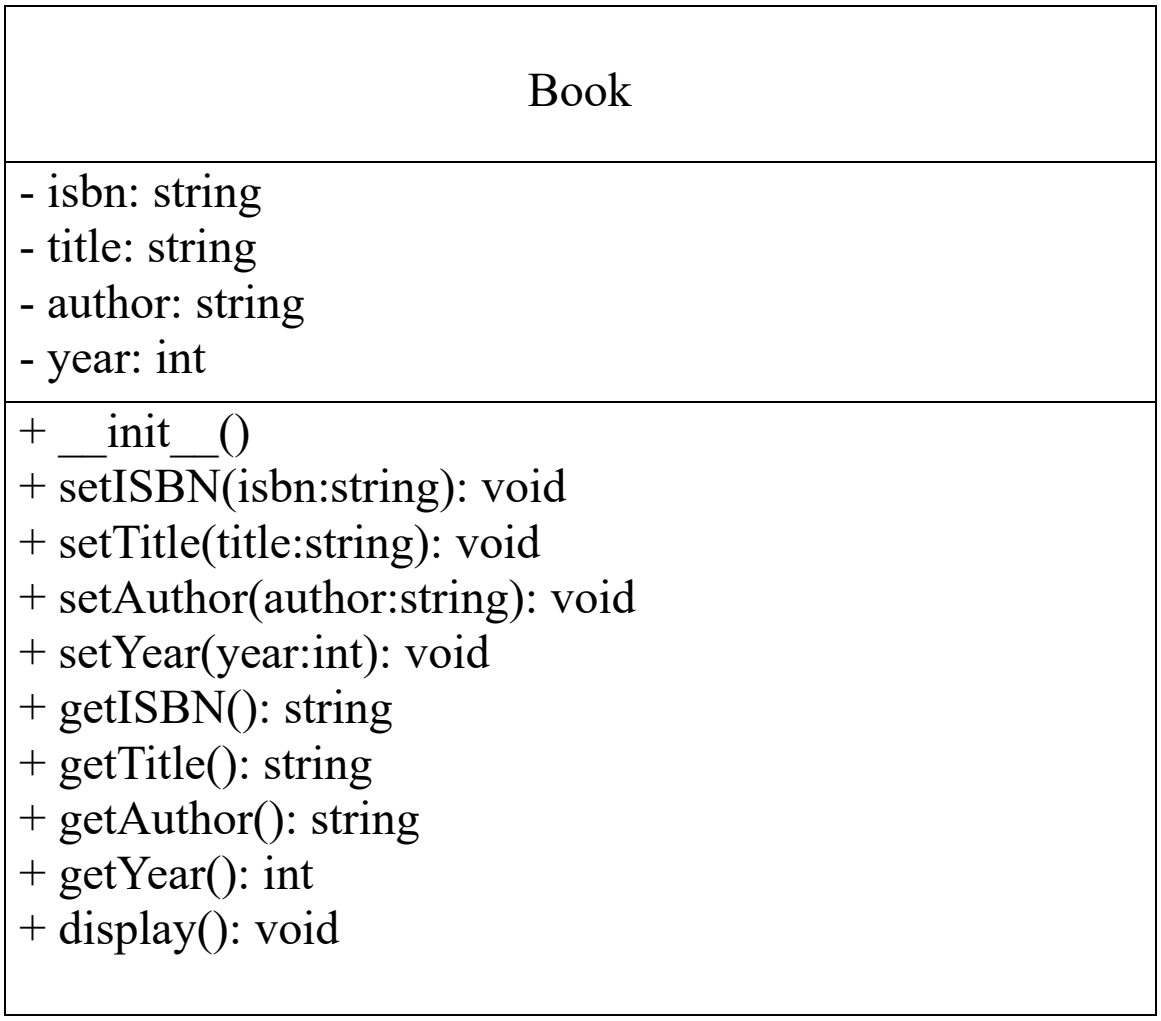
```python
class Shoe:
    #constructor
    def __init__(self,brand,size=None,price=1599):
        self.__brand = brand
        self.__size = size
        self.__price = price

    #Setters
    def setBrand(self,brand):
        self.__brand = brand

    def setSize(self,size):
        self.__size = size

    def setPrice(self,price):
        self.__price = price

    #getter
    def getBrand(self):
        return self.__brand

    def getSize(self):
        if self.__size == None:
            print("Please enter size!!!")
        else:
            return self.__size

    def getPrice(self):
        return self.__price

    def display(self):
        print("\nBrand: "+str(self.getBrand()).capitalize())
        print(f"Size: {self.getSize():}")
        print(f"Price: {self.getPrice():.2f}")

#Checking whether is valid brand or not
isValidBrand = False
while not isValidBrand: #when true
    shoeBrand= str(input("Enter shoe's brand name: "))

    if 2<=len(shoeBrand)<=10:
        isValidBrand= True
```

```
    else:
        print("You must enter a valid name between 2 and 10 chars length")

#Checking whether is valid shoe's size or not
isValidSize = False
while not isValidSize: #Price=True
    shoeSize=int(input("\nEnter shoe's size: "))

    if 35<=shoeSize<=45:
        isValidSize=True
    else:
        print("You must enter a valid size between 35-45")


#Checking whether is valid price or not
isValidPrice = False
while not isValidPrice: #Price=True
    shoePrice=float(input("\nEnter shoe's price: "))

    if 99.99<=shoePrice<=9999.99:
        isValidPrice=True
    else:
        print("You must enter a valid price between 99.99 and 9999.99")

#Creating object
shoe1= Shoe(shoeBrand,shoeSize,shoePrice)
#Displaying brand,size and price
shoe1.display()

#Creating second object
shoe2 = Shoe(shoeBrand,shoeSize)
shoe2.setBrand("Puma")
shoe2.setSize(37)
shoe2.display()
```

Output:


PS C:\TBC\Sem2\POP_Project> python -u "c:\TBC\Sem2\POP_Project\oeClass.py"
Enter shoe's brand name: prada
Enter shoe's size: 36
Enter shoe's price: 1499.99
Brand: Prada
Size: 36
Price: 1499.99
Brand: Puma
Size: 37
Price: 1599.00
PS C:\TBC\Sem2\POP_Project>
main* 9 2 Ln 76, Col 16 Spaces: 4 UTF-8 CRLF { } Python

2. Class Book UML diagram:



```python
class Book:
    #constructor
    def __init__(self,isbn,title,author=None,year=None):
        self.__isbn = isbn
        self.__title = title
        self.__author = author
        self.__year = year

    #Setters
    def setISBN(self,isbn):
        self.__isbn = isbn

    def setTitle(self,title):
        self.__title = title

    def setAuthor(self,author):
        self.__author = author

    def setYear(self,year):
        self.__year = year

    #Getters
    def getISBN(self):
        return self.__isbn

    def getTitle(self):
        return self.__title

    def getAuthor(self):
        if self.__author == None:
            return ("Author name not provided!!!")
        else:
            return self.__author

    def getYear(self):
        if self.__year == None:
            return ("Published Year not provided!!!")
        else:
            return self.__year
    # Book class instance/member method which displays the info
    def display(self):
        print("\nTitle: "+str(self.getTitle()).capitalize())
        print(f"ISBN Number: {self.getISBN()}")
        print(f"Author: {self.getAuthor().capitalize()}")
        print(f"Published Year: {self.getYear()}")
```

```
#Checking whether the inputs are valid or not
isValidTitle = False
while not isValidTitle: #when true
    title= str(input("Enter book's title: "))

    if 3<=len(title)<=20:
        isValidTitle= True

    else:
        print("You must enter a valid title between 3 and 20 chars length")

isValidISBN = False
while not isValidISBN: #when true
    isbn_no= input("Enter book's ISBN number: ")

    #checking it's length and and whether the given input is numeric or not
    if len(isbn_no)==13 or len(isbn_no)==10 and isbn_no.isdigit():
        #changing string into integer
        isbn_no=int(isbn_no)
        isValidISBN= True

    else:
        print("You must enter a valid ISBN number of 10 or 13 chars length")

isValidAuthor = False
while not isValidAuthor: #when true
    author= str(input("Enter book's author name: "))

    if 5<=len(author)<=35:
        isValidAuthor= True

    else:
        print("You must enter the valid Author name between 5 and 35 chars
length")

isValidYear = False
while not isValidYear:
    year= input("Enter the published year: ")
    #checking it's length and and whether the given input is numeric or not
    if 3<=len(year)<=4 and year.isdigit():
        #converting string into int
        year = int(year)
        #checking of year
```

```
        if 600<=year<=2025:
            isValidYear = True
    else:
        print("You must enter valid year between 600 to 2025")

#Creating objects
book1=Book(isbn_no,title,author,year)
book1.display()

book2=Book(isbn_no,title)
#using class method to access private variable
book2.setISBN(9781784752637)
book2.setTitle("To Kill a Mocking Bird")
book2.setAuthor("Harper Lee")
book2.display()
```

Output:

```
PS C:\TBC\Sem2\POP_Project> python -u "c:\TBC\Sem2\POP_Project\
okClass.py"
Enter book's title: Animal farm
Enter book's ISBN number: 9780030554346
Enter book's author name: george orwell
Enter the published year: 1945

Title: Animal farm
ISBN Number: 9780030554346
Author: George orwell
Published Year: 1945

Title: To kill a mocking bird
ISBN Number: 9781784752637
Author: Harper lee
Published Year: Published Year not provided!!!
PS C:\TBC\Sem2\POP_Project>
```