



Exit

1 #Implement a class called BankAccount
 that represents a bank account. The
 class should have private attributes
 for account number, account holder
 name, and account balance. Include
 methods to deposit money, withdraw
 money, and display the account
 balance. Ensure that the account
 balance cannot be accessed directly
 from outside the class. Write a
 program to create an instance of the
 BankAccount class and test the deposit
 and withdrawal functionality.#

```
3 v class BankAccount:
        def __init__(self, account_number,
    account_holder_name, initial_balance):
5
            self.__account_number =
    account number
6
             self.__account_holder_name =
    account_holder_name
            self.__account_balance =
7
    initial balance
8
9 ,
        def deposit(self, amount):
10 ~
            if amount > 0:
11
                 self.__account_balance +=
    amount
17
```

nain.py

Ш





Ln 1, Col 1 History 5

```
Voi) LTE .1 52%
  15:28 @ 🗷 🕓 •
                                       Exit
Challenge 2.1
9 ,
       def deposit(self, amount):
10 ~
            if amount > 0:
11
                 self.__account_balance +=
    amount
                 print("Deposited ₹{}. New
12
    balance: ₹
    {}".format(amount, self.__account_balanc
    e))
13 🗸
            else:
14
                 print("Invalid deposit
    amount.")
15
      def withdraw(self, amount):
16 ~
            if amount > 0 and amount <=
17 ,
    self.__account_balance:
                 self.__account_balance -=
18
    amount
19
                 print("Withdrew ₹{}. New
    balance: ₹{}".format(amount,
    self.__account_balance))
20 ~
            else:
21
                 print("Invalid withdrawal
    amount or insufficient balance.")
22
23 🗸
       def display_balance(self):
            print("Account Balance for {}
24
    (Account #{}): ₹
    {}".format(self.__account_holder_name,
                              Ln 1, Col 1 History 'S
                    main.py
                                          Run
        Ш
```

```
15:28 🗗 📵 🕓 🔹
                                  Voi) LTE .11 52%
                                       Exit
Challenge 2.1
21
                 print("Invalid withdrawal
    amount or insufficient balance.")
22
23 ~
        def display_balance(self):
24
            print("Account Balance for {}
    (Account #{}): ₹
    {}".format(self.__account_holder_name,
    self.__account_number,
    self.__account_balance))
25
26
    # Create an instance of BankAccount
    class
27
    account =
    BankAccount(account_number="123456789",
    account_holder_name="John",
    initial_balance= 5000.0)
28
29
    #Test deposit and withdrawal
30
    functionality
31
    account.display balance()
32
    account.deposit(500.0)
    account.withdraw(200.0)
33
34
    account.withdraw (20000.0)
35
    account.display_balance()
36
                              Ln 1, Col 1 History 'S
                    main.py
                                          Run
        \Pi
```



€ Exit

#Implement a class called Player that
represents a cricket player. The
Player class should have a method
called play() which prints "The player
is playing cricket. Derive two
classes, Batsman and Bowler, from the
Player class. Override the play()
method in each derived class to print
"The batsman is batting" and "The
bowler is bowling", respectively.
Write a program to create objects of
both the Batsman and Bowler classes
and call the play() method for each
object. #

```
2
 3
    # Define the base class Player
 5 v class Player:
      def play(self):
 7
             print("The player is playing
    cricket.")
 8
 9
    # Define the derived class Batsman
10 v class Batsman(Player):
       def play(self):
11 ~
12
             print("The batsman is batting.
    ")
13
                               Ln 1, Col 1 History 'S
```

challenge2.2.py:



Ш





```
15:27 🔁 📵 🛕 •
                                  Voi) LTE ... 52%
                                       Exit
Challenge 2.2
 5 v class Player:
        def play(self):
 7
            print("The player is playing
    cricket.")
 8
    # Define the derived class Batsman
 9
10 v class Batsman(Player):
        def play(self):
12
            print("The batsman is batting.
    ")
13
14
    # Define the derived class Bowler
15 v class Bowler(Player):
        def play(self):
16 🗸
17
            print("The bowler is bowling.")
18
19
    # Create objects of Batsman and Bowler
    classes
20
    batsman = Batsman()
21
    bowler = Bowler()
22
23
    # Call the play() method for each
    object
24
    batsman.play()
    bowler.play()
25
                              Ln 1, Col 1 History 'S
               challenge2.2.py
                                           Run
        Ш
```