



Challenge 2.1 :

[Exit](#)

```
1 class BankAccount:
2     def __init__(self,
  account_number,
  account_holder_name,
  initial_balance=0.0):
3         self.__account_number =
  account_number
4         self.__account_holder_name =
  account_holder_name
5         self.__account_balance =
  initial_balance
6
7     def deposit(self, amount):
8         if amount > 0:
9             self.__account_balance
  += amount
10             print(f"Deposited
  ${amount}. New balance:
  ${self.__account_balance}")
11         else:
12             print("Invalid deposit
  amount. Amount must be greater than
  0.")
13
14     def withdraw(self, amount):
15         if amount > 0:
16             if amount <=
  self.__account_balance:
```



Challenge 2.1 :

Exit

```
17 self.__account_balance -= amount
18         print(f"Withdrew
19 ${amount}. New balance:
20 ${self.__account_balance}")
21     else:
22         print("Insufficient
23 funds.")
24     else:
25         print("Invalid
26 withdrawal amount. Amount must be
27 greater than 0.")
28
29     def display_balance(self):
30         print(f"Account Number:
31 {self.__account_number}")
32         print(f"Account Holder:
33 {self.__account_holder_name}")
34         print(f"Account Balance:
35 ${self.__account_balance}")
36
37 # Create an instance of the
38 BankAccount class
39 account1 = BankAccount("123456",
40 "John Doe", 1000.0)
41
42 # Test deposit and withdrawal
43 functionality
44 account1.display_balance()
```

Ln 35, Col 25 History



main.py



Run





Challenge 2.1 :

Exit

```
20         print("Insufficient
funds.")
21     else:
22         print("Invalid
withdrawal amount. Amount must be
greater than 0.")
23
24     def display_balance(self):
25         print(f"Account Number:
{self.__account_number}")
26         print(f"Account Holder:
{self.__account_holder_name}")
27         print(f"Account Balance:
${self.__account_balance}")
28
29 # Create an instance of the
BankAccount class
30 account1 = BankAccount("123456",
"John Doe", 1000.0)
31
32 # Test deposit and withdrawal
functionality
33 account1.display_balance()
34 account1.deposit(500.0)
35 account1.withdraw(200.0)
```

Ln 35, Col 25 History



main.py



Run





Challenge 2.1 :

 Exit

```
Account Number: 123456  
Account Holder: John Doe  
Account Balance: $1000.0  
Deposited $500.0. New balance: $1500.0  
Withdrew $200.0. New balance: $1300.0
```



>_ Console



Run





Challenge 2.2 :

 Exit

```
1 class Player:
2     def play(self):
3         print("The player is
4         playing cricket.")
5 class Batsman(Player):
6     def play(self):
7         print("The batsman is
8         batting.")
9 class Bowler(Player):
10    def play(self):
11        print("The bowler is
12        bowling.")
13 # Create objects of both Batsman
14 # and Bowler classes
15 batsman = Batsman()
16 bowler = Bowler()
17 # Call the play() method for each
18 # object
19 batsman.play()
20 bowler.play()
```



Challenge 2.2 :

 Exit

```
The batsman is batting.  
The bowler is bowling.
```



>_ Console



Run

