```
€ Exit
Python (2)
1 v def fact_rec (n):
   if n==0 or n==1:
2 🗸
3
       return 1
4 ,
   else:
5
       return n*fact_rec (n-1)
6
  number = 2
7 res=fact_rec (number)
8
   print("the factorial of {}
   is {}".format(number,res))
                  Ln 1, Col 1 History 5
            🔷 main.py
                Run
```

```
€ Exit
Python (3)
 1 v def isleapyear(year):
2 🗸
       if(year%4==0 and
    year%100!=0)or year%400==0:
3
         return True
4 🗸
    else:
5
         return False
6 year=int(input("enter a
    year"))
7 v if isleapyear(year):
8
      print("{} is leap
    year".format(year))
9 velse:
10
      print("{}is not a leap
    year".format(year))
                   Ln 1, Col 1 History 5
            🔷 main.py
```

```
1 v class player:
        def play(self):
2 ~
3
          print("The player is playing
    cricket")
4
5 v class Batsman(player):
      def play(self):
6 🗸
7
          print("The batsman is playing
    batting")
8
9 v class Blower(player):
        def play(self):
10 🗸
11
          print("The blower is playing
    bowling")
12
13
    batsman = Batsman()
14
    blower = Blower()
15
16
    batsman.play()
17
    blower.play()
```

python3 main.py
The batsman is playing batting
The blower is playing bowling
•



```
1 v class BankAccount:
2 \ def __init__(self,account_number,
    account_holder_name,
    inital_balance=0.0):
3
        self.__account_number =
    account_number
4
        self.__account_holder_name =
    account_holder_name
        self.__account_balance =
5
    inital_balance
6
7 🗸
     def deposit(self, amount):
        if amount > 0:
8 ~
9
           self.__account_balance +=
    amount
          print("Deposit ₹{}. New
10
    balance: {}".format(amount,
        self.__account_balance))
11
12 _
        else:
13
          print("Invalid deposit
    amount")
14
                  Ln 1, Col 1 • Spaces: 2 History 5
```

main.py



```
def withdraw(self, amount):
15 <sub>~</sub>
         if amount > 0 and amount <=
16 V
    self.__account_balance:
17
           self.__account_balance -=
    amount
18
           print("Withdraw ₹{}. New
    balance: {}".format(amount,
           self.__account_balance))
19
20 \
        else:
           print("Invalid withdraw
21
    amount")
22
23 ~
      def display_balance(self):
         print("Account balance for {}
24
    (Account #{}): ₹{}".format(
             self.__account_holder_name,
25
    self.__account_number,
26
             self.__account_balance))
27
    account =
    BankAccount(account_number="4455769"
    ,account_holder_name="Kutdraleshwara
    n N",
                  Ln 1, Col 1 • Spaces: 2 History 5
                  main.py
```

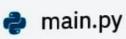






```
print("Account balance for {}
24
     (Account #{}): ₹{}".format(
             self.__account_holder_name,
25
     self.__account_number,
             self.__account_balance))
26
27
     account =
    BankAccount(account_number="4455769"
     ,account_holder_name="Kutdraleshwara
    n N",
     inital_balance=5000.0)
28
29
    account.display_balance()
30
    account.deposit(500)
31
    account.withdraw(300)
32
    account.display_balance()
33
```

Ln 1, Col 1 • Spaces: 2 History '5

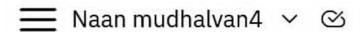














✓ Run 245ms on 17:44:04, 10/18 ✓

Account balance for Kutdraleshwaran N (

Account #4455769): ₹5000.0

Deposit ₹500. New balance: 5500.0

Withdraw ₹300. New balance: 5200.0

Account balance for Kutdraleshwaran N (

Account #4455769): ₹5200.0

: >_ Console :

Run



```
1 v class Student:
2
3 🗸
      def
    __init__(self,name,roll_number,c
    gpa):
4
       self.name=name
5
       self.roll_number=roll_number
6
       self.cgpa=cgpa
7
8 v def sort_students(student_list):
9
10
    sort_students=sorted(student_lis
    t, key=lambda student:
    student.cgpa,reverse=True)
11
12
       return sort_students
13
14 v students =[
15
      Student("mani", "A123", 7.8),
16
      Student("ram", "A124", 9.8),
17
      Student("vik", "A628", 8.8),
                🥏 main.py
                   Run
```



```
14 \vee students = [
15
      Student("mani", "A123", 7.8),
16
      Student("ram", "A124", 9.8),
      Student("vik", "A628", 8.8),
17
18
      Student("raj", "A383", 8.2)
19
    ]
20
21
    sorted_students=sort_students(st
    udents)
22
23 v for student in sorted_students:
24
        print("Name: {},Roll Number:
    {},CGPA: {}".
    format(student.name, student.roll
    _number,student.cgpa))
25
26
27
28
29
```

main.py







python3 main.py

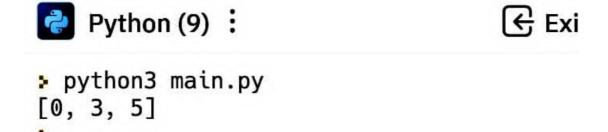
Name: ram,Roll Number:A124,CGPA: 9.8 Name: vik,Roll Number:A628,CGPA: 8.8 Name: raj,Roll Number:A383,CGPA: 8.2 Name: mani,Roll Number:A123,CGPA: 7.8

>

: >_ Console :

Run

```
1 v def
    linearSearchProduct(productList,
    targetProduct):
2
      indices = []
3
      for index, product in
4 ~
    enumerate(productList):
        if product == targetProduct:
5 ~
           indices.append(index)
6
7
8
      return indices
9
    products = ["shoes", "boot",
10
    "loafer", "shoes", "sandal",
    "shoes"1
    target = "shoes"
11
    target2 = 'apple'
12
13
    result =
    linearSearchProduct(products,
    target)
    result2 =
14
    linearSearchProduct(products,
    target)
    print(result)
15
16
                          Ln 15, Col 14 History K
                  main.py
                     Run
```



: >_ Console :

☐ ⊈ ► Run