Sanjay is a Manager in SBI bank, he wants to develop a python application for customer transactions such as deposit and withdrawal of money. Help Sanjay to develop python code to perform deposit and withdrawal of money, ensuring that a withdrawal does not exceed the account balance.

```
# Python program to create Bankaccount class
# with both a deposit() and a withdraw() function
class Bank_Account:
   def __init__(self):
      self.balance=0
      print("Hello!!! Welcome to the Deposit & Withdrawal Machine")

   def deposit(self):
      amount=float(input("Enter amount to be Deposited: "))
      self.balance += amount
      print("\n Amount Deposited:",amount)

   def withdraw(self):
      amount = float(input("Enter amount to be Withdrawn: "))
      if self.balance>=amount:
         self.balance-=amount
         print("\n You Withdrew:", amount)
      else:
         print("\n Insufficient balance  ")

   def display(self):
      print("\n Net Available Balance=",self.balance)

# Driver code

# creating an object of class
s = Bank_Account()

# Calling functions with that class object
s.deposit()
s.withdraw()
s.display()
```

Define a class CARRENTAL with the following:

(i) Class Members are: CarId of int type, CarType of string type and Rent of float type.

(ii) Define GetCar() method which accepts CredIt and Car Type.

(iii) GetRent() method which return rent of the car on the basis of car type, i.e. Honda = 3000, Ford = 4000, Toyota = 5000.

(iv) ShowCar() method which allow user to view the contents of cars i.e. id, type and rent.

```
class CARRENTAL:
    def __init__(self, car_id, car_type):
        """
        Initializes a CARRENTAL object.

        Args:
            car_id (int): The ID of the car.
            car_type (str): The type of the car (e.g., "Honda",
"Ford").
        """
        self.CarId = car_id
        self.CarType = car_type
        self.Rent = 0.0  # Initialize rent to 0

    def GetCar(self, credit, car_type):
        """
        Updates the car information.

        Args:
            credit (float): The credit amount (unused in this
implementation).
            car_type (str): The type of the car.
        """
        self.CarType = car_type
        self.CalculateRent()  # Calculate rent based on car type

    def GetRent(self):
        """
        Returns the rent of the car.
        """
        return self.Rent

    def ShowCar(self):
        """
        Displays the car's information.
        """
        print(f"Car ID: {self.CarId}")
        print(f"Car Type: {self.CarType}")
        print(f"Rent: ${self.Rent:.2f}")

    def CalculateRent(self):
        """
        Calculates and sets the rent based on the car type.
        """
        if self.CarType == "Honda":
            self.Rent = 3000.0
        elif self.CarType == "Ford":
            self.Rent = 4000.0
```

```
        elif self.CarType == "Toyota":
            self.Rent = 5000.0
        else:
            self.Rent = 0.0  # Default or invalid type

# Example usage
car1 = CARRENTAL(101, "Honda")
car1.ShowCar()  # Display initial information

car1.GetCar(1000.0, "Ford")  # Update car type
car1.ShowCar()  # Display updated information

car2 = CARRENTAL(102, "Toyota")
car2.ShowCar()
```