Project: Rental Shop for Customers

Navya Sree Guduri

MSITM, University of Texas at Dallas

MIS-6382: Object Oriented Program Python

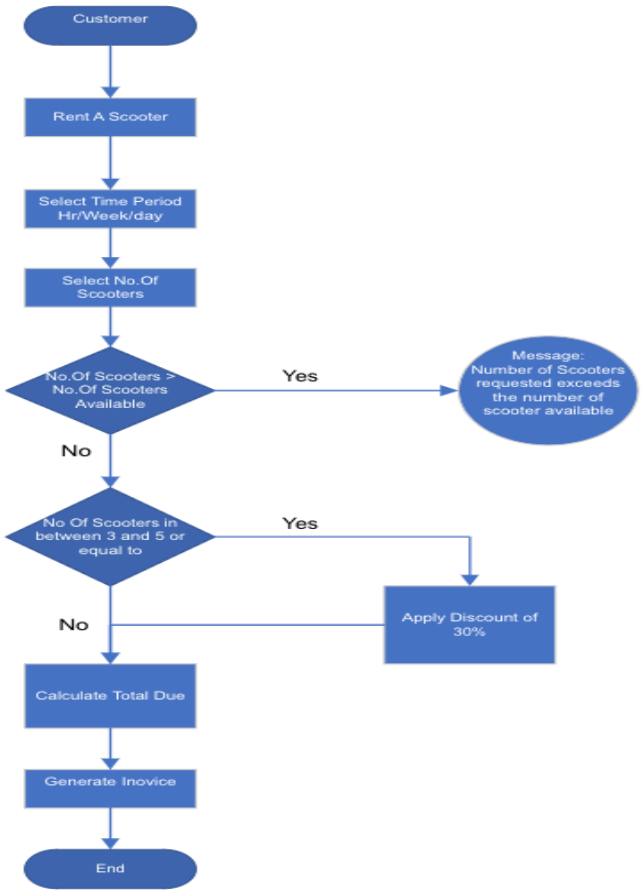
Prof. Vastal Kamleshbhai Maru

December 9, 2022

Rental Shop for Customers

Aiming to create a rental shop for customers. A customer can rent a scooter for an hour OR a day OR a week. Hourly price for renting a scooter is $5, daily price for renting a scooter is $20, and weekly price for renting a scooter is $50. A person can rent more than one scooter and apply promotional discounts for renting more than 2 scooters, meaning 3 to 5 number of scooters rented by a single person will receive 30% discount on the total price. A rental shop can issue an invoice when customers return a scooter. Total due is shown by calculating respective rates and time along with appropriate discount. A customer can select more scooters but not different time periods and that the request cannot exceed the number of scooters available.

Flow Chart:



Method:

```
class Shop:
    def __init__(self, num_scooters):
        self.num_scooters = num_scooters
```

Initiated a class ‘Shop’ and passing a parameter number of scooters available in the shop to rent

```
class Customer:
    def __init__(self, num_scooters, time):
        self.num_scooters = num_scooters
        self.time = time
        self.total_due = 0
```

Initiated a class ‘Customer’ and passing parameters number of scooter and time to choose by customer and initially the total amount due will be zero

```
    def calculate_price(self):
        hourly_rate = 5
        daily_rate = 20
        weekly_rate = 50
        discount_rate = 0.3
```

Declared the rates to calculate price
If Customer choose hourly or weekly or a day and number of scooters in between 3 and 5 then apply 30% discount on total due otherwise no discount applied

```
        if self.time == "hour":
            total_due = self.num_scooters * hourly_rate
            if self.num_scooters >= 3 and self.num_scooters <= 5:
                total_due *= (1-discount_rate)
            return total_due

        elif self.time == "day":
            total_due = self.num_scooters * daily_rate
            if self.num_scooters >= 3 and self.num_scooters <= 5:
```

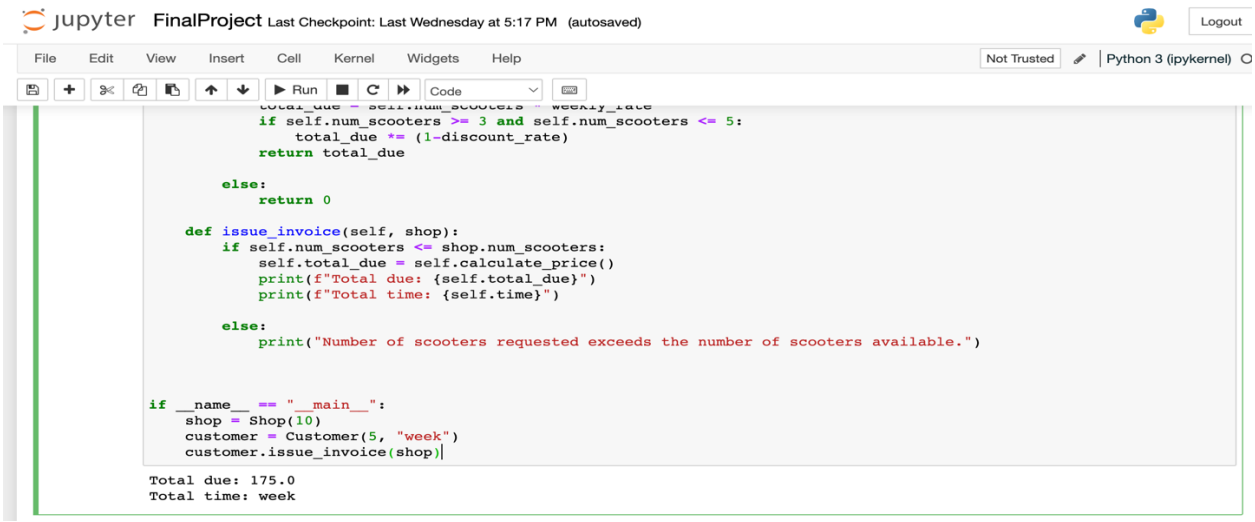
```
            total_due *= (1-discount_rate)
        return total_due

    elif self.time == "week":
        total_due = self.num_scooters * weekly_rate
        if self.num_scooters >= 3 and self.num_scooters <= 5:
            total_due *= (1-discount_rate)
        return total_due

    else:
        return 0

def issue_invoice(self, shop):
    if self.num_scooters <= shop.num_scooters:
        self.total_due = self.calculate_price()
        print(f"Total due: {self.total_due}")
        print(f"Total time: {self.time}")
    else:
        print("Number of scooters requested exceeds the number of scooters available.")

if __name__ == "__main__":
shop = Shop(10)
customer = Customer(5, "week")
customer.issue_invoice(shop)
```

This function is to check whether customer selected scooter less than or equal to number of scooters available to rent and generate invoice with total due and total time otherwise returns message “ Number of scooters requested exceeds the number of scooters available.") “

Unit test:

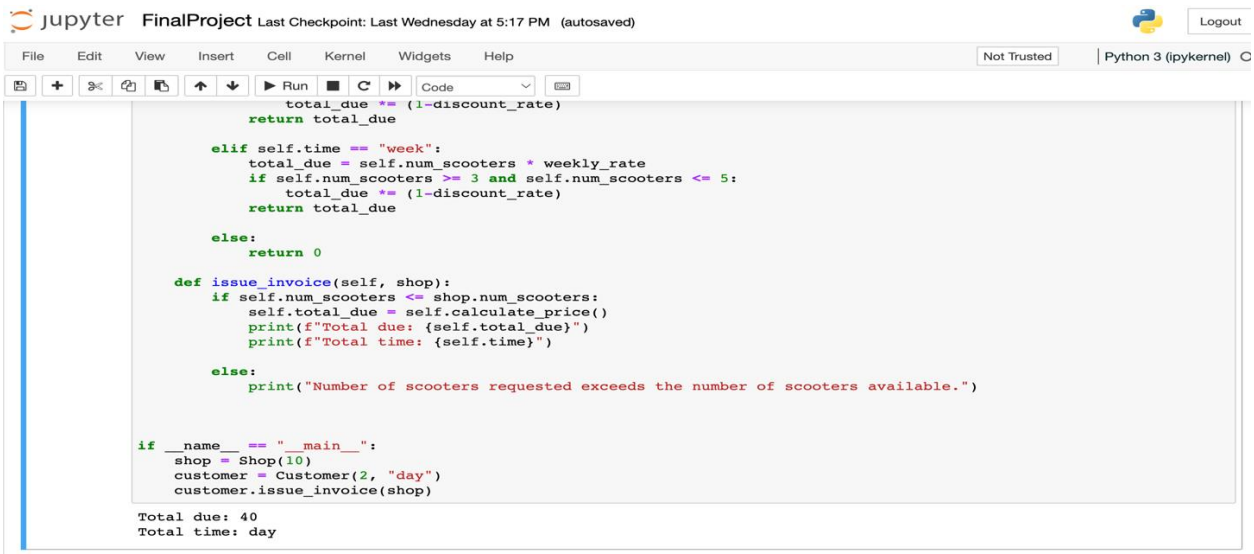
Testing the above code passing parameter value to shop as 10 (i.e; number of scooters available to rent a customer) and passing 5(num of scooters required by the customer) and its based on weekly

Results:

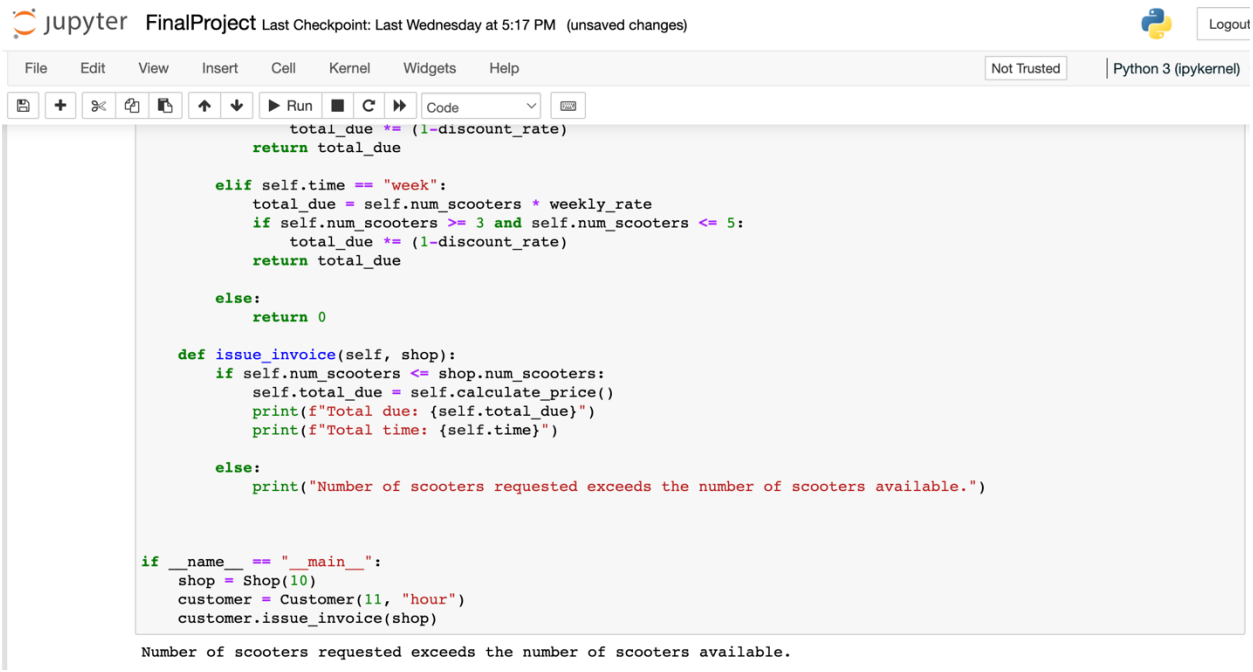


The customer rents 5 scooters for a week, which is the equivalent of 5 x 50 = 250.

Since the customer rents more than two scooters, they receive a 30% discount, resulting in a total due of 250 x (1-0.3) = 175.



The customer rents 2 scooters for a day, which is the equivalent of 2 x 20 = 40. Since the customer rents less than three scooters, they don't receive a 30% discount, resulting in a total due of 40.



The customer rents 11 scooters for an hour, Since the customer rents more than the scooters available in the shop, which is 10, doesn't allows the customer to rent scooters thus resulting a message "Number of scooters requested exceeds the number of scooters available.