VEHICLE RENTAL PROGRAM

Presented by: Jason Khoo

TABLE OF CONTENTS

ABOUT THE PROGRAM

- Developed to serve the needs of Friendly Vehicle Agency

Agency's Traits

- Small firm
- 3 vehicle types

Key Needs

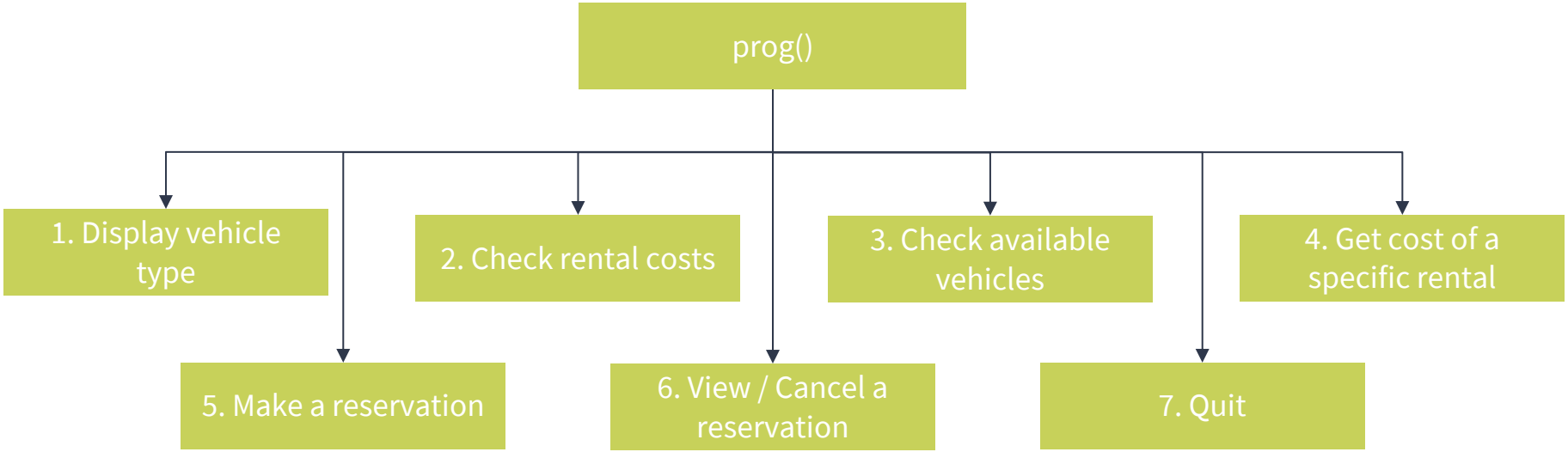
- User-friendly for user self service
- Efficient vehicle management system
- Effective reservation management

Program Functionalities

1. Check for available vehicles for rent
2. Request rental charges by vehicle type
3. Calculate cost of a specified rental
4. Make / Cancel reservations

PROGRAM DEMO / DESIGN

- Program demo on functionalities



PROGRAM DEMO / DESIGN

- Modular code design using functions and classes

```
**********************************************
* Welcome to the Friendly Vehicle Rental Agency *
**********************************************

<<< MAIN MENU >>>
[1] Display vehicle types
[2] Check rental costs
[3] Check available vehicles
[4] Get cost of specific rental
[5] Make a reservation
[6] View / Cancel a reservation
[7] Quit
```

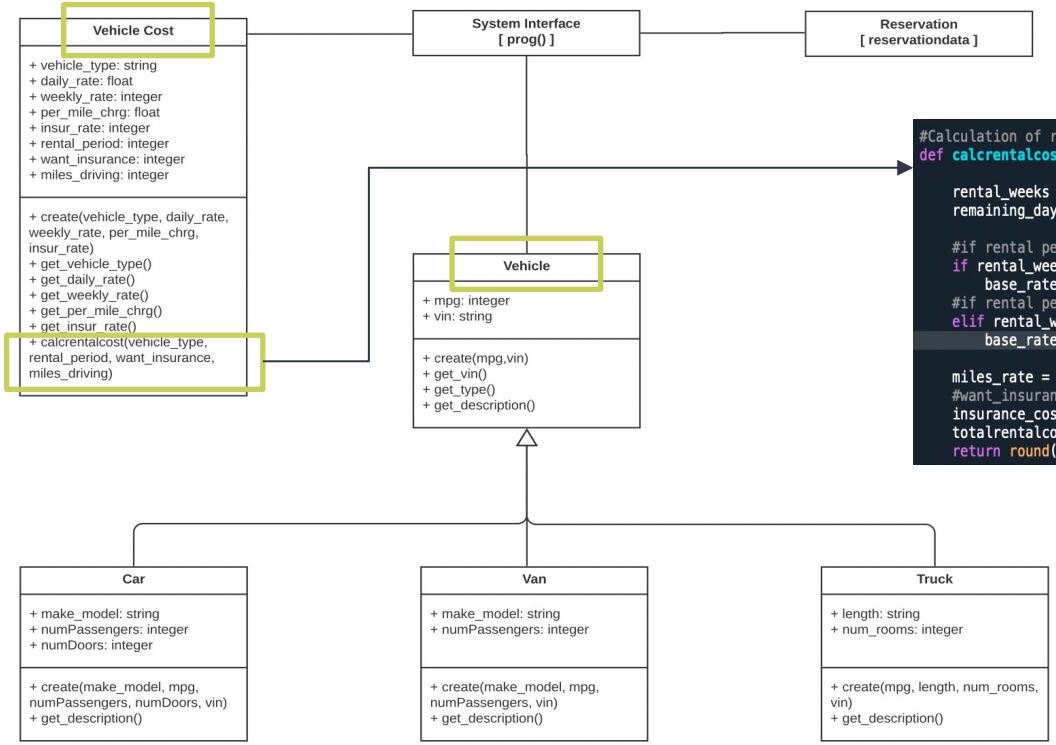
```
def prog():

    menu()
    test = True

    while test:

        option = input("Enter Option: ")

        if option == "1":
            print("Option 1 is selected.\n")
            vehtype()

        elif option == "2":
            print("Option 2 is selected.\n")
            test = False
            vehrentalcost()

            returnmainmenu = input("Enter [R] to return to main menu.\nOtherwise, enter

            if returnmainmenu == "R":
                test = True
                print("Returning back to the main menu ......\n")
                time.sleep(1)
                menu()

            else:
                print("Thank you for using the Friendly Rental Agency. See you soon!")

        elif option == "3":
            print("Option 3 is selected.\n")
            test = False
            checkavailvehicle()

            returnmainmenu = input("Enter [R] to return to main menu.\nOtherwise, enter

            if returnmainmenu == "R":
                test = True
                print("Returning back to the main menu ......\n")
                time.sleep(1)
                menu()

            else:
                print("Thank you for using the Friendly Rental Agency. See you soon!")
```

text-based using print statements

Sub functions codes

PROGRAM DEMO / DESIGN

- Modular code design using functions and classes




```python
#Calculation of rental cost based on a given vehicle type, rental period, opt for insurance and miles
def calcrentalcost(self, vehicle_type, rental_period, want_insurance, miles_driving):

    rental_weeks = rental_period // 7
    remaining_days = rental_period % 7

    #if rental period is less than 7 days -> daily rate applies
    if rental_weeks == 0:
        base_rate = rental_period * self.daily_rate
    #if rental period is more than 7 days -> weekly rate + daily rate applies (e.g. 9 days -> 1 week + 2 day)
    elif rental_weeks >= 0:
        base_rate = (rental_weeks * self.weekly_rate) + (remaining_days * self.daily_rate)

    miles_rate = self.per_mile_chrg * miles_driving
    #want_insurance = 1 (true) or 0 (false)
    insurance_cost = want_insurance * self.insur_rate * rental_period
    totalrentalcost = base_rate + miles_rate + insurance_cost
    return round(totalrentalcost,2)
```

PROGRAM DEMO / DESIGN

- File I/O (1. Vehicle data, 2. Vehicle cost data, 3. Vehicle reservation data)

1. Vehicle data

Load data

```
with open('vehicle_car_data.txt') as json_cardata:
    vehiclecardata = json.load(json_cardata)

with open('vehicle_van_data.txt') as json_vandata:
    vehiclevandata = json.load(json_vandata)

with open('vehicle_truck_data.txt') as json_truckdata:
    vehicletruckdata = json.load(json_truckdata)
```

2. Vehicle cost data

Load data

```
with open('vehicle_cost.txt') as json_vehcost:
    vehiclecostdata = json.load(json_vehcost)
```

3. Vehicle reservation data

Load data

```
with open('listofreservation.txt') as json_reservationlist:
    reservationdata = json.load(json_reservationlist)
```

Write data

Within make / cancel reservation:

```
with open('listofreservation.txt', 'w') as outfile:
    json.dump(reservationdata, outfile)
```

- Performed after updating dictionary values in reservation data

PROGRAM DEMO / DESIGN

- Data Structures

1. Vehicle data

vehicle_car_data.txt


```
{"make_model": ["Chevrolet Camaro", "Chevrolet Camaro", "Ford Fusion", "Ford Fusion Hybrid", "Ford Fusion Hybrid", "Chevrolet Impala", "Chevrolet Impala"],
"mpg": [30, 30, 34, 35, 32, 36, 30], "numPassengers": [4, 4, 5, 5, 5, 6, 6],
"numDoors": [2, 2, 4, 4, 4, 4, 4],
"vin": ["WG8JM5492DY", "KH4GM4564GD", "AB4FG5689GM", "GH2KL4278TK", "KU4EG3245RW", "QD4PK7394JI", "RK3BM4256YH"]}
```


2. Vehicle cost data

vehicle_cost.txt


```
{"Vehicle Type": ["Car", "Van", "Truck"],
"Daily Rate": [24.99, 35, 34.95],
"Weekly Rate": [180, 220, 425],
"Per Mile Charge": [0.15, 0.20, 0.25],
"Insurance Charge": [6, 7, 8]}
```


3. Vehicle reservation data

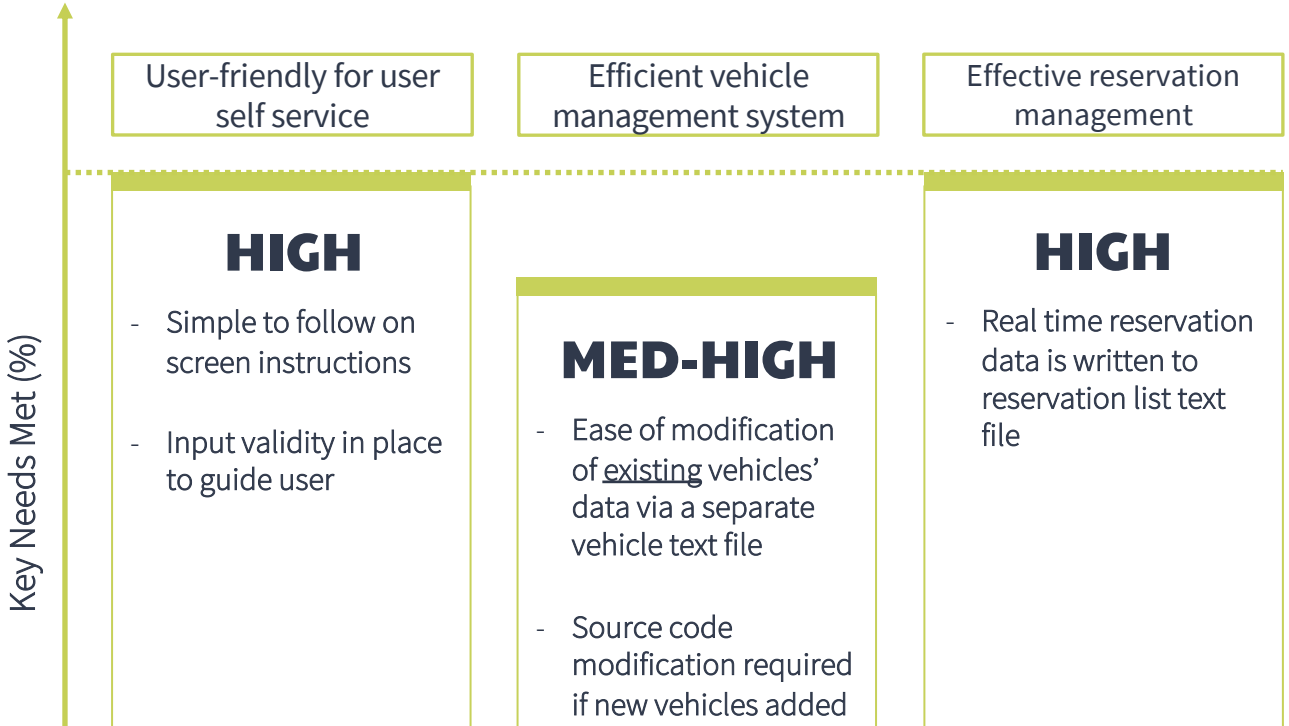
listofreservation.txt


```
{"Vehicle ID Number": [], "First Name": [], "Last Name": [], "Address": [], "Credit Card": [], "Mobile Number": []}
```


- Similar format for van / truck

PROGRAM EVALUATION

- Addressing key needs



PROGRAM EVALUATION

- Areas of improvement

	CODE OPTIMISATION	ADDITIONAL FUNCTIONALITIES
Key Action 1	Implementation of functions to minimise certain code repetition	Possible implementation of column sorting for available vehicles via sort( )
Key Action 2	Implementation of system interface and reservation module via OOP for greater integration	Possible implementation of GUI (e.g. tkinter) for interface
Key Action 3	Synchronise vehicle objects creation with vehicle data in text files (i.e. addition of a new car data in text file will automatically create a new car object)	

END OF PRESENTATION.

THANKS!