MIS 6382
Object Oriented Programming with Python
Fall 2019
Homework Four

Once you have written the code to fulfill the requirements of this assignment, you must test the assignment using the test data that is provided in the excel spreadsheet. Your final submission must include the empdata.dat file together with a single notebook (ipynb) file containing all the code.

Refer to the example in the Homework Four Example folder for help with this assignment.

I. In this assignment you will create an employee database application by fulfilling the requirements given below. You must first create the classes as described. Then write a driver program that will use these classes to build an application for an Employee Database. All employee data is stored in a file called empdata.dat.

II. Each time you run the program, the user is allowed to repeatedly select from the four choices described below:
   1. Add a new employee
   2. Print all employee data for all the employees in the file
   3. Compute and print the employee name and compensation of all employees
   4. Print the employee name with vehicle information vehicles with mileage greater than 78000 miles
   5. Exit the application

III. The first time you execute the program, your file will have no employees in it. Second time onwards, all employee data from previous executions of the program, should be available

Employee Application

- Create a base class Employee that has the following attributes:
  - Employee’s name (String)
  - Employee’s address (String)
  - Vehicle data (Vehicle).
- The child classes FullTimeEmployee, HourlyEmployee and Consultant that inherit from Employee have the following additional properties
  - FullTimeEmployee – salary (double).
  - HourlyEmployee - hoursWorked (int) and hourlyRate (double).
  - Consultant – hoursWorked (int) and ProjectType (valid values are 1, 2, and 3).
- All these classes have the __init__ method as well as the **get** and **set** methods. In addition, they have additional methods required to complete the application as described below.

- Compensation for each employee type is to be computed as follows:

- FullTimeEmployee: Compensation is salary minus taxes. Taxes are calculated based on the tax rate in the table below:

Salary	TaxRate
$45, 000 or less	18%
> $45,000 and <= $82,000	18% for the first 45000, 28% for the rest
> $ 82,000	18% for the first 45000, 28% for the amount between 45000 and 82000, and 33% for the rest

For example, someone whose salary is $123,000 will pay 18% on the first 45,000, 28% on the next (82,000 – 45,000) and 33% on the remaining (123,000 – 82,000)

- HourlyEmployee: Compensation is hoursWorked times hourlyRate for the first 40 hours. For hours in excess of 40 hours the hourly rate is 1.8 times the regular hourly rate.
  For example, someone whose hourlyRate is $12.50 and who has worked 48 hours will earn 40 * $12.5 + 8 * $12.5 * 1.8

- Consultant: Compensation is hourlyRate times the hours worked. HourlyRate for Consultants is computed based on the ProjectType as given in the table below:

• Project Type	• HourlyRate
• 1	• $55.00
• 2	• $70.00
• 3	• $85.00

- The Vehicle class is as described below: It has four instance variables – make, model, year of manufacture and mileage. It should have a constructor (**__init__** method) which accepts values for all of the instance variables. You may use aggregation or composition to include a Vehicle object to your Employee data.

- Now write a driver program to do the following:
  1. Accept input for new Employees
  2. Display employee information (including vehicle information) for all the employees that are there in the system.
  3. List the name of the Employees along with the compensation received by each.
  4. Display the employee name together with the make, model and mileage of all vehicles whose mileage is greater than 78000 miles.
  5. Exit the system after writing all information to a file.

  Any employee data from earlier executions of the application must be available in subsequent executions.

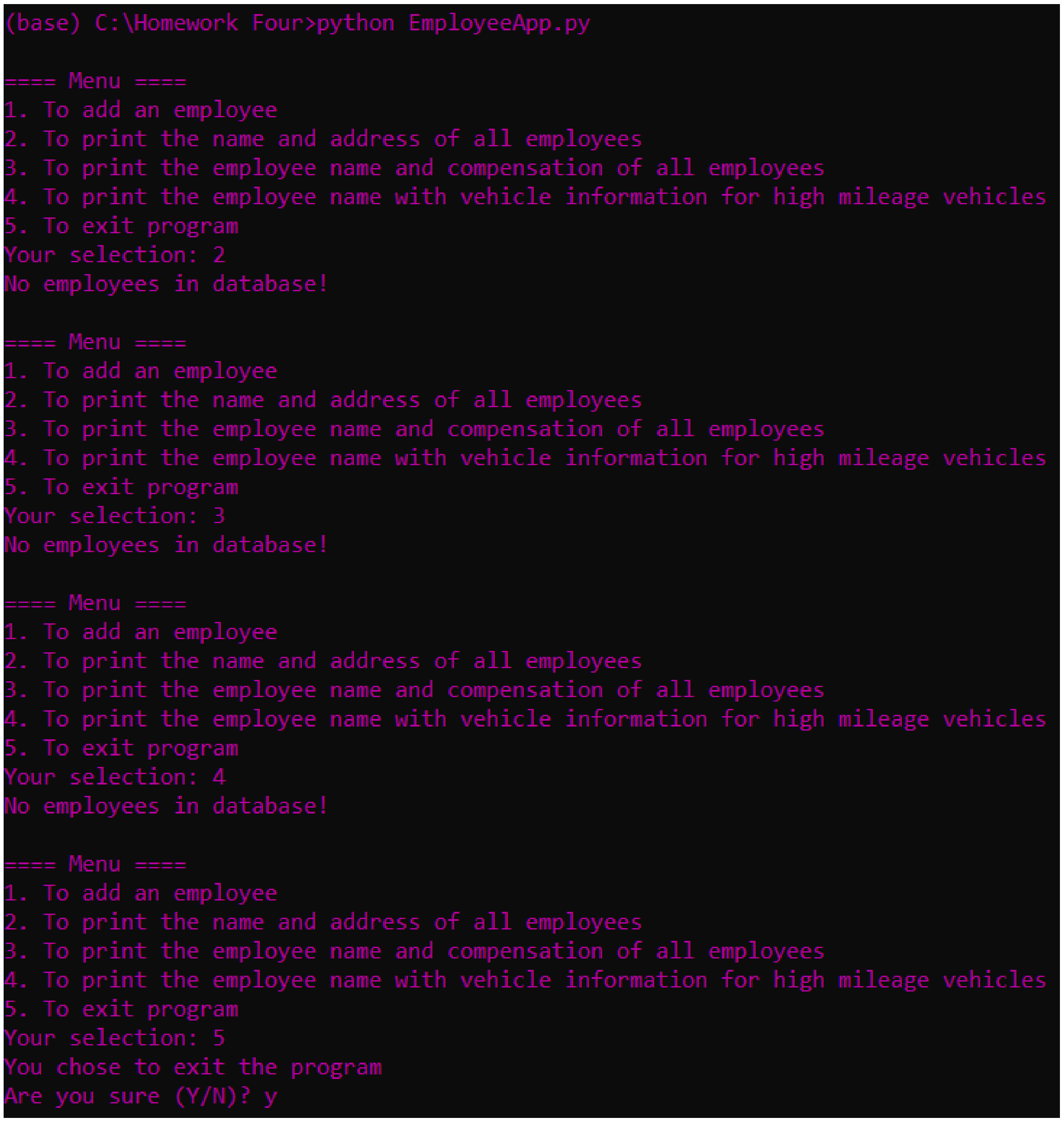
  - Additional Requirements:
    - Your program should use exception handling to validate user input for the following variables:
      - Numeric values for *year of manufacture, mileage, salary, hours worked* and *hourly rate*.
      - The only acceptable values for *project type* are 1, 2 or 3. If the user enters invalid values for any of the variables above, print an appropriate message and ask the user to re-enter a valid value.
    - Your program should also ensure that the user enters valid values for the menu options as well as the choice of employee type. If the user enters incorrect options for either the menu option or the pet type, your program should display an appropriate message and display the menu or the choice of employee types again.

II: Submission instructions

1. **Submit a single notebook containing all class definitions together with the driver code.**
2. **You must also submit the data file (empdata.dat) with the records containing all data for the employees given in the test data file. You must execute your program and enter data for all the employees listed in the test data file. The resultant empdata.dat will therefore contain all the data I have provided as test data. This will enable the TA to more easily grade your homework.**
3. **You will automatically lose 40% of the grade if your program does not compile.**

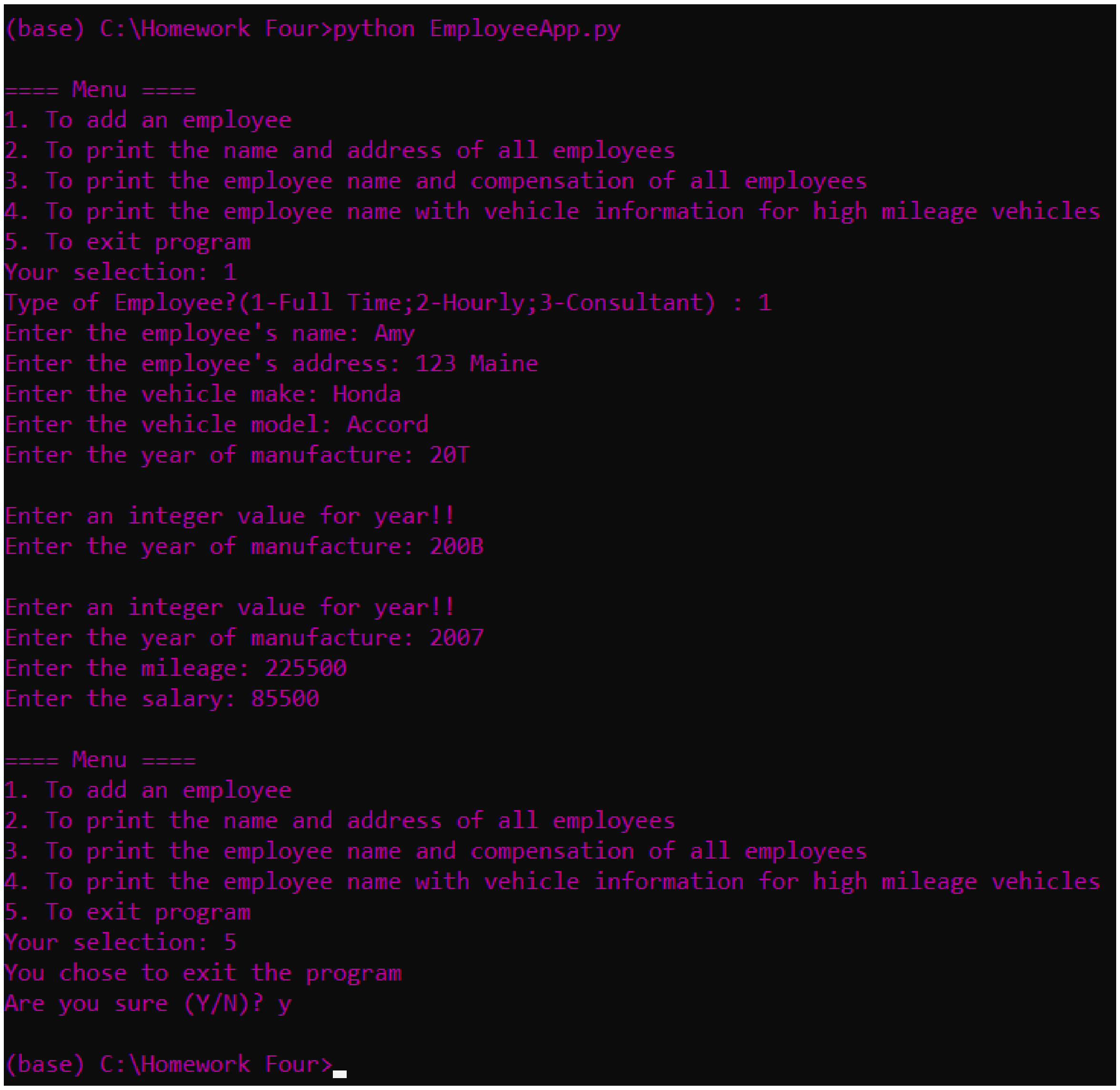
The following screen shots provide a sample interaction with the system.

I: If there are no employees in the database and any one of options 2, 3, or 4 are selected, the application should display an appropriate message as shown in the screen shot below.



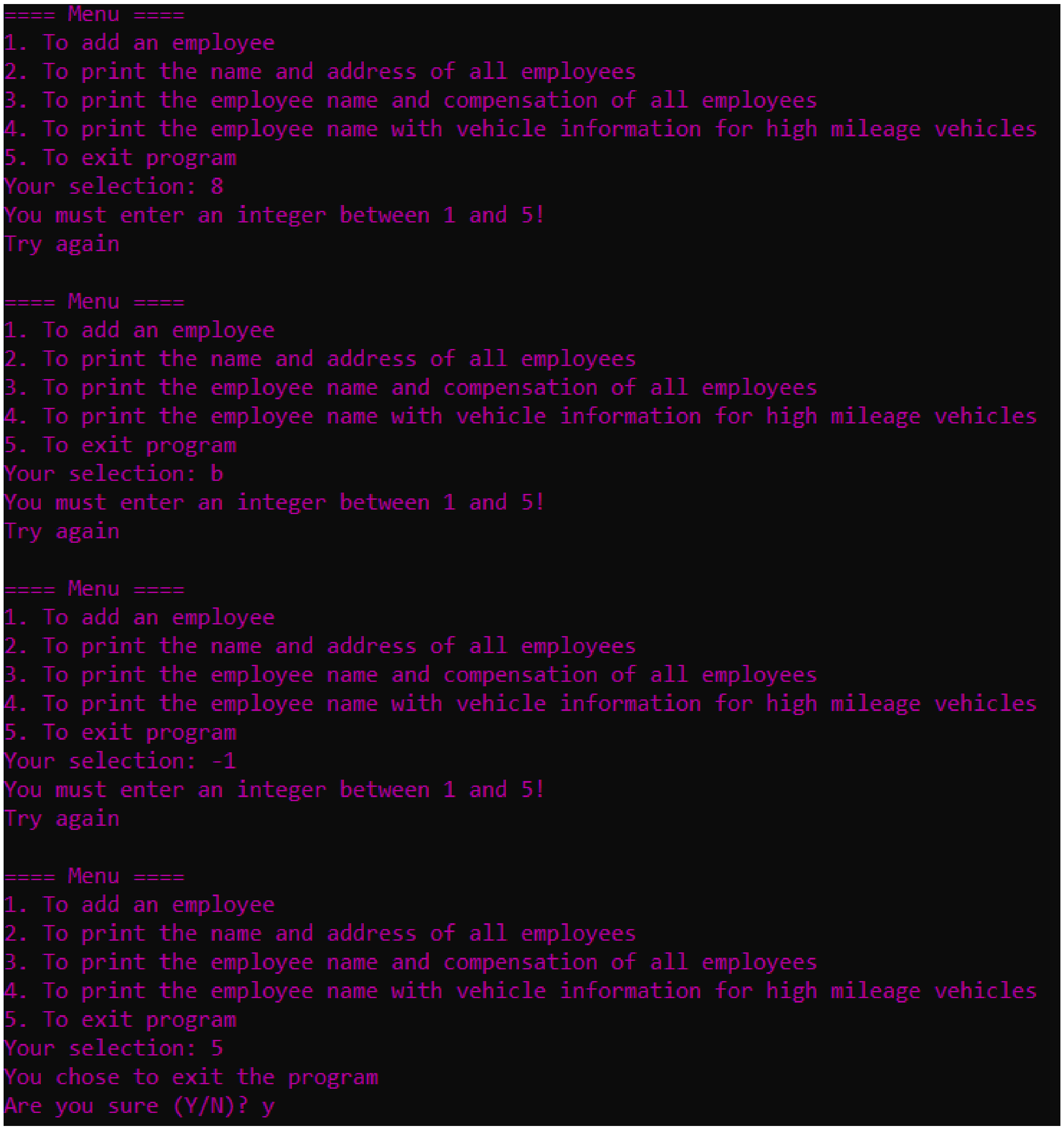
```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 2
No employees in database!

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 3
No employees in database!

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 4
No employees in database!

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 5
You chose to exit the program
Are you sure (Y/N)? y
```

II: If The user enters a non-numeric value for any one of the numeric variables, the application should display an appropriate message and repeatedly ask the user for a valid value as shown in the screen shot below.



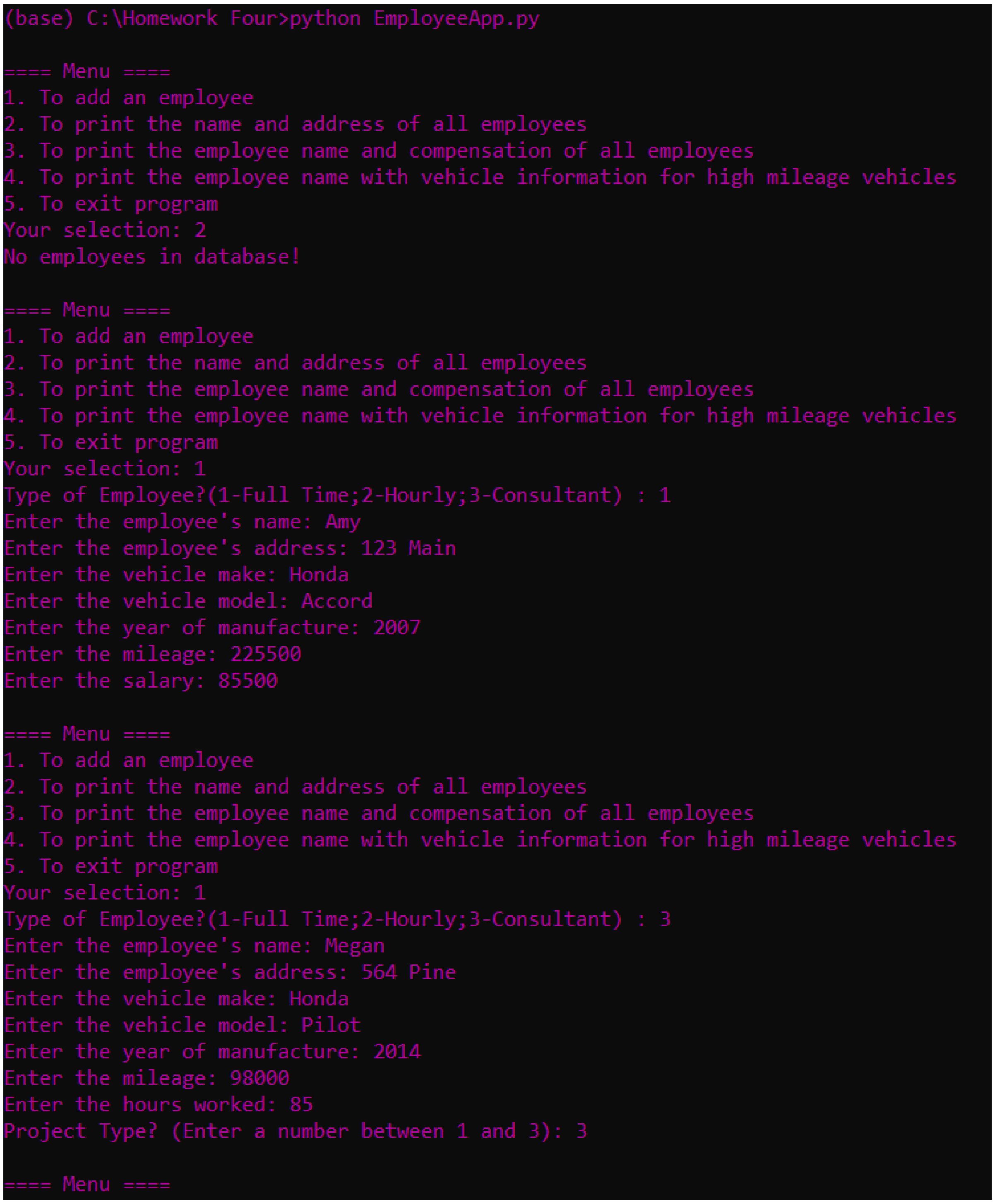
```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 1
Type of Employee?(1-Full Time;2-Hourly;3-Consultant) : 1
Enter the employee's name: Amy
Enter the employee's address: 123 Maine
Enter the vehicle make: Honda
Enter the vehicle model: Accord
Enter the year of manufacture: 20T

Enter an integer value for year!!
Enter the year of manufacture: 200B

Enter an integer value for year!!
Enter the year of manufacture: 2007
Enter the mileage: 225500
Enter the salary: 85500

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 5
You chose to exit the program
Are you sure (Y/N)? y

(base) C:\Homework Four>
```

III: If the user enters a value other than 1, 2, 3, 4, or 5 as a menu option, the application should display an appropriate message and repeatedly ask for a valid value.



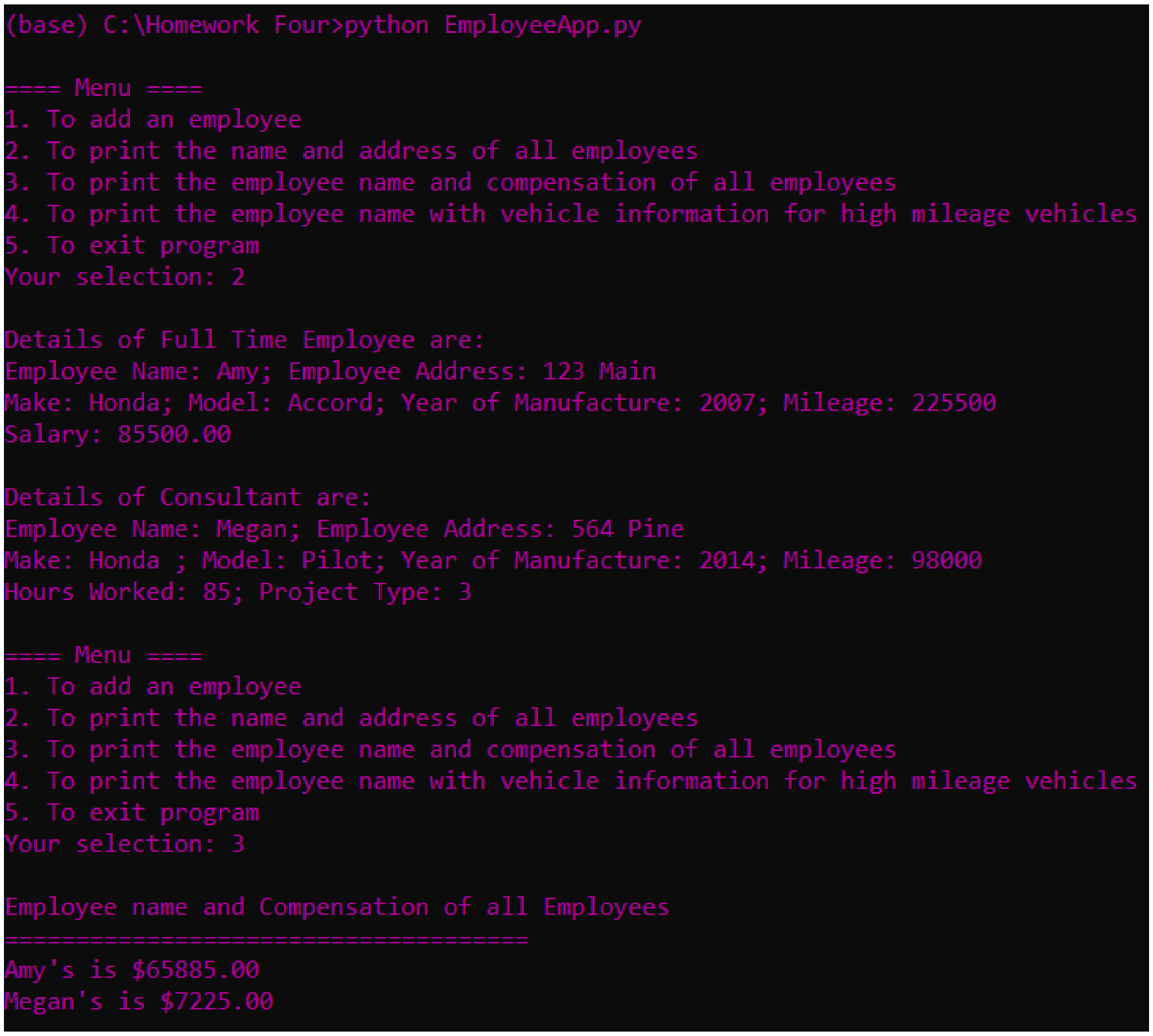
```
==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 8
You must enter an integer between 1 and 5!
Try again

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: b
You must enter an integer between 1 and 5!
Try again

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: -1
You must enter an integer between 1 and 5!
Try again

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 5
You chose to exit the program
Are you sure (Y/N)? y
```

IV: When the application is executed for the first time, the empdata.dat will not exist. It should be created, and data should be saved to this file before exiting the application. The screenshot below shows the case with no employees. Two employees are then added.



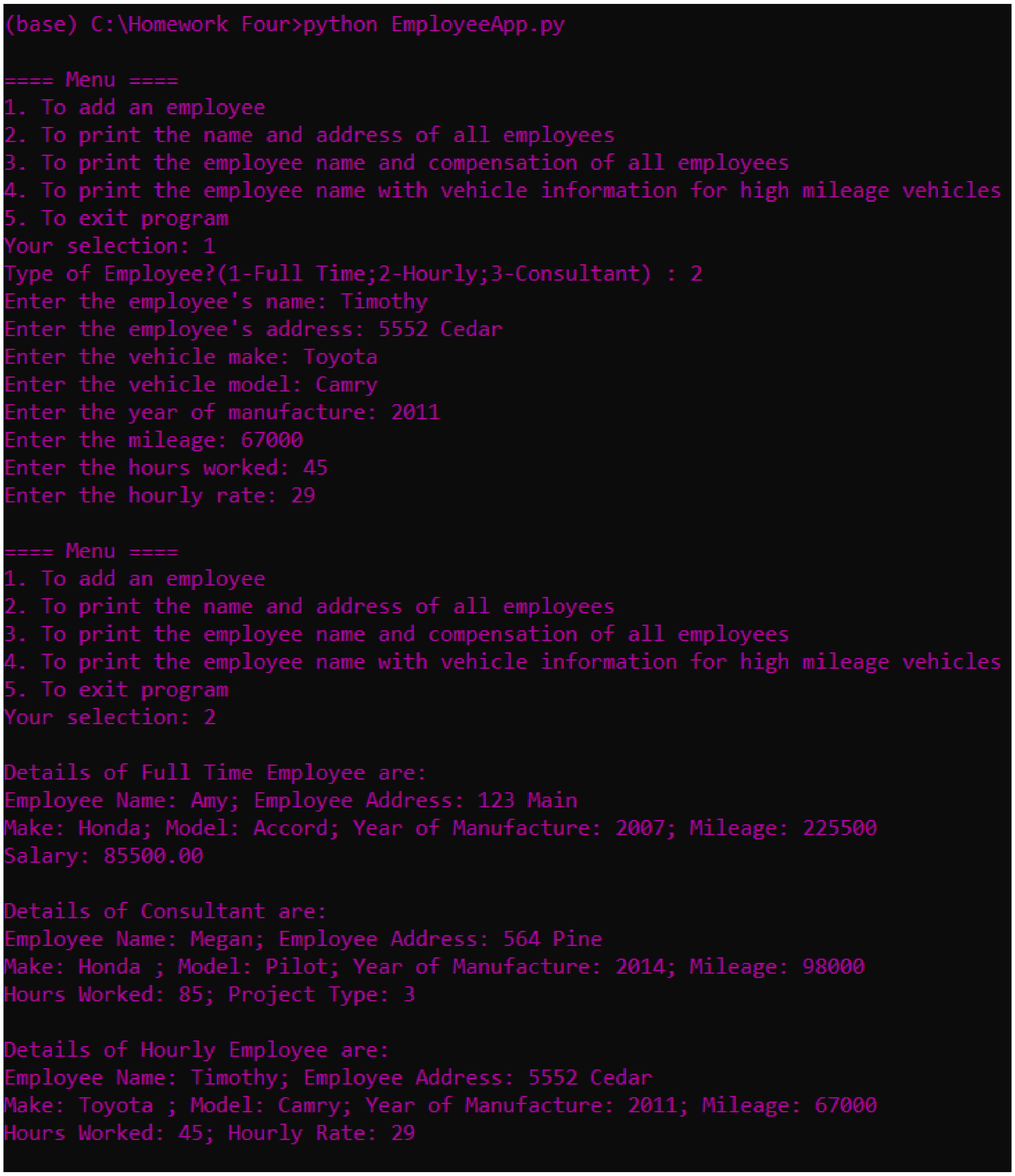
```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 2
No employees in database!

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 1
Type of Employee?(1-Full Time;2-Hourly;3-Consultant) : 1
Enter the employee's name: Amy
Enter the employee's address: 123 Main
Enter the vehicle make: Honda
Enter the vehicle model: Accord
Enter the year of manufacture: 2007
Enter the mileage: 225500
Enter the salary: 85500

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 1
Type of Employee?(1-Full Time;2-Hourly;3-Consultant) : 3
Enter the employee's name: Megan
Enter the employee's address: 564 Pine
Enter the vehicle make: Honda
Enter the vehicle model: Pilot
Enter the year of manufacture: 2014
Enter the mileage: 98000
Enter the hours worked: 85
Project Type? (Enter a number between 1 and 3): 3

==== Menu ====
```

V. When the application is executed the second and subsequent times, any employees that were added in earlier runs should be available in the file.



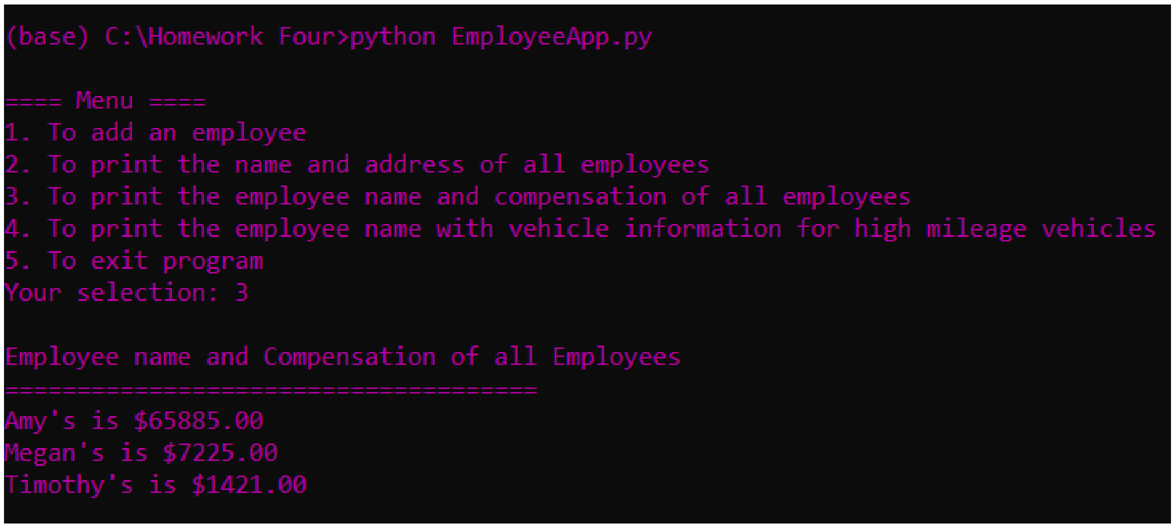
```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 2

Details of Full Time Employee are:
Employee Name: Amy; Employee Address: 123 Main
Make: Honda; Model: Accord; Year of Manufacture: 2007; Mileage: 225500
Salary: 85500.00

Details of Consultant are:
Employee Name: Megan; Employee Address: 564 Pine
Make: Honda ; Model: Pilot; Year of Manufacture: 2014; Mileage: 98000
Hours Worked: 85; Project Type: 3

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 3

Employee name and Compensation of all Employees
=====================================
Amy's is $65885.00
Megan's is $7225.00
```

VI. When new employees are added, the user should be only asked for data that is pertinent to that employee type as shown below.



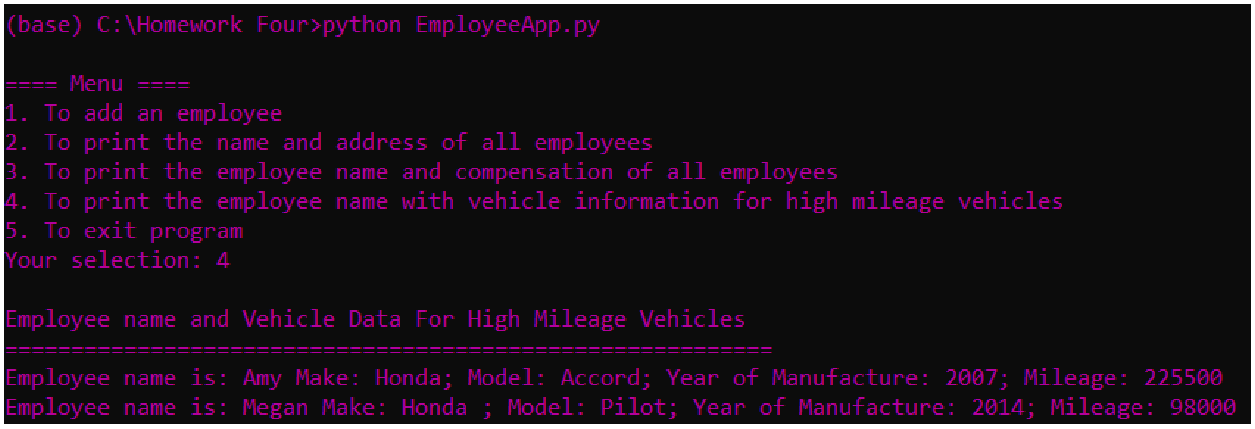
```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 1
Type of Employee?(1-Full Time;2-Hourly;3-Consultant) : 2
Enter the employee's name: Timothy
Enter the employee's address: 5552 Cedar
Enter the vehicle make: Toyota
Enter the vehicle model: Camry
Enter the year of manufacture: 2011
Enter the mileage: 67000
Enter the hours worked: 45
Enter the hourly rate: 29

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 2

Details of Full Time Employee are:
Employee Name: Amy; Employee Address: 123 Main
Make: Honda; Model: Accord; Year of Manufacture: 2007; Mileage: 225500
Salary: 85500.00

Details of Consultant are:
Employee Name: Megan; Employee Address: 564 Pine
Make: Honda ; Model: Pilot; Year of Manufacture: 2014; Mileage: 98000
Hours Worked: 85; Project Type: 3

Details of Hourly Employee are:
Employee Name: Timothy; Employee Address: 5552 Cedar
Make: Toyota ; Model: Camry; Year of Manufacture: 2011; Mileage: 67000
Hours Worked: 45; Hourly Rate: 29
```

VII. Options 2 and 3 should function as described above. Compensation should be correctly computed for each employee as shown below.



```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 3

Employee name and Compensation of all Employees
=====================================
Amy's is $65885.00
Megan's is $7225.00
Timothy's is $1421.00
```

VIII. Options 4 should function as shown below.



```
(base) C:\Homework Four>python EmployeeApp.py

==== Menu ====
1. To add an employee
2. To print the name and address of all employees
3. To print the employee name and compensation of all employees
4. To print the employee name with vehicle information for high mileage vehicles
5. To exit program
Your selection: 4

Employee name and Vehicle Data For High Mileage Vehicles
==========================================================
Employee name is: Amy Make: Honda; Model: Accord; Year of Manufacture: 2007; Mileage: 225500
Employee name is: Megan Make: Honda ; Model: Pilot; Year of Manufacture: 2014; Mileage: 98000
```

IX. When the user selects option 5, the data should be saved to the file empdata.dat before exiting the program.