Functions

Workshop 6 (out of 10 marks - 3% of your final grade)

In this workshop, you will code a C-language program with functions.

LEARNING OUTCOMES

Upon successful completion of this workshop, you will have demonstrated the abilities to:

- Define and implement functions
- Call functions with several input parameters and receive data back
- Understand reusability of functions for different use cases in the same program

SUBMISSION POLICY

The "in-lab" section is to be completed during your assigned lab section. It is to be completed and submitted by the end of the workshop period. If you attend the lab period and cannot complete the in-lab portion of the workshop during that period, ask your instructor for permission to complete the in-lab portion after the period. If you do not attend the workshop, you can submit the “in-lab” section along with your “at-home” section (with a penalty; see below). The “at-home” portion of the lab is due on the day of your next scheduled workshop (23:59).

All your work (all the files you create or modify) must contain your name, Seneca email and student number.

You are responsible to regularly back-up your work.

Late submission penalties:

- In-lab submitted late, with at-home: Maximum of 20/50 for in lab and Maximum of 50/50 for at home
- Workshop late for one week: in-lab, at-home and reflection must all be submitted for maximum of 50 / 100
- Workshop late for more than one week: in-lab, at-home and reflection must all be submitted for maximum of 30 / 100
- If any of in-lab, at-home or reflection is missing the mark will be zero.

IN-LAB: EMPLOYEE CLASS (50%)

Download or clone workshop 6 from https://github.com/Seneca-144100/IPC-Workshop6

Write this section of your code in rental.c provided with the visual studio template project inside in_lab folder. In this workshop segment, you will implement **rental charges** and **rental status display** functionality for vehicle rentals and will use C functions to add modularity to your solution.

Overview

The rental.c template file has already implemented the following:

- Displays the following menu list inside a do-while loop construct:
  1.) Rental Status
  2.) Apply Charges
  0.) Log out
- Captures user input for the above options and stores to an `int` variable named "option"
- Ability to iterate multiple menu choices (until 0 is entered) with required selection construct to direct process flow to the required logic/functionality (using switch). Refer to the comments for each case to identify the functionality required.
- Displays an error message for invalid menu option selections in the `default` case
- Initial output information for menu options 1 and 2 is provided with the relevant formatting.
- The template program has two rental accounts already defined and initialized with the following information using a `struct Rental` array named `vRent`. `struct Rental` has three member variables named `int id` , `double baseDay` and `double earnings`.

id	baseDay	earnings
123	9.95	0
124	19.95	0

- Case-2 has implemented the following, required to apply charges to the relevant rental account:
  - Prompts the user to input a vehicle `ID` number
  - Searches through all Rentals (`vRent` array) to find whether the Vehicle `ID` exists. If it exists, the matching array index is stored to a variable named `flag`. If it is not found, an error message is displayed.

You are required to complete the following functionality.

Define and Implement addPositive Function

- Prototype a C function called **addPositive** with two input parameters "`double amount` and `double newAmount`" before the main function
- Implement **addPositive** function anywhere below its prototype definition with two input parameters "`double amount` and `double newAmount`". This function adds the

values of both input parameters only if **newAmount** is a positive value; otherwise, **amount** value is unaltered. This function returns the resulting **amount** as a **double**.

Implement "Rental Info Display" functionality in Case 1

- Display "**id** and **earnings**" of the **struct Rental** array using a loop construct. Use the following formatting for the **printf** statement.

  `%8d%10.2lf` ←→ (*id, earnings*)

- After completing the display, enter a newline using a **printf** statement.

Implement "Apply rental charges" functionality in Case 2

- If the Rental **ID** is found in **vRent** array, then do the following, inside "`if (flag != -1)`" code block:
  - Prompt and capture the duration in days of the car rental.
  - Calculate the **base price** by multiplying the duration (days) and **baseDay** rate (e.g: 9.95 or 19.95) for the rental
  - Prompt and capture traveled distance in kilometers of the car rental.
  - Calculate the **kilometer cost** of the rental as per the following chart. **distRate1** and **distRate2** are defined at the beginning of the template file.

Description	Variable Name	Rate / km
Rate per Kilometer from 0km to 100km (inclusive)	**distRate1**	*0.69*
Rate per Kilometer above 100km	**distRate2**	*0.49*

  - Add the **"base price"** and **"kilometer cost"** and store to a temporary **double** variable named **charge**
  - Call the **addPositive** function supplying it with the current **earnings** for the rental and **charge** (calculated above) as arguments assigning the returned result back to **earnings**
  - Display a message showing the **base price**, **kilometer cost** and the **new earnings** (base + kilometer cost) with the following formatting:

    `%6.2lf %6.2lf %6.2lf` ←→ (*base price, kilometer cost, earnings*)

- Please refer to the sample output below for exact message contents and formatting

PROGRAM COMPLETION

Your program is complete, if your output matches the following output. Numbers/ text in red color shows the user's input.

```
***** Rental Vehicle Management App *****

1.) Rental Status
2.) Apply Charges
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 123
Enter Rental Period (in Days): 2
Enter kilometers driven: 125

Base   kmCost Total
====== ====== ======
 19.90  81.25 101.15

1.) Rental Status
2.) Apply Charges
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 124
Enter Rental Period (in Days): 3
Enter kilometers driven: 79

Base   kmCost Total
====== ====== ======
 59.85  54.51 114.36

1.) Rental Status
2.) Apply Charges
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 125
ERROR: Vehicle ID does not exist.

1.) Rental Status
2.) Apply Charges
0.) Log out

Please enter an option to continue: 1
```

```
-- Rental Vehicle Status --

ID#      Earnings
-------- ----------
     123     101.15
     124     114.36

1.) Rental Status
2.) Apply Charges
0.) Log out

Please enter an option to continue: 0
```

IN_LAB SUBMISSION:

To test and demonstrate execution of your program use the same data as the output example above or any information needed.

If not on matrix already, upload your **rental.c** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

```
~profname.proflastname/submit 144_w6_lab <ENTER>
```

and follow the instructions.

If the professor is not satisfied with your implementation, your professor may ask you to resubmit. Resubmissions will attract a penalty.

AT_HOME: (40%)

The rental2.c template file has already implemented the following: (rental2.c is an extension of rental.c)

- Changed the value of **noVehicles** to 3.
- Increased the content of the **vRent** array to 3 rentals as following:

id	baseDay	earnings
123	9.95	0
124	19.95	0
125	29.95	0

- Expanded the menu list to include options 3 & 4 after option 2 with the following
  - Apply Taxes to All Rentals
  - Apply Gift Card

- Have created two switch-cases for option 3 & 4 after case 2. Initial output information for menu options 3 and 4 is provided with the relevant formatting.
- Case-4 has also implemented the following required to apply Gift Card amount to the relevant rental account:
  - Prompt the user to input a vehicle **`ID`** number
  - Search through all Rentals (**`vRent`** array) to find whether the Vehicle **`ID`** exists. If it exists, the matching array index is stored to a variable named **`flag`**. If it is not found, an error message is displayed

You are required to complete the following functionality.

Copy Case-1 and Case-2 contents as required from rental.c to rental2.c along with your user variables. See comments in rental2.c for more information.

Define and Implement taxCalc Function

- Prototype a C function called **taxCalc** with two input parameters “**`double price`** and **`double rate`**” before the main function. See comments in rental2.c to see where to place the prototype function.
- Implement **taxCalc** function anywhere below its prototype definition with two input parameters “**`double price`** and **`double rate`**”. This function calculates the taxes for the **`price`** using the **`rate`** given. This function returns the resulting **`taxes`** as a **`double`**.

Define and Implement subtractPositive Function

- Prototype a C function called **subtractPositive** with two input parameters “**`double amount`** and **`double newAmount`**” before the main function. See comments in rental2.c to see where to place the prototype function.
- Implement **subtractPositive** function anywhere below its prototype definition with two input parameters “**`double amount`** and **`double newAmount`**”. This function subtracts **`newAmount`** from **`amount`** parameter only if **`newAmount`** is a positive-value; otherwise, **`amount`** value is unaltered. This function returns the resulting **`amount`** as a **`double`**.

Implement “Apply Discount” functionality in Case 2

- Display the following after the call to **addPositive** function with `earnings` and `charge`

  ```
  > Apply Discount: <
  ```

- Prompt the user to input ‘**Y**’ or ‘**y**’ to apply discount. If the input is either ‘**Y**’ or ‘**y**’ call **subtractPositive** function with current `earnings` for the rental and `discount` (`double discount` is defined at the beginning of the **main** function in rental2.c) as arguments and assign the returned result back to `earnings`.
- Modify the display message to include **discount status** and **new total** (base + kilometer cost – discount) with the following formatting

  `%6.2lf %6.2lf %10c %6.2lf` ←→ (*base price, kilometer cost, discount status, earnings*)

Implement “Apply Taxes” functionality in Case 3

- Iterate over all rental vehicles using a loop construct. For each vehicle, call **taxCalc** function with current `earnings` and `taxRate` (`double taxRate` is defined at the beginning of the main function) as arguments and assign the returned result to a temporary variable called `double taxes`
- Then call **addPositive** function with the current `earnings` and `taxes` (calculated above) as arguments and assign the returned result back to `earnings` for the respective rental.
- After completing the display, enter a newline using a `printf` statement.

Implement “Apply Gift Card” functionality in Case 4

- Display the following inside `if (flag != -1)` code block:

  ```
  > Enter Gift Card Amount: <
  ```

- Prompt the user to input a gift card amount
- Call **subtractPositive** function with current `earnings` and the above gift card amount and assign the returned result back to `earnings`.

Program completion

Your program is complete if your output matches the following output. Numbers/ text in red color shows the user’s input.

```
***** Rental Vehicle Management App *****

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 123
Enter Rental Period (in Days): 2
Enter kilometers driven: 125
Apply Discount: Y

Base   kmCost DiscStatus Total
====== ====== ========== ======
 19.90  81.25          Y  96.15

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 124
Enter Rental Period (in Days): 3
Enter kilometers driven: 60
Apply Discount: N

Base   kmCost DiscStatus Total
====== ====== ========== ======
 59.85  41.40          N 101.25

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 2

-- Rental Charges --

Enter vehicle ID: 125
Enter Rental Period (in Days): 1
Enter kilometers driven: 223
Apply Discount: Y

Base   kmCost DiscStatus Total
====== ====== ========== ======
 29.95 129.27          Y 154.22

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 1

-- Rental Vehicle Status --

ID#      Earnings
-------- ----------
     123      96.15
     124     101.25
     125     154.22

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 3

-- Apply Taxes To all Accounts--

ID#      Earnings   Taxes
-------- ---------- ------
     123     109.61  13.46
     124     115.42  14.18
     125     175.81  21.59

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 4

-- Apply Gift Card --

Enter vehicle ID: 125
Enter Gift Card Amount: 23
Discount Applied

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out
```

```
Please enter an option to continue: 1

-- Rental Vehicle Status --

ID#      Earnings
-------- ----------
     123     109.61
     124     115.42
     125     152.81

1.) Rental Status
2.) Apply Charges
3.) Apply Taxes to All Rentals
4.) Apply Gift Card
0.) Log out

Please enter an option to continue: 0
```

AT-HOME REFLECTION (10%)

Please provide brief answers to the following questions in a text file named **reflect.txt.**

1) What is a function and briefly discuss the need for functions in any language?
2) Comment on the modularity and reusability of **addPositive** and **subtractPositive** functions. (Hint: Each function was called twice for different purposes each time)
3) How do you send and receive data back to and from functions?

AT_HOME SUBMISSION:

To test and demonstrate execution of your program use the same data as the output example above.

If not on matrix already, upload your **rental2.c and reflect.txt** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

```
~profname.proflastname/submit 144_w6_home <ENTER>
```

and follow the instructions.

Please note that a successful submission does not guarantee full credit for this workshop.

If the professor is not satisfied with your implementation, your professor may ask you to resubmit. Resubmissions will attract a penalty.