Sivan Moodley

ST10084225

PROG6221 POE

Change Log

Change Log

Part 1

Scenario

In Part 1, I set up the skeleton of the program that will eventually become a fully-fledged WPF application in part three. I did this by:

- Creating a CLI program in C# that will be used as a budget planner.
- The program requires fields for:
- ➢ The user's gross monthly salary.
- ➢ Their estimated tax deducted monthly.
- ➢ Their estimated monthly expenses on groceries.
- ➢ Their estimated monthly expenses on water and lights.
- ➢ Their estimated monthly expenses on travel costs including petrol.
- ➢ Their estimated monthly expenses for cell phone and telephone billing.
- ➢ Any other expenses they might have in the month.

- The user will also be able to select if they'd like to rent or buy a property.
- If the user chooses to rent the property the following input must be given by the user:
- ➢ The purchase price of the property.
- ➢ The total deposit.
- ➢ The interest rate.
- ➢ The number of months given to them to repay(240 or 360).

- With this information the program will calculate the monthly repayment value of the property.
- If the user's monthly home loan repayment is greater than one third of their salary then, the program should notify the user that approval of their home loan is unlikely.
- The program will also calculate the amount of money the user will have left over after their monthly expenses.

System Design

- Created the main class and the display method.
- Created the abstract *Expenses* class and its abstract method *CalculateExpense*.
- Created the child classes of the *Expenses* class *RentalExpense* and *HomeLoanExpense, along with them implementing the CalculateExpense method*.
- Created the *PopulateArrayLists* class.
- Initialized the array lists in the *PopulateArrayLists* class.
- Created the *SumArray* method in the *PopulateArrayLists* class.
- Implemented user input prompts from the main class for all prior initialized fields.
- Generated *Get and Set* methods to bring abstraction to the user input.
- Implemented calculations in the main, *RentalExpense* and *HomeLoanExpense classes.*
- House Keeping and comments.
- Implemented an error handling system.

Part 2

Scenario

In part 2 I had to add the option for the user to buy a vehicle, if they choose to buy a vehicle, they will be prompted to enter the following information:

- The model and make of the car.
- The purchase price of the car.
- The total deposit.
- The interest rate.

As well as a function to print a report of the user's expenses in descending order. I also had to use a delegate to notify the user if their monthly expenses cost more than 75% of their salary.

System Design

- I created a new abstract class called CarExpenses along with its abstract method *CarCalcExpense*.
- I created a new child class of *CarExpenses* called *CarLoanExpense and implemented CarCalcExpense in it.*
- Added user input prompts in the main class to get the input needed to perform calculations.
- Generated *Get and Set* methods to bring abstraction to the user input.
- Implemented calculations in the *CarLoanExpense* class.
- Created a new delegate *notifyUser* in the main class.
- Instantiated an object of the delegate *del_notifyuser.*

- Invoke the delegate using the delegate object, in the *BudgetReport* method.
- Created a system to sort the expenses array list in descending order.
- Implemented descending budget report print.
- Added an index page and *KeySearch* comments for easier navigation.
- Implemented changes based on feedback
- House keeping
- Streamlined method call system, most major methods now return to the main after being called. This was done to improve code reusability for the next part of the POE.
- Renamed the *PopulateArrayLists* class to *PopulateDictionary.*
- Renamed the *PopulateDictionary* class object in the main class *populateDict*
- Changed the programs storage systems from array lists to a dictionary instead, it makes the process of printing the expenses and their names in descending order.
- Added a return to main menu feature
- Added a new branch to my GitHub Repository *PROG6221_POE_Part_2*.
- Synced and pushed current program version to GitHub.