Alexa Javellana
CMPT220
11/06/16

Project Milestone

In regards to my project, the first steps towards it's final stages have taken place. As indicated before in my proposal, my project is an application that serves as a random content generator. When run, the program asks for the user to input a numeric value in order to provide feedback, in this case being a recipe that would equate in value to what the user typed in. There are two main parts to this application's code that I will explain in this paper, being the RecipeRun class and the Recipe class, which contain all contents in order for it to run correctly. Each class has 3 or more elements to them, and I plan on adding more features as I progress in the development of my app.

Why I decided to choose making an application on this topic for my project is mostly because of it's relevancy to my current situation. As a college sophomore at Marist living in one of the townhouses, I have been faced with the everyday task to cook for myself, at least 2 times a day. I've reached a point to where I simply repeat recipes I have cooked before, and lack variety in my meals though it has only been 2 months into the school year (with around 6 months to go). There are definitely many other students out there with this same problem, including my housemates who tell me of their lack of culinary diversity, which prompted me to view creating a solution by addressing the issue in my project. The rest of this paper will be used to explain how the application runs, it's restrictions specified in the code being constructed, the content provided from the application to the user, any other miscellaneous but important features I plan on adding during it's construction and mentions of future developments.

To begin digesting how the program works, the UML diagrams are provided below, and the relationships between each element will be explained.

RecipeRun	
- userMoney: double	- The user input budget
- userRecipe: String	- The user's recipe from the budget
- isVegetarian: String	- Whether or not the user would like vegetarian recipes to be presented

Recipe	
- mealBudget: double	- The default budget of the meal
- budget: double	- The user's input budget of the meal
+Recipe()	- Recipe constructor
+Recipe(budget): double	- Equates the new user input to the mealBudget
+showNonVRecipe(): void	- Shows a non-vegetarian option out of two recipes in the program
+showVRecipe(): void	- Shows the singular vegetarian option for the user input budget

When the program is run, a prompt is printed that introduces the name of the application and asks for the user to input a double value that will act as variable userMoney. In the program, I created an if statement that restricts userMoney to be a value below 20.01. I set the maximum number to 20.01 because I reasoned that most college students wouldn't spend more than 20 dollars on a singular meal, since the average money spent on groceries per week between both men and women at 19-50 years old range from 40-60 dollars, at a low-cost budget (which most college students would define with since they're financially tied)[1]. If the user inputs a double greater 20.01 or any value that isn't proper for a double, the program will encounter an error or display a message that indicates an invalid budget has been inserted and exit the program. If the user puts in a proper value, userMoney will adopt this value and then be used to create a new Recipe class called userRecipe.

A new line of text will be shown, restating the new userMoney value in a sentence by using the Recipe class to show it (also known as mealBudget in the Recipe class), and then also ask if the user would like to see vegetarian options for their recipe. It will prompt for an input, either being yes or no. Dependent on what the user says, case insensitive, the program will display a recipe. If the user says "yes", a vegetarian recipe will be presented by invoking userRecipe.showVRecipe. If the user says "no", a non-vegetarian recipe will be presented by

[1] "Official USDA Food Plans: Cost of Food at Home at Four ...," , accessed November 7, 2016, https://www.cnpp.usda.gov/sites/default/files/usda_food_plans_cost_of_food/CostofFoodJul2014.pdf.

invoking userRecipe.showNonVRecipe. Which recipe is presented, though, depends on the value of mealBudget. In the Recipe class, the methods showVRecipe() and showNonVRecipe() evaluate mealBudget and println recipes based on what mealBudget equals. The intervals for the mealBudget recipes are separated from 1-5, 6-10, 11-15 and 16-20. This means that if mealBudget equals values from 1-5, 6-10, 11-15, or 16-20, certain recipes will be shown. In showNonVRecipe, there are two recipes for each interval, 8 recipes in total. Which recipe is shown in RecipeRun, though, depends on whether or not the userMoney input is odd or even, to give some sort of randomization. For showVRecipe, I have only chosen there to be one vegetarian recipe option for each interval, 4 recipes in total. So, if the user chooses yes to the isVegetarian boolean, they will only be presented one type of recipe according to their userMoney value. After the methods are invoked, the program ends and the user has received their recipe.

Since the program is pretty straightforward at this point without any hefty add-ons, most errors would occur if the user inputs something that is not the requested variable type i.e. inserting a string/word/phrase instead of a double value when prompted to enter a budget (userMoney). I would like to create error messages for when things like this occur, which I will try to implement in the future works of my project. Additionally, since it is so simple, I would like to use the rest of the time towards the projects development to be devoted to adding extra features, like more recipes for the showVRecipe method, updating the recipes to actually be recipes, and options like breakfast, lunch, and dinner. Nonetheless, given that this is it's alpha stage, the program has the most important features implemented in it at this point to at least give the user a general understanding of what the application has to offer if used.