Documentation for RPM - The Restaurant Price Manager

By: Shafin Mohammad

Basic Premise

The basic premise of RPM - The Restaurant Price Manager, is to help the user manage their time and money when they are paying their bill at a restaurant. RPM works by asking the user to go through a series of steps. To start users are prompted to enter in the dishes they ate as well as the costs. Users can add as many dishes as they like. After this, the user has 2 options, they can manage their costs by choosing to add different percentages of tips or discounts. Alternatively, they can proceed to just view their receipt, automatically adding tax. Finally, the user can split their bill with friends or family. Instead of wasting time calculating the total price with extras, RPM is designed to help you save time.

Key Features

A key feature of RPM is the modularity in the code, programmers can easily add or remove steps in the start select menu, edit or modify the program functions or even change the user parameters. RPM is a step-based program which lowers the amount of information that is presented to the user. Splitting the process into steps allows the user to feel less intimidated by the program and encourages them to user RPM more. The RPM was coded with upgradability in mind. Constants such as tax can be easily modified right from the top of the code. Additionally, the menu can be expanded to include more options. Following the steps provided below, you can add as many steps in the menu as you like! For ease of use, comments have been placed throughout the code to aid you in understanding and further developing the code.

<u>Three Simple Steps</u>

Step 1: Think of a menu that you would like to add on to the rpm and type it in as a println. Use the other menu options as a guide.
Step 2: Towards the end of the while loop, at the last else statement add another else if statement. In this statement type userInput == then a letter, once again use the other menu options as a guide.
Step 3: Code up a function you want to use below and then call it inside the else if statement. As another option, you can type up whatever code you like in the statement.

Another key feature of the RPM is the fact that all of the necessary features of a tip calculating program are present as well as other features the user may like. These features include the option to custom tip different percentages, reduce costs with discounts, change the tax rates and split the bill with friends and family.

The last feature about the RPM is the user experience, special prompts are displayed throughout the program when the user enters information. This makes using the RPM less intimidating to beginner users and encourages them to use the program more often. Some examples of these lucky “easter eggs” are: If the user enters a tip which is coincidentally larger than the math randomizer values of 30 and 90, the console prints WOW! That’s a BIG tip. Another example is after the user enters in their dishes the console prints MMMM! All of these dishes look soo goood! I have left a few more as secrets throughout and so continuous users of the RPM will notice them over time.

Controls / Game Mechanics

The controls of the RPM - The Restaurant Price Manager, are as followed:
Use the keyboard to type in your responses, utilize the brackets after each statement as a guide for what you should respond with. Alternatively, you can use the blue OK button present in the prompt as a continue button. If at any point you do not know what to press, as the instructions may be cut off, just press ok.

Limitations

A key limitation that the RPM has is a poor UI. As the whole program takes place in the prompt menu, there isn't much space to display text without having the user feel cluttered and confused. Instead of running the whole program in the prompt menu having a larger console to display the program would greatly enhance the function of the program. Another limitation the program has is the sequence in which the calculations are made. Since the program is coded, the user has no control over whether they would like to tip after paying tax or applying discounts. For now, the program runs in this order: Takes the subtotal and adds a tip, then reduces the price using any discounts, finally adds the tax to the final cost. The last limitation I found with the RPM is that the program can run worse if the user continuously switches between the menus and makes small changes every time. The prompts become delayed and the user is asked to press the blue ok button continuously.

Personal Comments

If I had more time to further elaborate on RPM - The Restaurant Price Manager. Firstly, I would try and fix the user interface. If the app were to be upgraded in the future, having a small prompt dialogue as the center bases of the whole program would reduce the function of the program drastically. Having to scroll down to read the instructions each time you leave a response can be challenging, increasing the chances of users switching to a different program altogether. For this reason, using a different code editor with a larger console or transferring the code to an app would allow the user to enjoy using RPM. Another feature I would like this app to have is the option to switch from percent to price for the discounts and tips. Instead of entering $5 as a discount the user would additionally have the option to write 10%. Further increasing the functionality of the RPM.