31.03.2022

FIX BUGS OF THE APP PROJECT

Prepared by: PRIYANKA DAS

Project objective:

As a developer, fix the bugs in the application using the appropriate algorithmic techniques.You have been assigned a few tasks during the sprint planning. Solving the bugs raised by the testing team is one among them. You are given the boilerplate code and are asked to complete it by fixing the bugs.

Bugs to be fixed:
Add the missing source code to the application based on searching technique. Find the appropriate comments to code for the searching technique.
Write source code for sorting the predefined array and ensure the functionality of the application. Find the appropriate comments to code for sorting the predefined array.
You can download the boilerplate code by executing the command below in your git bash.
git clone https://github.com/Simplilearn-Edu/Full-Stack---The-Desk-Application-.git.

We must use the following:

Eclipse/IntelliJ: An IDE to code for the application
Java: A programming language to develop the prototype
Git: To connect and push files from local system to GitHub
GitHub: To store the application code and track its versions
Search and Sort techniques: Select the relevant data structure algorithms to fix the bugs

About Project

The source code should be pushed to your GitHub repositories. You need to document the steps and write the algorithms in the Google Docs.
The link of your GitHub repository is must. In order to track your task, you need to share the link of the repository. You can add a section in the Google Docs. Document the step-by-step process involved in completing this task.
*/

About Project Code

The source code should be pushed to your GitHub repositories. You need to document the steps and write the algorithms in the Google Docs.
The link of your GitHub repository is must. In order to track your task, you need to share the link of the repository. You can add a section in the Google Docs. Document the step-by-step process involved in completing this task.

Ensures valid user input by catching an InputMismatchException which continues to the next iteration
to allow the user to try again.

Allow the user to pick another option if they make a non-existent choice by accident.
Sorts in ascending order using the existing functionality given to us by Collections.

Searching to see if an element exists in the List can be easily accomplished through using the indexOf method
The loop will not end unless option 6 is selected.

Empty the list if the user wants to delete all the expenses