CS162 - CSC10002

Solo Project
Data Structure Visualization

Department of Software Engineering-FIT-VNU-HCMUS

1

Description

The data visualization application provides an intuitive and user-friendly interface for the display and understanding data stored in basic data structures. Users can choose from a variety of data structures, including **static array**, **dynamic array**, **linked list (simply**, **doubly, or circular one)**, **stack**, and **queue**, and can easily **initialize**, **add**, **delete**, **update**, and **search data** within these structures.

When a data structure is selected, the application displays the existing data within the structure in a clear and easy-to-read format. Users can then perform various operations on the data using the following tools:

1. **Initialize**: the initialization function populates it with initial data. Users can choose to manually enter data into the structure or load data from an external file, or ask for randomized data. If input data is not provided, an empty data structure should be created.
2. **Add**: Users can add new data to the structure by entering the data into a form within the application. Once the data is added, the visualization is updated in real-time to show the new data point. The application should support 3 insertion modes: insert-to-the-first, insert-to-the-last and insert-to-the-middle.
3. **Delete**: Users can delete existing data points from the structure by selecting the data point and clicking on a delete button. The visualization is then updated to remove the deleted data point. The application should also support 3 deletion modes: delete-at-the-first, delete-at-the-last, delete-at-the-middle.
4. **Update**: Users can update existing data points within the structure by selecting the data point and making changes to the data in a form within the application. Once the changes are saved, the visualization is updated to reflect the updated data.
5. **Search**: Users can search for specific data points within the structure by entering search terms into a search bar within the application. The visualization is then updated to highlight the matching data points, making it easy to find the desired information.

They can also select different types of visualizations, depending on the type of data they are working with:

1. **Run-step-by-step:** allow the user to go to the next step, go to the previous step or go to the final step.
2. **Run-at-once:** when the user inputs data, the application visualizes quickly from the first step to the final step. Allow the user to change the visualizing speed.

The application should also highlight the corresponding source codes when visualizing operations.

The data visualization application also provides various customization options to help users tailor the visualization to their specific needs. For example, users can customize the color, size, and style of the visualization to suit their preferences.

Overall, the data visualization application provides a powerful tool for the display and understanding data stored in basic data structures.

2 Notes

2.1 Must use source code version control (git).

2.2 Must have C++.

2.3 Must have struct. Each struct has 2 files (.h, .cpp).

2.4 The source code must have at least 3 files:

- A header file (.h): struct definition, function prototypes/definition.
- A source file (.cpp): function implementation.
- Another source file (.cpp): main() function.

2.5 Your submission: **StudentID.zip**

- Report
  - Functional DS (functions that have been done / are doing / have not been done)
  - Program structure: structures, functions, ...
  - User's manual
  - Commit list copied from git history
  - Link of git
  - Link of demo video (YouTube).
- Source: contains all source code, resources (images, sounds, ...), and data.
- Release: contains executable files (exe), resources and data.