|  |
| --- |
| **Department of Computer and Software Engineering – ITU** |
| **SE200T: Data Structures & Algorithms** |

|  |  |
| --- | --- |
| **Course Instructor: Usama Bin Shakeel** | **Dated: 21st Aug 2024** |
| **Teaching Assistant: Zainab, Sadia & Ryan** | **Semester: Fall 2024** |
| **Session: 2024-2028** | **Batch: BSSE2023B** |

# **Assignment 1. Array Class using Dynamic Memory Allocation**

|  |  |  |
| --- | --- | --- |
| **Name** | **Roll number** | **Obtained Marks/35** |
| Muhammad Mukarram Raza | BSSE-23029-B |  |

Checked on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Submission:**

• Email instructor or TA if there are any questions. You cannot look at others’ solutions or use others’ solutions, however, you can discuss it with each other. Plagiarism will be dealt with according to the course policy.

• Submission after due time will not be accepted.

**In this assignment you have to do following tasks:**

**Task 1:** Ensure that you have installed all three softwares in your personal computer (Github, Cygwin & CLion). Now, accept the assignment posted in the classroom (e.g Google, LMS etc) and after accepting, clone the repository to your computer. Make sure you have logged into the github app with your account.

**Task 2:** Open Cygwin app, Move to your code directory with following command “cd <path\_of\_folder>”, <path\_of\_folder> can be automatically populated by dragging the folder and dropping it to the cygwin window.

Run the code through Cygwin, use command “make run”, to get the output of the code

**Task 3:** Solve the given problems, write code using **CLion** or any other IDE.

**Task 4:** Keep your code in the respective git cloned folder.

**Task 5:** Commit and Push the changes through the Github App

**Task 5**: Write the code in separate files **(as instructed**). Ensure that file names are in lowercase (e,g **main.cpp**).

**Task 6:** Run ‘**make run**’ to run C++ code

**Task 7:** Run ‘**make test**’ to test the C++ code

Write code in functions, after completing each part, verify through running code using **“make run”** on Cygwin. Make sure to test the code using **“make test”.**

***Problem Statement: Array Operations***

*Implement the* **MyArray***class with dynamic array handling, this task is in continuation to your Lab1 task. Ensure the array resizes when needed (when inserting beyond capacity). Also implement error handling for invalid indices. Use the menu-driven approach to interact with the user and call the appropriate methods.* ***This implementation will also be using templates.***

***Class Attributes:***

* *data*
* *size*
* *capacity*

***Class Methods:***

* *insertAtIndex*
* *removeAtIndex*
* *sortArray*
* *display*
* *uniqueElements*
* *Menu*

***Task 1: Insert At Index***

*Inserts an element at a specific index in the array. If the index is out of bounds, display an error message. Resize the array if necessary.*

***Example:***

***Input:*** *Enter element to insert: 4*

***Input:*** *Enter index to insert at: 2*

***Output:*** *Array [10, 5, 4, 1, 7, 9, 0, 2]*

***Task 2: Remove At Index:***

*Removes an element from a specific index in the array. If the index is out of bounds, display an error message. Adjust the array size accordingly.*

***Example:***

***Input:*** *Enter index to remove from: 2*

***Output:*** *Array [10, 5, 1, 7, 9, 0, 2]*

***Task 3: sortArray:***

*Sorts the array in ascending order or descending order based on boolean parameter if true - ascending else descending*

***Example:***

***Output:*** *Array [0, 1, 2, 5, 7, 9, 10]*

***Task 4: display:***

*Displays the current elements of the array.*

***Task 5: uniqueElements:***

*This function will ensure that any repeated element in the array would be removed.*

***Task 6: menu:***

*Displays a menu with the following options:*

***Enter 1*** *to insert at a given index*

***Enter 2*** *to remove from a given index*

***Enter 3*** *to sort the array*

***Enter 4*** *to display the array*

***Enter 5*** *to remove duplicates*

***Enter 0*** *to Exit*

**Assessment Rubric for Assignment**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Performance metric** | **CLO** | **Able to complete the task over 80% (4-5)** | **Able to complete the task 50-80% (2-3)** | **Able to complete the task below 50% (0-1)** | **Marks** |
| 1. Realization of experiment | 3 | Executes without errors excellent user prompts, good use of symbols, spacing in output. The testing has been completed. | Executes without errors, user prompts are understandable, minimum use of symbols or spacing in output. Some testing has been completed. | Does not execute due to syntax errors, runtime errors, user prompts are misleading or non- existent. No testing has been completed. |  |
| 1. Conducting experiment | 2 | Able to make changes and answer all questions. | Partially able to make changes and few incorrect answers. | Unable to make changes and answer all questions. |  |
| 1. Computer use | 4 | Document submission timely. | Document submission late. | Document submission not done. |  |
| 1. Teamwork | 4 | Actively engages and cooperates with other group member(s) in an effective manner. | Cooperates with other group member(s) in a reasonable manner but conduct can be improved. | Distracts or discourages other group members from conducting the experiment |  |
| 1. Laboratory safety and disciplinary rules | 2 | Code comments are added and do help the reader to understand the code. | Code comments are added and do not help the reader to understand the code. | Code comments are not added. |  |
| 1. Data collection | 2 | Excellent use of white space, creatively organized work, excellent use of variables and constants, correct identifiers for constants, No line-wrap. | Includes name, and assignment, white space makes the program fairly easy to read. Title, organized work, good use of variables. | Poor use of white space (indentation, blank lines) making code hard to read, disorganized and messy. |  |
| 1. Data analysis | 3 | Solution is efficient, easy to understand, and maintain. | A logical solution that is easy to follow but it is not the most efficient. | A difficult and inefficient solution. |  |
| **Total (out of 35):** | | | | |  |