

**Recep Tayyip Erdogan University**

**Faculty of Engineering and Architecture**

**Computer Engineering**

CE205- Data Structure

**Homework-1 (Week-2)**

**Fall Semester, 2021-2022**

|  |  |
| --- | --- |
| Instructor | Asst. Prof. Dr. Uğur CORUH |
| Contact Information | [ugur.coruh@erdogan.edu.tr](mailto:ugur.coruh@erdogan.edu.tr) |
| Google Classroom Code | **y3qryym** |
| Publish Date | **15.10.2021** |
| Due Date | **22.10.2021 23:59** |

**Complete the following homework requirements, prepare them in the format given in the description below until the deadline and time, and upload them to the classroom's related assignment.**

**Grades:**

|  |  |
| --- | --- |
| Problem-1 | 90 points |
| Problem-2 | 10 points |
| **Total** | **100** points |

***Development Environment***

In this problem, you will develop a C++ static library, C# class library, test and console application for performance measurements and reporting.

You will add your application to GitHub (you can use the GitHub desktop app for this). I have prepared a template code for you, please use it <https://github.com/ucoruh/CE205LinkedListTemplate/tree/master>

You should provide working code without any bugs.

Be careful about project types. The project type will be the library. If you use the template, you will have already generated projects for you

Do not forget to provide README.md files with clear descriptions

Clearly explain your codes with comments and generate Doxygen documentation.

***Code Sharing***

You will create your project with git version management support, and you will add a gitignore file for your project to decline unnecessary files. You will push your lite code to your GitHub account, and you will share this link in the classroom. Also, you will submit your compressed code to the classroom with the report. The report is necessary for formal validations.

Be careful to submit codes without unnecessary files such as intermediate build files, executable outputs, etc. these files are ignored via virus protection tools.

***Grading Criteria***

1. *Github Code Sharing*
2. *Classroom Code and Report Sharing*
3. *Code Comments and Doxygen Code Generation*
4. *Project Types*
5. *Code or Project Bugs (Not Running or Complaining)*
6. *Unit Tests and Assertions*
7. *Algorithm Solution Methods and Explanations*

**Problem-1: (90 points)**

In this problem, you will implement the following linked list data structures and their operations in C++ and C#. Also, you will provide the test code of their operations. In the console application, you will measure linked list insertion, deletion, and list (show) operations time complexity (time measurement), space complexity (memory measurement), and you will compare and report your results in the console application. You can use 1000 item insertion, deletion, and list for each linked list.

Each linked list will have the following functions

1. Insert
2. Delete
3. List
4. Search
5. AllocatedMemory
6. AverageInsertTime
7. AverageDeleteTime

You will implement the following linked list and functions below for each linked list

1. Single Linked List (15 points)
   1. Insert
   2. Delete
   3. List
   4. Search
   5. Length
   6. AllocatedMemory
   7. AverageInsertTime
   8. AverageDeleteTime
   9. AverageListTime
   10. AverageSearchTime
   11. ResetAverageInsertTime
   12. ResetAverageDeleteTime
   13. ResetAverageListTime
   14. ResetAverageSearchTime
2. Circular Linked List (15 points)
   1. Insert
   2. Delete
   3. List
   4. Search
   5. Length
   6. AllocatedMemory
   7. AverageInsertTime
   8. AverageDeleteTime
   9. AverageListTime
   10. AverageSearchTime
   11. ResetAverageInsertTime
   12. ResetAverageDeleteTime
   13. ResetAverageListTime
   14. ResetAverageSearchTime
3. Double Linked List (15 points)
   1. Insert
   2. Delete
   3. List
   4. Search
   5. Length
   6. AllocatedMemory
   7. AverageInsertTime
   8. AverageDeleteTime
   9. AverageListTime
   10. AverageSearchTime
   11. ResetAverageInsertTime
   12. ResetAverageDeleteTime
   13. ResetAverageListTime
   14. ResetAverageSearchTime
4. XOR Linked List (15 points)
   1. Insert
   2. Delete
   3. List
   4. Search
   5. Length
   6. AllocatedMemory
   7. AverageInsertTime
   8. AverageDeleteTime
   9. AverageListTime
   10. AverageSearchTime
   11. ResetAverageInsertTime
   12. ResetAverageDeleteTime
   13. ResetAverageListTime
   14. ResetAverageSearchTime
5. Skip List (15 points)
   1. Insert
   2. Delete
   3. List
   4. Search
   5. Length
   6. AllocatedMemory
   7. AverageInsertTime
   8. AverageDeleteTime
   9. AverageListTime
   10. AverageSearchTime
   11. ResetAverageInsertTime
   12. ResetAverageDeleteTime
   13. ResetAverageListTime
   14. ResetAverageSearchTime

You will create unit test for each operation for each linked list

1. Insert
2. Delete
3. List
4. Search
5. Length
6. AllocatedMemory
7. AverageInsertTime
8. AverageDeleteTime
9. AverageListTime
10. AverageSearchTime
11. ResetAverageInsertTime
12. ResetAverageDeleteTime
13. ResetAverageListTime
14. ResetAverageSearchTime

In the console application, you will compare AllocatedMemory and Average Insert, Delete, List, and Search Times by using these functions.

**Problem-2: (10 points)**

Implement Strand Sort in C++ and create its unit test for testing

Good Luck!