School of Computing, Electrical and Applied Technology

ISCG6426
Data Structures & Algorithms

Assignment

Semester 1, 2021

Due Date: June 13th 2021 11:59PM

Total Marks: 40
Course Weighting: 40%

Learning outcomes covered in this assignment

1. Apply object-oriented design and implementation techniques.
2. Interpret the trade-offs and issues involved in the design, implementation, and application of various data structures with respect to a given problem.
3. Explain the purpose and answer questions about data structures and design patterns that illustrate strengths and weaknesses with respect to resource consumption.
4. Assess the impact of data structures on algorithms.
5. Analyse the scalability of data structures and algorithms in terms of both space and time complexity.

Assignment instructions:

- You will work individually on this assignment.
- Select a data structure or algorithm from the Appendix. Implement and create an interactive game or visualization which uses it in an educational manner. Once you've selected one, discuss it with your lecturer and get their approval.

Assignment Submission:

- Create a GitHub repository for the assignment and share the repository information with your lecturer. On the assignment due date the lecturer will clone this repository as your submission.
- If you forget to push your most recent changes by the due date, then the lecturer will use whichever version available on GitHub when the deadline is reached for marking.

Task 1: Implement a data structure / algorithm [17 Marks]

Select a data structure or algorithm from the appendix and get the lecturer's approval to work on it. Ask your lecturer for functionality you will be required to implement.

Correctly implement the chosen data structure. The implementation must use randomly generated data.

Task 2: Code an interactive visualization [15 Marks]

Choose an algorithm or data structure from the appendix and get the lecturer's approval to work on it.
Using the *SFML.NET* library (NuGet package or https://www.sfml-dev.org/download/sfml.net/) or another approved library/framework, create an interactive game or visualization demonstrating how the chosen data structure or algorithm works.

Work closely with your lecturer during this phase to ensure that your implementation is accurate. Failing to do so may result in loss of marks.

Your implementation is required to accurately represent the chosen data structure/algorithm in an educational manner.

It should support both keypresses and mouse input. It must implement a draw() and update() method in the visualization.

Task 3: Analysis & Commentary [8 Marks]

Create a file named README.md in your project directory. In this file, write the following:

- A paragraph documenting issues you encountered in the design or implementation of your chosen data structure/algorithm.

- Briefly explain the strengths and weaknesses of your data structure or algorithm with respect to resource consumption. Under what conditions does it perform the best or worst?

- List a real-world application of your chosen data structure or algorithm.

- A sentence or two on the asymptotic worst-case time and space complexity of your chosen data structure/algorithm.

Marking Schedule

Task	Marking Criteria	Marks	Given	Comments
1	Correct DS/A implementation	15		
	Uses random data	2		
2	Accurate visualization	7		
	Handles mouse interaction	2		
	Handles keypress interaction	2		
	Has module documentation	2		
	Implements draw() method	1		
	Implements update() method	1		
3	Has implementation notes	2		
	Discusses strengths/weaknesses	2		
	Lists an application	2		
	Time/space complexity analysis	2		
Total Marks		**40**		

Late Submission of Assignments:

Assignments submitted after the due date and time without having received an extension through Affected Performance Consideration (APC) will be penalised according to the following:

- 10% of marks deducted if submitted within 24hrs of the deadline,

- 20% of marks deducted if submitted after 24hrs and up to 48hrs of the deadline,
- 30% of marks deducted if submitted after 48hrs and up to 72hrs of the deadline,
- No grade will be given for an assignment that is submitted more than 72hrs after the deadline.

Assistance to other Students:

Students themselves can be an excellent resource to assist the learning of fellow students, but there are issues that arise in assessments that relate to the type and amount of assistance given by students to other students. It is important to recognise what types of assistance are beneficial to another's learning and also what types of assistance are unacceptable in an assessment.

Beneficial Assistance:

- Study Groups
- Discussion
- Sharing Reading Material
- Reading the available online and library resources

Unacceptable Assistance:

- Working together on one copy of the assessment and submitting it as own work .
- Giving another student your work.
- Copying someone else's work, this includes work done by someone not on the course.
- Changing or correcting another student's work.
- Copying from books, the Internet etc. and submitting it as own work; anything taken directly from another source must be acknowledged correctly; show the source alongside the quotation.
- Don't copy code from a website or video tutorial and pretend you made it or slightly change it. This will be an instant fail (0%).

Appendix 1

Choose one of the following to implement for your assignment.
Contact your lecturer for approval. As this is an open-source assignment, each student needs to work on something different.

1. Double-Ended Queue (deque)
2. Priority Queue
3. Sorting
    a. BubbleSort
    b. Insertion Sort
    c. Selection Sort
    d. QuickSort
    e. MergeSort
    f. HeapSort
4. Linked-List

   a. Singly
   b. Doubly
   c. Circular
5. Searching
   a. Binary search
   b. Depth-first-search {Pre, In, Post}-order
   c. Breadth-first search
6. Trees
   a. Binary search tree
   b. AVL Tree
   c. Red-Black Tree
   d. B-tree
   e. Merkle-tree
   f. Trie
   g. Quadtree
7. Heap
   a. Min
   b. Max
   c. Fibonacci
   d. Binary
8. Hashing
   a. Chaining
   b. Open addressing
   c. SHA256
9. Graph shortest-path
   a. Dijkstra's
   b. Bellman-Ford
   c. A* search
   d. Floyd-Warshall
10. Network maximum flow
   a. Ford-Fulkerson
   b. Edmonds-Karp
11. Graph cycle detection (Floyd's)
12. Directed Acyclic Graph (DAG)
13. Traveling Salesman Problem (TSP)
14. Minimum spanning tree (Boruvka's, Prim's, Kruskal's)

If you choose to do something not on this list you need to get approval from your lecturer. Communicate with your lecturer during the course to ensure your application meets the required guidelines and marking schedule.