**School of Computing and Information Technology**

Data Structures and Algorithms ISCG6426

**Percentage Mark:** 40% of final course marks

# Mini-Project

## 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.

## Instructions

- You will work individually on this assignment.
- Select a data structure or an algorithm from the appendix below. Implement and create an interactive game or visualization which uses the chosen data structure's functionality/working principle.

**Project Information:**

- SFML.NET starter information. Use the NuGet package instead of manual install. https://www.sfml-dev.org/download/sfml.net/
- NuGet Package information - Best to add package directly from Visual Studio. https://www.nuget.org/packages/SFML.Net/
- Official SFML tutorials, written in C++ but functionality is the same for C# https://www.sfml-dev.org/tutorials/2.5/
- Official C# Documentation and examples (download the repo): https://github.com/SFML/SFML.Net
- SFML.NET Hello World! Example code: https://www.gamedev.net/blogs/entry/2269532-sfml-c-net-core-rotating-hello-world/

## Delivery

1. Create a .NET framework Console solution using any version of Visual Studio. (VS 2022 recommended)
2. Add your name and student no. as comments at the top of Program.cs file
3. Save all files in the project and add the whole project to a zip file.
4. Name the file with the following format: studentid_firstname_lastname_Project.zip
5. Upload the zip file to the submission link on Moodle.

## Task 1: Implement a data structure / algorithm [15 Marks]

- Select a data structure or algorithm from the appendix
- Correctly implement the chosen data structure. The implementation must use randomly generated data.

## Task 2: Code an interactive visualization [17 Marks]

- Choose an algorithm or data structure from the appendix
- Select a programming framework or library that supports interactive visualizations. For .NET projects, options like SFML.NET are recommended. Alternatively, Windows Form or Unity can be an option.
- Your implementation is required to accurately represent the chosen data structure/algorithm in an educational manner.
- It should support both **keypresses** and **mouse input**.

## Task 3: Analysis & Commentary [8 Marks]

Create a file named README.txt 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<br><br>Uses random data | 13<br><br>2 | | |
| 2 | Programming framework<br><br>Accurate visualization<br><br>Handles mouse interaction<br><br>Handles keypress interaction<br><br>Has module documentation | 5<br><br>5<br><br>2<br><br>2<br><br>3 | | |
| 3 | Has implementation notes<br><br>Discusses: strengths/weaknesses<br><br>Lists real-world applications<br><br>Time/space complexity analysis | 2<br><br>2<br><br>2<br><br>2 | | |
| **Total Marks** | | **40** | | |

## Late Submission of Assignments:

Assignments submitted after the due date and time without having received an extension through Special Assessment Circumstances (SAC) 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 later than 72hrs after the deadline.

## Special Assessment Circumstances:

A student, who due to circumstances beyond his or her control, misses a test, final exam or an assignment deadline or considers his or her performance in a test, final exam or an assignment to have been adversely affected, should complete the Special Assessment Circumstances (SAC) form available from the Student Central.

When requesting an SAC for an assignment, the SAC must be submitted (along with work completed to date) within the time frame of the extension requested; i.e. if the Doctor's certificate is for one (1) day, then the SAC and work completed must be submitted within one (1) day.

## Assistance to other Students:

Group work and collaboration are great learning methods, but assessments evaluate individual achievement of course outcomes. Therefore, while knowledge building and preparations for examinations and assessments through collaboration are encouraged, it is important to be aware 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.

## Appendix 1

**Choose one of the following to implement for your assignment.**

1. Stack
2. Queue
3. Sorting
    - **a.** Bubble Sort **b**. Insertion Sort
    - **c**. Selection Sort **d**. QuickSort
4. Linked-List
    - **a.** Singly
    - **b.** Doubly
    - **c.** Circular
5. Searching
    - **a.** Binary search
6. Trees
    - **a.** Binary search tree – Depth-first search {Pre, In, Post}-order and/or Breadth-first search
7. Heap
8. Graph
    - **a.** Dijkstra's algorithm **b**. A* search algorithm
    - c. Kruskal's algorithm