Module:	Algorithms & Data Structures
CA:	CA2 - Project
Value:	20% of Module
Due Date:	See Moodle

Objectives:

- To practice the design and implementation of a C++-based non-linear data structure.
- To gain experience with the use of C++ unit testing.
- To demonstrate the use of advanced C++ concepts (templates, functors, and predicates).

Core Requirements:

Stage 1:

Implement a templated Tree Map data structure in C++. The java TreeMap class uses a special variation on a binary tree called a Red-Black Tree, however for this assignment you may use the Binary Search Tree created in class. This is available on GitHub via the Moodle link.

Your implementation must contain the following methods:

void clear()	Removes all entries from the Map
bool containsKey(K key)	Returns true if this map contains a mapping for the specified key.
V& get(K key)	Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
BinaryTree<K> keySet()	Returns a Set view of the keys contained in this map.
void put(K key, V value)	Associates the specified value with the specified key in this map
int size()	Returns the number of key-value mappings in this map
Bool removeKey(K key)	Removes the item denoted by the given key.
V& operator[K key]	Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.

Part 2:

Using only your own implementation of the TreeMap and BST tree (SET) create an application that reads in a text file and organises all of the unique words contained within the file by their first letter.

When the user runs the programme, they should be able to view

- The list of letters for which there were words in the file.
- All of the words associated with a given letter.

Modify the print methods of the Binary Tree Starter Code for display

Stage 3:

Create a CSV file containing at least 100 rows of data on any topic you wish. The topic you choose should have a least 5 fields of at least 2 different data types. You may use online data generation tools to generate your data.

Stage 4:

Write a C++ application which will allow a user to view and search the data generated in stage 3. Your application should allow the user to perform each of the following:

1. Create an index of the data based on any, user-specified field within the data. This should also show the user the unique values within that field and the number of rows containing that value.

2. View a subset of the data based on a user-given value for any one of the fields.

All data should be clearly and neatly presented. Efficient memory management must be applied.

Repository & Design Requirements:

You are required to create a GitHub student repository for your project. This project must have the following properties:

- Name format (i.e. "ADS_2024_CA2_studentinitials1_studentinitials2")
- The project repository must be shared (read access) with your lecturer(s) on the module.

You will be awarded a maximum of **5 marks** for the regularity and commenting (clarity, use of illustrative emojis) of your project commits.

Note: Your project will NOT be graded in the absence of this requirement.

Submission Requirements:

1) **All Source code must be submitted via Moodle. This must include a README containing a single link to your GitHub repo.**
2) The assignment **must** be entirely the work of each student. Students are **not** permitted to share any pseudocode or source code from their solution with any other individual or group in the class. Students may **not** distribute the source code of their solution to any student outside of their group in **any** format (i.e. electronic, verbal, or hardcopy transmission).
3) Plagiarised assignments **will** receive a mark of zero. This also applies to the individual allowing their work to be plagiarised.
4) **The use of generative AI tools is strictly forbidden.**
5) Any plagiarism **will** be reported to the Head of Department and a report will be added to your permanent academic record.
6) Late assignments will **only** be accepted if accompanied by the appropriate medical note. This documentation **must** be received within 10 working days of the project deadline. The penalty for late submission is as follows:
    - Marked out of 80% if up to 24 hours late.
    - Marked out of 60% if 24-48 hours late.
    - Marked out of 40% if 48-72 hours late.
    - Marked out of 20% if 72-96 hours late.
    - Marked out of 0%, if over 96 hours late.
7) Each student **must** complete and sign a single assignment cover sheet. Please submit this with your moodle submission.
8) An **interview** may take place to assess the student's work on this project. This interview will be scheduled within **10 working days** of the project deadline. **Failure** to attend this interview will result in a grade of **0%** in this component. Your final grade is **directly related** to your performance in this interview. Each student will be asked several **questions** related to the **concepts** covered in, and their **contribution** to, the project. Failure to adequately **answer** a question will result in a **penalty** applied to the final mark for that student.