

National College of Ireland

**Higher Diploma in Science in Computing**

**HDSDEV\_SEP, HDCSDEV\_INT, HDSDEV\_JANOLY1\_O**

**Release Date-Time: May 15th 09:00**

**Submission Date-Time: May 18th 23:55**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Algorithms and Advanced Programming**

**Dr. William Clifford**

**Jack O’Neill**

**Total marks:** 100 marks with 50% contribution to your final result.

**Learning outcomes:**

LO-1: Use iterative and recursive techniques to design and implement sorting and searching algorithms

LO-2: Conduct in depth algorithm analysis in terms of time complexity and present the results of analysis.

LO-3: Evaluate algorithms, identify an algorithm from a range of possible options, and implement the algorithm to solve computational problems in particular contexts.

LO-4: Identify and apply best practices including exception handling and design patterns in the implementation of software solutions to solve real-world problems.

**Instructions**

1. This is an open book TABA.
2. You are allowed to use your class notes. If the lecture notes are used as a resource for this assignment, do not copy them directly but rather paraphrase them (write in your own words).
3. **You are not allowed to discuss your solution with other students during the examination or seek outside help from another person. If it is found that a student has discussed his/ her solution with other people not approved by the college, the case will be referred to the disciplinary committee for further actions.**
4. You need to submit the solution of this assignment on your Moodle page at the TERMINAL ASSIGNMENT Link. You should submit the PDF (or DOC) file by the end of your exam time as well as any other files (.java files, not .class files) contained within a zip folder.
5. You can also draw a diagram/ figure on a piece of paper, take a picture with your mobile phone camera then you submit the photo on Moodle. Please ensure that you write below your drawing the question number the drawing corresponds to.
6. Use a single column layout document.
7. Font size for the body of the text should be 12-point Times New Roman/ Arial.
8. Include student name, student ID and course name at the top of the first page.
9. The question number being addressed must be clearly indicated in the document.
10. Attempt **all** Questions, they all add to a total of 100 marks.
11. This is an individual personal assignment, co-operation or collaboration among students is strictly not allowed and may result in disqualification. Students may be asked to outline/explain in person the reason for any approach taken or solution provided.
12. Note: do not use any special functionalities in java not covered in class (specifically lambda functions). If you are unsure of what is or is not allowed, then please ask your lecturer or post on teams.

**Question 1. (35 marks)**

Using a data file you have **not** previously worked on for the CA (either staff.csv, stock.csv, or films.csv), complete the following tasks. A java application should be written to complete the following tasks (these can be separate applications or just a single one).

1. Read a CSV file into memory using the Buffered reader (note the original way to read these files in the provided code for the CA used the Scanner class instead of buffered reader).

**(5 marks)**

1. Write a recursive method to compute the sum of every **even-indexed** row of either the wage, weight, or length column, depending on whether you have read the staff.csv, stock.csv, or films.csv respectively.

**(2.5 marks)**

1. Write an iterative method to find the minimum value in the respective column from part b). Consider only the odd indexes.

**(2.5 marks)**

1. Write a sorting method for all possible columns in the data file. Your application should allow the user the option to sort the array using any column (This can be done through either the Scanner class or using the swing windowing library).

**(12.5 marks)**

1. Create a multi-threaded solution, where each thread sorts a copy of the data in a different order depending on the column being sorted, i.e., thread 1: sorts based on column 1, thread 2: sorts based on column 2, and so on. A copy of each order should be saved to a different file. For example, if sorting the films file based on column 1 save it as sortedFilms\_C1.csv (it doesn’t have to be csv, it can be whatever file format you choose).

**(12.5 marks)**

**Question 2. (20 marks)**

Consider what the Big O run-time complexity would be for an unsorted array, sorted array, and linked list for each of the following tasks (explain your reasoning):

1. Retrieving the first item from this data structure.
2. Calculating the sum of every second indexed item.
3. Retrieving the index of a specific value from this data structure.
4. Calculating the product of all values less than some specified value, x.
5. Setting the final indexed value to zero or null in this data structure.

**(12.5 marks)**

1. How could you test the run-time of these algorithms and their respective data structures in a real system on your own personal machine? You can provide examples as well as a description if this helps.

**(7.5 marks)**

**Question 3. (20 marks)**

1. Discuss **in your own words** the following with respect to multi-threading. Locks, deadlock, starvation, and synchronization. Provide example of when each may be appropriate or can occur (i.e., when to use locks and synchronization, and what causes deadlocks and starvation).

**(12.5 marks)**

1. What are possible solutions to deadlock and starvation?

**(7.5 marks)**

**Question 4. (25 marks)**

1. Write a Java program that accepts a new record (with all the six fields) and adds it at the end of the record array from Question 1. with a new consecutive id number (like emp\_no, film\_id, or stock\_no after the last item). **(12.5 Marks)**

**Example Input for staff**:(“Joe”, “Bloggs”, “Male”, “IT”, 10000, 50)

**Example Input for film**: (“comedy”, “David Frankel”, “Marley and me”, 2.3, 2.5)

**Example Input for stock**: (12.2, 120.3, “furniture”, 100.0, ”L-Sofa”)

1. Write a Java Exception that handles special cases and communicates to users to correct the cases. A typical special case is that the first\_name field cannot be empty or cannot contain digits only. The exception should generate “Employee first\_name cannot be empty. It cannot have only digits! Please correct this” message. **(12.5 Marks)**

**Staff example:**

**Example Invalid input**: first\_name = 2343322

**Example Invalid input:** first\_name =””

**Film example:**

**Example Invalid input**: length = “really long”

**Example Invalid input:** length = -5

**Stock example:**

**Example Invalid input**: product\_type = “anything other than the 3 products mentioned”

**Example Invalid input:** product\_type = 1995-12-05