Faculty of Engineering, Environment and Computing

210CT Programming, Algorithms and Data Structures

Assignment Brief 2017/18

Module Title Programming, Algorithms and Data Structures	Ind/Group	Cohort (Sept/Jan/May) September	Module Code 210CT
Coursework Title (e.g. CWK1) Programming Coursework			Hand out date: 12.10.2017
Lecturer Dr. Diana Hintea			Due date: 01.12.2017
Estimated Time (hrs): 40 Word Limit*: Not applicable	Coursework type: Individual written report assessed via VIVA		% of Module Mark 50%

Submission arrangement online via CUMoodle: Deadline 6pm on the 1st of December 2017 – TurnitIn link
File types and method of recording: Word or PDF
Mark and Feedback date: VIVAs finish on the 15th of December, grades and feedback released 8.01.2018
Mark and Feedback method: Individual feedback sheet

Module Learning Outcomes Assessed:

1. Understand and select appropriate algorithms for solving a range of problems.
2. Design and implement algorithms and data structures for novel problems.
3. Reason about the complexity and efficiency of algorithms.
4. Demonstrate the use of object oriented programming features.

Task and Mark distribution (mark distribution is presented separately in the VIVA marking scheme):

Basic Exercises

Week 1

1. Write a method that combines two strings, by taking one character from the first string, then one from the second string and so on. Once one string has no characters left it should carry on with the other string. For example, for strings "day" and "time" should give the result "dtaiyme".
2. Check if a 3 digit number is an Armstrong number. An Armstrong number of three digits is an integer such that the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since 3**3 + 7**3 + 1**3 = 371. Input: a number, such as 371 Output: “Yes” if the number is an Armstrong number, “No” if the number is not an Armstrong

number.

Week 2

3. Read the degree of two polynomials and their coefficients, all integers, from the standard input. The polynomial is of the form $(x) = pn * x\,n + \cdots + p1 * x\,1 + p$, where $p0 \neq 0$.

   a) Write the pseudocode for adding two polynomials.
   b) Given the following pseudocode, first understand how it works, then implement the code for multiplying the two polynomials in the programming language of your choice.
   c) Write the pseudocode for computing the derivative for each of the two polynomials.
   d) What is the time complexity (expressed using Big-O notation) for each of the above operations (a, b and c)?

4. Write the pseudocode and code for a function that determines whether an array is a palindrome (reads the same in both directions). What is the time complexity (expressed using BigO notation)?

Week 3

5. Write a program that reads n words from the standard input, separated by spaces and prints them mirrored (the mirroring function should be implemented recursively). What is the time complexity of the algorithm? Use the BigO notation to express it.
6. Write a recursive version of linear search on an array of integers. What is the time complexity of the algorithm? Use the BigO notation to express it.

Week 4

7. Manually arrange the sequence [2 7 9 4 1 5 3 6 0 8] in ascending order using insertion sort, bubble sort and selection sort, showing at each step the new configuration of the sequence. How many comparisons and how many element moves were used by each method? Which is the best performing method for sorting this array of integers? Which would be the worst arrangement of this sequence?
8. Use binary search in implementing a guessing game. One thinks of a number between 1 and 20000, the program attempts to guess the number and feedback is given whether my number is higher or lower. The program then makes a new guess and so on until it guesses the right number.

Week 5

9. Based on the Python code or the C++ code provided in class as a starting point, implement a function that deletes duplicate value nodes from the list.

Week 6

10. Implement a function that deletes a node in a binary search tree in a language of your choice.

Week 7

11. Implement an unweighted, undirected graph structure in the programming language of your choice. Implement the BFS or DFS traversal for this graph.
12. For the same graph, implement a function named havePath(v, w), where $v \in V$ and $w \in V$, to check if there is a path between the two given nodes.

Week 8

13. Adapt the previous graph structure to support weighted connections and implement Dijkstra's algorithm.

Advanced Exercises

Week 1

1. The factorial function, n! is defined thus for n a non-negative integer: 0! = 1 n! = n * (n-1)! (n > 0). We say that a divides b if there exists an integer k such that k×a = b. The input to your program consists of several lines, each containing two non-negative integers, n and m, both less than 231. For each input line, output a line stating whether or not m divides n!, in the format shown below.
2. A lorry can carry at most n kilograms. The name of the materials, the amount of material in kilograms and the material price per kilo are known. Compute a load composition in such a way that the value (price) of the load is maximum.

Week 2

3. Write a program to solve the problem of the eight queens: i.e. find their placement on the chess board so that they have do not have the possibility to attack each other.

Week 3

4. Consider having n cubes, each being characterized by their edge length and their colour. Use the cubes to build a tower of maximum height, under the following conditions: a) Any two neighbouring cubes must be of different colours. b) The edge length of a cube is lower or equal than the edge length of the cube placed below it.

Week 4

5. Consider a nxm matrix, with elements decimal digits (natural numbers between 1 and 9), representing colours. A connected set associated to an element is the set of elements that may be reached from this element, by successive moves on a same row or column preserving the same colour. It is to determine the size and the colour of the biggest connected set. In case of multiple solutions, display them all.
6. Adapt the quick sort algorithm to find the mth smallest element out of a sequence of n integers.

Week 5

7. Find the smallest sum of n integer numbers taken from different diagonals parallel with the main diagonal, and including the main diagonal of a matrix A of size n × n.
8. Read a paragraph containing words from an input file. Then create a doubly-linked list containing the distinct words read, where the words of the same length are placed in the same list, in alphabetical order.

Week 6

9. Implement a system that maintains information for school classes. For each of the students in a class the following information is kept: a unique code, student name, birth date, home address, id of class he is currently in, date of enrolment, status (undergrad, graduate). For keeping track of the students, the school secretary would use a computer program based on a search tree data structure. Write a program to help the secretary, by implementing following the following operations:

   - Find a student by his/hers unique code, and support updating of the student info if found.
   - List students by class in lexicographic order of their names.
   - List all students in lexicographic order of their names.
   - List all graduated students.
   - List all undergrads by their class in lexicographic order of their names.
   - Delete a student given by its code.
   - Delete all graduates.

Week 7

10. For an undirected graph G = (V, E), where node names are positive integers, implement a graph construction method, and a function named isConnected to check whether or not the graph is connected.

11. Implement the structure for a weighted, directed graph G, where nodes consist of positive integers, then implement an algorithm to find the longest simple path.

Week 8

12. The input to this problem is a pair of strings of letters A = a1. . . am and B = b1. . . bn. The goal is to convert A into B as cheaply as possible. The rules are as follows:
    - For a cost of 3 you can delete any letter.
    - For a cost of 4 you can insert a letter in any position.
    - For a cost of 5 you can replace any letter by any other letter.

    For example, you can convert A = abcabc to B = abacab via the following sequence: abcabc at a cost of 5 can be converted to abaabc, which at cost of 3 can be converted to ababc, which at cost of 3 can be converted to abac, which at cost of 4 can be converted to abacb, which at cost of 4 can be converted to abacab. Thus the total cost for this conversion would be 19. This is almost surely not the cheapest possible conversion.

13. A connected undirected graph G = (V, E) is given by its cost matrix, where all costs are positive, and ≤ 65534. Determine a simple cycle passing through all nodes (a Hamiltonian cycle) of minimal cost.

Notes:

1. You are expected to use the CUHarvard referencing format. For support and advice on how this students can contact Centre for Academic Writing (CAW).
2. Please notify your registry course support team and module leader for disability support.
3. Any student requiring an extension or deferral should follow the university process as outlined here.
4. The University cannot take responsibility for any coursework lost or corrupted on disks, laptops or personal computer. Students should therefore regularly back-up any work and are advised to save it on the University system.
5. If there are technical or performance issues that prevent students submitting coursework through the online coursework submission system on the day of a coursework deadline, an appropriate extension to the coursework submission deadline will be agreed. This extension will normally be 24 hours or the next working day if the deadline falls on a Friday or over the weekend period. This will be communicated via email and as a CUMoodle announcement.
6. Please only submit text-based code for the official online submission (no screenshots) otherwise a mark of 0 will be awarded.
7. For each of the weeks, you can choose either the Basic tasks or the Advanced tasks, but you cannot combine them within the week itself.