**COMP1003**

**Algorithms, Data Structures and Mathematics**

**20 CREDIT MODULE**

**ASSESSMENT: 100% Coursework W1: 30% Set Exercises**

**W2: 70% Report**

**MODULE LEADER: Thomas Wennekers**

**MODULE AIMS**

* To give students a basic understanding of algorithmic design and practice
* To give students a basic understanding of mathematical principles underlying Computing

**ASSESSED LEARNING OUTCOMES (ALO):**

1. Explain the importance of algorithmic design in optimizing use of computing resources.
2. Review fundamental mathematical and logical principles that underlie modern computer science.
3. Identify suitable structures and algorithms to implement programming tasks.
4. Synthesize the solution to a real-world task as a combination of two or more standard algorithms.

**Overview**

This document contains all the necessary information pertaining to the assessment of *COMP1003 Algorithms, Data Structures and Mathematics*. The module is assessed via **100% coursework**, across two elements: *30% Set Exercises* and *70% Report*.

The sections that follow will detail the assessment tasks that are to be undertaken. The submission and expected feedback dates are presented in Table 1. All assessments are to be submitted electronically via the respective DLE module pages before the stated deadlines.

|  |  |  |
| --- | --- | --- |
|  | Submission Deadline | Feedback |
| Set Exercises (30%) | **20/03/2023 4pm** | Within 20 working days |
| Report (70%) | **18/05/2023 4pm** | Within 20 working days |

Table 1: Assessment Deadlines

All assessments will be introduced in class to provide further clarity over what is expected and how you can access support and formative feedback prior to submission. Whilst the assessment information is provided at the start of the module, it is not necessarily expected you will start this immediately – as you will often not have sufficient understanding of the topic. The module leader will provide guidance in this respect.

**Assessment 1: Set Exercises**

Answer the following questions. Hand-in a single PDF file that contains solutions to the questions in the same order they appear below. Solutions must show your working and have explanations. Highlight final results of calculations, for example, by underlining them. You can submit type-set answers (for example using LaTeX or similar) or hand-written answers. The latter may be scanned or photographed. If you submit hand-written answers, make sure they are tidy and well readable.

**Q1 Functions**

[Total 20 marks]

Consider the function f : **R** 🡪 **R** , f(x) = (x-2)2(x+2)

1. Turn f into a polynomial.

[2 marks]

1. Compute the derivative and second derivative of f.

[4 marks]

1. Compute the zero crossings, minima and maxima of f (if any).

[4 marks]

1. Compute the indefinite integral of f.

[3 marks]

1. Compute the integral of f between -4 and 3.

[2 marks]

1. Determine the limits of f(x) when x approaches +infinity and –infinity.

[2 marks]

1. Draw the graph of f indicating the zero crossings, minima, maxima and limits.

[3 marks]

**Q2 Probability**

[Total 20 marks]

A sample space consists of 9 elementary outcomes e1, e2, …, e9 whose probabilities are

P(e1) =P(e2) = 0.08, P(e3) = P(e4) = P(e5) = 0.1, P(e6) = P(e7) = 0.2, P(e8) = P(e9) = 0.07

Suppose A = {e1, e2, e5, e8}, B = {e2, e5, e8, e9}

1. Compute the Union, Intersection, Set difference A/B and B/A, and the Symmetric Difference between A and B.

[5 marks]

1. Calculate P(A), P(B) and P(A Ո B)

[3 marks]

1. Using the addition law of probability, calculate P(A U B)

[3 marks]

1. List the composition of the event A U B, and calculate P(A U B) by adding the probabilities of the elementary outcomes.

[3 marks]

1. Calculate P(B’) from P(B), also calculate P(B’) directly from the elementary outcomes of B’.

[3 marks]

1. Calculate the probability of the symmetric difference between A and B.

[3 marks]

**Q3 Order and Metric**

[Total 20 marks]

1. Prove that divisibility over the natural numbers induces a partial order.

[10 marks]

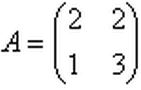
1. Prove that the lexicographic ordering on alphanumeric strings is an order or disprove this if it is not true.

[10 marks]

**Q4 Matrices and Vectors**

[Total 20 marks]

Consider the matrix M, vector v and scalar l below:



v = (1, -2)’ (where ‘ indicates matrix transposition)

l = -3

1. compute lv, lM, Mv, v'v, v'M, MM, vv', and det(M)

[8 marks]

1. Compute the eigenvalues and eigenvectors of M and normalise the eigenvectors. Check orthogonality using scalar products. What are the angles between pairs of eigenvectors?

[12 marks]

**Threshold Criteria**

To achieve a pass (40%+), 40% of the total marks for this CW element

To achieve a merit (60%+), 60% of the total marks for this CW element

To achieve a distinction (70%+), 70% of the total marks for this CW element

**Assessment 2: “Report”**

**The Set data type**

In computer science, a "set" is an abstract data type that stores certain data, without any particular order, and no repetition. The data may be Boolean values, numbers, characters, or even other data structures. In this task we simply assume they are positive integers: 1, 2, 3, 4, 5, etc. Therefore, {1, 4, 7} would be a valid set, but {-3, 0, 1} or { 1, 2, 2, 3 } would not be valid sets, because the first one contains 0 and a negative number, and the second one contains the 2 twice. The "empty set" is the set that does contain no element, denoted as {}.

a) Some simple operations on sets are:

clear\_set ( S ) : removes all elements from the set; returns nothing

is\_empty( S ) : checks whether the set S is empty and returns the Boolean value true or false, accordingly.

size( S ) : returns the number of elements in S, for example, size ( { 1, 2, 3 } ) = 3, or size ({}) = 0. In the second example "{}" is the set that contains no elements, the "empty set".

capacity( S ) : returns the maximum number of values that S can hold. This depends on the implementation.

is\_element\_of ( x, S ) : checks whether the value x is in the set S and returns true or false, accordingly. For example, 2 is in the set {1,2,3}, but 5 is not.

print( S ) : prints a list of the elements of S in some arbitrary order.

b) adding and removing are operations to construct sets from individual elements

add (S, x) : this function adds the element x to a set S, if it is not present already. It returns no value.

remove(S, x) : this function removes the element x from the set S. It returns no value.

c) Other operations on Sets are

copy\_set ( S ) : returns a copy of a set S

is\_subset( S, T) : tests whether the set S is a subset of set T, i.e. if all elements of S are also in T, and returns a Boolean value, accordingly.

intersection( S, T) : returns the "intersection" of sets S and T. These are the elements that occur in both, S and T, i.e., intersection ( S, T ) = {2} for S = {1,2} and T = {2, 3, 4}.

symmetric\_difference ( S, T ) : returns the "symmetric difference" of S and T, i.e., the union of all values in S that are not in T and of all values in T that are not in S.

**Task (Total: 90 marks)**: Implement the set-operations listed above in C#. Make your implementation as computationally efficient as possible!

Marks: a) 20 marks

b) 20 marks

c) 20 marks

An additional 10 marks are allocated for good and efficient use of data structures and algorithms.

An additional 10 marks are allocated for some reasonable error checking and/or handling of special cases, as well as some reasonable testing of your functions in a Main-program, or separate test-routines.

An additional 10 marks are allocated for good programming style, like proper layout, identifier names, commenting, modularisation, and the like.

Do not use object-oriented programming other than perhaps method-free classes for the data containers. Do not use the C# Collection library. There are numerous examples in the lectures and labs showing what you can use.

**Format of coursework to be handed in:**

Hand in a single file for part B. This can be a zip-file, if more than one document needs to be included.

For tasks requiring text or figures use .doc, .docx or .pdf document formats. If they are clear and legible, figures can be drawn by hand and scanned, or using any graphical tool available. They should be in PDF format or directly included in a document file.

For tasks requiring coding provide **a single C#** file containing your program, *suitably laid out and commented*. Do not submit a full C#-solution (compare with the examples in the lectures and exercises). The code should run without modification, otherwise marks will be reduced.

Do not code graphical user interfaces -- use a console application. GUIs will not attract more marks, but very likely make the code less readable. This would reduce marks because readable code follows good programming practice.

Do not use programming constructs on top of the basic C core functionality of C#. Keep your code simple and clear. The module’s DLE page contains an introduction into the C-core of C#. There are also plenty examples in the course material. If in doubt, ask!

**Threshold Criteria:**

To achieve a pass (40%+), 40% of the total marks for this CW element

To achieve a merit (60%+), 60% of the total marks for this CW element

To achieve a distinction (70%+), 70% of the total marks for this CW element

**General Guidance**

**Extenuating Circumstances**

There may be a time during this module where you experience a serious situation which has a significant impact on your ability to complete the assessments. The definition of these can be found in the University Policy on Extenuating Circumstances here:

<https://www.plymouth.ac.uk/uploads/production/document/path/7/7741/Extenuating_Circumstances_Policy_and_Procedures.pdf>

Students with valid EC’s may claim either for an extension period or for non-submission. Valid for Non-Submission means you will be asked to do a new piece of work. Please see below.

**Referral**

Please note that if you claim for non-submission and are offered a referral, you will be required to complete a NEW piece of work for the module. The new piece of work will assess that you have met the learning outcomes for the module but in a way that will be different to the original set piece. The referral is not a repeat or extension of the original coursework.

Carrying out a new piece of work means you will not be able to keep the marks already gained during the module. Eg: if you pass the set exercises (CW1) but do not submit the main work (CW2) AND are offered a referral, you will not keep the set exercises grade.

**Plagiarism**

All of your work must be of your own words. You must use references for your sources, however you acquire them. Where you wish to use quotations, these must be a very minor part of your overall work.

To copy another person’s work is viewed as plagiarism and is not allowed. Any issues of plagiarism and any form of academic dishonesty are treated very seriously. All your work must be your own and other sources must be identified as being theirs, not yours. The copying of another persons’ work could result in a penalty being invoked.

Further information on plagiarism policy can be found here:

Plagiarism: <https://www.plymouth.ac.uk/student-life/your-studies/essential-information/regulations/plagiarism>

Examination Offences: <https://www.plymouth.ac.uk/student-life/your-studies/essential-information/exams/exam-rules-and-regulations/examination-offences>

Turnitin (<http://www.turnitinuk.com/>) is an Internet-based 'originality checking tool' which allows documents to be compared with content on the Internet, in journals and in an archive of previously submitted works.  It can help to detect unintentional or deliberate plagiarism.

It is a formative tool that makes it easy for students to review their citations and referencing as an aid to learning good academic practice. Turnitin produces an ‘originality report’ to help guide you. To learn more about Turnitin go to:

<https://guides.turnitin.com/01_Manuals_and_Guides/Student/Student_User_Manual>

**Referencing**

The University of Plymouth Library has produced an online support referencing guide which is available here: <http://plymouth.libguides.com/referencing>.

Another recommended referencing resource is [Cite Them Right Online](http://www.citethemrightonline.com.plymouth.idm.oclc.org/); this is an online resource which provides you with specific guidance about how to reference lots of different types of materials.