ASSIGNMENT BRIEF

Programme: Computer Science, Software Engineering, Games Development

Module code/title: 1CB101

Assessment mode: Portfolio

Assessor(s): Andrew Guest

DEADLINE	Noon Tue 8th Jan 2019 (provisional)
HOW SUBMITTED	Electronic – through gitlab and moodle

Assessment
This module is assessed by a portfolio. The assessment will be handed out in four parts over weeks 7,8,10 and 11. This document will be updates as each part is handed out. Each part is worth twenty five marks (for a total mark out of 100) and will consist of two exercises. The marks available to each exercise varies so please read the briefing carefully. The sections below will detail each part of the assessment (as they are issued) and break down how each is marked. The assessment will be submitted through Moodle and GitLab. A project repository should be created for each part of the portfolio (for a total of four) and each repository should contain your solutions to the two exercises for that part. Once you have fully completed all four parts of the portfolio and uploaded them to the repository you will need to complete a submission form and upload it to Moodle. The completed submission form will contain four links to your four repositories of solutions. (The submission form will be made available once all four parts of the assessment have been issued). NB Late submission rules apply to the repositories. Late changes made within a week of the deadline (between noon Tue 8th Jan and noon Tue 15th Jan) will receive a capped mark of 40%. Submission more than a week late will not be marked.

Part One

- Clone yourself a copy of the gitlab repository https://git.yorkdc.net:8888/a.guest/assessment01-18.git
- It has a class and a unit test for that class
- You need to complete the methods in the class.
- The unit tests will let you check you have got a working solution
- NB hardcoding the methods to return the correct answers to the unit test rather than implementing the requested methods is **not a valid solution**
- Make sure you create a new repository in your gitlab for your solution. You will be submit your code for marking by sending me a link to your repository.
  - You should not send me the link now. Wait until you have completed all four parts of the assessment and submit all four links through Moodle. Instructions on how to do this will be given later.

01 - A Gentle Start
Worth 5 marks

Debug this method so it passes the unit test.
Add comments beside any changes you make

01 – A Gentle Start	**Total Marks Available 5**
Method compiles and runs	1
Method passes unit test	1
Correct changes made to code	2
Changes commented adequately	1

02 - Adding Across A Range
Worth 20 marks

Complete the method sumNumbersBetween so that it adds together all the integers between start and end, but not including start or end
This method should only deal with 0 and positive integers
This method should return -1 if it cannot calculate the result
It cannot calculate a result if :-

a) start > end
b) start or end = -1
c) there are no numbers between start and end

You should comment your code explaining what each part does

02 – Adding Across A Range	**Total Marks Available 20**
Method compiles and runs	2
Appropriate algorithm chosen (regardless of whether or not it works)	4
Method passes unit test	5
Method handles bad input correctly	5
Code commented adequately	4

Part Two

- Clone yourself a copy of the gitlab repository
- https://git.yorkdc.net:8888/a.guest/assessment02-18.git
- It has a class and a unit test for that class
- You need to complete the methods in the class.
- The unit tests will let you check you have got a working solution
- NB hardcoding the methods to return the correct answers to the unit test rather than implementing the requested methods is **not a valid solution**
- Make sure you create a new repository in your gitlab for your solution. You will be submit your code for marking by sending me a link to your repository.
  - You should not send me the link now. Wait until you have completed all four parts of the assessment and submit all four links through Moodle. Instructions on how to do this will be given later.

03 - Scrabble Score

Complete this method so that it returns the scrabble score for the word in aWord
In scrabble each letter is worth a number of points and each word is worth the sum of the scores of the letters in it. For this assignment we will ignore double/treble letter/word bonuses. The English language points per letter can be found at https://en.wikipedia.org/wiki/Scrabble_letter_distributions

You will need to come up with a way of connecting each letter to its score and a way of identifying each letter in the word.

03 – Scrabble Score	**Total Marks Available 10**
Method compiles and runs	1
Appropriate algorithm chosen (regardless of whether or not it works)	3
Score calculation /Method passes unit test	4
Code commented adequately	2

04 - Password Validator

Complete this method to validate that the String password is a valid password
A password is valid if it is
- between 8 and 16 characters long (inclusive)
- made up of letters (upper or lower), numbers, and the following characters !£$%
- has at least one lower case letter, one upper case letter and a number
- does not contain the phrases 'password' or 'passwd'

04 – Password Validation	**Total Marks Available 15**
Method compiles and runs	1
Appropriate algorithm chosen (regardless of whether or not it works)	2
Validation of password /Method passes unit test	10
Code commented adequately	2

Part Three

- Clone yourself a copy of the gitlab repository
- https://git.yorkdc.net:8888/a.guest/assessment03-18.git
- It has a class and a unit test for that class
- You need to complete the methods in the class.
- The unit tests will let you check you have got a working solution
- NB hardcoding the methods to return the correct answers to the unit test rather than implementing the requested methods is **not a valid solution**
- Make sure you create a new repository in your gitlab for your solution. You will be submit your code for marking by sending me a link to your repository.

The simplest form of encryption is the rotation cipher (also known as Caeser's Cipher). An offset value is chosen to encrypt a message. Each letter is replaced by the letter that that number of places away from it. If an offset value of 1 is chosen, each letter is replaced by the one after it, so a becomes b, b becomes c, etc. If a value of -2 is chosen a becomes y, b becomes z and so on

05 - encryptedCharacter
Complete this method so that it returns the encrypted character for theChar when and offset of theOffset is used. So if theChar='m' and theOffset is 3 the method will return 'p'. Lower case characters remain lower case, upper case remain upper case. Any other characters are returned unchanged

05 – encrypted Character	**Total Marks Available 15**
Method compiles and runs	1
Appropriate algorithm chosen (regardless of whether or not it works)	2
Character encryption / Method passes unit test	10
Code commented adequately	2

06 - encryptedMessage
Complete this method so that it uses encryptedCharacter to return the encrypted version of theMessage using theOffset.

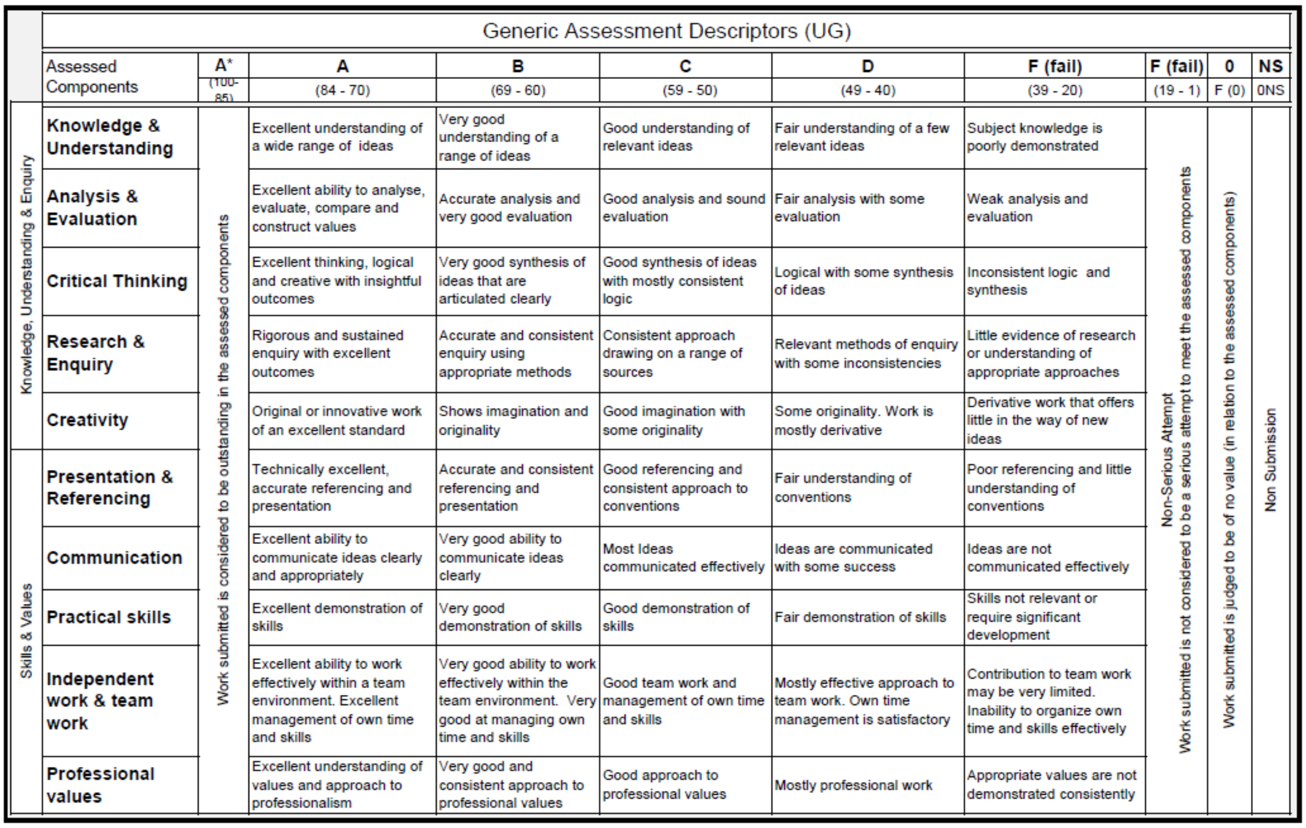
06 – Encrypted Message	**Total Marks Available 10**
Method compiles and runs	1
Appropriate algorithm chosen (regardless of whether or not it works)	3

Message Encryption /Method passes unit test	4
Code commented adequately	2

Part Four • Clone yourself a copy of the gitlab repository • https://git.yorkdc.net:8888/a.guest/assessment04-18.git • It has a class and a unit test for that class • You need to complete the methods in the class. • The unit tests will let you check you have got a working solution • NB hardcoding the methods to return the correct answers to the unit test rather than implementing the requested methods is **not a valid solution** • Make sure you create a new repository in your gitlab for your solution. You will be submit your code for marking by sending me a link to your repository.

FEEDBACK (how & when?)	Feedback will be provided through Moodle by Tuesday 30th January

University Generic Assessment Description



Generic Assessment Descriptors (UG)

	Assessed Components	A*	A	B	C	D	F (fail)	F (fail)	0	NS
		(100-85)	(84 - 70)	(69 - 60)	(59 - 50)	(49 - 40)	(39 - 20)	(19 - 1)	F (0)	0NS
Knowledge, Understanding & Enquiry	**Knowledge & Understanding**	Work submitted is considered to be outstanding in the assessed components	Excellent understanding of a wide range of ideas	Very good understanding of a range of ideas	Good understanding of relevant ideas	Fair understanding of a few relevant ideas	Subject knowledge is poorly demonstrated	Non-Serious Attempt Work submitted is not considered to be a serious attempt to meet the assessed components	Work submitted is judged to be of no value (in relation to the assessed components)	Non Submission
	Analysis & Evaluation		Excellent ability to analyse, evaluate, compare and construct values	Accurate analysis and very good evaluation	Good analysis and sound evaluation	Fair analysis with some evaluation	Weak analysis and evaluation			
	Critical Thinking		Excellent thinking, logical and creative with insightful outcomes	Very good synthesis of ideas that are articulated clearly	Good synthesis of ideas with mostly consistent logic	Logical with some synthesis of ideas	Inconsistent logic and synthesis			
	Research & Enquiry		Rigorous and sustained enquiry with excellent outcomes	Accurate and consistent enquiry using appropriate methods	Consistent approach drawing on a range of sources	Relevant methods of enquiry with some inconsistencies	Little evidence of research or understanding of appropriate approaches			
	Creativity		Original or innovative work of an excellent standard	Shows imagination and originality	Good imagination with some originality	Some originality. Work is mostly derivative	Derivative work that offers little in the way of new ideas			
Skills & Values	**Presentation & Referencing**		Technically excellent, accurate referencing and presentation	Accurate and consistent referencing and presentation	Good referencing and consistent approach to conventions	Fair understanding of conventions	Poor referencing and little understanding of conventions			
	Communication		Excellent ability to communicate ideas clearly and appropriately	Very good ability to communicate ideas clearly	Most Ideas communicated effectively	Ideas are communicated with some success	Ideas are not communicated effectively			
	Practical skills		Excellent demonstration of skills	Very good demonstration of skills	Good demonstration of skills	Fair demonstration of skills	Skills not relevant or require significant development			
	Independent work & team work		Excellent ability to work effectively within a team environment. Excellent management of own time and skills	Very good ability to work effectively within the team environment. Very good at managing own time and skills	Good team work and management of own time and skills	Mostly effective approach to team work. Own time management is satisfactory	Contribution to team work may be very limited. Inability to organize own time and skills effectively			
	Professional values		Excellent understanding of values and approach to professionalism	Very good and consistent approach to professional values	Good approach to professional values	Mostly professional work	Appropriate values are not demonstrated consistently			

Assessment Part One

This is the first part (of four) of the assessment for 1CB101 Introduction To Programming 2018.

Marking Guide

Solutions to Part One will be given a mark out of 25.

Five marks are available for 01 – A Gentle Start and twenty marks are available for 02 Adding Across A Range. These marks are broken down in the table below.

01 – A Gentle Start	**Total Marks Available 5**
Method compiles and runs	1
Method passes unit test	1
Correct changes made to code	2
Changes commented adequately	1
02 – Adding Across A Range	**Total Marks Available 20**
Method compiles and runs	2
Appropriate algorithm chosen (regardless of whether or not it works)	4
Method passes unit test	5

Method handles bad input correctly	5
Code commented adequately	4