UNIVERSITY OF JOHANNESBURG

FACULTY OF SCIENCE

ACADEMY OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

MODULE CSC1A10
Introduction to algorithm development (C++)
CAMPUS **APK**

EXAMINATION
PAPER B

DATE: 2016-05-28

ASSESSOR(S) **DR DA COULTER**

INTERNAL MODERATOR **MR A MAGANLAL**

DURATION 3 HOURS **MARKS 100**

SURNAME, INITIALS (or ID NUMBER):______________________________

STUDENT NUMBER: ______________________________

COMPUTER NR: ______________________________

CONTACT NR: ______________________________

NUMBER OF PAGES: 3 PAGES

REQUIREMENTS: NON-PROGRAMMABLE CALCULATORS ARE PERMITTED

Marker:			Submission overseen by:	
Sort Rank	Result	Moderation	Correction	**Submission**
				CD:
				USB:
				EVE:

Mark sheet		
Surname:		
Initials:		
Computer:		
Competency	*Description*	*Result*
C0	Program Design	/10
C1	Boiler plate code • Standard namespace (1) • System library inclusion (3) • Indication of successful termination of program (1)	/5
C2	Coding style • Naming of variables (1) • Indentation (1) • Use of comments (1) • Use of named constants (1) • Program compiles without issuing warnings (1)	/5
C3	Functional Abstraction • Task decomposition (5) • Reduction of repetitive code (5)	/10
C4	Separate Compilation • Header file (1) • Guard conditions (2) • Inclusion of header file (1) • Appropriate content in header file (1) • Use of programmer defined namespace (5)	/10
C5	User Interaction • Menu System (5) • Appropriate use of input, output and error streams (5)	/10
C6	Command Line Argument Handling: • Appropriately overloaded main function (1) • Handling incorrect argument counts (1) • Use of supplied arguments (3)	/5
C7	Error Handling • Use of assertions (2) • Use of conventional error handling techniques (3)	/5
C8	Pseudo-random number generation (5)	/5
C9	Dynamically allocated two dimensional array handling • Allocation (5) • Initialisation (5) • Deallocation (5)	/15
C10	Algorithm implementation • Logical Correctness (5) • Effectiveness / Efficiency of approach (5) • Correct use of appropriate selection / iteration structures (5) • Correct output (5)	/20
B	Bonus	/10
Total:		**/100**

Markers Signature:____

I declare that I am eligible to write this summative assessment according to the rules and regulations of the Academy of Computer Science & Software Engineering, the Faculty of Science and the University of Johannesburg. I declare that the work submitted is my own and that I have verified the correctness of my electronic submissions.

I UNDERSTAND THAT NON-COMPILING CODE CANNOT BE AWARDED A PASSING MARK

Student Signature:____

EQUATION HUNTER

The Utopian Department of Education has approached you to create an educational game as a turn based, text console application in C++:

					●		
				43			
					45		
		42					
	47		40				
49							

*6 * 7 = ?*

Player (Black Circle) Empty squares (open space) Possible answers (number)

In the game you will need to move a player controlled character around a two dimensional playing area. The player will need to collect the correct answer from a set of possible answers to a randomly generated simple arithmetic equation. Only one of the answers is correct. Your logic must be placed in the `MathSpace` **`namespace`**.

Initialisation:

- The size of the playing area is given as a command line argument.
- The player is placed in row 1 of a random column
- A random question is generated made up of two random numbers between 1 and 10 (inclusive) which either multiplied, divided, added or subtracted together.
- The correct answer and a random number of incorrect answers are placed randomly on the map on any row after row 2. All answers are rounded off to the nearest integer.
- All remaining cells contain empty space.

Moving:

- The player may move north (up), south (down), east (right), or west (left). The player may not move outside of the game area.
- If the player moves on top a cell containing an incorrect answer the equation is replaced and new answers are placed in the playing area.

End-game:

- The game ends when the player moves over the correct answer.

Using your knowledge of good software engineering principles and C++ you must design and implement such a simulation as follows. Consider the competencies as laid out in the mark sheet.

- C0 – Create a program design. Your UML must model the generation of the equation and placement of answers.
- C1 – Use your knowledge of basic C++ program structure and make sure to utilise the appropriate system libraries.
- C2 – Your program must be readable by human beings in addition to compiler software.
- C3 – Demonstrate your knowledge of the divide and conquer principle using functions.
- C4 – Your program must make use of programmer defined source code libraries.
- C5 – Create a menu system which will ask the user which action they wish to take.
- C6 – The user must provide the number of rows and columns used by the simulation (range checked based on terminal width).
- C7 – Provide assertion based error handling as well as conventional error handling.
- C8 – Random numbers are used when initialising the 2D arrays.
- C9 – Use dynamic 2D arrays to implement your simulation. The main array may be output to screen using printable ASCII characters.
- C10 – Pay careful attention to checking the legality of moves.
- Bonus – Make use of C++11/14 features, structures, and/or enumerations in your code.