THE UNIVERSITY OF WAIKATO
Te Whare Wānanga o Waikato

2022 SCHOLARSHIP EXAMINATION

DEPARTMENT	Computer Science
COURSE TITLE	Year 13 Scholarship
TIME ALLOWED	FIVE hours with a break for lunch at the discretion of the supervisor
QUESTIONS	There are TWO questions in the paper. Candidates are to answer BOTH questions. Answer as much of each question as you can. Note that Question 2 is significantly more difficult than Question 1. Plan your time to allow a good attempt at each.
INSTRUCTIONS	Candidates may use any text or manual or online programming language documentation for reference during the examination. Candidates may not copy code from the internet or consult anyone other than the examiners during the examination
DETAILS	Both questions pose problems which you are asked to solve by writing computer programs. They may also ask for written answers for some problem parts. In programming you may work in the programming language of your choice. However, the examiners need to be able to read your program text and if at all possible, test run it. If problems arise from your choice of programming language, we may contact you after the examination, for clarification. Written answers to parts of questions can be submitted in text files; included as comments in your program text; or as photographed or scanned images of hand written documents. Remember also that partial marks may be awarded for programming ideas written down, but not yet implemented.
CALCULATORS PERMITTED	Yes

1. **A tiny lexicon** (Careful and Accurate Programming)

Your programming work in this question will be assessed on three criteria:

(a) Completeness and accuracy of the program. It may be that this problem statement does not state exactly what the program should do under all circumstances. If you find a situation of that nature, choose a solution and write down, either on paper or in the comments of your program what the difficulty was and how you chose to resolve it.

(b) Good presentation. That is, it should make good use of programming language facilities, be well organised, neatly laid out, and lightly commented.

(c) Careful checking. Wherever possible check input from the program user in case they have made errors.

In this question, you are asked to write a program that collates and processes a small collection of words. Imagine that you have some text, or a conversation, and you are interested in learning which words are used, how often they are used, and which words are used most and least often. Your task is to write a program which interacts with a user allowing that user to enter the words from a piece of text or conversation.

Your program should allow the user to enter a series of words, one word at a time. Any input must be one word long (i.e. does not include spaces), must contain alphabet characters only (i.e. a-z), and must contain lowercase characters only. Your program should then process and analyze the words in four different ways, as follows.

- ⇨ Print all of the words on one line, comma separated
- ⇨ Find the shortest word(s), print them on one line, comma separated
- ⇨ Find the longest word(s) , print them on one line, comma separated
- ⇨ Find the most commonly used word(s), print them on one line, comma separated

Finally, your program should display a list of the unique words including the count of each word (see sample output on the next page)

The transcript of a sample interaction with such a program is given on the next page. In the transcript, information entered by the user is shown in **bold** type. You don't have to follow this style of data entry or format results in the same way. The sample is just here to show the kind of interaction expected of your program.

QUESTION 1 CONTINUES ON THE NEXT PAGE

```

The Tiny Lexicon

How many words would you like to enter: 5
  Please enter word 1: harry
  Please enter word 2: potter
  Please enter word 3: is
  Please enter word 4: in
  Please enter word 5: gryffindor

Would you like to enter any more words? (y/n): y

How many words would you like to enter: 5
  Please enter word 1: harry
  Please enter word 2: is
  Please enter word 3: not
  Please enter word 4: in
  Please enter word 5: slytherin

Would you like to enter any more words? (y/n): n

Would you like to print your words? (y/n): y
  harry, potter, is, in, gryffindor, harry, is, not, in, slytherin

Would you like to find the shortest word(s)? (y/n): y
  is, in

Would you like to find the longest word(s)? (y/n): y
  gryffindor

Would you like to find the most commonly used word(s)? (y/n): y
  harry, in, is

Would you like to print your wordlist including word counts? (y/n): y
  gryffindor (1)
  harry (2)
  in (2)
  is (2)
  not (1)
  potter (1)
  slytherin (1)

```

2. **Battleship** (Problem Solving and Programming)

Your programming work in this question will be assessed on three criteria:

(a) Your approach to the problem. We will be looking at your work for evidence that you found good ways of storing the necessary data, and devised algorithms for finding and displaying the requested results. ***Please hand in any notes and diagrams which describe what you are attempting to program, even if you don't have time to code or complete it. You may include comments in your program, or write a description of your program to hand in.***

(b) The extent to which your program works and correctly solves the problem.

(c) The extent to which you use results from your programming to explore the problem presented.

You may find that the programming language you use makes it difficult to produce output as shown in the example implementation steps below. If this is the case, feel free to build your program in a way that suits your circumstances.

Note: Six text files have been provided to you: battleship_grid(1).txt, battleship_grid(2).txt, battleship_grid(3).txt, battleship_grid(rectangle).txt, battleship_grid(large).txt, and battleship_grid(bad).txt. Image files have also been provided to illustrate the layout described in each text file.

Battleship is a strategic guessing game. It is played on a grid, where battleships are marked but hidden. The player must guess grid positions to attempt to 'sink' the battleships. If they guess a position where a battleship is located, then it is a 'hit'. If they guess a position where a battleship is not located, then it is a 'miss'. Battleships are always one square wide, but can be any number of squares long. They can be positioned either horizontally or vertically. Once a player gets a 'hit' in one square, they can then guess the squares around that location to sink the rest of that battleship. The game is over when all grid squares of all battleships have been sunk.

```
+---+---+---+---+         +---+---+---+---+         +---+---+---+---+
| B | B |   |   |         |   |   |   |   |         | X | X | . | . |
+---+---+---+---+         +---+---+---+---+         +---+---+---+---+
|   |   |   | B |         |   |   |   |   |         |   |   |   | X |
+---+---+---+---+         +---+---+---+---+         +---+---+---+---+
|   | B |   | B |         |   |   |   |   |         | . | X |   | X |
+---+---+---+---+         +---+---+---+---+         +---+---+---+---+
|   |   |   | B |         |   |   |   |   |         |   | . | . | X |
+---+---+---+---+         +---+---+---+---+         +---+---+---+---+
```

In this question, you are asked to write a program to display and play a game of battleship, using only text display. The question presents the problem in stages for you to program. We suggest that you build your program in the order given. This will make it likely that you have parts working at the end, even if you don't have time to complete the whole program. We also strongly suggest that you read through all the stages before starting to program. Stage I is the final stage, in which you have the most freedom to explore algorithm ideas.

The stages of this problem involve building and changing a program. Instructions will be given in some detail for the first stages. Later stages require that you develop the code yourself. When you are making a major change, you should save a working version of your program. This will help us see what you have achieved, especially if you have difficulties with the altered version. Where stages ask you to try different ways of displaying the game, you can write different display procedures within the same program to make sure that all of your answers are still visible to the examiner.

QUESTION 2 CONTINUES ON THE NEXT PAGE

Setting up the Game

Stage A: Building a basic grid

Battleship is played on a grid. Write a program to draw a 4 x 4 grid using text characters. The result should be similar to the figure below.

```
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
```

Stage B: Building a dynamic grid

There's no guarantee that our board is going to be 4 x 4. Extend your program to build a grid of any height and width. Our goal is to work towards a dynamic battleship game. The results when creating a 3 x 3 grid should be similar to the figure below left, while the results when creating a 5 x 2 grid should be similar to the figure below right.

```
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
|   |   |   |
+---+---+---+
```

```
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
|   |   |   |   |   |
+---+---+---+---+---+
```

Stage C: Adding the battleships

The next thing we need to do is add battleships to our grid. Given a 4 x 4 grid, add battleships to the following positions:

- X = 0, Y = 0, length = 2, orientation = horizontal
- X = 3, Y = 1, length = 3, orientation = vertical
- X = 1, Y = 2, length = 1, orientation = horizontal

The results should be similar to the figure below, where B marks each section of a battleship.

```
+---+---+---+---+
| B | B |   |   |
+---+---+---+---+
|   |   |   | B |
+---+---+---+---+
|   | B |   | B |
+---+---+---+---+
|   |   |   | B |
+---+---+---+---+
```

QUESTION 2 CONTINUES ON THE NEXT PAGE

Stage D: Adding the battleships dynamically

Just like our grid, the number of and locations of battleships should be dynamic. Extend your program to read grid size and battleship information from file. Note: 6 files have been provided for you: *battleship_grid(1).txt*, *battleship_grid(2).txt*, *battleship_grid(3).txt, battleship_grid(rectangle).txt, battleship_grid(large).txt, and battleship_grid(bad).txt*

For example, the file called *battleship_grid(1).txt* contains the grid size and battleship information for the example grid shown in Stage C. The lines of the file are as follows:

1. Grid width
2. Grid height
3. Number of battleships
4. X position of the first battleship
5. Y position of the first battleship
6. Length of the first battleship
7. Orientation of the first battleship
   (coded as either “H” for horizontal or “V” for vertical
8. X position of the second battleship
9. Y position of the second battleship
10. Length of the second battleship
11. Orientation of the second battleship
12. and so on

The images below show the expected output from *battleship_grid(1).txt*, *battleship_grid(2).txt*, and *battleship_grid(3).txt. battleship_grid(rectangle).txt* can be used to test a rectangular grid, *battleship_grid(large).txt* can be used to test a large grid, and *battleship_grid(bad).txt* can be used as an example of a bad input file. Image files have also been provided to illustrate the layout described in each text file.

```
+---+---+---+---+
| B | B |   |   |
+---+---+---+---+
|   |   |   | B |
+---+---+---+---+
|   | B |   | B |
+---+---+---+---+
|   |   |   | B |
+---+---+---+---+
```
battleship_grid(1).txt

```
+---+---+---+---+
| B |   | B |   |
+---+---+---+---+
| B |   | B |   |
+---+---+---+---+
| B |   |   |   |
+---+---+---+---+
|   |   | B | B |
+---+---+---+---+
```
battleship_grid(2).txt

```
+---+---+---+---+
| B |   | B | B |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
| B | B | B |   |
+---+---+---+---+
|   |   |   | B |
+---+---+---+---+
```
battleship_grid(3).txt

Playing the Game

Stage E: Game play

Now that you have set up the grid, you can implement game play. When the game starts, the following steps should be taken:

1. Select one of the input files (*battleship_grid(1).txt – battleship_grid(3).txt*) and use it to setup the game
2. Welcome the user to your game and ask them their name
3. Display the board, but with all battleships hidden
4. Ask the user to enter an x and y position for their guess

QUESTION 2 CONTINUES ON THE NEXT PAGE

5. Redisplay the board, updated with the user's guess
    a. If the position is a 'hit', i.e. there is a battleship at that position, then display an 'X'
    b. If the position is a 'miss', i.e. there is not a battleship at that position, display a '.'
6. Continue the game until all grid squares of all battleships have been hit. Inform the user when they have won

The output below (and Appendix 1) shows an example of gameplay.

```
Welcome to Battleship!

Please enter your name: Hermione

Welcome Hermione!
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 0
Y position: 0

It's a hit!
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 0
Y position: 0

That position has already been guessed, please guess again
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 1
Y position: 1

It's a miss
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position:
```

QUESTION 2 CONTINUES ON THE NEXT PAGE

Stage F: Game play continued

Right now the user can take as many guesses as they like. This means that they will always win the game, even if they guess every position on the grid. Our game is dynamic (our grid can be any size) so limiting the user to a certain number of guesses isn't ideal. For example, if we limit them to 16 guesses, then this will allow them to guess every square in a 4 x 4 grid, and will only give them a very small number of guesses on a 100 x 100 grid. Extend your program to limit the number of guesses that the user is allowed. Determine the guesses in a way that will act proportionate to the grid size. Your answer should include a written description of the method you used, and your reasoning for selecting this method.

Stage G: Game play: two player

Battleship is usually a two-player game. Extend your program to allow two players to play together. Note: each player would have their own board and the players would take turns guessing. Allow the user to select whether they would like to play the one-player or two-player version of your game. Appendix 2 shows an example of two-player game play.

Stage H: Game play: playing the computer

Two player games often allow you to play against the computer. Extend your program so that Player Two is the computer. In this case, each time it is the computer's turn, it should simply select a position randomly. Allow the user to select whether they would like to play the one-player, two-player, or two-player (with computer) version of your game. Appendix 3 shows an example of two-player game play.

Advanced gaming AI

Stage I: AI for game play

Please note, Stage I is the final stages, in which you have the most freedom to explore algorithm ideas. You may complete either part (i), or part (ii), or both.

(i) Stage H asked you to implement a very simple AI player. Extend your program to include an AI player that is more clever. Your answer should include a written description of the method(s) that you try, and your reasoning for trying them.

(ii) So far, the size, orientation, and location of battleships has been determined by input files. Extend your program to include dynamic battleship placement, i.e. the length, orientation, and location of battleships is determined 'on the fly' by your program and may be different each time your program is run. Your answer should include a written description of the method(s) that you try, and your reasoning for trying them.

Appendix 1

One-player Example

```
Welcome to Battleship!

Please enter your name: Hermione

Welcome Hermione!
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 0
Y position: 0

It's a hit!
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 0
Y position: 0

That position has already been guessed, please guess again
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position: 1
Y position: 1

It's a miss
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
Please pick an x and y position
X position:

...
```

Appendix 2

Two-player Example

```
Welcome to Battleship!

1. One player
2. Two player
Please enter your selection (1 or 2): 2

Welcome to Two-Player Battleship!

Please enter Player One's name: Harry
Please enter Player Two's name: Ron

Welcome Harry and Ron!

Harry's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Ron's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

It's Harry's turn
Harry, please pick an x and y position
X position: 0
Y position: 0

It's a hit!

Harry's Grid
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Ron's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
It's Ron's turn
Ron, please pick an x and y position
X position: 1
Y position: 1

It's a miss

Harry's Grid
+---+---+---+---+
| X |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Ron's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

It's Harry's turn
Harry, please pick an x and y position
X position: 1
Y position: 0

It's a hit!

Harry's Grid
+---+---+---+---+
| X | X |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Ron's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

It's Ron's turn
Ron, please pick an x and y position
X position: 0
Y position: 0

It's a hit!

...
```

Appendix 3

Two-player with computer Example

```
Welcome to Battleship!

1. One player
2. Two player
3. Play against the computer
Please enter your selection (1, 2, or 3): 3

Welcome to Two-Player (against the computer) Battleship!

Please enter Player One's name: Ginny
Player Two will be the computer

Welcome Ginny and Computer!

Ginny's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Computer's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

It's Ginny's turn
Ginny, please pick an x and y position
X position: 1
Y position: 1

It's a miss

Ginny's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Computer's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
It's Computer's turn
Computer, please pick an x and y position
X position: 1
Y position: 3

It's a miss

Ginny's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+

Computer's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+

It's Ginny's turn
Ginny, please pick an x and y position
X position: 3
Y position: 3

It's a hit!

Ginny's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   | X |
+---+---+---+---+

Computer's Grid
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   |   |   |   |
+---+---+---+---+
|   | . |   |   |
+---+---+---+---+

It's Computer's turn
Computer, please pick an x and y position
X position: 0
Y position: 0

It's a hit!

...
```