

Module 18 HCI & GUI Programming

Module title	Computer Programming
Module NFQ level (only if an NFQ level can be demonstrated)	7
Module number/reference	BSCH-HGP
Parent programme(s)	Bachelor of Science (Honours) in Computing Science
Stage of parent programme	Award stage
Semester (semester1/semester2 if applicable)	Semester 1
Module credit units (FET/HET/ECTS)	ECTS
Module credit number of units	5
List the teaching and learning modes	Direct, Blended
Entry requirements (statement of knowledge, skill and competence)	Learners must have achieved programme entry requirements.
Pre-requisite module titles	BSCH-CP, BSCH-OOP, BSCH-SD1, BSCH-SD2, BSCH-PDS
Co-requisite module titles	None
Is this a capstone module? (Yes or No)	No
Specification of the qualifications (academic, pedagogical and professional/occupational) and experience required of staff (staff includes workplace personnel who are responsible for learners such as apprentices, trainees and learners in clinical placements)	Qualified to as least a Bachelor of Science (Honours) level in Computer Science or equivalent and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.
Maximum number of learners per centre (or instance of the module)	60
Duration of the module	Two Academic Semesters, 12 weeks teaching
Average (over the duration of the module) of the contact hours per week	3
Module-specific physical resources and support required per centre (or instance of the module)	One class room with capacity for 60 learners along with one computer lab with capacity for 25 learners for each group of 25 learners

Analysis of required learning effort		
	Minimum ratio teacher / learner	Hours
Effort while in contact with staff		
Classroom and demonstrations	1:60	18
Monitoring and small-group teaching	1:25	18
Other (specify)		
Independent Learning		
Directed e-learning		
Independent Learning		44
Other hours (worksheets and assignments)		45
Work-based learning – learning effort		
Total Effort		125

Allocation of marks (within the module)					
	Continuous assessment	Supervised project	Proctored practical examination	Proctored written examination	Total
Percentage contribution	40%	60%			100%

Module aims and objectives

The aims of this module are to empower the learner with the theory and practice of modern GUI development utilizing HCI principles.

The objectives of this module are:

- To familiarize the learner with human computer interaction principles.
- To give the learner a grounding in the creation of standard GUI components.
- To encourage the learner to discover appropriate combination of standard GUI controls to solve a problem.
- To support the learner in developing custom controls to tackle more challenging problems.
- To enable learners to create comprehensive GUI applications incorporating standard and custom controls to solve complex problems.

Minimum intended module learning outcomes

On successful completion of this module, the learner will be able to:

1. Create programs controlled by GUIs
2. Apply HCI principles to GUI implementation
3. Implement GUIs using the Module, View and Controller design paradigm
4. Address biological and behavioural factors which influence user GUI interactions when developing GUI application
5. Explain how implemented GUIs embody good interface design

6. Identify the need for custom controls and demonstrate their implementation
7. Apply the features of event-driven programming

Rationale for inclusion of the module in the programme and its contribution to the overall MIPLOs

Understanding the principles of Human Computer Interaction to better aid Graphical User Interface design is a fundamental skill in computing science. As part of this programme learners develop skills in problem analysis, algorithmic development, and application implementation. As part of this module learner will

- Achieve increased usability of their code by nontechnical users through encapsulation, abstraction and providing access to application functionality through a graphical user interface.
- Gain an appreciation for the needs of a user and how they can be accommodated into an event driven programming environment.
- Understand the mechanisms of human perception and behaviour and how the GUI can be optimized to allow the user to leverage the intended application functionality seamlessly.

Appendix 1 of the programme document maps MIPLOs to the modules through which they are delivered.

Information provided to learners about the module

Learners receive a programme handbook to include module descriptor, module learning outcomes (MIMLO), class plan, assignment briefs, assessment strategy and reading materials.

Module content, organisation and structure

Introduction and motivation

- Overview of the subject.
- Why do we need HCI?
- How do we evaluate the usability of systems?

User Biology & Influences on GUI design

- Vision
- Visual Processing
- Reading
- Attention
- Memory
- Motivation to learn
- Behaviour patterns

- Colour Theory

Presentation of information to the User

- Layout guidelines
- Flow of information
- Methods for displaying different categories of information

Model View Controller

- Theory
- Practical implementation

Event-driven programming overview

- Principles of event-driven programming
- Passing by reference verses passing by value.

Accepting information from the user

- Field entry
- Validation of data
- Restricting input options
- Multiple inputs
- Gathering information from the mouse

Testing and verification

- Principles of testing
- Different testable aspects of an application
- Prototyping
- User testing / QA testing

Module teaching and learning (including formative assessment) strategy

The module is taught as a combination of lectures and lab sessions. The lecture sessions discuss and explain to learners the theoretical underpinnings HCI and GUI. The practical lab sessions give learners an opportunity to build GUIs which embody the principles outlined in the lecture materials. Learners will also experience first-hand how they interact with the computer and GUI components solidifying their conviction that the HCI principles presented in the lecture materials are well founded. Through experimentation with varying GUI designs they will explore this nuanced field gaining from feedback from fellow classmates.

Assessment is divided into three elements. First there will be there take home assignments that assess the learner's competency in specific areas of the syllabus, while giving them increasing larger problems to solve. There are biweekly very minor class tests to function as a regular revision exercises to ensure that the theory remains

in the fore front of learning minds when performing implementation. Each minor test examines an increasingly large portion of the module. Finally, there is a group project to be completed in the second half of the semester which tests the learners' understanding of the theoretical material and allows them to realize a fully-fledged GUI based application which embodies core HCI principles.

Timetabling, learner effort and credit

The module is timetabled as one 1.5-hour lectures and one 1.5-hour lab per week.

Continuous assessment spreads the learner effort to focus on small steps before integrating all steps into the overall process of GUI program design and implementation.

There are 36 contact hours made up of 12 lectures delivered over 12 weeks with classes taking place in a classroom. There are also 12 lab sessions delivered over 12 weeks taking place in a fully equipped computer lab. The learner will need 44 hours of independent effort to further develop the skills and knowledge gained through the contact hours. An additional 45 hours are set aside for learners to work on worksheets and assignments that must be completed for the module.

The team believes that 125 hours of learner effort are required by learners to achieve the MIMLOs and justify the award of 5 ECTS credits at this stage of the programme.

Work-based learning and practice-placement

There is no work based learning or practice placement involved in the module.

E-learning

The college VLE is used to disseminate notes, advice, and online resources to support the learners. The learners are also given access to Lynda.com as a resource for reference.

Module physical resource requirements

Requirements are for a classroom for 60 learners equipped with a projector, and a 25-seater computer lab for practical sessions with access to Python Development Environment and associated libraries (this may change should more suitable technologies become available).

Reading lists and other information resources

Recommended Text

Johnson, J. (2014) *Designing with the Mind in Mind*. Amsterdam: Morgan Kaufmann.

McKay, E. N. (2013) *UI is Communication: How to Design Intuitive, User Centered Interfaces by Focusing on Effective Communication*. Amsterdam: Morgan Kaufmann.

Summerfield, M. (2015) *Rapid GUI Programming with Python and Qt*. London: Prentice Hall.

Secondary Reading:

Dix, A. (2003) *Human-computer Interaction*. Harlow: Pearson.

Itten, J. (2006) *Design and Form*. Hoboken: Wiley.

Johnson, P. (1992) *Human-computer Interaction: Psychology, Task Analysis and Software Engineering*. London: McGraw Hill.

Specifications for module staffing requirements

For each instance of the module, one lecturer qualified to at least Bachelor of Science (Honours) in Computer Science or equivalent, and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.. Industry experience would be a benefit but is not a requirement.

Learners also benefit from the support of the programme director, programme administrator, learner representative and the Student Union and Counselling Service.

Module Assessment Strategy

The assignments constitute the overall grade achieved, and are based on each individual learner's work. The continuous assessments provide for ongoing feedback to the learner and relates to the module curriculum.

No.	Description	MIMLOs	Weighting
1	Assignment 1 and Assignment 2 are 2 take home assignments that will foster development of the skills of: <ul style="list-style-type: none"> • Program design • GUI design • Processing User input • Event handling • Objects • Pass by reference 	1,2,3,4,6,7	25%
2	Biweekly very minor class tests to function as a regular revision exercises to ensure that the theory remains in the fore front of learning minds when performing implementation. Each minor test examines an increasingly large portion of the module.	2,4,5	15%
3	Project to be completed with a partner.	1,2,3,4,6,7	60%

All repeat work is capped at 40%.

Sample assessment materials

Note: All assignment briefs are subject to change in order to maintain current content.

Assignment 1

Introduction

In this assignment, you are required to produce a working application, which will perform conversion calculations from imperial to metric units. The GUI should follow the basic structure indicated in Figure 1a & 1b but you are encouraged to make improvements to the interface by using some of the principles from the lectures.

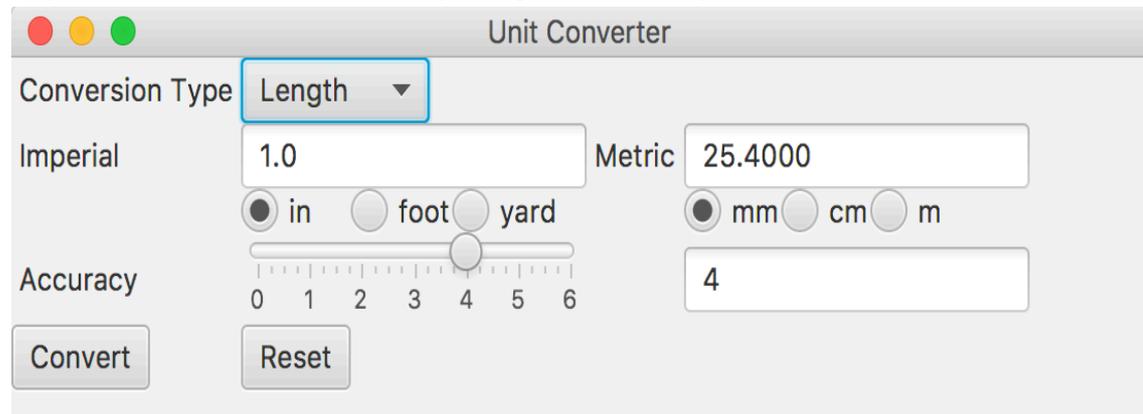


Figure 1a: Default state when application is started or *length* is selected

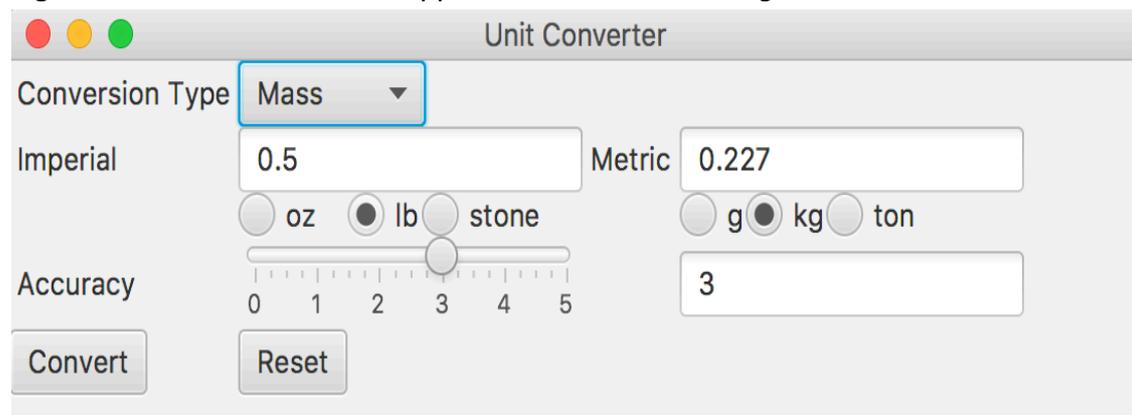


Figure 1b: Default state when *mass* is selected

Template File

- You will be given a template java file to add to.
- The lecturer solution is approximately 290 lines of code including comments.
- There are many lines of code, imports etc. missing.
- The amount of space in the template files does not indicate the number of lines of code required or the exact locations of missing code.
- Using the same method and variable names as those provided in the template will make it easier for you to receive support from your fellow classmates and lecturer.
- You will learn more if you comment your code in a detailed fashion prior to writing it.

Applications to give you ideas

Here is another interface which does similar work, feel free to compare it with the prescribed task but do not alter your interface to look like this one. More flexibility with interface design will be encouraged for the project.

<https://www.diydata.com/information/metricimpcnversions/calculator/calculator.php>¹

User Tasks and Marking Scheme

Task No.	Detail	Marks (100)
1	Upon loading the application should display all the relevant controls in a similar layout.	20
2	Upon loading the controls should contain default values as displayed in Figure 1a/1b including <ul style="list-style-type: none">- 2nd row <i>TextFields</i>: length/mass values for imperial/metric- 3rd row <i>RadioButtons</i>: imperial/metric measurement units- 4th row <i>Accuracy Slider</i> and <i>Accuracy TextField</i>	5
3	On changing the conversion type from Length to Mass <ul style="list-style-type: none">a) Imperial mass should be 0.5b) Imperial lbs <i>RadioButton</i> should be selectedc) Metric kg <i>RadioButton</i> should be selectedd) <i>Accuracy Slider</i> should range from 0 to 5, have 3 preselected and have ticks every 1 units.e) The preselect <i>Accuracy Slider</i> value should appear in the <i>Accuracy TextField</i>. These values are displayed in Figure 1b, when Length is selected the values should return to those displayed in Figure 1a.	20 (5 x 4 marks)
4	If the <i>Imperial TextField</i> does not contain a numerical value and the convert button is pressed an appropriate error message should be displayed in the <i>Imperial TextField</i> and the <i>Metric TextField</i> should become blank.	5
5	If a new valid numerical value is typed into <i>Imperial TextField</i> and the <i>Convert Button</i> pressed then the conversion value should appear in the <i>Metric TextField</i> .	20
6	On changing the <i>RadioButton</i> selections the <i>Metric TextField</i> should present the new conversion.	5

¹ Last accessed 24/07/2018

7	On moving the <i>Accuracy Slider</i> the values in the <i>Accuracy TextField</i> should change and the number of decimal places displayed in the <i>Metric TextField</i> should change.	5
8	The <i>Imperial TextField</i> should be the only editable one.	5
9	When the <i>Reset</i> button is pressed the application should initialize the controls based on the current state of the <i>Conversion Type ComboBox</i> and appear like Figure 1a/1b.	5
10	Additional layout improvements using Gestalt Principles, the functionality of the application must not be reduced.	5
11	Additional functionality improvements make sure to clearly highlight it in the UI and comment it in the code so I can see it!	5

Submission Notes

- You will be given approximately 2 weeks to do this assignment.
- The deadline for this assignment is displayed on Moodle.
- You are required to submit a single zip file to the Moodle using the file name conventions detailed in the *Assessment, Submissions and Requirements* Lecture.
- Penalties that will be applied should your submission fail to meet certain standards. These are detailed in the above set of lecture notes.

Additional Notes

The JavaFX lab book will only give you the pieces to construct widgets and perform event handling on them. You are required to generate the rest of the application yourself. It will be a long time before you will see if your application is working fully, but be sure to test each additional code block before proceeding. As a suggestion whenever you add event handlers or methods you should test them will simple print statements to the console to ensure that you are receiving events correctly.

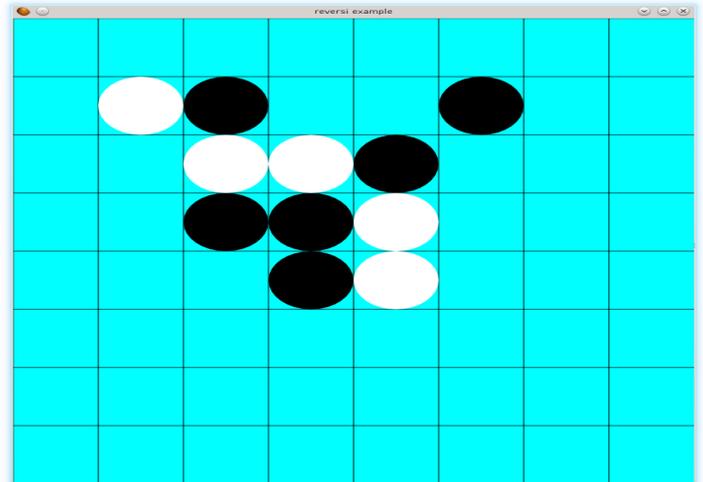
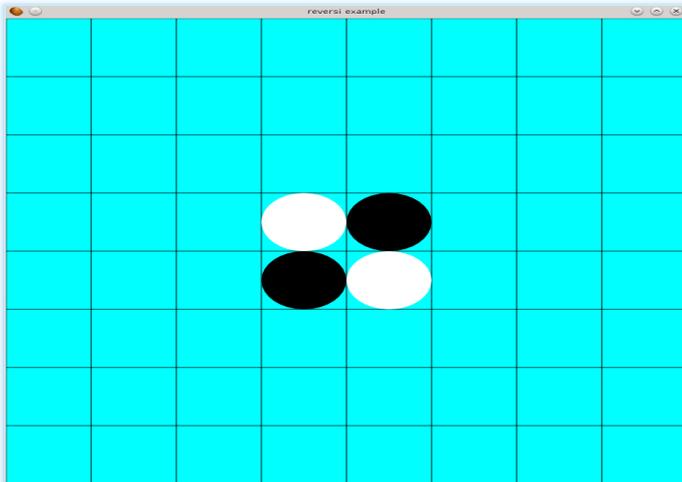
Assignment02: Building a working game of Reversi

Introduction:

NOTE: please read the assignment in full before you start coding. Failure to do so will result in you missing out important information.

In this assignment, you will be tasked with building a fully working game of Reversi (also known as Othello). The object of the game is to capture as many of your opposing player's pieces while trying to fill the board. The player with the most pieces at the end

is the winner. Below are screenshots that show the initial state of the game and a game currently being played. This will test your ability to react to user input in the form of keyboard and mouse events and also test your drawing abilities. Note that there is a number of simple algebraic formulas that you will define along the way so it helps to have pen and paper handy to do drawings to understand the concepts.



There is a source file provided on Moodle that contains all the fields, methods, and classes you need to build this application. It is highly recommended that you use this file as the shell code for this application is 270 lines long. The code you will be adding will be roughly 250 lines long.

Tasks and MarkingScheme:

It is recommended that you follow the order listed below and that you read the notes for additional information.

1. in the Reversi class modify the main() method to launch the JavaFX application (1%)
2. in the Reversi class modify the start() method to set a title on the primaryStage, set a scene that is 800x800 pixels with sp_mainlayout as the root and show the window (1%)
3. in the Reversi class modify the init() method initialise sp_mainlayout, it should initialise rc_reversi and add it to sp_mainlayout. (1%)
4. in the ReversiControl class modify the constructor() to initialise rb_board and add it as a child of the ReversiControl. You should add a listener for mouse clicks that will call the placePiece() method of rb_board with the coordinates of the click. You should also add a listener for key presses that will call the resetGame() method of rb_board whenever the space key is pressed (5%)

5. in the ReversiControl class modify the resize() method to call the superclass resize method and call the resize method of rb_board. (1%)
6. in the ReversiBoard class modify the constructor to allocate memory for the following 7 arrays (numbers in brackets are the required sizes): (4%)
 - render (8x8)
 - horizontal (8)
 - vertical (8)
 - horizontal_t (8)
 - vertical_t (8)
 - surrounding (3x3)
 - can_reverse (3x3)

After this call the methods for initialising the lines and background, initialising the render array and resetting the game.

7. in the ReversiBoard class modify the initLinesBackground() method to (8%)
 - create a cyan rectangle and add it to the children of this ReversiBoard.
 - Generate 8 horizontal lines and their associated translate objects, horizontal and horizontal_t respectively. Attach the translate object to the line and set the x1,y1,y2 value to be 0. add each line as a child of this ReversiBoard.
 - Repeat the same process above for vertical lines and set x1,x2,y1 to have a value of 0
8. in the ReversiBoard class modify the initialiseRender() method to create render objects in the render array and construct them with a value of 0 to represent an empty space (2%)
9. in the ReversiBoard class modify the resetRenders() method to call the setPiece() method of each render object with a value of 0 (1%)
10. in the ReversiBoard class modify the resetGame() method to (4%)
 - reset the render array.
 - It should set the pieces at coordinates (3,3) and (4,4) to have a value of 1 and set the pieces at (3,4) and (4,3) to have a value of 2.

- set `in_play` to true, `current_player` to 2, opposing to 1, and default the scores to 2 all.

11. in the `ReversiBoard` class modify the `resize()` method to (2%)

- call the superclass `resize` method.
- Determine the width and height of each cell in the board based on the new width and height (store in `cell_width`, and `cell_height` respectively)
- resize the background rectangle to take the full width and height of the window.
- Resize and relocate the horizontal and vertical lines (you have methods for this)
- resize and relocate the pieces (you have a method for this)

At this point if you run your code you should see a completely cyan background on your window.

12. in the `ReversiBoard` class modify the `horizontalResizeRelocate` method to set a new `x2` value on each horizontal line. You should also modify the associated `translate` objects of each horizontal line to space out the Y values of all lines by `cell_height`. (1%)

At this point you should see evenly spaced horizontal lines dividing your cyan background into 8 horizontal regions.

13. in the `ReversiBoard` class modify the `verticalResizeRelocate` method to set a new `y2` value on each horizontal line. You should also modify the associated `translate` objects of each horizontal line to space out the X values of all lines by `cell_width`. (1%)

At this point you should see evenly spaced vertical lines, combined with horizontal lines that divide your cyan background into an 8x8 grid.

in the `ReversiBoard` class modify the `pieceResizeRelocate` method to resize each `ReversiPiece` to be `cell_width` by `cell_height` and relocate each `ReversiPiece` to be a certain number of `cell_width`'s across and a certain number of `cell_height`s down. (2%)

in the `ReversiPiece` class modify the constructor to (4%)

take a copy of the passed in player integer

initialise `piece` and `t`, add `t` as a transform of `piece`.

If the player is player 1 then set the fill colour of the piece to be white, player 2 should be set to black then add the piece to the list of children of the ReversiPiece.

If the player is player 0 then this piece should be set to be invisible

in the ReversiPiece class modify the `resize()` method to call the superclass method and set the center and radius of the piece to be half the width and half the height of the new width and height that was passed in. Note that there are separate radius values in the X and Y directions. (1%)

in the ReversiPiece class modify the `relocate()` method to call the superclass method and change the x and y values on t. (1%)

After this point you should get the initial screenshot on the left.

in the ReversiPiece class modify the `swapPiece()` method to change a white piece to a black piece and vice versa (2%)

in the ReversiPiece class modify the `setPiece()` method to change the colour of a piece depending on the value passed in. If the type is zero the piece should be invisible otherwise it should be set to be visible. (4%)

in the ReversiPiece class modify the `getPlayer()` method to return the player value. (1%)

in the ReversiBoard class modify the `getPiece()` method to return the piece that is at the position (x,y) if the indices are out of bounds then return -1 to indicate that this position does not exist. (1%)

in the ReversiBoard class modify the `placePiece()` method to (9%)

determine which cell the user has clicked on. i.e. it should take x and y and convert it to a value in the range [0,8) (for the sake of this assignment we will call these converted values cx and cy)

if the game is not in play then return immediately

if the ReversiPiece at (cx,cy) does not have a type of zero then return immediately.

Determine what pieces surround the cell at (cx,cy), if there is not an adjacent opposing piece then return immediately.

Determine if a reverse can be made in any direction. If not then return immediately.

If all the preceding tests pass then place a piece on the board and reverse in every direction possible. Update the scores swap the players and see if the game is finished.

Print the scores to console and state which player is next to place a piece

in the ReversiBoard class modify the determineSurrounding() method to populate the surrounding array with all of the pieces that surround the cell (cx,cy). Use the getPiece() method to avoid ArrayIndexOutOfBoundsException exceptions (2%)

in the ReversiBoard class modify the adjacentOpposingPiece() method to determine if any of the surrounding cells of (cx,cy) have an opposing piece in them. If you find one then return true otherwise return false. (4%)

in the ReversiBoard class modify the isReverseChain() method to see if the direction (dx, dy) contains a chain that can be reversed by a piece in (cx, cy). The values dx and dy have a range of [-1,1] and indicate the direction in which a reverse should be checked. It is assumed that (cx + dx, cy + dy) contains an opposing piece as checked by the adjacentOpposingPiece method above. A valid reverse chain is one where there is one or more opposing pieces in an unbroken sequence, terminated by one of the current player's pieces. If this condition is satisfied then return true, otherwise return false. (3%)

in the ReversiBoard class modify the determineReverse() method to check if there is a reverse chain in any of the eight directions. If one or more reverse chains exist then return true otherwise return false. Note that you should not stop after you find the first chain, you should check all 8 directions always. The results of each check should be stored in the can_reverse array. (6%)

in the ReversiBoard class modify the reverseChain() method to take a position (cx,cy) and a direction (dx,dy). You should reverse all opposing pieces until you reach a piece belonging to the current player. It is assumed that the can_reverse array has been properly set from the determineReverse() method. (2%)

in the ReversiBoard class modify the placeAndReverse() method to place the piece at (cx, cy) and reverse in all possible directions. (5%)

in the ReversiBoard class modify the swapPlayers() method to swap the values of current_player and opposing (1%)

in the ReversiBoard class modify the updateScores() method to count the number of pieces belonging to both players (5%)

in the ReversiBoard class modify the determineWinner() method to compare the scores of both players and determine the winner. The winner is the player with the most pieces. As there is an even number of cells on the board it is possible that a draw may occur. Print out messages to the console stating which player has won and state that the game has ended. (5%)

in the ReversiBoard class modify the canMove() method to determine if the current player can make a valid move on the board. This will require you to check every empty cell. If a move is possible then return true immediately otherwise return false if no move is found. (5%)

in the ReversiBoard class modify the determineEndGame() method to determine if the game has ended. The game has ended if one of the following conditions occur. If any of these occur then determine the winner and set in_play to false (5%)

there are no empty cells left on the board

one player has lost all of their pieces

if the current player cannot make a move then switch to the opposing player. If the opposing player cannot make a move either then the game has ended. If the opposing player can make a move keep playing the game.

Notes:

You will be given two and a half weeks to do this assignment. The deadline for this assignment will be the 14th of November 2017 at 23:55. You are required to submit a single archive file to the moodle that has a filename of <lastname>_<firstname>_<student number>_<IDE>.zip In that zip file I should find a single folder containing your entire project and it should be named <lastname>_<firstname>_<studentnumber>

The Javafx book will only give you the pieces to set event handlers and to do basic drawing. You are required to generate the rest of the application yourself. It will be a long time before you will see if your application is working correctly. As a suggestion whenever you add small pieces of code you should test them will simple print statements to the console (or use a debugger) to ensure that you are recieving events correctly. Note that there are penalties that will be applied should your submission fail to meet certian standards. These are detailed in the first set of lecture notes. But as a recap in short they are

- -30% should your code fail to compile
- -10% should your archive be of a non permitted format
- it should be one of zip/rar
- and should have the filename given above
- -10% for a wrong project name

In addition to the note above that states that you should develop this from item 1 through to item 33 it should be noted that items 25 and 26 should be worked on together at the same time, as well as items 27 and 28.

HGP Sample Class Tests

MCQ1 – graded

1. A cold wait occurs in an event loop when
Select one:

- a. The program is rendering components.
- b. Your computer crashes
- c. There are two events attempting to happen at the same time.
- d. There is no more events in the events list and the application has no further rendering to perform.

2. Before cold wait
Select one:

- a. Keyboards and mice became unresponsive.
- b. The CPU was less heavily used.
- c. GUI applications would continuously poll the keyboard and mouse for events many times in a second.
- d. Battery powered devices lasted longer.

3. In the event driven programming model which of the following is TRUE.
Select one:

- a. The running of the program is determined by the generation of events by the user.
- b. We create a class. Write a main method. Run the program. Wait for the code in the main method to finish.
- c. The timing of the user generated actions must be known in advance of running the program.
- d. The program follows a specified execution path and ignores any events generated by the user.

4. Redisplay events result in _____
Select one:

- a. the GUI to rerender itself on the screen.
- b. the last error message to be displayed.
- c. the monitor to degauss.
- d. the last input by the user to be displayed.

5. The event loop follows the following steps.
Select one:

- a. The user clicks the button and the event loop wakes up. The loop determines which button was clicked. The loop goes into cold wait.

- b. The user clicks the button and the event loop wakes up. The loop executes all event handlers. The application carries out actions in response to the events.
- c. The user clicks the button and the event loop wakes up. The loop determines which button was clicked and will call the appropriate event handler. The application carries out actions in response to the events.
- d. The user clicks the button and the event loop wakes up. The application carries out actions in response to the events.

6. The event list is a _____ .

Select one:

- a. LIFO datastructure
- b. FIFO datastructure
- c. Stack
- d. FIDO datastructure

7. The general structure of a busy wait loop is

Select one:

- a.

```
while(true){
    if(peekEvent()){
        x = nextEvent()
        if(x == quit){
            break
        }
        process x
    }
}
```
- b.

```
if(peekEvent()){
    x = nextEvent()
    if(x == quit){
        break
    }
    process x
}
else{
    idle()
}
```
- c.

```
while(true){
    if(peekEvent()){
        x = nextEvent()
```

```

        if(x == quit){
            break
        }
        process x
    }
    else{
        idle()
    }
}
d. while(false){
    if(peekEvent()){
        x = nextEvent()
        if(x == quit){
            break
        }
        process x
    }
    else{
        idle()
    }
}

```

8. Which of the following is FALSE when speaking about the implementation of the event loop.

Select one:

- a. When working with Win32 APIs or X Windows you can write event loops.
- b. No GUIs permit you to write your own event loop.
- c. Some GUIs permit you to write your own event loop
- d. With GUIs this is usually embedded inside the GUI toolkit itself and is not exposed to the developer.

9. Which of the following is NOT a valid mouse event

Select one:

- a. Mouse clicked
- b. Mouse pressed
- c. Mouse typed
- d. Mouse released

10. Which of the following is NOT true

Select one:

- a. The only time your GUI will become active is when the user causes an event in your application.

- b. If the control makes no change to the application state or does not inform the user then it should not be there.
- c. Every single control should have an effect on the overall application state or should display information to the user
- d. The GUI should remain hidden from view unless the user has recently interacted with it.

MCQ2 - graded

1. GUI is short for_____.

Select one:

- a. Graphics User Interface
- b. Graphical User Interface.
- c. Graphical Users Interfaces.
- d. Grapical User Interaction.

2. GUIs should be informative, which means they _____.

Select one:

- a. Require the use of a manual
- b. Are easy to understand and do not require explicit instructions.
- c. Are as complicated as possible
- d. Tell you exactly what they are doing

3. GUIs use a different model of programming called the _____ model.

Select one:

- a. system driven
- b. user driven
- c. task driven
- d. event driven

4. An _____ is a block of code that executes in response to a mouse click.

Select one:

- a. Event Loop
- b. Event Construct
- c. Event Handler
- d. Event Timer

5. When an event occurs your GUI application should first determine _____.

Select one:

- a. Which user initiated the event
- b. What type of event occurred
- c. What time it is
- d. The execution time of loop

6. If I attempt to divide by zero, in the worse case scenario if this possibility is not catered for, it may result in ____.

Select one:

- a. a calculation not being attempted
- b. an application crash
- c. an incorrect result
- d. a system crash

7. A normal stack data structure is ____.

Select one:

- a. FOLI
- b. LIFO
- c. FIFO
- d. FOFI

8. You render a stack of overlapping windows ____.

Select one:

- a. in random order
- b. from bottom to top
- c. from top to bottom
- d. from the middle to the top

9. Rendering overlapping windows is similar to how an artist constructs a painting as he will draw the ____ first, then the ____ ground and finally the ____ground.

Select one:

- a. middle,back,fore
- b. middle,fore,back
- c. back,middle,fore
- d. foreground,middle,back

10. To determine what window (GUI) owns a given event we use the stacking order in conjunction with _____ tests.

Select one:

- a. bounding box
- b. bounding package
- c. bounding rectangle
- d. bounding square

11. The window at the top of the stack will ____.

Select one:

- a. guaranteed to be in control of keyboard events.
- b. guaranteed to be in control of all mouse events.
- c. guaranteed to be in control of mouse events.
- d. guaranteed to be in control of all keyboard and mouse events.

12. The first step in determining the window (GUI) that owns a mouse event is _____.

Select one:

- a. determine if the user clicked the left or the right hand mouse button
- b. determine what window owns the event
- c. determine the global coordinates of where the user clicked on the screen
- d. determine if the user used the scroll wheel.

13. Only a limited number of layouts are used in Widgets, the reason for this is that the visual system loves _____ and patterns.

Select one:

- a. disorder
- b. order
- c. chaos
- d. direction

14. One of the design guidelines, you should use when designing a GUI, is to rely on _____ rather than recall.

Select one:

- a. retention
- b. the mind map
- c. memory
- d. recognition

15. Regarding Visual Perception, the overall _____ of a sentence can also influence the words we see.

Select one:

- a. rendering
- b. capacity
- c. comprehension
- d. wording

16. Perception is influenced by goals and plans for the future. Anything not related to goals is filtered out _____.

Select one:

- a. subsentiment
- b. at the end

- c. subknowingly
- d. subconsciously

17. The German word for shape is _____.

Select one:

- a. Destalt
- b. Hestalt
- c. Gestalt
- d. Festalt

18. The _____ principle relates to the relative distance between objects in a display, which affects our perception of whether and how the objects are organised into subgroups

Select one:

- a. nearness
- b. closeness
- c. opposition
- d. proximity

19. The _____ principle relates to objects that are like one another appear grouped (all other things being equal).

Select one:

- a. similarity
- b. sameness
- c. parallel
- d. likeness

20. The human mind always tries to resolve ambiguity OR fill in missing data to perceive whole objects. This is called the _____ principle.

Select one:

- a. discontinuos
- b. enduring
- c. continuity
- d. persistance

MCQ3 - graded

1. There are 3 main classes of events in JavaFX, namely mouse, keyboard and _____ events.

Select one:

- a. touch
- b. tap
- c. feel
- d. tip

2. There are 3 events associated with keys on a keyboard, key _____, key released and key typed.

Select one:

- a. touched
- b. tapped
- c. pressed
- d. pushed

3. Mouse events have a single pointer and can be of _____ event types

Select one:

- a. 2
- b. 4
- c. 3
- d. 1

4. For both key and mouse events, there may be modifier keys (_____, ALT and SHIFT) active.

Select one:

- a. insert
- b. control
- c. backspace
- d. enter

5. The top left hand corner of a widget, is considered the origin of a widget with coordinates (____, ____).

Select one:

- a. 1,0
- b. 0,0
- c. 0,1
- d. 1,1

6. A visual _____ enables to focus on relevant information. This for example includes, dividing information into distinct sections.

Select one:

- a. graph
- b. ladder
- c. hierarchy
- d. system

7. If presenting numerical information to an end user long numbers should be _____.

Select one:

- a. be discarded
- b. should be presented as a single value
- c. not be used
- d. broken into smaller groups of numbers

8. People read from _____ in the English language.

Select one:

- a. back to front
- b. bottom to top
- c. right to left
- d. top to bottom

9. The more terse and _____ the presentation of information the more quickly and easily users can understand and comprehend it.

Select one:

- a. structured
- b. disorganised
- c. dissimilar
- d. unstructured

10. The book "" _____ "" is a book by Jeff Johnson that discusses how the human mind and perception is related to GUI design.

Select one:

- a. Designing with the human in mind
- b. Designing with perception in mind
- c. Designing with the brain in mind
- d. Designing with the mind in mind

11. In GUI design, design rules often describe _____ and not actions.

Select one:

- a. interpretation
- b. symptoms

- c. goals
- d. rules

12. When designing GUIs there will be tradeoffs and compromises, in order to determine the right balance it will (generally) be necessary to _____.

Select one:

- a. rely on your own opinion
- b. get users to carry out a lot of testing
- c. ask your friends what they think
- d. keep the design as minimalist as possible

13. Good interface design, requires a little knowledge of ____ psychology.

Select one:

- a. attachment
- b. relationship
- c. cognitive
- d. conflict

14. A normal stack data structure is _____.

Select one:

- a. LIFO
- b. FIFO
- c. FOLI
- d. FOFI

15. A stack for a windows manger differs from a normal stack data structure as it must be possible that an element _____ can be moved to the top of the stack.

Select one:

- a. anywhere in the stack
- b. in position of 2 (only)
- c. from the bottom of the stack (only)
- d. from the middle of the stack (only)

16. The window that has _____ (i.e. has the users attention) is at the top of the stack.

Select one:

- a. concentration
- b. focus
- c. centre
- d. fixation

17. Redisplay events cause _____

Select one:

- a. the last error message to be displayed.
- b. the last input by the user to be displayed.
- c. the monitor to degauss.
- d. the GUI to rerender itself on the screen.

18. A ____ wait is where the application process is put into a sleep state, such that it will not consume CPU cycles

Select one:

- a. lukewarm
- b. hot
- c. warm
- d. cold

19. GUIs should be informative, which means they _____.

Select one:

- a. Require the use of a manual
 - b. Tell you exactly what they are doing
 - c. Are as complicated as possible
 - d. Are easy to understand and do not require explicit
- This website makes use of cookies to enhance browsing experience and provide additional functionality. Details

20. To a developer / end user GUIs consist of a single window and a selection of different _____.

Select one:

- a. Input boxes
- b. Controls
- c. Dialog Boxes
- d. Buttons

This website makes use of cookies to enhance browsing experience and provide additional functionality. Details

MCQ4 - graded

1. The majority of the resolution of the human eye is at the center, where the _____ resides.

Select one:

- a. fobea
- b. fopea
- c. fomea
- d. fovea

2. Clarks research in 1998 had a computer tracking the eye position of users, text was displayed where the user was focused with random noise elsewhere. The study found that the _____.

Select one:

- a. the user could read normally and was not distracted by the noise.
- b. user could read the text but at a much slower pace
- c. user could not read the text
- d. the user was completely distracted by the noise

3. The area in our eye where there is zero visual information is called the _____.

Select one:

- a. favoe
- b. blind spot
- c. weak spot
- d. vision free spot.

4. A humans vision is optimized to detect _____ not brightness.

Select one:

- a. contrast
- b. tinting
- c. depth
- d. colour

5. The smaller or thinner objects are the _____ it is to distinguish colours.

Select one:

- a. easier
- b. harder
- c. simpler
- d. more straight-forward

6. The most common form of colour blindness is _____.

Select one:

- a. yellow-blue
- b. red-green
- c. red-blue
- d. white-orange

7. Humans are wired for language but not for _____.

Select one:

- a. reading
- b. visualizing
- c. seeing
- d. speaking

8. Feature driven reading can be disrupted by _____.

Select one:

- a. displaying text on a noisy background.
- b. displaying text on a clear background.
- c. displaying text using good colour contrast
- d. displaying text in an easy to read font.

9. Humans are goal oriented which means they want to _____.

Select one:

- a. complete tasks within a timeframe of 30 seconds
- b. complete tasks within a timeframe of 5 minutes
- c. complete tasks in the minimum amount of time possible
- d. complete tasks in the maximum amount of time possible

10. The top left hand corner of a widget, is considered the origin of a widget with coordinates (__, __).

Select one:

- a. 1,0
- b. 0,0
- c. 1,1
- d. 0,1

11. There are two types of rendering mode with regards to computer graphics, _____ mode and retained mode.

Select one:

- a. asap
- b. instant
- c. here and now
- d. immediate

12. In JavaFX, scale, rotate and translate are examples of _____.

Select one:

- a. rearrangements
- b. transpositions
- c. reversals
- d. transformations

13. Humans tend to parse complex scenes or objects in a way that will reduce visual complexity. This is called the _____ principle.

Select one:

- a. evenness
- b. proportion
- c. balancing
- d. symmetry

14. The human visual system automatically tries to complete partially drawn figures so that they are perceived as whole objects. This is called the _____ principle.

Select one:

- a. closeness
- b. closing
- c. closure
- d. cessation

15. _____ is concerned with objects that move.

Select one:

- a. Common Fate
- b. Common Knowledge
- c. Common Principles
- d. Common Objective

16. One of the design guidelines, you should use when designing a GUI, is to always _____ short term load.

Select one:

- a. minimise
- b. maximise

17. The German word for shape is _____.

Select one:

- a. Hestalt
- b. Destalt
- c. Gestalt

d. Festalt

18. The human mind always tries to resolve ambiguity OR fill in missing data to perceive whole objects. This is called the _____ principle.

Select one:

- a. continuity
- b. enduring
- c. persistence
- d. discontinuos

19. A normal stack data structure is _____.

Select one:

- a. FOLI
- b. FOFI
- c. FIFO
- d. LIFO

20. A stack for a windows manger differs from a normal stack data structure as it must be possible that an element _____ can be moved to the top of the stack.

Select one:

- a. in position of 2 (only)
- b. from the bottom of the stack (only)
- c. anywhere in the stack
- d. from the middle of the stack (only)

MCQ5 - graded

1. Activating a memory involves reactivating the same neural _____.

Select one:

- a. brain wave
- b. formation
- c. pattern
- d. sensation

2. The information stored in short term memory is _____.

Select one:

- a. small and limited
- b. unknown
- c. large and detailed

d. large and limited

3. The resolution at the periphery of the human eye is _____.

Select one:

- a. super high definition
- b. high definition
- c. the same as a digital camera
- d. only a few dots per inch

4. Images on the periphery of our vision are ____ quality.

Select one:

- a. good
- b. low
- c. excellent
- d. high

5. A humans vision is optimized to detect _____ not brightness.

Select one:

- a. depth
- b. tinting
- c. contrast
- d. colour

6. If a problem occurs it is better to tell an end user _____ than stating what the problem is.

Select one:

- a. to reboot
- b. disconnect from the network
- c. to shutdown
- d. how to fix the problem

7. Text in applications should be _____.

Select one:

- a. difficult to read
- b. text should not be on a plain background
- c. be displayed on a patterned background
- d. text should be displayed on a plain background

8. For both key and mouse events, there may be modifier keys (____, ALT and SHIFT) active.

Select one:

- a. enter

- b. insert
- c. control
- d. backspace

9. The top left hand corner of a widget, is considered the origin of a widget with coordinates (__, __).

Select one:

- a. 1,1
- b. 1,0
- c. 0,1
- d. 0,0

10. A visual _____ enables to focus on relevant information. This for example includes, dividing information into distinct sections.

Select one:

- a. ladder
- b. hierarchy
- c. system
- d. graph

11. The human visual system automatically tries to complete partially drawn figures so that they are perceived as whole objects. This is called the _____ principle.

Select one:

- a. closeness
- b. closing
- c. cessation
- d. closure

12. In GUI design, design rules often describe _____ and not actions.

Select one:

- a. symptoms
- b. goals
- c. interpretation
- d. rules

13. The German word for shape is _____.

Select one:

- a. Hestalt
- b. Destalt
- c. Gestalt
- d. Festalt

14. A stack for a windows manger differs from a normal stack data structure as it must be possible that an element _____ can be moved to the top of the stack.

Select one:

- a. from the middle of the stack (only)
- b. anywhere in the stack
- c. in position of 2 (only)
- d. from the bottom of the stack (only)

15. You render a stack of overlapping windows _____.

Select one:

- a. from bottom to top
- b. in random order
- c. from top to bottom
- d. from the middle to the top

16. When the size of a window changes the _____ event is called.

Select one:

- a. restructure
- b. rerender
- c. resize
- d. redraw

17. In JavaFX, what is the name of the event type associated with a mouse button is _____.

Select one:

- a. OnTouch
- b. OnTip
- c. OnTick
- d. OnClick

18. HCI is short for _____.

Select one:

- a. Humane Computer Interaction
- b. Human Computer Interaction
- c. Human Computer Intraface
- d. Human Computer Interface

19. GUIs are primarily concerned with _____.

Select one:

- a. None of the above
- b. Making the interface as complex as possible

- c. Making things more complicated for the end user
- d. Simplifying things for the end user

20. GUIs should be informative, which means they _____.

Select one:

- a. Are as complicated as possible
- b. Are easy to understand and do not require explicit instructions.
- c. Require the use of a manual
- d. Tell you exactly what they are doing

MCQ6 – graded

1. GUIs are primarily concerned with _____.

Select one:

- a. Making the interface as complex as possible
- b. Simplifying things for the end user
- c. None of the above
- d. Making things more complicated for the end user

2. GUIs should be intuitive, which means they _____.

Select one:

- a. Are easy to understand and do not require explicit instructions.
- b. Require the use of a manual
- c. Are as complicated as possible
- d. Are stripped down as much as possible.

3. The model of programming in which a program runs from beginning to end is not suitable for GUIs as they must run _____.

Select one:

- a. For a minimum of 5 minutes
- b. For a specified period of time.
- c. For a longer period of time
- d. Indefinitely

4. A cold wait occurs in an event loop when

Select one:

- a. There are two events attempting to happen at the same time.
- b. Your computer crashes
- c. There is no more events in the events list and the application has no further rendering to perform.
- d. The program is rendering components.

5. When the size of a window changes the _____ event is called.

Select one:

- a. rerender
- b. resize
- c. restructure
- d. redraw

6. In GUI applications, If you want the end user to select a single option, a _____ should be used.

Select one:

- a. RadioButton
- b. Textbox
- c. MenuList
- d. DropDown

7. The first step in determining the window (GUI) that owns a mouse event is _____.

Select one:

- a. determine if the user used the scroll wheel.
- b. determine if the user clicked the left or the right hand mouse button
- c. determine the global coordinates of where the user clicked on the screen
- d. determine what window owns the event

8. The _____ principle relates to the relative distance between objects in a display, which affects our perception of whether and how the objects are organised into subgroups

Select one:

- a. proximity
- b. nearness
- c. closeness
- d. opposition

9. If presenting numerical information to an end user long numbers should be _____.

Select one:

- a. not be used
- b. should be presented as a single value
- c. broken into smaller groups of numbers
- d. be discarded

10. Humans tend to parse complex scenes or objects in a way that will reduce visual complexity. This is called the _____ principle.

Select one:

- a. proportion
- b. balancing
- c. eveness
- d. symmetry

11. For both key and mouse events, there may be modifier keys (____, ALT and SHIFT) active.

Select one:

- a. enter
- b. insert
- c. backspace
- d. control

12. There are two types of rendering mode with regards to computer graphics, _____ mode and retained mode.

Select one:

- a. asap
- b. immediate
- c. here and now
- d. instant

13. In JavaFX, scale, rotate and translate are examples of _____.

Select one:

- a. transpositions
- b. transformations
- c. rearrangements
- d. reversals

14. The smaller or thinner objects are the _____ it is to distinguish colours.

Select one:

- a. easier
- b. more straight-forward
- c. harder
- d. simpler

15. The most common form of colour blindness is _____.

Select one:

- a. red-green
- b. yellow-blue
- c. white-orange
- d. red-blue

16. In relation to GUI design, Error information should _____.

Select one:

- a. be located anywhere on the GUI
- b. be close to where the user entered input
- c. be far from where the user entered input
- d. be in the smallest font size possible so its unreadable

17. When presenting error information it makes it easier for the user to see it if an error _____ is used.

Select one:

- a. dialog
- b. menu list
- c. box
- d. symbol

18. One of the core implications for GUI design of short term memory is that UIs should _____.

Select one:

- a. help users remember everything
- b. help users forget everything
- c. help users remember essential information
- d. help users forget essential information

19. Memories in the brain are stored in a _____ manner.

Select one:

- a. distributed
- b. unknown
- c. condensed
- d. scattered

20. Which of these statements is true regarding long term memory.

Select one:

- a. stored information is equivalent to a HD recording
- b. all information is stored perfectly
- c. there is no information lost after it is stored
- d. it is not perfect

Project

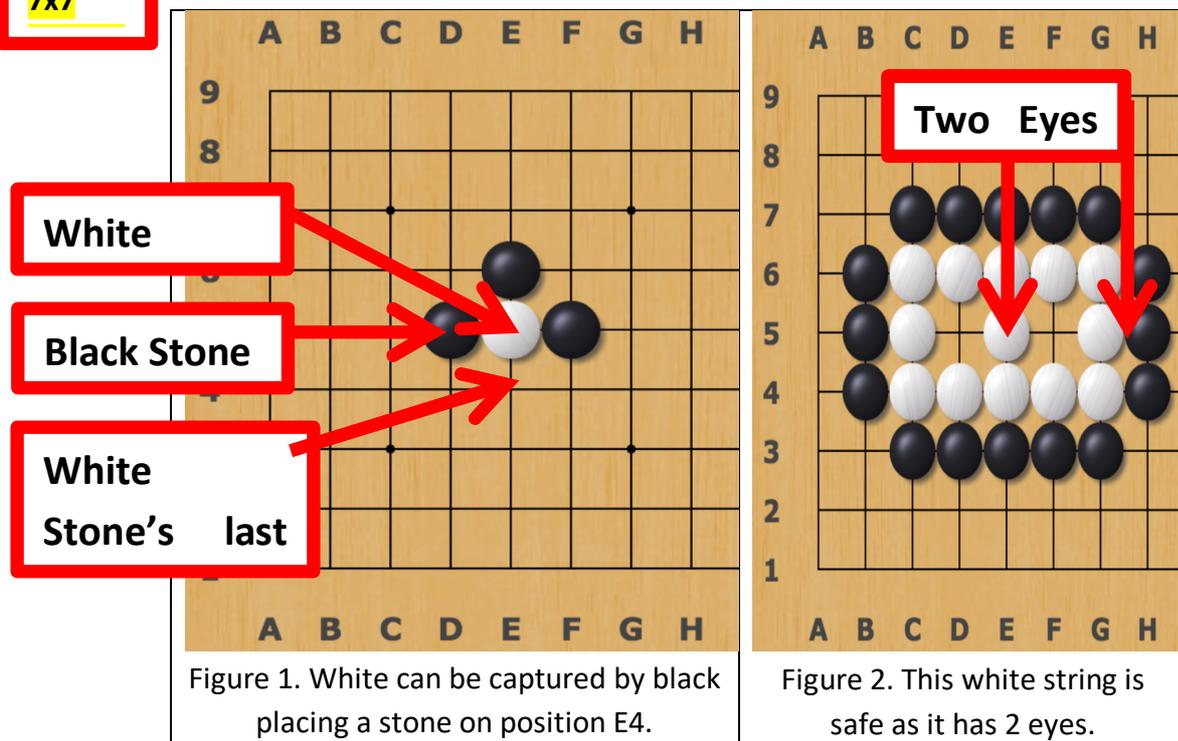
1. Introduction.

N.B.:

- Please read the project description in full before you start coding. Failure to do so will result in you missing out on important information and reducing your ability to avail of marks. Please attempt to work in line with the file structure provided. Very large departures from this template may result in loss of marks.

x9
board
use a
7x7

In this project, you will be tasked with building a fully working game of Go.



2. Resources

- **Moodle**
 - [This Description](#)
- **From your lab work**
 - [Example20](#). The solution is heavily based on this and it is essential to have a detailed working knowledge of Example20 before proceeding with this project.
 - [Reversi](#). The skills you learned in the design of your Reversi application will stand to you. You may like to begin with a working version of Reversi.

3. Explanation of the Game

Has been omitted to reduce documentation length.

4. Comparison between Go and Xs & Os

Comparison between project and previous course work has been omitted to reduce documentation length.

5. Deliverables

- The full list of deliverables is
 1. This document with the following sections completed
 - a. Summary of Division of Work Word
 - b. Screen Shots of Working Application
 2. Folder containing all the java source files, images etc. relevant to your application
- You are **both** required to submit a single **identical** archive (zip) file to Moodle that has a filename of **<lastname1>_<firstname1>_<studentnumber1>_<lastname2>_<firstname2>_<studentnumber2>_go.zip** If the files are not identical (MD5 hash etc.) then you will be called to interview.
- In that zip file, I should find a single folder containing all your deliverables and it should be named **<lastname1>_<firstname1>_<studentnumber1>_<lastname2>_<firstname2>_<studentnumber2>_go**

N.B. work is expected to **BE YOUR OWN OR YOUR PARTNERS**. **PLAGIARISM** will not be tolerated under any circumstances.

6. Tasks & Marking

Below is a list of tasks which must be completed and their associated marks

Task Number	Description	% of Grade
1	Generate an application that is forced to be square in size. It should display a full Go board of side size 7.	10
2	Add code and menus/buttons/labels to your application to <ol style="list-style-type: none">a) How to play your game including rulesb) Show how many prisoners each player has takenc) Show how much territory is controlled by a playerd) Show whose turn it ise) Allow player to passf) Allow the game to be reset	20 (6 x 3.3)
3	Implement placement of stones using mouse clicks.	10
4	Implement placement of stones in valid locations	10

	only – suicide rule	
5	Implement placement of stones in valid locations only – KO rule	10
6	Implement capture of stones – single stone	10
7	Implement capture of stones – multiple stone	10
8	Implement winner detection, the game should then end immediately with an appropriate notification. Two passes.	10
9	Additional features select one of the following: <ul style="list-style-type: none"> - 2 timers 1 for each player to implement speed Go. Each player should have 2 minutes to make moves. The 1st players timer should start to count down when the game is started, 2nd players timer counts down when 1st player has completed his move and so on. If a player runs out of time then they will lose the game. - Animation of moves - Implement a handicapping system - Other additional feature of your choice with similar complexity 	10
<p>Please Note that Marks will be Deducted for the following:</p> <ul style="list-style-type: none"> • Code that fails to compile -30% • Submission in wrong format. Your project should be submitted in zip format only -10% • Submission with the wrong filename -10% 		

7. Working in Pairs

Guidance on how learners can work more successfully in pairs has been omitted to reduce documentation length.

8. Suggested Development Plan

Guidelines on how to work of the 5 week period has been omitted to reduce documentation length.

9. Division of Work Log

Reporting guidelines on how to record the contribution of each partner to the project has been omitted to reduce documentation length.

10. Screen Shots of Working Application

Detail of requirement has been omitted to reduce documentation length.