Project report: Pinn Booranamaitree

My project is to write a Minesweeper game. The game should function properly and be easy to play without errors or bugs. The player should be able to follow the easy instructions of the game while having fun. The code should be written properly with correct conventions and use some advanced techniques in the game. That means I have to plan the project carefully, think about the relevant implications of the game, and do testing and trialing to ensure the best possible outcome.

# Part 1: Planning

# 1.1 Planning tools & techniques

When I start my project, the first thing I do is plan the whole project so I know how much work there is and how much I have to do to meet the deadline.

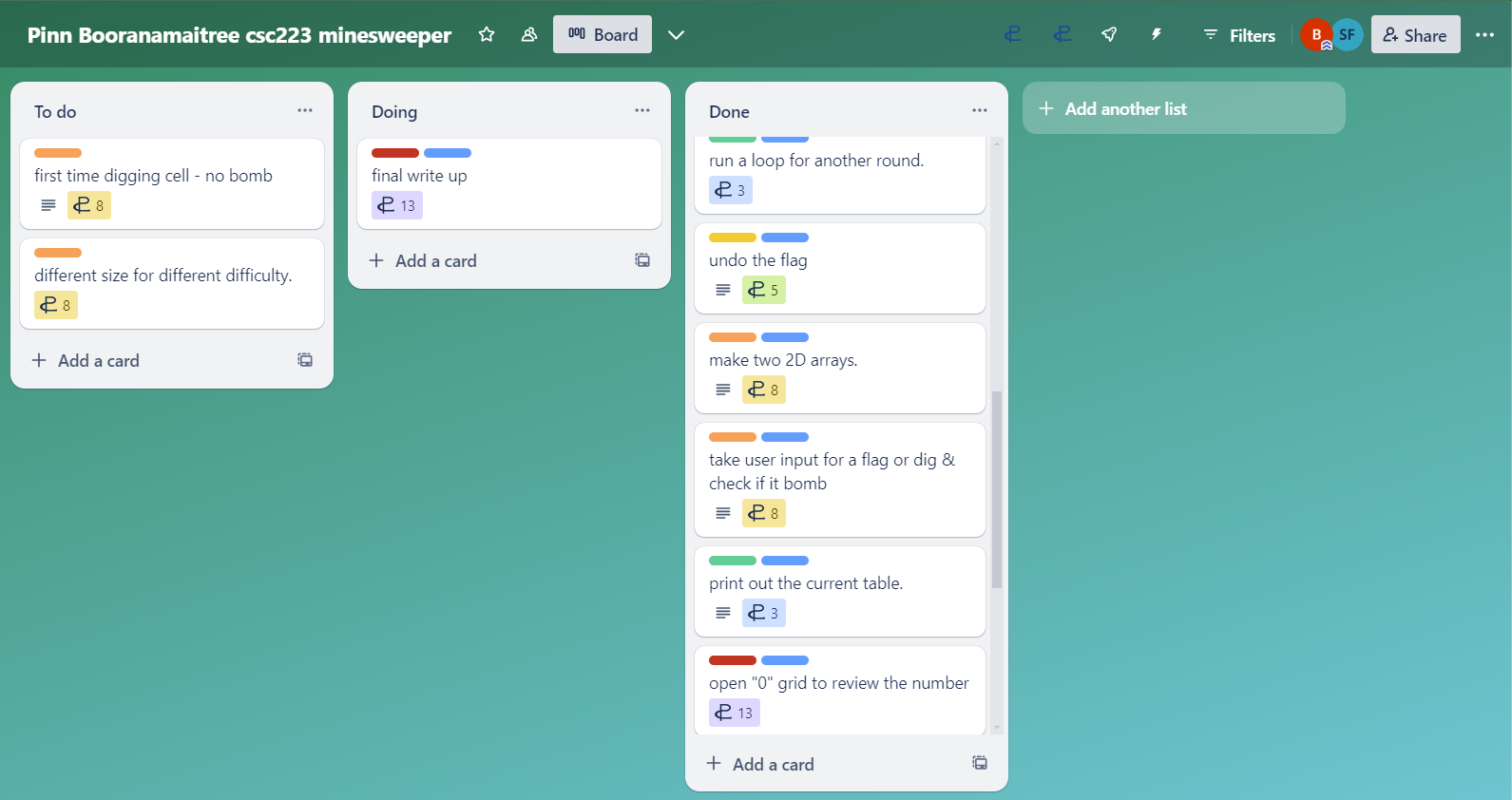
I use agile methodology to plan the project. The first thing I did was create a Kanban board using an online software called Trello. I use Trello because our teacher suggested this program to us and it is a software that has a Kanban board template that is easy to use and this software is also free. The term kanban comes from a Japanese word for *instruction card,* and it basically works by breaking down our project into little components and putting them into a card,the board itself has three columns to do, doing, and done. In Trello, we can assign each card a number indicating how much work is in that component, we can also add any label to the card such as the difficulty of the task or if it’s an MPV (minimum viable product) task, this will help us to visualize the amount of work effectively.

After I was done with the Kanban board I created a burn-down chart for the project. The chart will help me visualize my workflow and see if I am behind on the work or not. I created the chart by adding the total amount of work I have to do from the Kanban board, then splitting my entire term into 8 sprints and dividing the amount of work equally into each sprint.

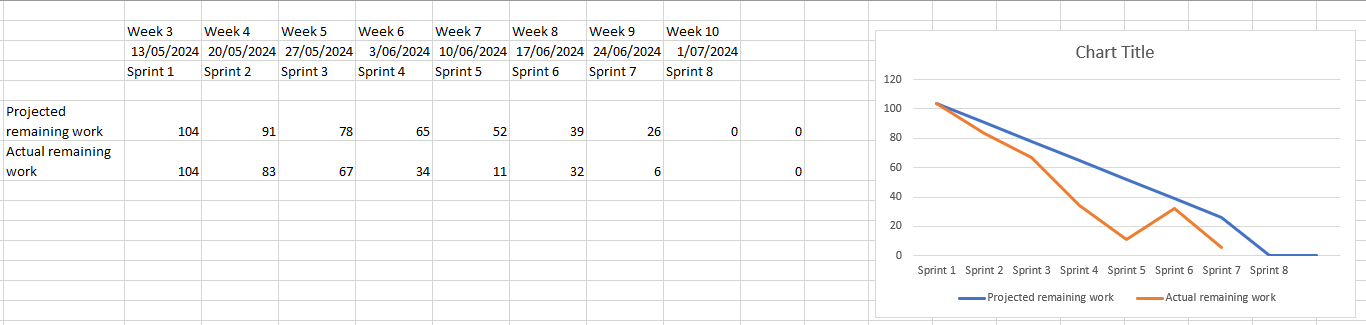
At the beginning of every week(sprint) I did a Kanban weekly commitment which is a form that I fill in, this includes what I did last week. Did I meet my goals? What are the next things I have to do? What is the problem that I struggle with and how did I address them? This helps me to keep track of all my work and have a clear plan for the week.

### 1.1.1 Actual use of the planning tools

This is my Kanban board Using Trello software. I have a lot of components that I break down and the amount of work ranges from 3, 5, 8, and 13 for a very big task. I added a label to every card so it is easier to visualize the task, I made it so that green is for the easy task, yellow is the medium task, orange is the big task, and red is the super big one.



This is my burn-down chart. The blue line is the path that I should be following in order to get the work done in time. There is a gradient change in sprint 8 because I am taking the last week of school so I have to adjust the burndown rate according to my deadline. The red line is the actual amount of work that I am able to do each week. It is going great at 1-5 sprint because I am going faster than the blue line but during sprint 6 I added more tasks because I forgot about all the writing parts which is also a lot.

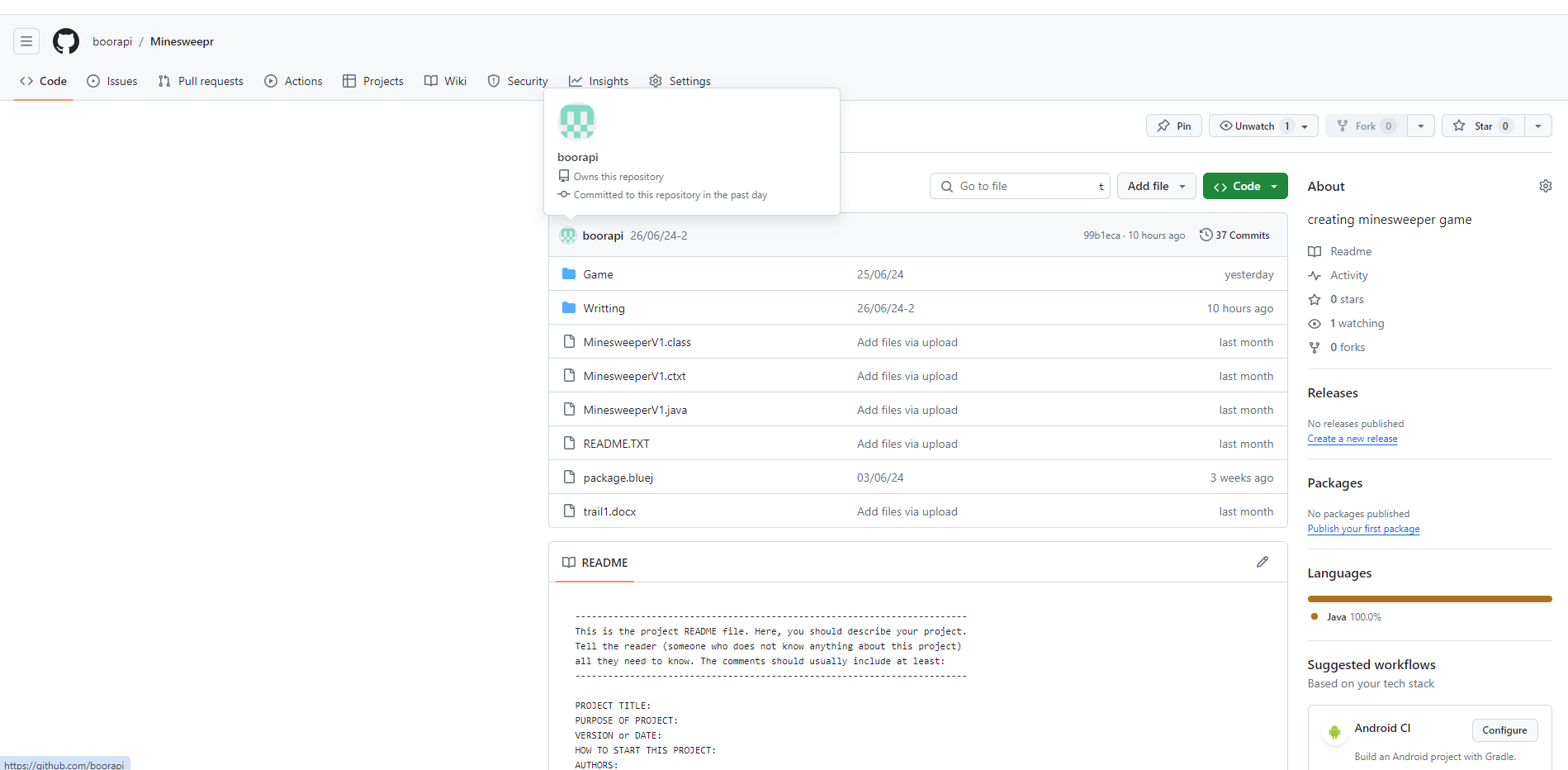


These are some of the examples of my weekly commitment. I did this very week to help me track the project so I know what is going on every week.

### 1.1.2 Version management

I use GitHub as my version control tool. GitHub is a version control software that is free for student so it is very good. GitHub helps me with my project a lot, it allows me to upload my code to online respiratory that I can access anywhere if I have internet which means I can work on my project at home on my computer and at school at the same time. Version control in GitHub is very helpful with my project, it allows me to commit my code to the respiratory and still have the old version of it saved in the history, which means that I can revisit any version of my code if there is something wrong or I can go back to my previous version if I do not like my current one. In GitHub when you do a commit it allows you to add any comments to that commit, I usually comment about what I did/finished in that version so I know what I have done in each commit, I also add a date in there. I went back to see the old commits all the time when there is something wrong so this is very helpful for me. In GitHub, I can also create a branch from the main, it allows me to copy the main branch and try some new codes on it, and when I am satisfied with the new part I can marge it to the main branch or if I do not like it, I can just delete that branch. Overall GitHub is a great version control tool for my project and it helps me with various things.



## 1.2 Project components

The game that I wrote is a game called Minesweeper. It is a logic-based game that features interactable tiles with hidden mines scattered throughout the board. The goal of the game is to clear the board without detonating any of the mines, with help from clues about the number of reboring mines in each field. If the player chooses to dig a tile that contains mine, they will lose but if the cell has no mine the hint number will be revealed and if that tile number is zero (which means that there are no mines around it) the tiles will expand out revealing all the empty cell and the number around it.

I did not use GUI in my game so I had to write a text-based game. The player will interact with the board tile by typing the x and y coordinates in and they can choose different options such as dig or flag.

### 1.2.1 Problem decomposition

When planning the project, I started by breaking the entire project into smaller components. I break the smaller components from three major sections that I know my game will need, the display (UI) part, the logic/calculation part, and the input-taking and user interaction with the game. By using Trello I wrote all the small components into a card and assigned each card the difficulty and the amount of work that needed to be done. I also added an MVP label to the card that is needed. MVP stands for minimum viable product which means the minimum work the game needs for it to work. This will help me to prioritize the task and will help the project to finished at the deadline. Trello uses Kanban structure so I can move the card around when it is in process, to do, or already done, from this I can track how much work I did each week and then I can use the data to plot my burndown chart.

This is more detailed about different sections and some of the components in it.

**Display (UI)**

10x10 board - a display board that the player should able to interact with. Refresh the screen -there should be a while loop for each turn and after the turn the display should be refreshing and the update display should be printed out. Instruction – Print the instructions out so the player knows what to do and what are the commands.

**Logic and calculation**

Count mine around each cell – a method that I passed in x and y coordinates then the method checks how many mines are around that tile. Revealing the cell when it is 0 – if the tile that passed in has no mines around it (0) the tile should expand out revealing all the 0 tile and the number around it. check win/lose condition – if the player digs a mine the game should end and tell the player that they have lost, if the player uses all the flags the game should end and the program should check for a win or lose.

**Input-taking and user interaction**

Take the input for the coordinates – ask the player for input for the x-axis as an integer between 1-10, ask the player for the input for the y-axis as a character from a - j then convert the character to an integer. Input for the command – print out all the command options and make an if-else statement for each command. Error checks the input – if there is a valid input make the program loop through until it get a valid input (this is not MVP).

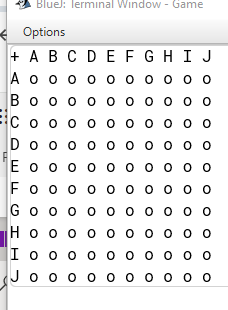
Trialing Document

|  |  |
| --- | --- |
| Date | What is the thing I am trialing |
| 14/05/24 | GUI of the game. |

*Copy and paste the table above before filling it out*

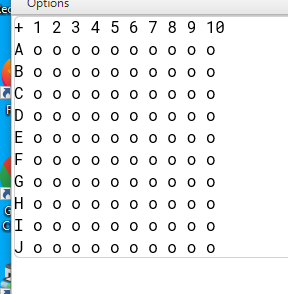
### Version A

### Coordinate x and y have the same label which is hard to understand.



### Version B

X and y coordinates have different types of label (alphabet and number).



Person: Mr. Fairhall (my teacher)

Date: 14/05/24

Comment: From version A if the user puts the coordinate of e.g. [D][C] the user might not know which grid they interacted with (because there are two possibilities DC or CD)

### Evaluation and next steps

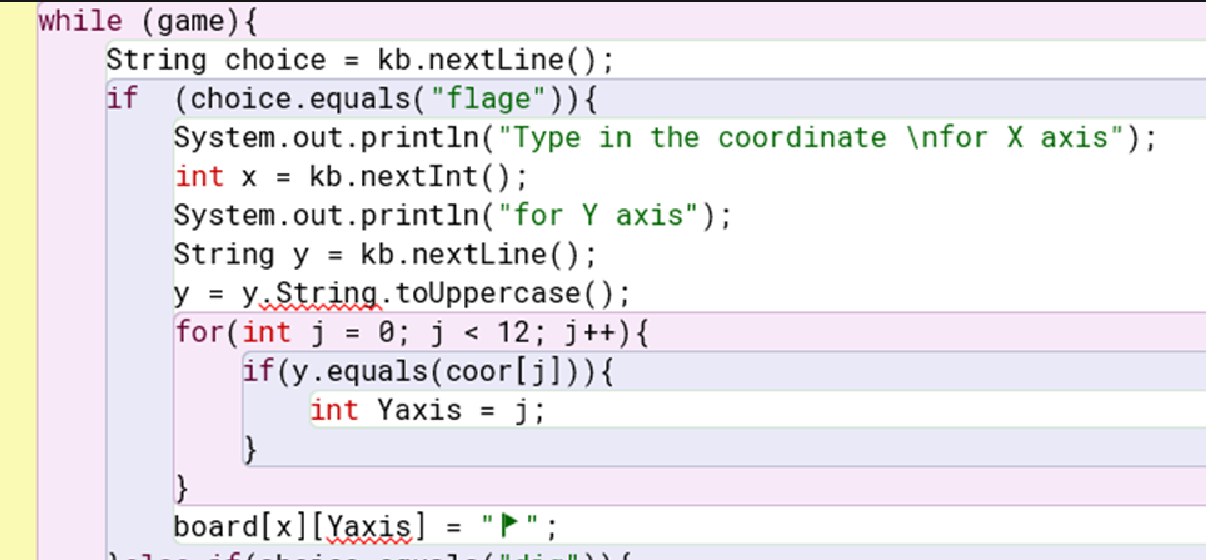
Based on this trialing and the feedback, I have decided to use version B because its easier to understand and use.

|  |  |
| --- | --- |
| Date | What is the thing I am trialing |
| 21/05/24 | Find out the Y coordinate as an integer for the arrays (because our y coordinate is alphabet) |

*Copy and paste the table above before filling it out*

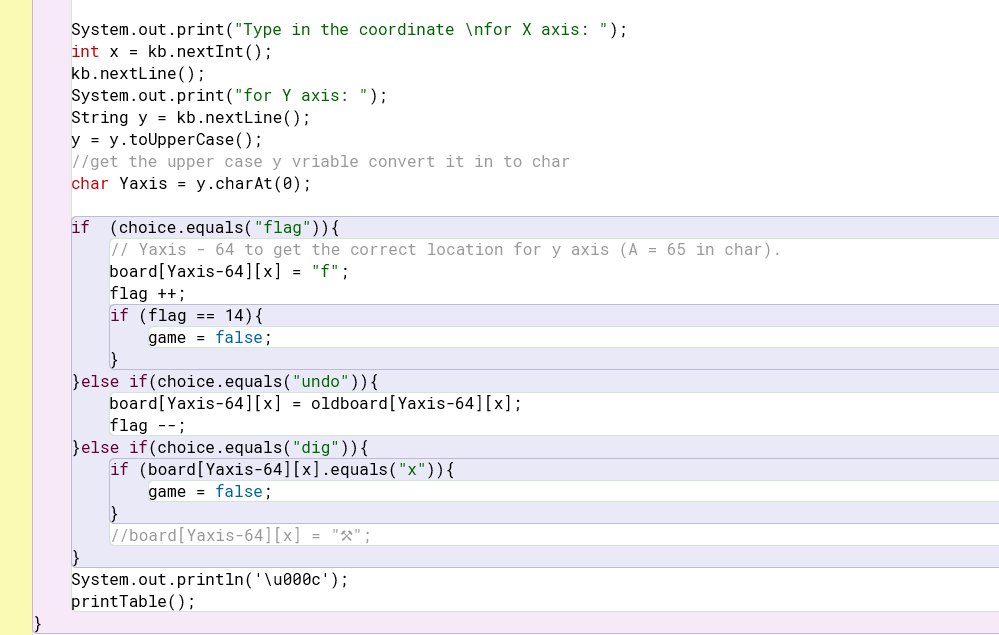
### Version A

To find the y coordinate I use a for loop to loop through the arrays of y coordinate [a, b, c, d…] and keep track of how many times we loop through [ j ] then set it to equal to the y axis coordinate which over complicate it. The better way to do it is to use char.



### Version B

Because char is an ASCII it has a value in int as well (A = 65) so I converted a string of y axis to char then minus it with 64 to get the y coordinate. Now it works fine with minimal code (compared with loop through the coordinate as string).



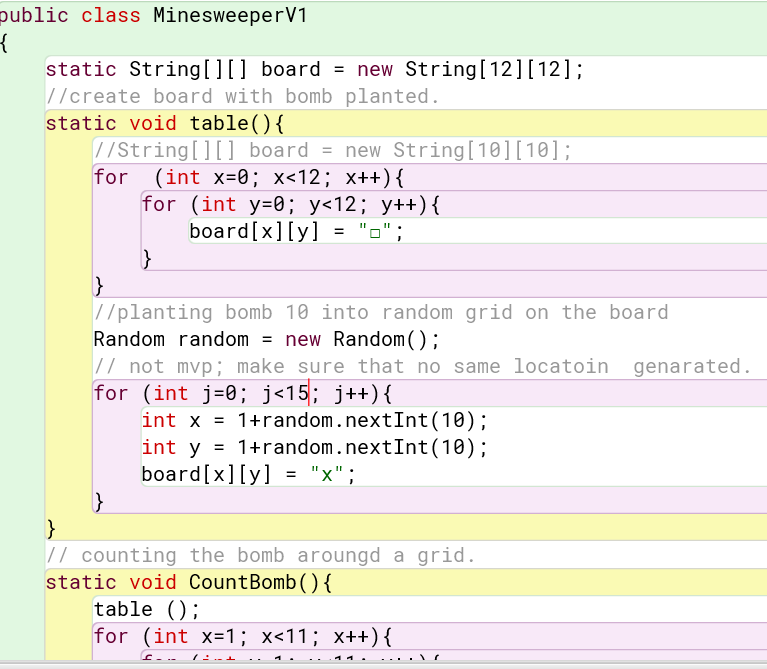
Evaluation and next steps

Based on this trial I decided to use version B because version B is faster, more efficient, and requires less code for the same output.

|  |  |
| --- | --- |
| Date | What is the thing I am trialing |
| 5/06/24 | Different ways of counting a bomb around each cell. |

### Version A

To count how many bombs are around a cell I created a method that takes the input of x and y coordinates then I ran a check on the cell around it ( [x-1, y-1] [x, y-1] [x+1, y+1]… ) if it contains a bomb or not but this will not work if the cell that we are checking is on the corner or at the rim of the board so what I do is create a bigger board (12x12) and only use the middle (10x10) part of it so if the cell that is being checked is at the edge of 10x10 board it will still works fine because the board is 12x12 so there will be no index out of range error.



### Version B

I used a normal-size board (10x10) and passed the surrounding coordinates ( [x-1, y-1] [x, y-1] [x+1, y+1]… ) to another method called countbomb() which takes x and y coordinates as an argument. If the x and y coordinates are out of range the method will return 0 but if that cell is not out of range and contains a bomb the method will return 1 to the main Checkbomb method which will keep track of the amount of the bomb.



Evaluation and next steps

Based on this trial, I have decided to use version B because the board size of the board is being used throughout the entire game. If I use a size 12x12 board when I print the board out or generate a bomb inside the board the coordinates have to be in a range of 1 – 10 and the number of board lengths is not consistent throughout the game which made the game quite hard to deal with. Also, I think version B is easier to understand for everyone in case anyone wants to develop this project.

|  |  |
| --- | --- |
| Date | What is the thing I am trialing |
| 11/06/24 | User try the game 1 |

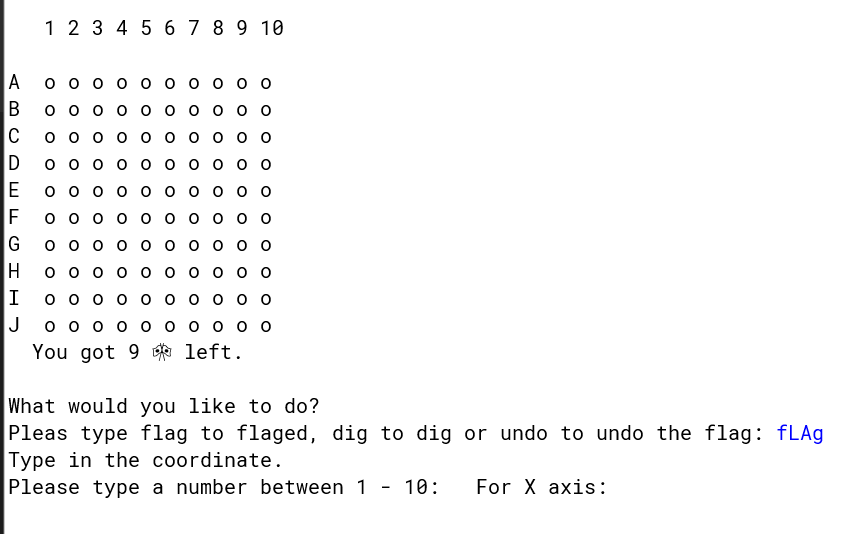
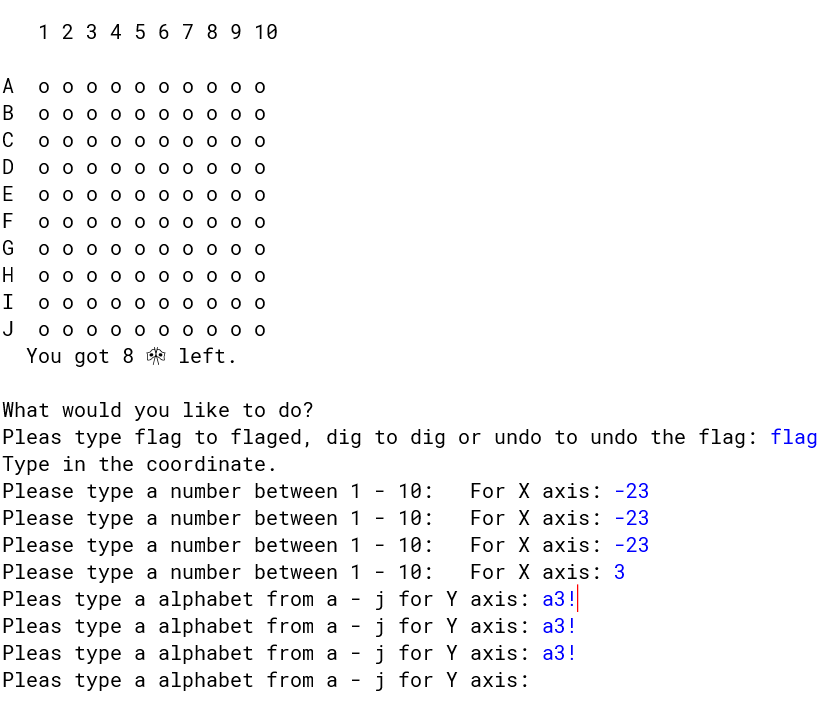
### Feedback

Person: Huy (my classmate)

Date: 11/06/24

Comment: According to Huy, the game is really hard to play because if the player types something wrong, the game will end with an error immediately.

I put error checking into my game by passing every input into a method and will loop through until I get the valid input. if the player types something wrong the program will loop through until it gets the valid input. This will also work when the user inputs a wrong case e.g. [fLaG] the program will still work fine.



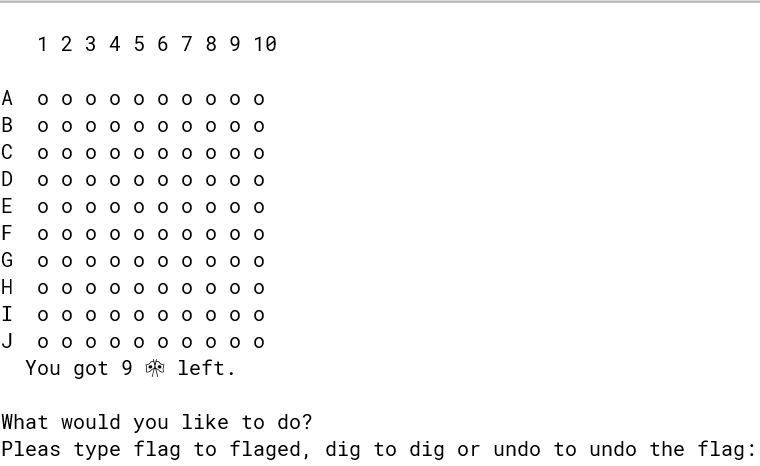
|  |  |
| --- | --- |
| Date | What is the thing I am trialing |
| 13/06/24 | Users try the game 2 |

Person: Aksel (my friend that try the game)

Date: 3/05/24

Comment: According to Aksel he suggests that I put an undo flag option in the game because if the player accidentally places a flag at a wrong spot or places it wrong then finds out later they cannot undo it.

I have decided to add an undo option to my game, now the player can undo the flag if they make any mistake.



## Relevant implications

Your project has an impact on people. These are encapsulated in the relevant implications.

### Relevant implication #1 Usability

*Describe the relevant implication in general terms*

Usability refers to the user's ability to interact with the product or the system easily and should be able to achieve their goal efficiently and effectively. There are many aspects when talking about usability such as simplicity, intuitiveness, and user satisfaction. In software or game development usability ensures that the product is accessible, engaging, and straightforward, minimizing any potential error or confusion. Mainly the game should work properly with minimum viable product.

*Explain how it impacts on your project.*

Usability is a crucial key to a successful game. A game that is difficult to understand and navigate or play can lead to user frustration and negative experience and might cause the player to not continue playing the game. Ensuring high usability means that the game should have clear and easy-to-follow instructions for the player. The player should be enjoying the game experience, and the game should also be accessible to a diverse group of players with varying levels of prior gaming experience.

*Discuss what you have done to address this implication.*

Clear instructions: I have designed very straightforward and easy-to-follow instructions in the game which guide the player step by step throughout the game. The program will also tell the player if they did anything wrong e.g. “Please type a number between 1-10” or “Please remove the flag be for dig the cell”.

User interface design: I did a lot of trailing on this and came up with a very easy to navigate user interface with labels on the x and y-axis. The program refreshes every time the player finishes in that round and prints a new update display.

Trailing: I also do a trial on different ways to write code. For example, I came up with two ways of checking bombs around a single cell. Then I can choose the version that is better suited for my program and has better usability which will make the program run better and/or faster.

### Relevant implication #2 Sustainability and future-proofing

*Describe the relevant implication in general terms*

Sustainability and future-proofing in game development refer to creating a program that is not only efficient and effective but also remains functional, maintainable, and scalable over time. By writing clean code and easy to understand variables and having well-documented work will allow an easy update and development in the future and the program will not need an overhaul.

*Explain how it impacts on your project.*

Sustainability and future-proofing are a very important concepts when doing a game design. The game has to be sustainable over time with minor errors and don’t need any maintenance. My project is finished for now but in the future, me or someone else might want to continue with the project or add more features to the game, so the game structure has to be easy to understand so that any programmer can understand it quickly. My game has to have a well-structured code so that there is no need to overhaul the program when the program needs changes.

*Discuss what you have done to address this implication.*

Code structure: I try my best to use the correct format and convention with comments and variable names in my code. I have commented on every method in my code and on the lines that have complex conditions so that if someone were to see it, they would understand which part of the code does what. I use a clear and easy to understand variable name e.g. checkBomb method to check a bomb.

Version control: I use GitHub to track different versions of my code so that I or anyone else can come back and look at it. This will help in the future because it keeps all the history of the different versions of my program.

Document: I have a document that was created when I did the game such as testing trailing and trello.

The testing and trailing will help the future developer to understand why I did what I did and the result of it. Trello … future developers can see how I create the program step by step and how I break it down into different components.

*Describe the relevant implication in general terms*

End user consideration is mainly on positive user experience. This includes ensuring the game can run smoothly without bugs or errors, the game should have an easy-to-use interface and well design game control and clear feedback to the player such as when there is an incorrect input or the game outcome (win/lose).

*Explain how it impacts on your project.*

End-user consideration is very important to my game because it directly affects player satisfaction. If the game is full of bugs or runs into errors all the time this can affect the user experience and they will not want to play my game. If the game has a poor interface or the control is confusing this could also cause the player to have a negative experience, so considering how the user will experience the game is very important for my game (and most other games).

*Discuss what you have done to address this implication.*

Testing and trailing: I did a lot of testing and trialing on the game. The testing is to make sure that there are no bugs or errors in the game especially in the boundary case so the game can run smoothly. I also did a trial to get feedback from the player and the player will tell me what I should change or add to the game so that the game is easier to play.

Game control/ Error handling: I did a very clear game control. To interact with a cell in the game the player just has to type the coordinates of the x and y axes in. The different options are also clear to follow e.g. “type dig to dig or flag to flagged”. I have implemented robust error handling in my program. If the player types some invalid input the program will tell the player and loop through until it gets a valid input. I also added a feature so that if the player uses capitalized letters randomly the game will still work fine.

Colum and row. Contrast turn base game reaction time.

# Part 2: Testing

Test subject: Input taking/interacting with the board

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test | Expected | Actual | Date | Notes |
| Expected cases | Inputting coordinates for a particular cell.  [1, a] | The player can interact with cell [1, a] (in this case flag the cell) |  | 15/05/24 | Coordinate for the y-axis is a string but it has to be an integer to locate a particular cell e.g. board[3][3] |
| Inputting coordinates for a particular cell.  [1, a] | *The player can interact with cell [1, a] (in this case flag the cell)* |  | 15/05/24 | Change the Y coordinate from string to char because char can refer to a number. |
| Inputting different options  [Dig]  [6, e] | After selecting the cell, that cell should be revealed. |  | 15/05/24 |  |
| Inputting different options  [flag]  [9, f] |  |  | 15/05/24 |  |
| Inputting different options  [undo]  [9, f] | The cell that is [F]  Should go back to “o”. | ➡️ | 15/05/24 |  |
|  |  |  |  |  |
| Invalid cases | Inputting coordinates for a particular cell. But using invalid input  [-23, a3!] | Loop through until get a valid input. |  | 11/06/24 |  |
| Inputting an Invalid option  [ fl4, fl%, di, di#, un!, und@ ] | Loop through until get a valid input. |  | 11/06/24 |  |
| null | It should also loop through until it is valid. |  | 11/06/24 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Boundary cases | Inputting  Coordinate at the edge of the table.  [Flag]  [1, a], [10, a]  [1, j], [10, j] | Should be able to work fine. |  | 11/06/24 |  |
| Coordinate of y-axis with capital letters.    [flag]  [5, G] | Should be able to work fine. |  | 11/06/24 |  |
| Inputting different options with random capitalized letters.  [fLAg] | Should be able to work fine. |  | 11/06/24 |  |
| Undo a cell that have no flag  [undo]  [5, h] | The program should tell the player that they can’t undo this cell. |  | 11/06/24 |  |
| Flag the cell that has already been open  [flag]  [1, g] | The program should tell the player they can’t place a flag here. |  | 11/06/24 |  |

Test subject: Revealed a number

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test | Expected | Actual | Date | Notes |
| Expected cases | When opening an empty cell, it should expand out. |  | \*\*stack overflow error. | *2/06/24* | The recursive function that checks the cell around one cell checks each other back and forth causing an infinite loop. |
| When opening an empty cell, it should expand out. |  |  | 2/06/24 | If the cell already checks and is empty change that cell to “0” Then if the cell is “0” return because it has already been checked. |
| When opening a cell that contains a number  [flag]  [7, g] | It should reveal the number. |  | 2/06/24 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Invalid cases |  |  | Invalid cases such as out-of-range coordinates or wrong input for the y-axis (not a-j) when digging are already tested in the input taking part. |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Boundary cases | When digging a cell that has already been flagged  [dig]  [8, a] | Should not reveal the cell. |  | 7/06/24 | It works fine but it should also tell the player that there is a flag there so the player knows. |
| When digging a cell that has already been flagged  [dig]  [8, a] | The program should also tell the player that they have to remove the flag first. |  | 7/06/24 |  |
| When digging a cell that has already been flagged and that cell contain a bomb  [dig]  [9,a] | The player should not lose because there is a flag and they dig. Even though the bomb is there but there is also a flag so the program should tell the player first. |  | 7/06/24 | It should work fine but there is some error in the code. |
| When digging a cell that has already been flagged and that cell contain a bomb  [dig]  [8,d] | The player should not lose because there is a flag and they dig. Even though the bomb is there but there is also a flag so the program should tell the player first. |  | 7/06/24 | The error is because I check if there is a bomb in that cell before if there is a flag so if there is a bomb the player will lose straightway.    I fix it by changing the odder of if/if else statement so the program checks if there is a flag first then check if there a bomb.    It works fine now. |

Test subject: End of game/win/lose

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Test | Expected | Actual | Date | Notes |
| Expected cases | When all the flags are used and the flag is at the correct location. | Tell the player that they won. |  | 10/06/24 |  |
| When all the flags are used and not all the flag is in the correct location | Tell the player that they lost and say how many bombs they missed. |  | 10/06/24 | The flag is at the wrong place but it still says the player wins. There is something wrong |
| When all the flags are used and not all the flag is in the correct location | Tell the player that they lost and say how many bombs they missed. |  | 11/06/24 | Before fixing the error    After fixing the error    The error is because the if statement for checking the flag will run if display[x][y] = “F”. But I use “f” instead the program did not run the check at all. |
| If the player digs a bomb  [dig]  [4, f] | Tell the player that they dug a bomb and they lose. |  | 10/06/24 |  |
|  |  |  |  |  |
|  |  | - |  |  |
| Invalid cases |  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Boundary cases |  |  | - |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 

## 2.4 Changes resulting from trailing and testing

Did you make any major changes as a result of testing? If so, what, and why?

One of the major changes in my program was when I did a trial on different ways of counting bombs around each cell. The first way was creating a bigger board of 12x12 and using the inner part of the board to prevent index out of range and the second way was to create another method that takes in the surrounding x and y coordinate as an argument and checks if that cell contains a bomb or not and it will return 1 if there is a bomb or 0 if there is no bomb or the index is out of range. Based on this trial I decided to go with the second version because as I said it is way easier to keep the board size the same all the time because we use this variable a lot throughout the whole game which will make the code less complicated and easier to understand to everybody in case anyone wants to develop it further.

The other change that I made after I did a trial with the user was to make my program more robust and easier to play. In my first version, I did not do any error checking at all so when my friend tried to play it and type something thing wrong the whole game just stopped and an error text was shown on the screen and they had to start all over again. So based on this trial, I decided to add error checking into my program, by passing every input into a method and looping through until the program gets a valid input. I also programmed that if the user use wrong capitalize the program will still work. On another trial with another user, they suggested I put an undo option in the game so that if the player found out later that they flagged a wrong cell the undo option would come in handy.

I make a lot of changes in my program according to my testing results. For example, I did a lot of testing on revealing numbers on a cell after the player dug it. The test shows that when I try the boundary case for example digging the cell that has already flagged nothing should happen and the program should tell the player that they cannot dig here and have to remove the flag first but instead, the game ends, and the player is lost. This should not happen, so after I found the error in the code I went back to my program and fixed it. I would not have found this out if I had not done the testing so testing helps me to improve my game.

In my opinion, trialing is helping me to get a better outcome for my program because after I get feedback from my friends and after I run the trial myself, I can see a different way of doing things and different problems. This helps me see the bigger picture of my program and helps me decide which solution is the best. Testing also plays a crucial part in my program. It helps me identify the problem, keep track of it, and address it correctly which lead to higher quality outcomes.

# 3. Summary

All the planning, trialing, and testing has led to a great final program. Talk about that, summarise how each of these (both individually and in combination) have contributed to that excellent final product that you have produced.

Provide evidence from your project to back up your discussion.

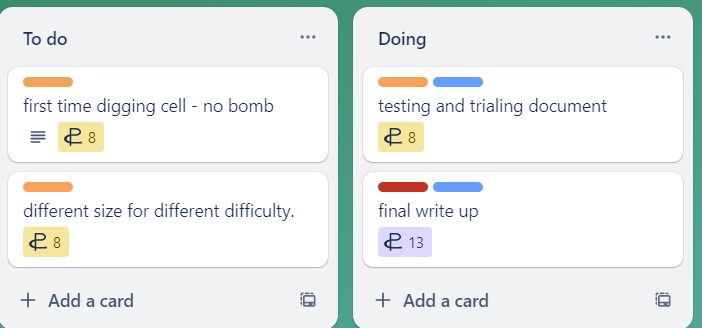
**Planning**

Planning helps me a lot during the process of finishing this project. I made a very clear plan before I started the project which helped me visualize the entire project and the deadline and all the tools played a significant role in my project.

The Kanban board and burndown chart help me prioritize the work so it meets the deadline. During sprint 6 I realized that I forgot to add all the writing tasks to my Kanban board the writing tasks were a lot of work so I had to add them to my Kanban board and I also had to adjust my burndown chart. This affects my burndown rate a lot and I have to adjust my burndown chart again because I will be away for the last weeks of the term, which means I will lose a whole sprint. So, I have to manage my time and tasks and prioritize the work because it is more important to finish the game and the writing rather than finish just the game with the game having all the features that are not MPV and I did not finish the writing. If I did not have the Kanban board or the burndown chart, I would not know that I cannot finish everything in time and I would try to finish the game (with all the features) first and not finish the writing work.

*My final burndown chart*

*My Kanban board. I did not do the last two cards in to do column because it is not an MVP cart (MVP card have blue labels). I also add the writing part.*



GitHub as a version control software helps me keep all the history of my code and I can go back to see my old file which sometimes I need when there is something wrong with my code. It also helps in terms of saving files into clouds so I can access it anywhere and this make it easy for me to work both at home and school and save me a lot of time.

**Trailing**

Trailing also helped me get a better outcome for my program. I did a trailing on different ways to write a code then I chose the best way and used it (all of this information is in the trailing document). This helps improve the program and the logic in it. I also trailed with players to get the players feedback and adjust the game according to the players. This helps improve my game in terms of user experience so the players will enjoy the game more (all of this information is also in the trailing document).

**Testing**

Doing testing on my game also helps me to improve the game. I did testing on almost every component of my game. This helped me to find all the errors and bugs in the game and fix them. If I had not done the testing I would not found these errors which would have led to a game that is full of errors and bugs and might not be playable (all the testing information can be found in the testing document). This shows how testing helps me to get a better outcome of my game.

In summary, I am very satisfied with my project the project is fully complete, and the game itself is fun to play with easy-to-follow instructions and minor or almost no errors. This is because of all the work on planning testing and trailing and all the tools I use so I think this project is a success for me.