Yazan Bawaqna

CS152-B

Report 10

Minesweeper game development

Developing a minesweeper game via python

Abstract:

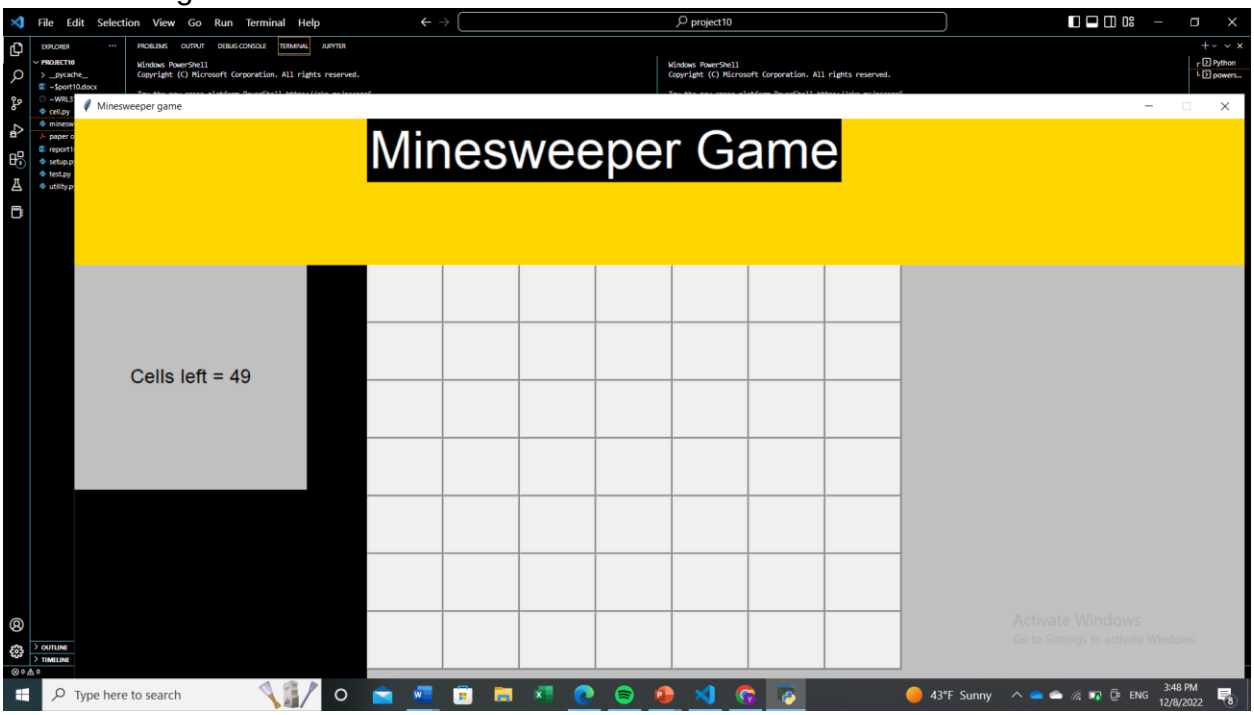
This project aims to develop a minesweeper game based on object-oriented programming via python. The game was used to develop a social experiment that measures how fast the user scores his second win in compared with the first win. The code is rich in programming tools that include graphic visualization via Tkinter and ctypes libraries, classes, recursion, and many more. Starting with the most prominent feature, which is the Tkinter graphic library, the program uses this library to set up frames, which are the windows of the space where the program will run. For convenience, I divided the program into different frames to allow easier arrangement. Furthermore, the Tkinter library was used to import both labels and buttons, which are used for outlining specific info about the game and for the creation of cells respectively. The usage of classes was utilized via the creation of a “Cell class”. Cells in the game are the places where the user clicks, further info is provided in the next section. To elaborate, In Python, we can create objects using classes. A class of an object just carries the characteristics of that object. Imagine a cow class, for instance, its characteristics would be its height, weight, color, etc. Moreover, the class carries all the functions for a certain class. In our case, functions include the right and left clicks on the mouse. Finally, I will explain the concept of recursion. For this project, I created a countdown before the user lost due to time constraints. Recursion is the concept of recalling the same function, so for the countdown, the function will keep calling itself every second, and every time the time limit goes down by 1. When the value reaches 0 the function aborts the program, and the user loses.

Project description:

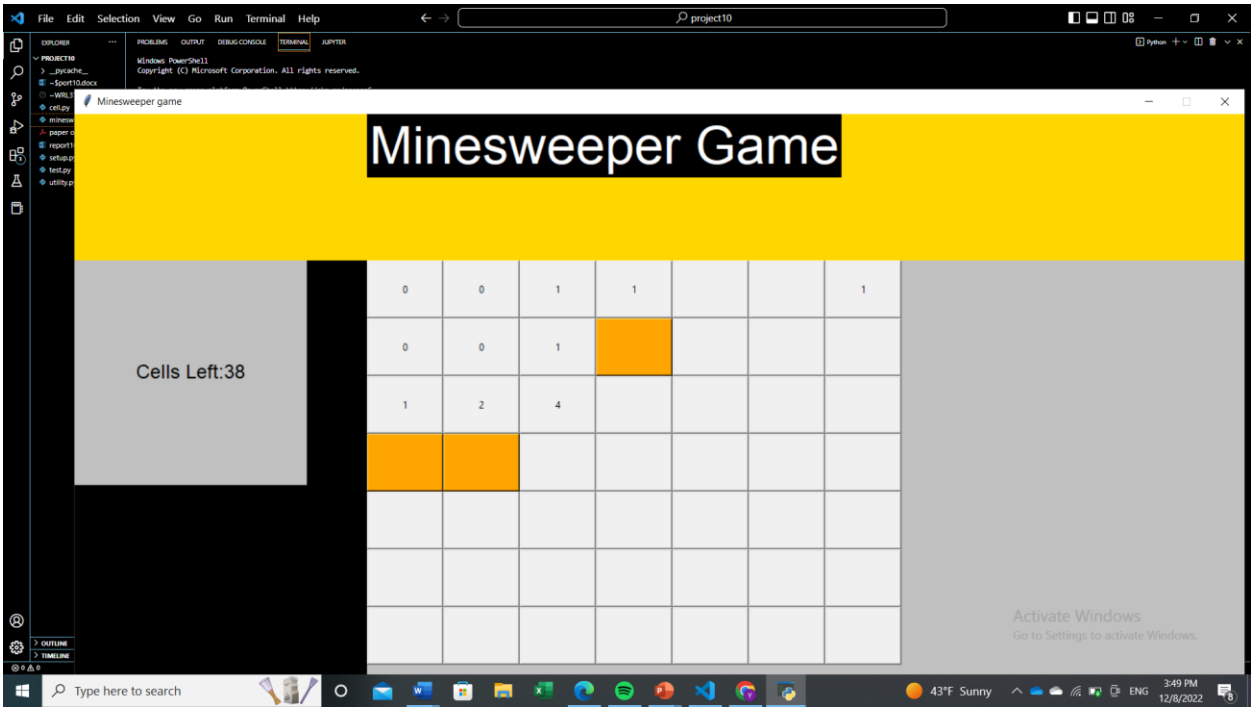
The goal of the minesweeper game is to open all the cells that do not contain a mine inside of them, and if a mine is clicked or the time runs out the user loses. In a programming language, I defined the win as when the number of mines equals the number of unopened cells, sending a winning message. When a user clicks on a cell, the cell reveals how many mines surround it, and if the user suspects the existence of a mine, he can a use right click to mark a flag. For convenience, the Sid sidebar contains the number of cells left, which allows the user to see his progress. The way this works is that we user clicks on a cell, and the buttons for the cells unbind, hence the cell is marked as opened. The way mines and cells are arranged is random. However, the total number of mines in the game is determined in the setup file which is the grid dimension divided by 4. The setup file allows the user to change the difficulty of the game by increasing or decreasing the grid dimensions or the number of mines. Lastly, it is important to know that for the user to open the needs both a degree of luck and probability skills to predict the positions of mines.

Project presentation:

Screeshot1-game start



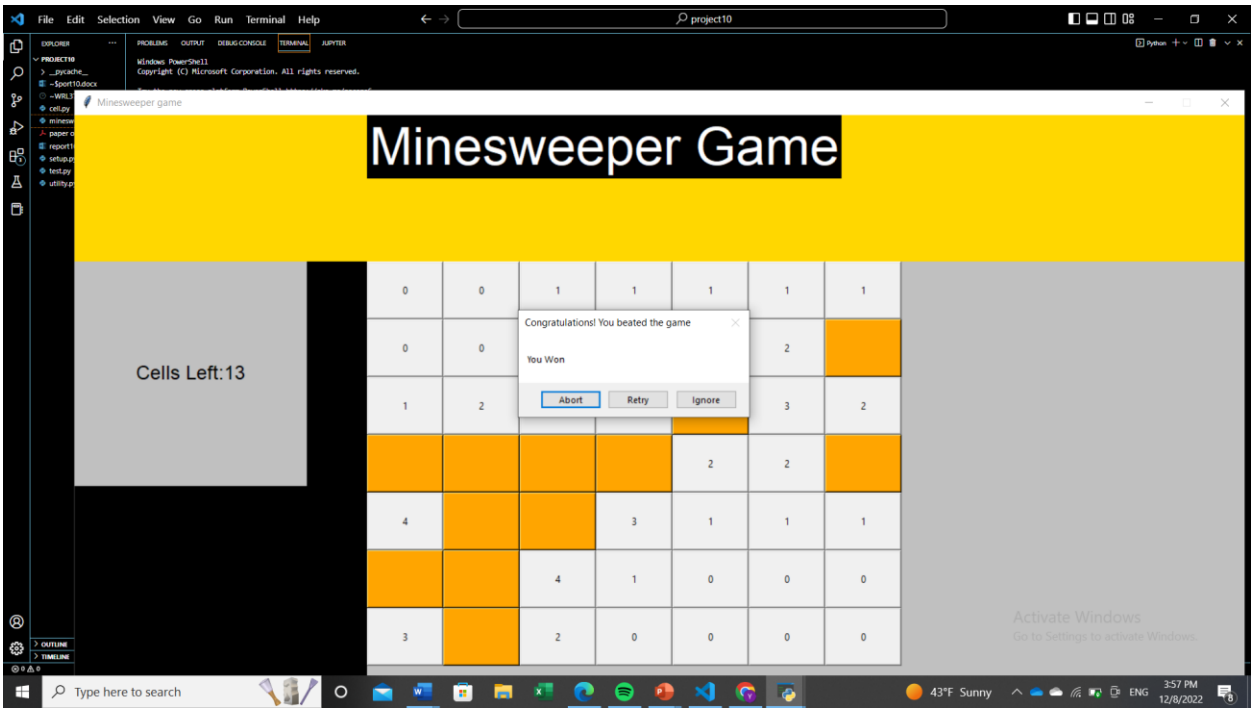
Screenshot2-Gameplay:



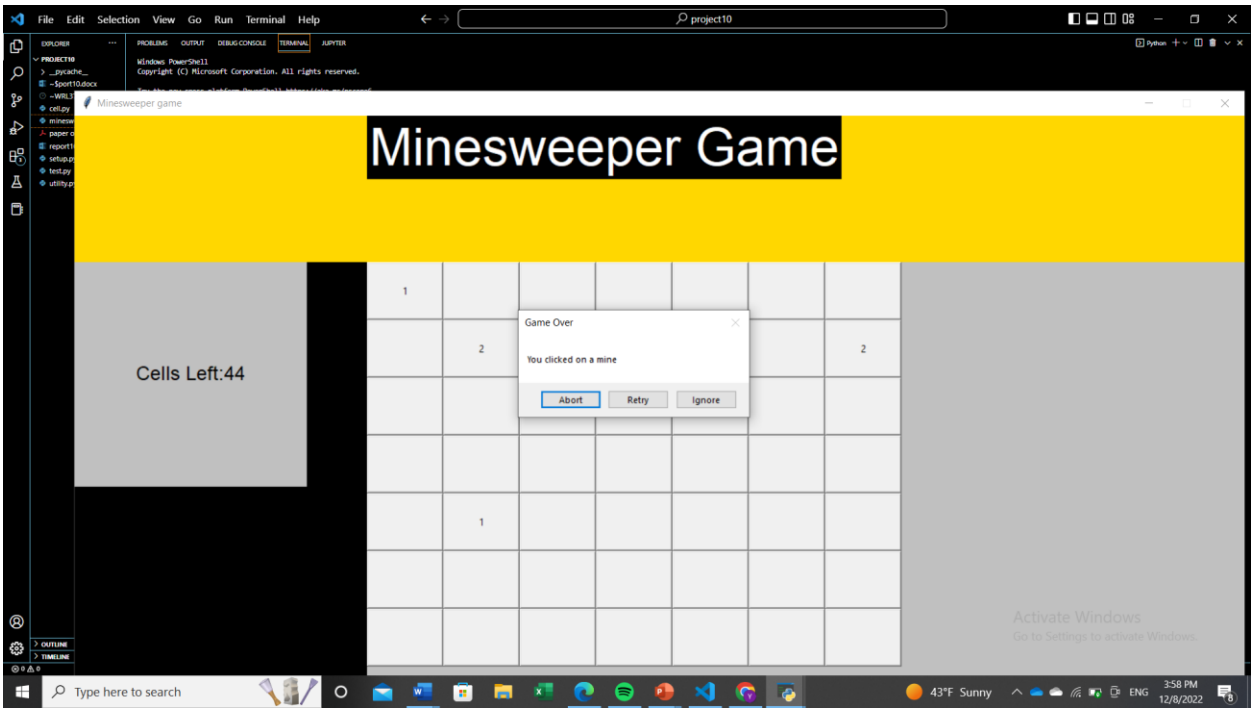
We can see that the number of cells left decreased

The orange squares are the flags, which I input wherever I suspect the existence of a mine.

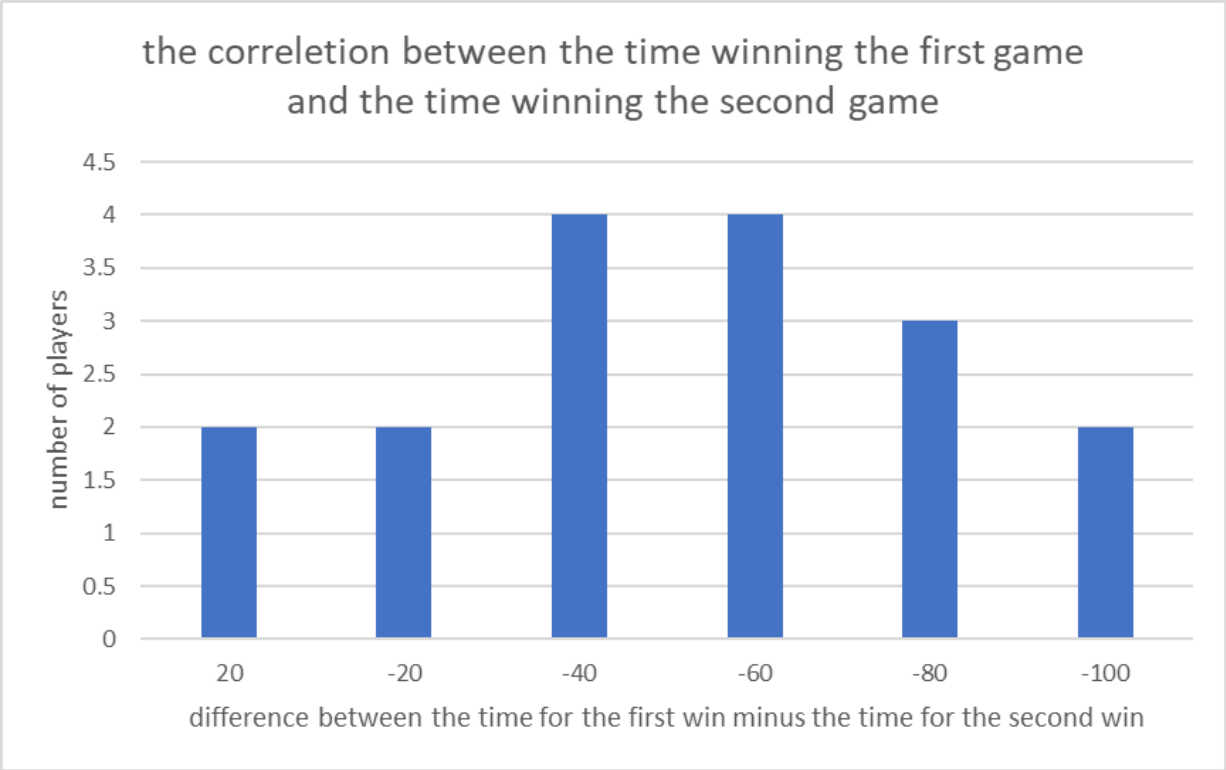
Screenshot3-winning



Screenshot4-losing



Graph1-



Evaluation:

To provide a holistic analysis, I will discuss the usage of objected-oriented programs, the lack of dictionaries and inheritance, and the usage of recursion. Objected-oriented programming is all about efficiency and convenience. If used probably, other programs will have an easy time evaluating the code to be reused and provide them with accessibility for inheritance by developing child and parent classes. However, my program did not take full advantage of classes by not using inheritance or using dictionaries. If I had the opportunity to redevelop my code again for another person or school project, I would separate my cells from safe cells by acquiring them in different classes with the cell class being their parent class. With no inheritance, cod could not be compiled and hence does not help data encapsulation. Multi-dimensional lists on the other hand, offered a more organized, easy to read and efficient code.

Limitations:

The first main issue that affected both the experiment and the game is the fact the user may lose from the first click if the user clicked on a mine. An improvement I have considered is to input the mines into the game after the user's first cell choice. The

issue with this, however, is that the game will lose a little aspect of its randomity. The other issue was seen in the experiment since the user may encounter a stuck phase, and many randomities caused the correlation between the users winning the second time faster lower. However, I noticed that many players (my friends who played the game multiple times) start to approach an imaginary maximum time. That was very interesting to study, and I may continue this experiment for curiosity. Additionally, I was unable to exhibit the countdown (time until the user loses by time constraint) which caused many players to lose due to a lack of attention to the time. Lastly, the database was taken from only 10 people who are my friends, which is ultimately not a random data sampling. Perhaps the lack of both gender, age, and experience variation made the data insufficient to conclude.

Reflection:

Since it is my first time implementing a whole project by myself, I found the experience enriching. I came up with the idea via programming discussion websites where people suggested that developing a game is extremely rewarding in terms of newly acquired skills. I went to explore the best library to graphically visualize my project where I found out the tremendous choice Tkinter library has to offer. Furthermore, I learned a set of new skills like using “@” for “static method” and “property” which allow the function to be used singularly and become an attribute respectively. Also, this project was my first time using the time library which helped me frame my countdown.

What was fascinating about the project is how easy it is to level up the complexity of the project by importing outside-sourced libraries. Although ignorant as I may seem. I never comprehended the volume of outside libraries we can import into our code. While I was doing the project, I went to check for some online libraries out of curiosity. To my surprise, I came to know that there are endless options of codes to be reused at all levels of complexity. The power of this is that it allows beginners like me to have the ability to invent advanced projects, like sensors or 3D graphing. What’s most beneficial is that code reuse here allowed me to decrease my code lines by at least 75%, which is what is needed for a code to be efficient.

Works Cited:

Manish Shivanandhan 2 hours ago, Mugilan Ragupathi 21 hours ago, Dionysia Lemonaki a day ago, Beau Carnes a day ago, Keyur Paralkar 2 days ago, Anthony Behery 2 days ago, Marco Venturi 2 days ago, et al. “FreeCodeCamp Programming Tutorials: Python, JavaScript, Git & More.” freeCodeCamp.org. Accessed December 8, 2022. https://www.freecodecamp.org/news.

Yeo, Leonard. “This Is How to Create a Simple Minesweeper Game in Python!” Medium. The Startup, September 24, 2020. https://medium.com/swlh/this-is-how-to-create-a-simple-minesweeper-game-in-python-af02077a8de.

Replit. “How to Program Minesweeper in Python! (Fully Explained in Depth Tutorial).” replit. Accessed December 8, 2022. https://replit.com/talk/learn/How-to-program-MineSweeper-in-Python-fully-explained-in-depth-tutorial/9397.

David Istrati. contributed to debugging my code.

Imported libraries:

Tkinter from *

time

math

sys

random

ctypes