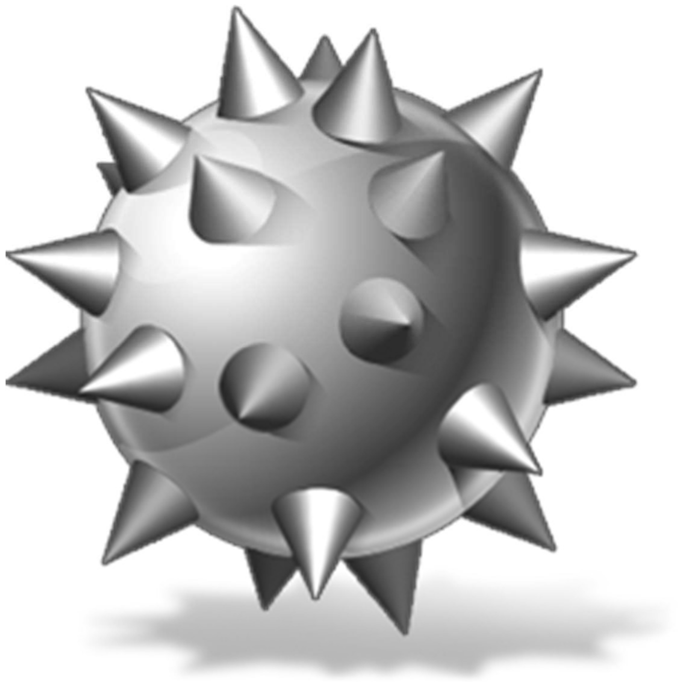
Project on

GUI Based Python Game

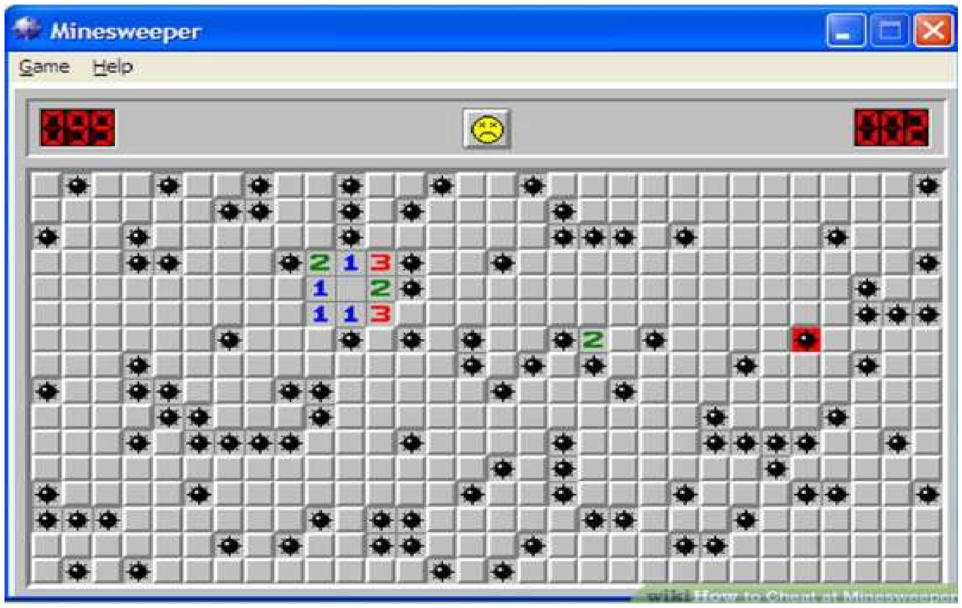
Minesweeper



Group members:

- Kartik Gupta (IMT2016128)
- Bukka Nikhil Sai (IMT2016091)
- Venkat Sai Srikar Adapa (IMT2016029)

Contents   Page no.



Objective

Our objective is to create GUI based minesweeper game in python language.

Purpose of Document

This document will define the design of the game minesweeper. It contains specific information about the expected input, output, modules, classes and function.

About Game

The game begins with a 9x9 matrix (or 16x16) of tiles that cover some mines. Some tiles hide mines, most do not. User play the game by clicking on tiles to reveal what lies beneath. If he exposes a mine, he blows himself up and the game is over. The goal is to expose all the tiles that do not cover bombs in as little time as possible.

User is aided along the way when he uncovers tiles by numbers that appear in squares that are adjacent to a bomb. When a square is adjacent to zero bombs, the square appears blank. When one or more bombs are adjacent to a square (either left, right, up, down, or diagonally adjacent) then the number of adjacent bombs is displayed.

To help user keep track of which tiles you have inferred that cover bombs, you flag squares by turning flags on. When flags are turned on and he click on a tile you are flagging it as a bomb and not exposing what is underneath.

Specifications

The game starts with a pop up asking the player which level he/she wants to play. If a player is new to the game, there is an instruction button below the level buttons. If the player clicks on the instruction button instructions will be displayed with an ok button. The player should click the ok button after reading the instructions after which it returns.

There are two levels easy and hard. If user selects the easy level a frame with 9x9 grid opens in which all mines are covered. The frame also contains a status bar showing number of mines and number of flags. There is a timer which starts as soon as the game starts. There is a quit button at the top of the frame if you want to quit in the middle of the game. If a player wants to uncover a mine he/she should click the left mouse button and if he/she wants to place a flag he/she should click the right mouse button. After completion of the game a message pops up and the message depends on whether you won or lost. Then it asks if you want to play the game again or not. If you click no, then it again asks if you are sure to exit. If you press exit, then the game ends displaying "Thank you". If the player clicks yes to play again it returns to the display where the level is asked and the same happens with the hard level. The only change in hard level is it consists of a 16x16 grid and contains more mines.

Design

Modules:

The application can run only in the presence of the modules mentioned below. They are: -

a) Tkinter
b) Random
c) Time

a) Tkinter: It provides a wide range of options which allows us to create graphics in which we can interact with the user. It provides options to create frames, message boxes, buttons, shapes, import images and many more. There are modules which are supported by tkinter such as tinker.scrolledtext , tkinter. colorchooser, tkinter. commondialog, tkinter.messagebox , tkinter.simpledialog and many more. We may include some of this module in our program.

b) random():Function random() generates a random float uniformly in the semi-open range [0.0, 1.0).Similar to tkinter it also contains some modules like random.getstate , random.setstate ,random.getrandbits(k) (Returns a Python integer with *k* random bits) and many more. We can also use these sub modules.

c) Time module is used for importing time to start timer and stop timer and display the time while playing the game.

Classes:

(1). Main class:

* Levels will be created in different files (easy level file and hard level file) with different classes and then it will be imported to the "main" file.

* The "main" class takes classes in other files as input. These classes initialize the frame, append the images to the class and then creates instructions and levels (easy, hard) buttons and adds them to the frame. When a level is selected, it calls the respective function of that button and clears the frame from buttons and then they call the respective class according to the selected level. After completion of the game a function is called asking the player if he wants to play again.

Functions in the main class:

a) Function to call different levels which are attached to the buttons in the frame.

b) <u>Lose function:</u> It tells the player that he lost the game. The function makes sure that it asks the player if he wants to play again.

c) <u>Win Function:</u> It is similar to the lose function. It tells the player that he has won the game and it makes sure to ask the player if he wants to play again.

(2). A class imported from easy level file starts initializing the frame and then appends images to that class. Then a grid is initialized with mines produced at random places using random function. Buttons are then created in dictionary which stores :

a) button image
b) whether mine present
c) state (opened, closed, flagged)
d) button number
e) coordinates of button
f) number of mines nearby.

Then it checks the number of mines nearby and it stores in the dictionary.

Functions in “Easy” and “Hard” classes:

a) check mines:
It is used to the number of check mines and store it in the buttons

b) l_click and r_click functions:
When the user presses left mouse button then l_click function is called which either displays the image of a bomb if it is present or the it displays number of bombs nearby. When the user presses right mouse button the r_click is called and the button is either flagged or unflagged. If it is flagged the image of the flag is displayed.

(3) A class imported from hard level file also contains the same functions as the classes imported from easy level file. These are modified according to size of the grid.

Work Plan

We have divided the work equally and such that the team members working on it can work independently of other modules to best extent possible. One member is working on the “Main” class and other two members are working on the basic algorithm and GUI based programme of the levels which are the “Easy” and “Hard” classes. The one who is developing the “Main” class will also try to work on the timer for the game.

References

a) Learning Python (5th Edition)

b) Core Python Programming (Pearson)

c) New Boston YouTube Videos for GUI

d) Code chef (online teachings)

e) Stack overflow website for some doubts

f) Tutorials point for Object oriented programming

g) Help from seniors

********** THANK YOU ************