Final Report

# Introduction

## Problem Description

This a game trying to mimic all the features in a traditional Nintendo Bombman Game. It’s a strategic, maze-like video game. It supports both single-player model and multi-player model. It has more than one monster type moving on the map. However, with more advanced tool in python, now we can largely improve the graphic feature. We don’t need to put up with the window-95 like GUI any more. In this implementation, the graphic features will be extremely highlighted.

## Objectives for Project

* To create a multi-player version of Bombman Game
* To create a way that allow users to customize the game in a certain level
* To create a more good-looking GUI than tkinter
* To implement high scores

## Overview of Approaches Taken

* Using pygame as a game engine. It better supports images.
* Using sprite class in pygame to create monsters and characters to better deal with collision.
* Using more pictures to replace the ugly button and tkinter widgets

## Organization of the rest of the report

The report introduces detailed features of the product design, including the music tracks, high scores, a map editor, different player modes, instructions, game saving features and bunas tiles. Then, it introduces features actually developed in the game.

# Detailed Features of the Product Design

## Music Track

The original version has a lot of beautiful and pleasure music track. Some of them will be introduced to the game if they are available on the web. I wish these features can remind players childhood memories.

## High Scores

This is a quite common feature for a stand-alone version game. It gives the top scores that several players have created on the local machine. It should contain the player’s name and his score, ranging from the highest to the lowest. It also can be saved to the disk, won’t be lost after terminate the game.

## Map Editor

This is an additional feature added to the game. In the original version of the game, it only has 10 fixed maps. Once a player finished it, they can soon lost interest on this game. However, a simple map editor can probably change this. Allowing user to customize their own map can easily double the fun.

## Player Mode

In the original version, it has 10 fixed map for both single-player mode and multi-player mode. Now, since we have a map editor, I can simply file to random generate the game map in single-player mode and multi-player version every time. Goal of both modes are to eliminate all the monsters on the map.

## Instructions

This game should provide clear instructions page to indicate how to play on the pc since it doesn’t support game controllers. It has to provide a reasonable key map that enables users to play on the keyboard. Besides, the key map should be intuitive and comfortable to use.

## Save Game

This game should provide a way let users to save the game and can resume later. Sometimes, players need to leave for a while to pause the game or even exit the game. It should provide such features.

## Bonus Tiles

There are a lot of bonus tiles in the original game. They can increase lives or bomb range or increase the movement speed. I choose to implement some of them due to the time issue.

# Detailed Features Developed

## Music track

I provide some music track in the game. When a player enters the game, it will have welcome music. During the game, there will be sound effect when the bomb blows.

## High Scores

I implemented a score board that lists the top six highest score records. When a player win the game, it will ask the player to enter the name and it will save the records to the hard disk.



## Map Editor

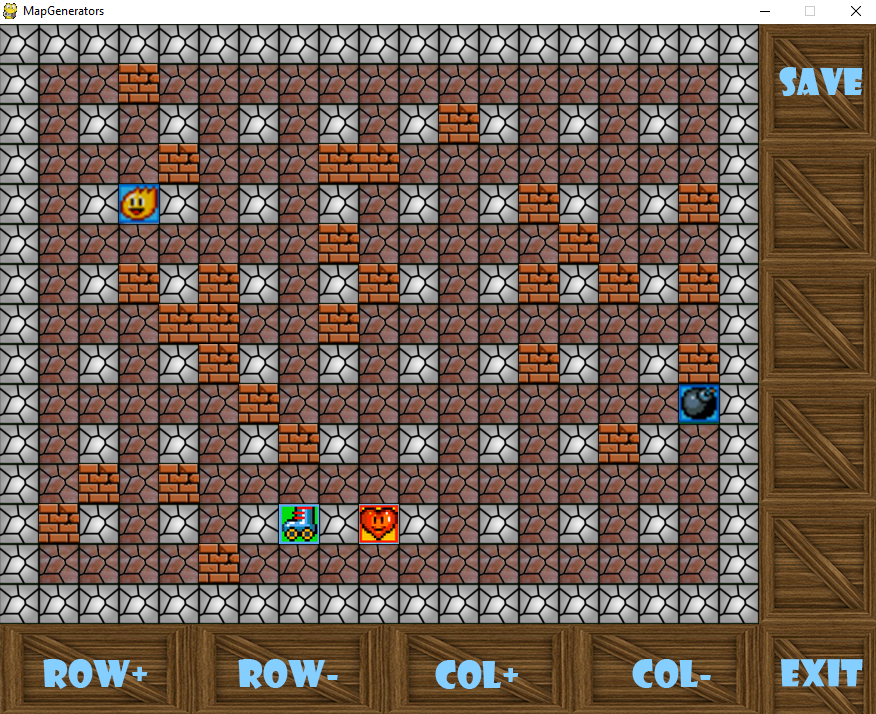
The map editor I implemented has more than one feature.

First, it allows user to change the scale of the map. The button at the bottom can be used to increase or decrease the row number and column number.

Secondly, it allows user to change the tile on the map. It allows the user to change any tile on the map by just click the tile he wants to change. It has bricks, walls, bomb-number-increase, bomb-range-increase and so on. Each bonus will be automatically covered by a piece of brick wall tile.

Thirdly, it provides the save function. It allows the user to save the map to the hard disk, which allows the user to load later.

Operations: When you creating the map, you can change the tile type by left-clicking the tile and change the scale by clicking the button at the bottom.



## Player Mode

This game provides 2 modes, namely single player mode and multiplayer mode. In the single-player mode, the player avatar will show up at the left-top corner of the map. It will generate a random number of group of enemies up to 12. The single player mode will change the direction that from

|  |
| --- |
|  |
| Figure Multi--player GUI |

## Instructions

Instructions should provide sufficient information for users to play the game.

What is shown in Figure 2 is the key map for single-player mode. It uses the arrow keys to control the avatar and space to deploy the bomb.

The map editor GUI is shown in Figure 3. It provides “save”, “row+”, “row-”, “col+”, “col-” and “change the tile type” functions. The user can increase row and column number by clicking the “row+”, “row-”, “col+” and “col-”. The row number and column number must be odd. The user can change the tile type by clicking the tile.

|  |
| --- |
|  |
| Figure Single player mode keymap |

|  |
| --- |
|  |
| Figure Map Editor GUI |

The multi-player key map is shown in Figure 4, to separate the key map between player 1 and player 2. The WASD to use the avatar 1 to move around and use the Left-Control to deploy the bomb; the arrow keys to control the Avatar 2 to move around and use Right-Alt to deploy the bomb. It basically allows two users to control different avatars and deploy bombs. Besides, the movement control keys are pretty intuitive.

|  |
| --- |
|  |
| Figure Multi-player Mode Key Map |

## Bonus Tiles

As in the original version, the program provides several bonus tiles to improve the avatar’s availability, such as increase the bomb number the avatar can carry with, improve the avatar’s power range of the bomb, increase the movement speed and increase the lives.

I provided the four mentioned above:

|  |  |
| --- | --- |
| Icon | Description |
|  | Increase bomb number by 1 |
|  | Increase the character movement speed |
|  | Increase the bomb range by 1 |
|  | Increase the avatar life by 1 |

# Assessment of How Well Approach Works for Problem

It works well.

## Image Process

By using pygame, I get better performance when dealing with images. At the same time, it provides more methods to cope with images. It enables me to rotate / mirror the entire the image. Therefore, I can provide better-look GUI rather than the one Tkinter provides.

## Music Track

Also, because the program is using pygame, it enables me to add original sound track to the game.

## Pickle Module

By using pickle, it allows us to pickle the object in the program and store with status to the hard disk and resume the game.

## Pygame Sprite

Apart from the image processing superiority, this is the most important reason to use Pygame library. It provides a lot of interesting features when dealing with collisions. In this program, I simply use the hierarchy shown as follows:

I put the common features between Player and Enemy to the Character, which inherit from the pygame.sprite.Sprite. Since both players and enemies are sprites, I can use the collide\_rect to dealt with when they run into each other. The cooler part is that we can isolate them if we put them to different groups. In that way, we can know if this is a collision that happened between a player with another player or it’s a collision between an enemy and a player, which we should give penalty to the player. With groups, we can do these on the group level, rather than on the individual object. In this way, we can gain a better design pattern. All the object only needs to hold its status, no logic inside.