BATTLE SHIP:

Programmer’s Guide MP2

Manuel, Troy Vinenz

Sumabat, John Carlo

Llanto, Tim Kristian

*(Grp30)*

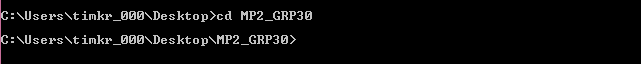
*Dec.04, 2018*

CONTENTS

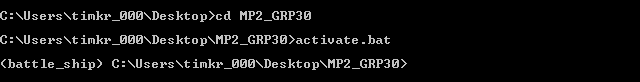
1. How to Start the Game...............................................................................................3
2. Requirements to Run the Game.............................................................................4
3. Functions........................................................................................................................5
4. Flow of execution.........................................................................................................7
5. Sources..........................................................................................................................14
6. How to Start the Game
7. Open the terminal.



1. Find the game’s directory and type it on the terminal. The game can be found inside MP2\_GRP30 folder.



1. As this version uses an external module, a virtual environment (venv) is created that needs to be activated before starting the game. This can be done by opening “activate.bat” file, located in the folder with the game, in the terminal (This works only for Windows)﻿. If you are on Mac or Linux, you must still make a new venv and install the required modules.



1. Open the game by typing python main.py. A new window should pop-up showing the game.
2. Requirements to Run the Game

* Python should be installed in the host computer.
* Python PIP for the installation of Python packages
* Pyglet (version1.3.2) which should install if Virtual Environment(venv) is activated in the terminal as it is already indicated inside the Virtual Environment(Windows)
* A new isolated venv if on Mac or Linux using the required module indicated in requirements.txt

1. Functions

Listed below are worth mentioning and unique function(s) used in the engine of the game as well as their role in the game execution.

engine.py proves all the necessary functions for the game and for the AI (artificial intelligence) opponent. Modified functions for the AI have ‘ai\_’ in their name.

1. grid()

Creates a 10x10 grid for both the player and the AI.

1. shipcheck() and ai\_shipcheck()

Checks if the coordinates of the ship will fit within in the board. It also checks if the ship intersects with another ship.

1. place() and ai\_place()

Places all coordinates of the sip goes into a list called ‘occupied’ or ’ai\_occupied’.

1. shipset() and ai\_shipset()

Checks which ship is being placed and what’s their size. Then it performs shipcheck() and place().

1. hitcheck() and ai\_hitcheck()

Checks whether if the coordinates chosen hits a ship or misses.

1. ai\_coordinates()

Chooses the coordinates of the AI. It takes into account the size of the smallest ship left alive and makes sure whatever coordinate chosen, said smallest ship can fit. If no such coordinate is found (a portion of a ship is destroyed and thus the size of the smallest ship becomes irrelevant), it picks a random coordinate.

1. ai\_repeat()

If the AI successfully picks a coordinate with a player’s ship, the AI will randomly make a decision. This decision is a number from 0 to 3, with each number representing a specific direction (up, down, left, and right) to choose the next coordinate. If the next coordinate results in a hit, the AI will then continue to use that direction until it comes across a non-empty coordinate (hit or miss).

1. no\_moves()

Checks if the AI has any possible move left by checking how many coordinates are non-empty.

1. ai\_hit()

This is what the AI does in its turn. It runs ai\_coordinates() to pick a coordinate and checks, using ai\_hitcheck(), if it's a hit or miss. If the coordinates chosen is a hit, it will run ai\_repeat().

1. ship\_unset()

This function executes and edits the human board occupied coordinates if ship is dragged away if already in the board.

1. mouse\_position\_is\_in()

This function checks if the coordinates of the cursor of the mouse is within a specified box or area.

1. ai\_set\_ships()

This places the desired coordinates inside the human occupied coordinates

1. play\_sound()

Plays a sound that is not looping.

1. play\_music()

Plays a sound that is looping.

1. record\_if\_highscore()

This function checks if the player’s score surpasses the current highscore and records if it does.

1. restart\_program()

This function restarts the whole python program.

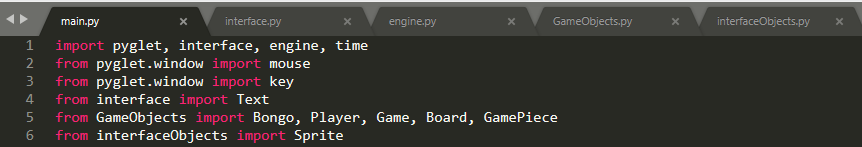
1. ship\_unset()

This function executes and edits the human board occupied coordinates if ship is dragged away if already in the board.

1. ship\_unset()

This function executes and edits the human board occupied coordinates if ship is dragged away if already in the board.

1. Flow of Execution

The game starts when the program *main.py* is executed in the terminal.

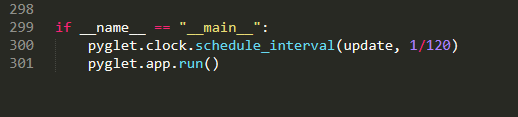
Several modules are first imported to make the game work. Some of these are self-created modules:

GameObjects: imports the programmer-defined objects used in the game .

engine: imports the python code that handles the game logic.

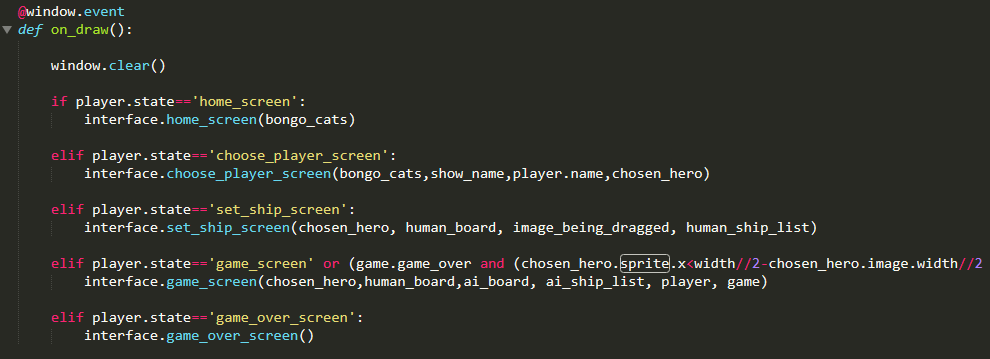
interface: imports the python code that handles the game’s Pyglet interface.

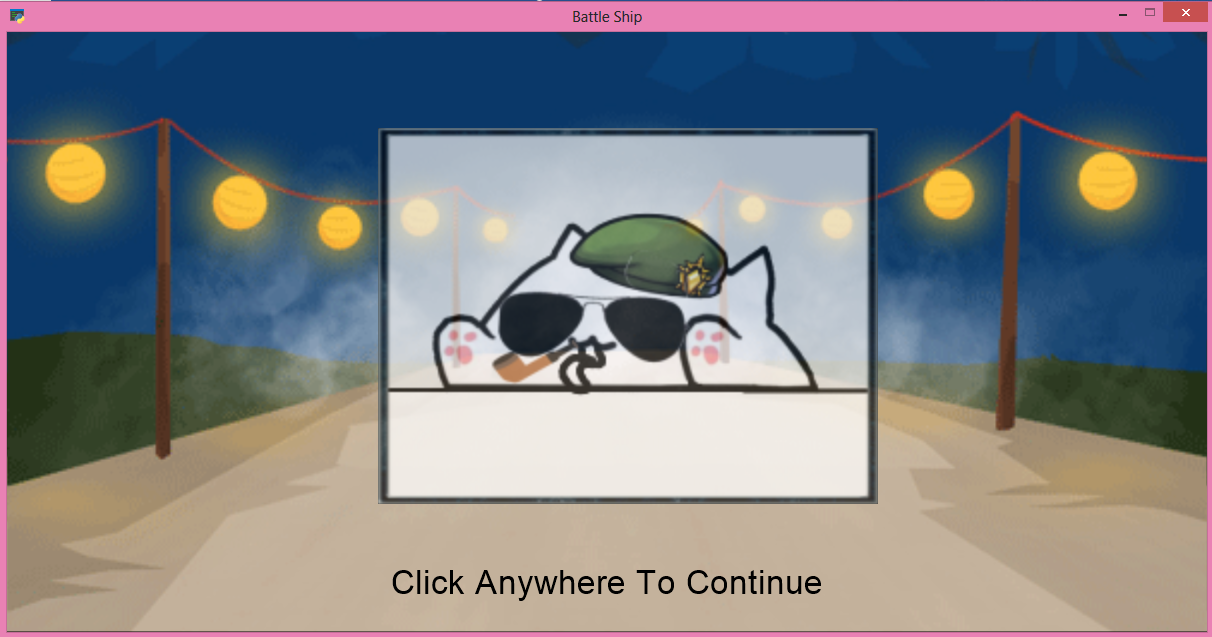
The program would then initialize all the necessary variables and objects that would be used throughout the game under the main module­ —including the initialization of the Pyglet module. A scheduled interval using pyglet is set to 1/120 so that the frame interface updates 120 times per second.



Also inside main.py, all other events are situated that includes mouse events (pressing, dragging, releasing, & motion) and keyboard events (pressing a key)

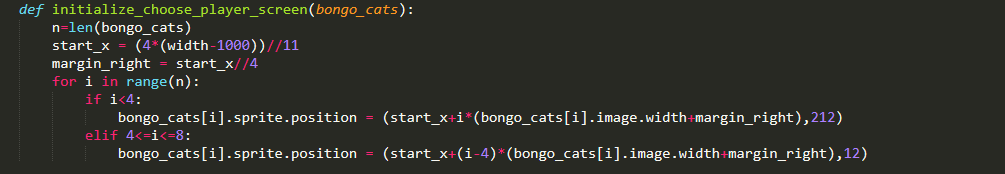
Almost all of the events are subdivided under the category of the ‘state’ attribute of the ‘player’ object. The programmers did this to lessen the time it takes to update the code when there is an event. The player ‘states’ are named according to which part of the screen is the user is. The on\_draw() function includes all the ‘states’ of the game:



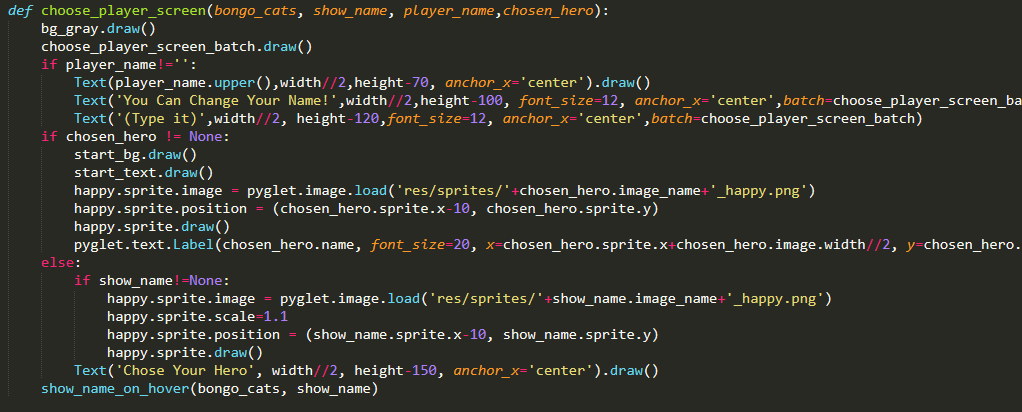
When the game is started (main.py is executed in the terminal), the home screen shows up: 

Here the player can view an animation of the different heroes that are currently available in the game. Clicking anywhere within the window executes the next state, ‘choose\_player\_screen’

It is important to note that every state has its corresponding initialization function and another function that is continuously executed for the interface of the game. The division of what is required to run only once to those that are needed to repeatedly be executed saves a lot of graphic memory that enhances the performance of the game. Below is the example for the state ‘choose\_player\_screen’ :

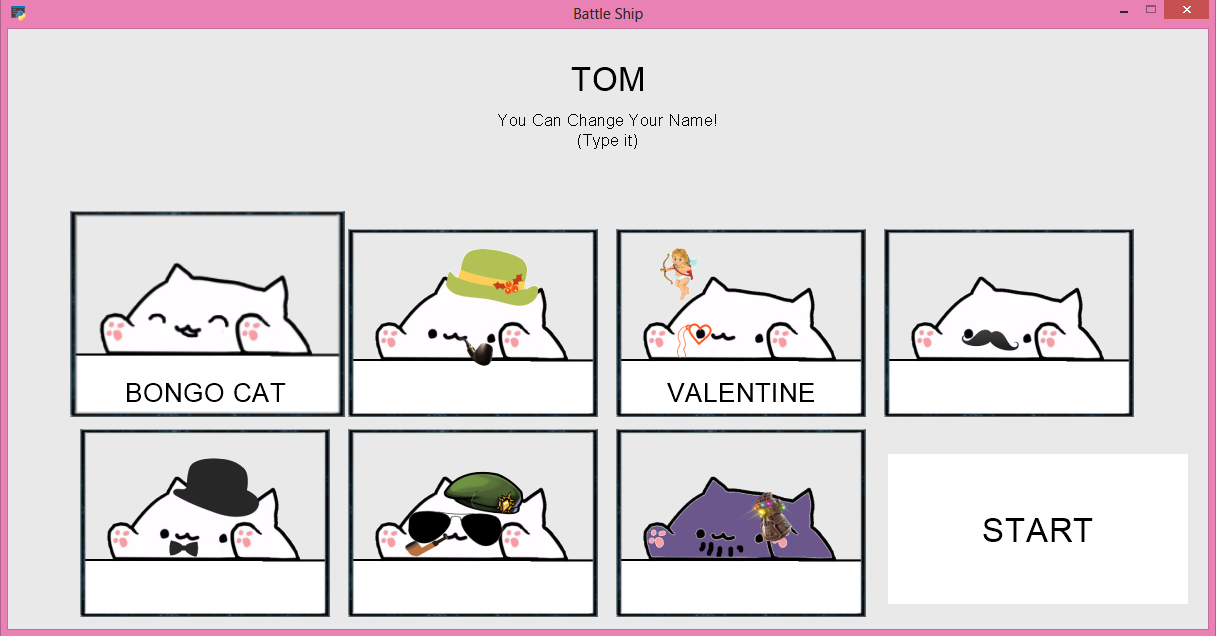


#run only once the player enters the ‘choose player screen’



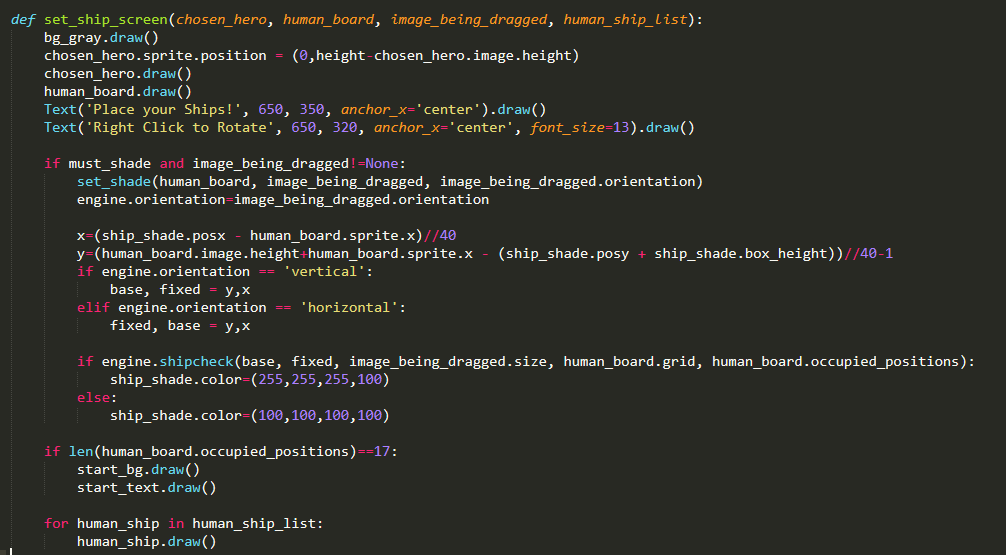
#run repeatedly once the player enters the ‘choose player screen’

The code above shows a display of the different characters available in the game. The display output is shown below:

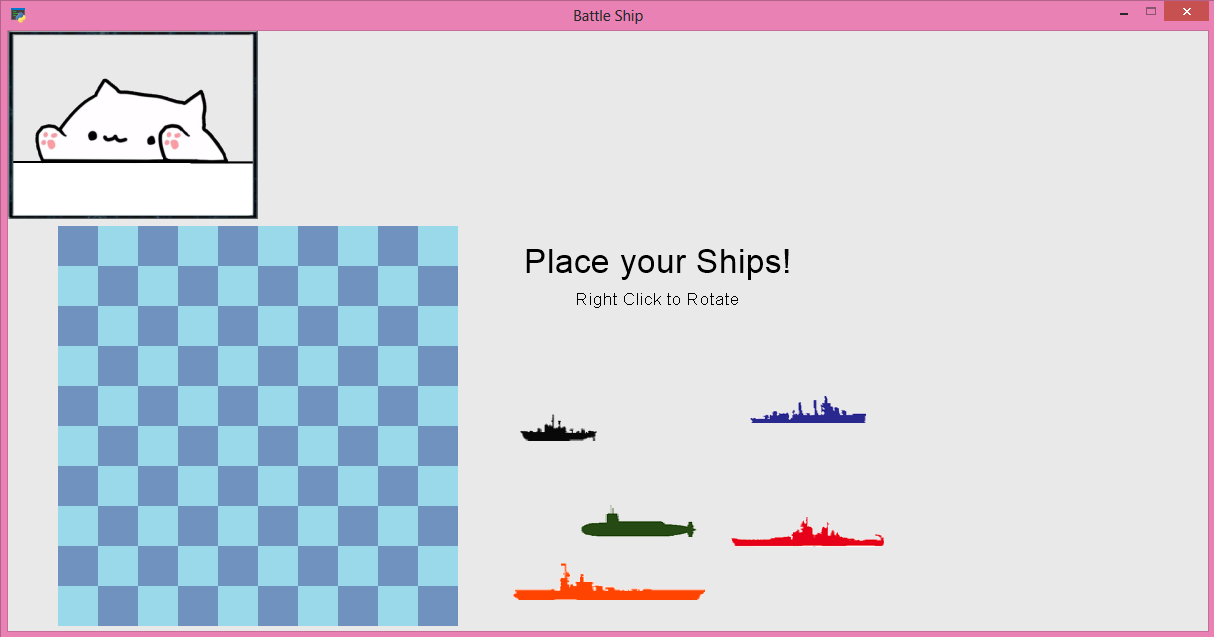


Once the player picked his/her hero and had decided for the name, the start button will show up that signals the player is now ready to start the game.

The third state is the set\_ship\_screen\_state where the object human board and the human ship objects are introduced in the game. The state uses mouse dragging and right clicking events to place the ship in the board and to rotate them, respectively. The function set\_ship\_screen() in the interface uses another unique function called set\_shade() that computes and displays the position of the shading of the ship that is being dragged. The code that represents what objects are to be drawn in the window are shown below:

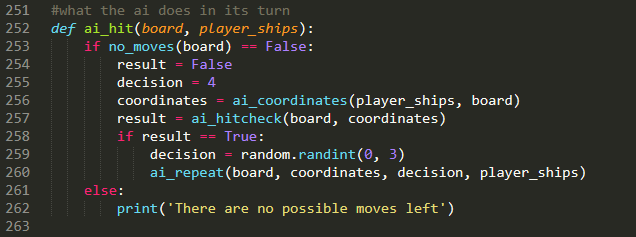


This function outputs a display shown below:

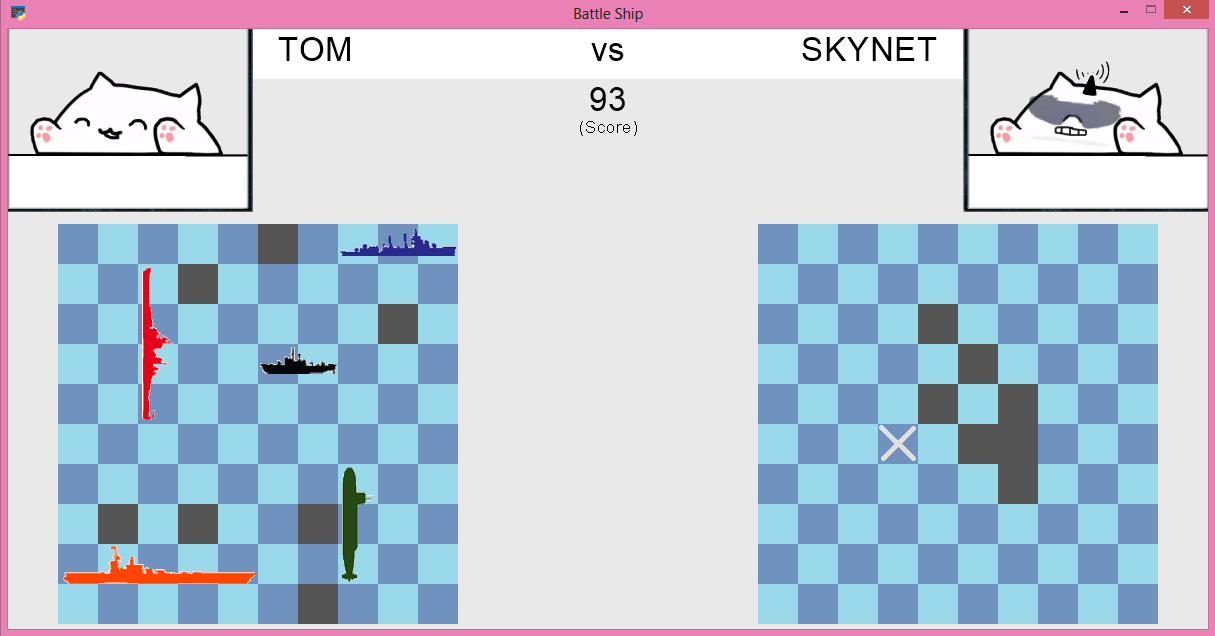


The next state is the game\_screen state where most of the logic of the game takes place. While in this state, the program will wait for the user input in the form of a click in the board of the AI. And yes, we created an AI, but it is still in its infancy. But there shall come a time where this AI would grow and would surpass human intellect. :)

The engine module contains all of the necessary logic for the computation for the AI’s move. It is programmed to hit randomly until it successfully hits a ship where it would again randomly hit a box vertically or horizontally adjacent to it. If successful, the number of laser fired by the AI determines the number of successful attempts in hitting a human ship. Below is the ai\_hit() function in the engine, where the AI decides on where to hit:

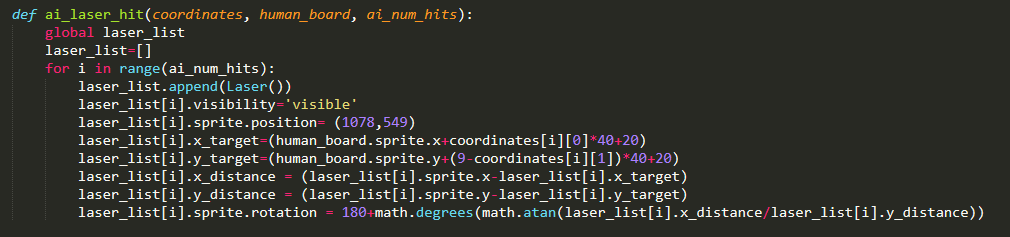


The board updates every time the human and the AI take turns. The main game display is shown below:



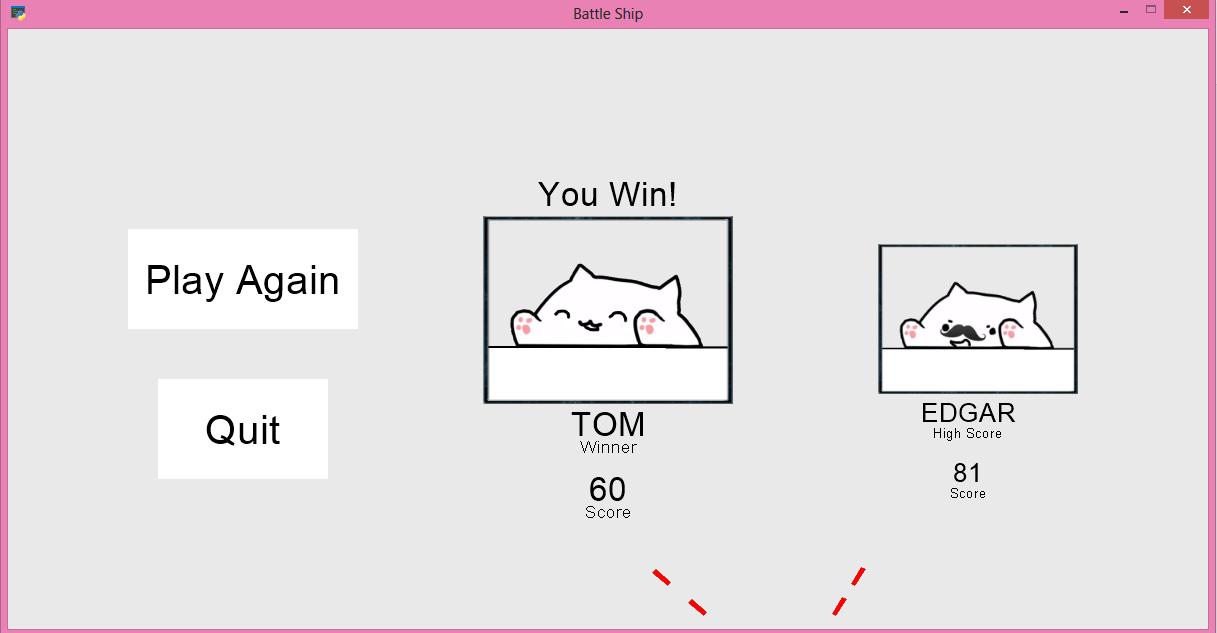
The names of the player and the AI are shown in the top bar. Below the top bar is the score of the player. To make an intuitive scoring based game in Battle Ship, we first set the scores of every game equal to 100. In every turn the player misses a hit, he loses a point while retaining the points if a hit is successful. And the player’s score sets to 0 if the AI wins the game.

Within the game\_screen state, every time the AI takes a turn, an animation of the AI shooting a laser to its desired target can be seen. The function that is situated in the interface used for the animation of the laser is shown below:



Within the game\_screen state, every time the AI takes a turn, an animation of the AI shooting a laser to its target plays.

If the game is over, the player’s name, score, and chosen hero are displayed in the screen and the high scorer’s name, score, and chosen hero will be displayed as well. The program will check if the player had beaten the current high scorer at that time and will overwrite it if so in the form of a text file. The game\_over\_screen is shown below:



1. Sources

All the other Accessories of Cats

<http://editor.pho.to/edit/>

Bongo Cat - Bongo Cat Template 1 (Raised Paws) & Template 2 (Lowered

Paws)Retrieved From <https://knowyourmeme.com/photos/1411203-bongo-cat>

Downey, A. (2015). Think Python: How to Think Like a Computer Scientist.

9 Washburn Ave Needham MA 02492: Green Tea Press.

Explosion Gif, Laser Sound, and Laser Image- (From User:Totex)

<https://github.com/totex/Space-Invaders-in-Pyglet>

Gif Home Background

<http://skillsthatfox.tumblr.com/post/22631819741>

Intro Sound (From User: adeluc4)

<https://freesound.org/people/adeluc4/sounds/125347/>

Main Documentation Page of Pyglet - All the good stuff!

<https://pyglet.readthedocs.io/en/pyglet-1.2-maintenance/index.html>

Order of Sprite (From User: Torxed)

<https://stackoverflow.com/questions/46960665/pyglet-render-sprites-in-order-depending-on-y-value-using-batches>

Photoshop Online

<https://www.photopea.com/>

Python Tutorial: virtualenv and why you should use virtual

environments. Retrieved From

<https://www.youtube.com/watch?v=N5vscPTWKOk>

Randomization used in the Game

<https://docs.python.org/3.0/library/random.html?fbclid=IwAR2RciBPcgG4eOWOJCJMfBHIa8UZQ4TevJw_Giu_LN6EHPsl-XUf-wYfFAI>

Ship Images

<https://www.pinterest.ph/pin/453174781226288273/>

Thanos gauntlet

<https://fortnite.gamepedia.com/Infinity_Gauntlet>

venv — Creation of virtual environments. Retrieved From

<https://docs.python.org/3/library/venv.html>