****

Project Plan and Umbrella Activities

Hannah Roach

Dill Junye

Elijah Roberson  
CSC 478: Software Engineering Capstone  
11/1/2018

**CAPSTONE PROJECT**

**TABLE OF CONTENTS**

Page #

[1.0 Project Plan and Umbrella Activities 3](#_Toc532139703)

[1.1 Scope Statement, Tools and Standards 3](#_Toc532139704)

[1.2 Org Chart (5 pts) 4](#_Toc532139705)

[1.3 Gantt Chart (10 pts) 4](#_Toc532139706)

[1.3 Configuration Management Plan 7](#_Toc532139707)

[1.3 Weekly Status Report 7](#_Toc532139708)

# Project Plan and Umbrella Activities

## 1.1 Scope Statement, Tools and Standards

The program should work as a stand-alone application with no interaction among other software. There will be one player. No artificial intelligence will be used in the game. The game will be written in Python and will have a Graphical User Interface (GUI).

How the Game Works

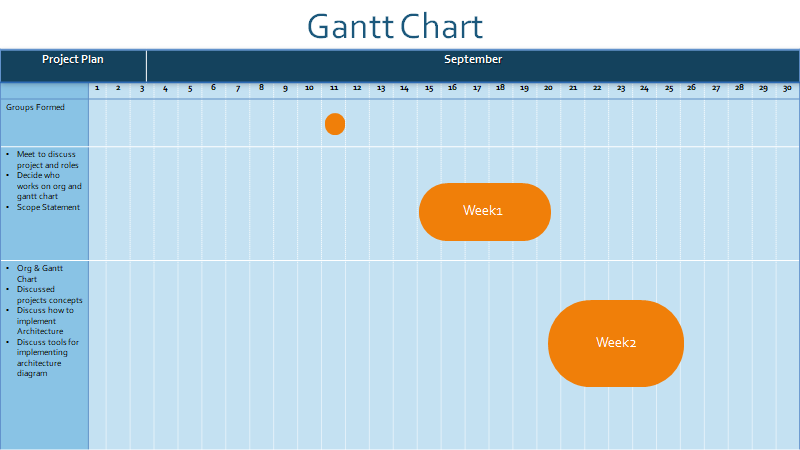
* There is 1 user player who acts as the ‘shooter’.
* The objective of the shooter is to eliminate the zombies by shooting them.
* The shooter dies if a zombie reaches him (touches him).
* The game is over when you die.

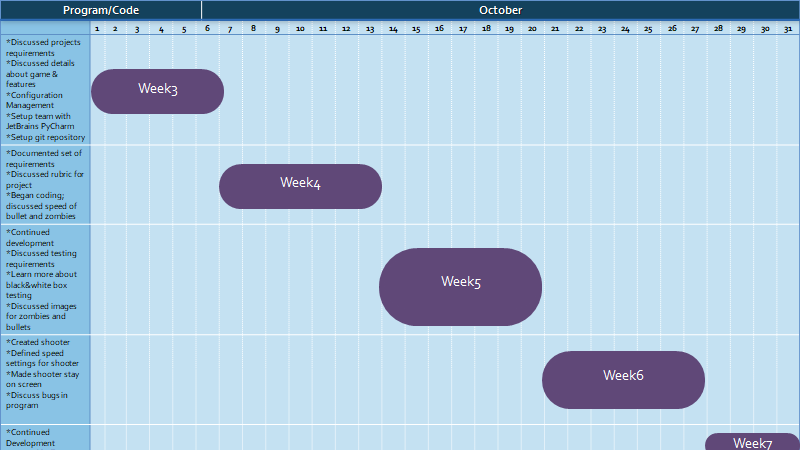
Other Useful Information

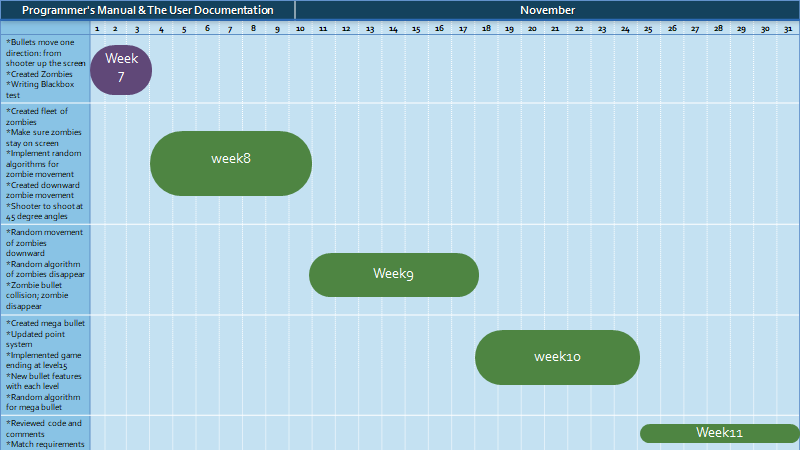
* The game gets progressively harder each level.
* Level of difficulty is determined by changes to the zombie’s quantity and size.
* For each new level, the zombies will shrink in size by 6%.
* The number of new zombies on the screen is determined by the size of the zombies.
* It takes 1 hit to kill a zombie with a regular bullet.
* Super bullets can kill multiple zombies.
* The shooter gets 4 lives.
* Score of each game session is recorded based on total zombies killed.
* To move to the next level, the shooter must kill all of the zombies.
* Each zombie is worth 1 point.
* The zombies move at a random progression forward.
* If a zombie hits the side of the screen, it changes directions.
* Zombies are staggered towards the shooter so that some zombies will approach you faster than other zombies.

## 1.2 Org Chart (5 pts)

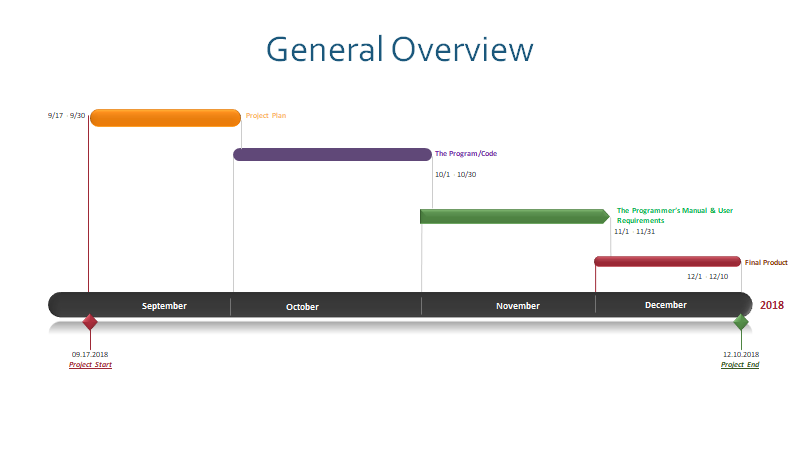
## 1.3 Gantt Chart (10 pts)











## 1.3 Configuration Management Plan

To maintain the Zombie Invasion game, our team used a simple versioning system. With each new development, we tested the program, zipped the working copy of our program, and appended the new version number; e.g., “v1”, “v2”, etc. We all installed Slack, which enabled us to easily communicate our progress. Each week we met to review what changes had been implemented and assign new tasks. We took turns developing new features, so as to avoid overwriting each other’s work. We then shared our zipped program though Slack and we all downloaded the updated version. To manage storage, the lead coder pushed our final program to a shared git repository. We initially discussed using this repository for version control but found that it was easier to simply send each other zipped files.

## 1.3 Weekly Status Report

**Week 1**

* Meet to discuss project and project roles
* Decided which team mates would implement the Org and Gant Chart
* Wrote the Scope Statement

**Week 2**

* Created the Org and Gant Chart
* Discussed concept for project; e.g., what type of game to create, what features to implement, discussed code responsibilities
* Discussed how to implement architecture
* Discussed tools for implementing architecture; UML diagrams

**Week 3**

* Discussed project requirements
* Discussed more details about the game and additional features
* Created Configuration Management Plan
* Setup team with JetBrains PyCharm
* Set up git repository

**Week 4**

* Documented set of requirements
* Discussed rubric for project
* Began coding; discussed speed of zombies and bullets

**Week 5**

* Continued development
* Discussed testing requirements
* Takeaway to learn more about black box and white box testing
* Discussed images for zombies and shooter

**Week 6**

* Continued development
* Created shooter; shooter moves left to right
* Defined speed settings for shooter
* Made sure shooter stays on the screen
* Discussed bugs in program

**Week 7**

* Continued development
* Created bullets
* Bullets move one direction; from shooter up the screen
* Created zombies
* Writing black box tests

**Week 8**

* Continued development
* Created fleet of zombies
* Make sure zombies stay on the screen
* Implemented random algorithms for zombie movement
* Created downward movement of zombie
* Shooter to shoot at 45 degree angles

**Week 9**

* Continued development
* Random movement of zombies downward
* Random algorithm for which zombies appear
* Zombie bullet collision; make zombie disappear

**Week 10**

* Created Mega Bullet with capability of shooting multiple zombies at one time
* Updated point system
* Implemented game ending after level 15
* New bullet features with each level
* Random algorithm for mega bullet after level 13

**Week 11**

* Reviewed code and comments
* Match requirements to methods in code
* Worked on creating executable file.
* Reviewed requirements and tests

**Week 12**

* Completed User’s Manual
* Completed Programmer’s Manual