**Assignment 2**

***Functional Specification***

**PROG1226 Comparative Programming Languages**

|  |  |
| --- | --- |
| Josh Cooper / 1039652 |  |
| Stephen Johnson |  |
| Computer Programming Enhanced Concepts |  |
| November 24, 2014 |  |

# Table of Contents

[Table of Contents 2](#_Toc404584655)

[Requirements Summary 3](#_Toc404584656)

[System Requirements Summary 3](#_Toc404584657)

[Use Case Design 4](#_Toc404584658)

[Activity Diagram Design 5](#_Toc404584659)

[Assumptions and Dependencies 6](#_Toc404584660)

[Feature Cuts and Unsupported Scenarios 6](#_Toc404584661)

# 

# Requirements Summary

The assignment is required to incorporate a number of programming practices. In no particular order those requirements are as follows: string handling, file I/O, parameter passing, exception handling, and be designed using an Object Oriented Paradigm. There was also an additional official unofficial requirement of rendering graphics of some form or another.

# System Requirements Summary

Visual Studio 2013 or equivalent C++11 Compiler

JDK 8u11

Python 3.0

Windows 7

1.6GHz CPU

1024MB RAM

1024x768 display

# 

# Use Case Design

The user interacts with the application via DoInput. Then DoInput either signals that an action must be performed or performs the action itself. The game object of which DoInput belongs has two methods running simultaneously, Draw and Update. Draw uses the screen manager to make drawing screen objects easy. Update uses the Invasion Manager, Screen Manager, and Bullet Manager to facilitate the updating of data for different aspects. Screen Manager will update sprites, whereas Invasion Manager will update enemy positions, and bullet manager manages everything involving bullets including physics.

# Activity Diagram Design

The user runs the executable and this immediately starts into blit3d’s run method which loops until told to stop by the user (the user quits). Run spawns a thread to run the game::update in and then goes into the execution of game::draw itself in the main thread of execution. These two threads continue on via occasional communication via shared Boolean variables. Update runs through its execution until all the time-slices have been calculated. After each time slice game::draw receives the green light to draw. When it finishes drawing game::update gets its green light to go ahead and run another update.



# Assumptions and Dependencies

Assumptions are that the player enjoys dodging swarms of bullets. The game depends on OpenGL being installed on the player’s system.

# Feature Cuts and Unsupported Scenarios

A game over for when invaders collide with the player (eg. Reach the bottom) has been cut from the game.