Project Proposal

Concept:

This project will be a game of space invaders. A game where you control a tank at the bottom of the screen and you have to use the tank to destroy aliens that are moving from side to side and down before they reach the bottom of the screen, each time they move down, their movement speed will increase. You must also avoid the bombs and shots from the aliens that will take one of your lives away (you will only have 3), to help with this there will be 4 barriers you can hide behind but they will slowly be destroyed as they get hit by more and more shots. Once a level has been cleared the player will gain a life and the next level will load (which will be the same but harder by having the aliens move faster).

Target Audience:

Age: 8-16

Gender: Any

Interests: playing video games, computers, classic arcade games.

Other: Must own a computer, will have to be relatively skilled with their computer as the game uses python to run so they will have to have python installed or be able to install it.

Requirements Met:

Validating inputs:

The player will only move on left and right arrow presses and shoot on spacebar presses.

The score-board will only allow alphabetical characters.

Interfacing with stored data:

The program will store the top 5 in a text file and will taking in that data and manipulate it when displaying the score-board it will then also save the top 5 in the same file.

Binary search or sorting algorithms:

A Binary search and sorting algorithm will be used on the scores when considering adding a new score.

Array of objects:

An array of objects will be used to store the high score data while being manipulated.

Feasibility:

Technical:

* The game will need to be object orientated both to make it easier to program and to meet some of the requirements. Python will be used for this as it is object orientated, simple to install and I already know the syntax.
* The game will need to have a graphical interface, for this Pygame and Pyganim (python libraries) will be used, this requires python and Pim (a software installation package) to be installed. This is feasible as Pim comes included in the most recent python package and installing Pygame and Pyganim only takes a one-line command.

Economic:

* The Project is economically feasible as it will not cost anything to product.

Legal:

can the solution be created and adhere to existing laws

Schedule:

(is there enough time to complete the project, are the right people and resources available when required to deliver the project on time

Requirement Specification

Functionality:

Aliens:

There will be 5 layers of aliens.

The aliens will be held in an 2D array of objects.

3 Types of aliens:

10pts:

These constitute the lower 2 layers.

20pts:

These constitute the 2 layers above the 10pts aliens.

30pts:

These constitute the top layer.

There will be 11 aliens on each row.

There will also be a bonus point 'mother ship' that will occasionally travel across the top of the screen, above all the aliens this will be worth 100pts.

Bombs:

There will be 2 types of projectile, bolt (fast) and arrow (slow).

Movement:

The aliens will all move across the screen, then when they reach the end they will shift down a bit and their movement speed will increase.

Player:

The player will only be able to move right and left. Shots will only come from the centre of the player character, this makes the game harder. The player will have 3 'lives' in total, each time the player 'dies' there will be a short explosion animation.

Barricade:

There will be 4 barricades in total, each time a shot hits a barricade a 'chunk' will be taken out of it.

Projectiles:

There will be 3 types of Projectiles in total:

The 2 bomb types of the Aliens, mentioned earlier.

The last type is the type the player will be able to 'shoot'.

Score-board:

Throughout the game the score will be kept track of and shown at the very top of the 'canvas'. Once the game is over the user will be asked to input 3 letters to be their player name and once they submit the name the score will be compared to a file containing the top 5 scores, if it is higher than any of the scores it will replace them/slot in and shift down the list, removing the new 11th highest score. Any repetitions of the scores will cause the names to be added together in a list style, e.g. YAN, BEN. This will not occur if the name if the same, if so no change will be made.

\*\*\*Make test plan\*\*\*

Implementation:

This project will be made using the object-oriented programming language Python.

To make the GUI I will be using the package Pyganim/Pygame.

Aliens:

There will be 3 types of aliens, but they will all have the same underlying behaviour.

This means the best way to implement this is by having an 'Alien' super-class, containing the code that controls the behaviour that appears in all the alien (movement, shooting, hit detection, death animation, etc...), and then using this class to create all 3 subclasses which will contain the type specific info (points awarded sprite images, etc...).

There will also be a 'mothership' sprite this will have its own class as its behaviour is different to the other aliens.

Player:

The player will have its own class that will be completely self-contained.

The player will be able to move right and left using the arrow keys and shoot by pressing the spacebar.

The death animation will consist of 2 images switching back and forth a couple times before the sprite disappears and the respawn method is called.

Barricades:

The barricades will be part of the background image (coloured green), then when the projectile detect they're touching the colour green 'above' the co-ordinates of the player, the image of the projectile will change to one of 2 black masks and will stop moving, thus making part of the barricade black.

Projectiles:

There will be 3 types of projectile in total but they will all have the same underlying behaviour, therefore I will make a super-class called Projectiles containing the code that controls movement, then I will have 3 sub classes containing the code controlling the direction, speed, image, etc...

Score-board:

The scores will be stored in a plain text file and will be sorted using an array of objects.

To check scores the find minimum, search and sort algorithms will be used.