# Requirements Document

The Oregon Trail with Python(s) – Red Team

# Software Introduction:

The purpose of our project is to recreate the popular text-based adventure decision game from the ‘70s “The Oregon Trail” using python. This will not be a 1:1 recreation but simply inspired by it. This version will have python puns included in gameplay to reference the language the game is built with.

This is a linear path game where you try to make it to the next checkpoint before running out of food. Each decision you make only affects how much time it takes to get to the next checkpoint or your food rations. If you take too long, you’ll run out of food and die. There are also special scenarios where you’ll get more food, lose food, or randomly die. It all is related to the food countdown or randomized death.

# User Requirements:

We hope to create a satirical version of the game that has a retro feel to bring a nostalgic feeling to previous players but also be appealing to newcomers. The following are user requirements that we plan to implement:

**Mandatory Requirements:**

1. Base game mechanics consisting of a player trying to traverse from checkpoint to checkpoint that is presented with random scenarios along their journey. They must manage their food and money in order to survive and arrive at the destination.
2. A user interface that has a retro feel similar to the original game with sprites for different scenes and scenarios.
3. This game will be compatible with most personal computers with an installation of Python 3.0 or newer.

**Possible Future Enhancements:**

1. Additional game mechanics and player options.
2. Bug fixes.
3. Balancing between playable classes.

# System Architecture:

The system architecture will consist of many interconnected classes. This will all be developed in Python. There will be a main player class that will interact with either a scenario manager or checkpoint manager. The player class will keep track of the money and food that the user has as well as the choice to travel or hunt. When the player makes a choice in the game, it will be delivered to the scenario manager which will handle the scenario and determine the outcome. The graphics will be developed with the TKinter library. Each scenario will have their own sprite with a part of the screen dedicated to a description of what is happening in the game. At times, the user will be shown options to interact with the game.

# System Evolution:

The game will start development in week 3 after all project planning has been completed in the first 2 weeks. There will be 2 weeks of development for the game engine itself. Then, the team will move on to scenario building in week 5. This also is a 2 week sprint. Concurrently, graphics will begin in week 5, as well, for 2 weeks. Testing will be run from week 6 to week 8, starting with the game engine which will be completed first and proceeding to the whole game which should be completed by week 7. After a thorough testing is complete, we will present the project to the client for review.

# Systems Development Life Cycle

We will be using the waterfall model for this project, as this project needs to be delivered within a fixed timeframe. The first stage is specification, where we will define the game mechanics and all other attributes and decide responsibilities for each team member. We’ll lay out a schedule of activities needed to create the game and plan when to complete them. Next, is the development phase. This is when we will be coding the game and developing the graphics. Then, we’ll proceed to validation where we will perform testing on all of the game pieces. From there, we will move on to evolution where we will make fixes to the faults we found during testing and release the game again for a final product.

# Appendices:

The software that this team will use is an operating system (preferably windows), Python, and Visual Studio Code. Other IDE software may also be used in place of Visual Studio Code. The software language we chose for this project is Python.

The hardware requirements for this project are relatively basic. We will only need a Windows computer, a monitor, and peripherals such as a keyboard and mouse.