

## Wandering in the Woods Game

### Group 4

Jeff Amorose, Jonathan Amadio, Omon Carter, Aaron Czulanda, Tesean Ferguson, yihong Ye

04/21/2021

04/25/2021

04/28/2021

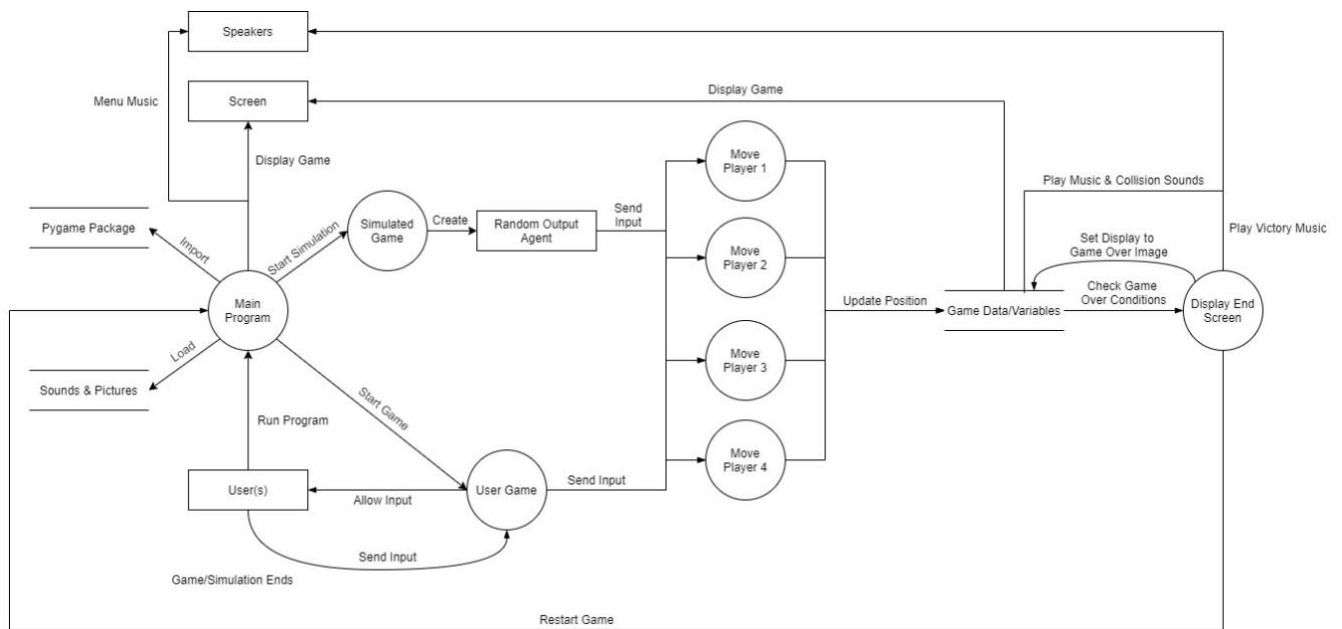
05/01/2021

## Programmer's Guide

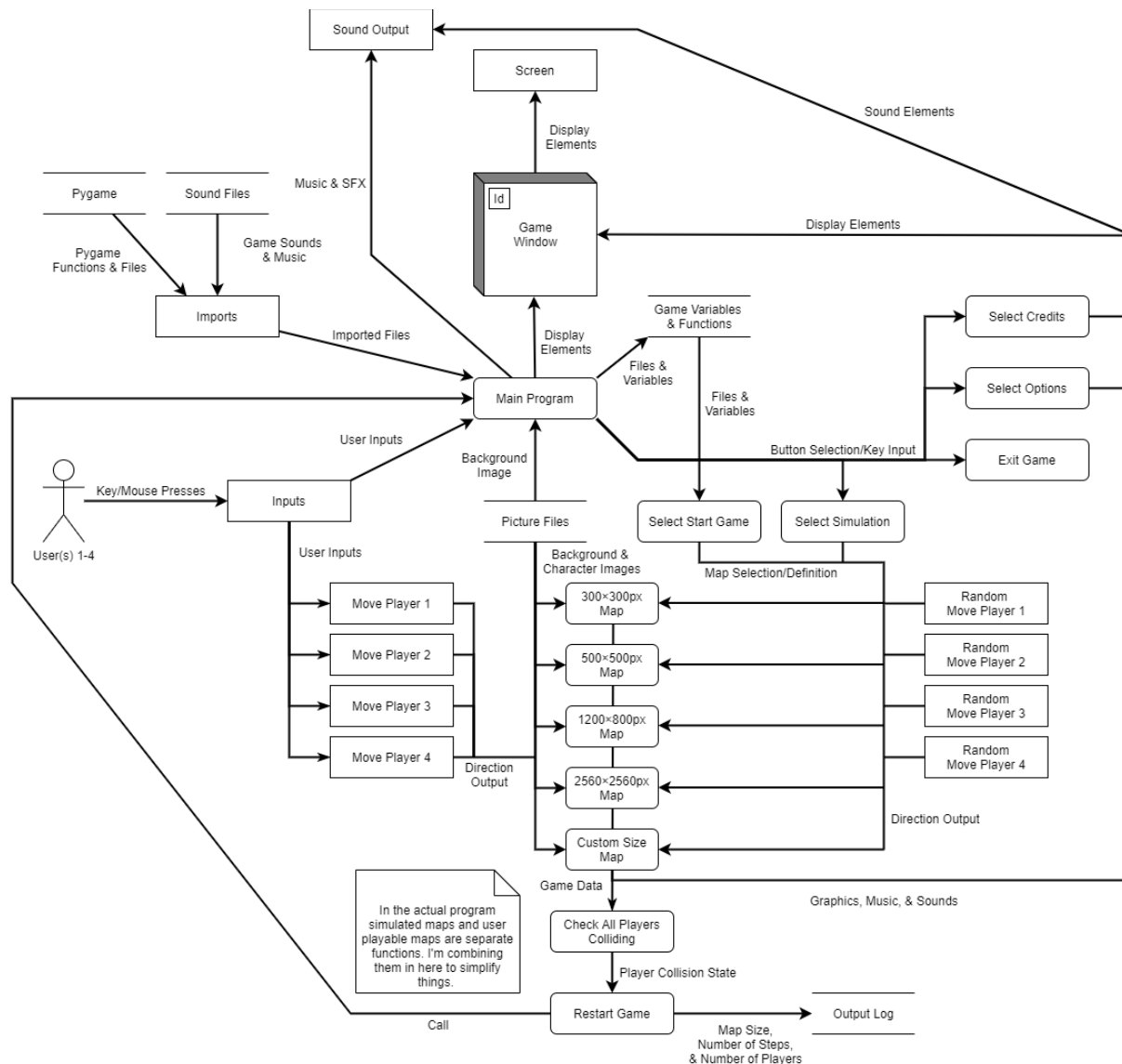
### Section 1: Assumptions about the Programmer

The programmer is going to have to have Python and Pygame installed to run the code, and they should use PyCharm, or another python IDE, to maintain it. This project was written using Python and is cross-platform. Github was used to for code exchange and storing, and the code was developed using the PyCharm environment. All images and sounds are included in the folder, and they are free to use. As for the results, they will be sent into a pre-assigned txt file in the folder. Then the data in the txt file will be used to find the average run time, shortest run time, and longest run time. From there, students can analyze the data to determine how different settings, like player count and map sizes, effect run time of the characters meeting each other.

### Section 2: High Level Design



### Section 3: More Detailed Designs



### Section 4: Installation Instructions

Installation instructions are relatively straight forward and are meant to be easily understood by someone who does not have a technical background. The steps needed to download and run the program are listed below:

1. The first step is to download Python onto your computer. This can be done by clicking this link <https://www.python.org/>, which leads you to the Python website. From there, you will want to click the download tab and select your computer's operating system. The option will be Mac OS X or Windows. Once the operating system is picked, you will be directed to another page with different Python options for you to download. You will want to click the download link that says, "Latest Python 3 Release - Python 3.9.4". This will start your download. Once finished, a window

will appear, and you will need to follow the installation instructions for downloading applications on your specific device. For Window devices, you will be given the option to add Python to PATH. You will need to click the box associated with this option.

2. The next step is to download an integrated development environment (IDE) to run the program through. Any python IDE would work, but the most common one is PyCharm. To download PyCharm, you will need to click this link <https://www.jetbrains.com/pycharm/>. From there, you will want to click the blue download button on the top right part of the window which will lead you to a download page. This page will have download options for different operating systems. Choose the operating system you used in step one and then click the download button under the “community” option. Your download will start, and you will be directed to installation instructions for downloading applications on your specific device. PyCharm should now be downloaded on your device along with python.
3. The next step is to download the gaming software for Python called pygame. To download pygame, you will need to open the terminal for mac or the command prompt for windows, or whatever the command window is for your operating system. Once in the command window, you will need to type the following command:

For Windows: `py -m pip install -U pygame --user`

For Mac OS X: `python3 -m pip install -U pygame --user`

4. Now that all the proper software is installed, the next step is to download the actual folder that contains the program and game. This can be done by clicking the following link that leads you to GitHub, where the program is stored: <https://github.com/yalerthebuilder/lostInForrest>. From this page, you will want to click the green button that says “Code”. From there, click the “Download ZIP” button. The download will start, and you will have a folder called “lostInForrest-main”. Move this folder to your desktop.
5. Next, you will want to open pyCharm. Once pyCharm is opened, go to “file” and then “open”. Find the “lostInForrest-main” folder on your desktop and open it. The folder should then appear in pyCharm. It contains everything needed to run the program and play the game. Expand the folder and double click on the file “main.py”. Once it is open, you will see a screen with code on it. Right click on the code and click the button “Run” menu. This will start the game. If you want to analyze multiple games that are ran, you will want to double click the “analyze.py” button. You will again see a page of code. Right click on the code and click “Run” menu.

This is a base tutorial on how to download the appropriate software to run the program and then the game. Please refer to the user manual on how to navigate through the menu to pick different game modes

## **Appendix A: Implementation Code**

Menu.py:

There are universal functions that can be used by all different modes such as player1 movement, player2 movement, player 3 movement, player4 movement for updating the x and y coordinate of the player. And there is also drawing function can be used to display player, map image and update them at real time(100FPS). Collision detection functions are implemented inside of movement function, if a player is hitting the wall, the collision function will be called. When each stimulation is over, there will be a function named record () being called to write down the stimulation

data into a text file for analyzing use. All different modes of the map have their own function and all of them are in the menu function that they can be directly called from.

For stimulation part, the player movement is a little bit different from manual movement. Instead of read key input, it randomly selects a movement for the player. It also has a collision detection function inside of it as well. The universal variables color, sound, fps, move speed, and font are used in every mode. Additionally, pygame is the framework that connects all the parts together.

Analyze.py:

There is a function named data sorter used to find the average, longest, and shortest path based on the list of step data in the .txt file in the results folder. The menu contains two functions, one is to analyze the stimulation data and the other is to analyze the user manual input data. All data is sorted and displayed under the given categories.

## **Appendix B: User Manual**

Upon running the program, as described in the programmers guide under installation instructions, the following menu will be displayed:

1. Start
  2. Option
  3. Credit
  4. Simulation
1. For Start Page:
    - This is a controllable game mode
    - User can go back to menu by press 'esc' key
    - User can select given map sizes or create their own
    - Users then can choose between two, three, and four players
    - Once game is over, results will be sent to the results folderThe character controls are below:
    - Patrick (user 1): is able to press keyboard "w", "s", "a", "d" to move up, down, left, right
    - SpongeBob (user 2): is able to press keyboard up, down, left, right to move up, down, left, right
    - Sandy (user 3): is able to press keyboard "1", "2", "3", "4" to move up, down, left, right
    - Squid (user 4): is able to press keyboard "6", "7", "8", "9" to move up, down, left, right
  2. For Option Page:
    - User can go back to menu by pressing "esc" key
    - User can change the volume setting mode by clicking appropriate buttons
  3. For Credit Page:
    - User can go back to menu by pressing 'esc' key
    - User can view the credit of this program
  4. Simulation:
    - User can go back to menu by pressing 'esc' key
    - User can select given map sizes or create their own

- Users then can choose between two, three, and four players
- Once number of players is chosen, the simulation will run
- Once simulation is over, results will be sent to the random.txt file

## **Appendix C: Test Plan**

### **Section 1: Introduction**

The wandering in the woods game is simple simulation that shows the random movement of two characters on a screen. The characters move throughout the screen until they meet up, or land on the same point on the screen. The base wandering in the woods game consists of two players and a set square screen playing field. We have advanced the game structure to add more features to the game and to allow users more freedom in creating the playing field and running random simulations. The purpose of our game is to give different grade groups a chance to play the wandering in the woods game at their level of education and knowledge. Our game is designed to be user friendly for all grade groups and even teachers. We also wanted to make our game attractive to the younger generation, so we chose a SpongeBob theme for characters. The game is an educational tool that allows teachers and parents to educate kids on mathematical functions in a fun manner. We set up the game in a way that allows kids to choose their grade group which then prompts them to choose additional features that they want in their game or simulation. The user guide that is attached describes in detail on how kids can choose which game they want to play. The overall idea is that the older the kids are, the more complex the game is.

For grades k-2, we wanted to be as simple as possible. When kids choose the k-2 option, they are given a set square grid and only two characters, placed at the upper right and lower left corners of the grid. We did give the option for this grade group to choose between three different square sized grids. Once they choose one of these grids, the game will automatically start, and the two characters will start to move randomly around the grid until they meet each other. Once they meet each other, the number of moves it took for the two characters to meet each other gets displayed. For grades 3-5, we added the ability for users to create their own grid. The grid could be of any size and could be rectangular, not just square. We also added the ability for this group to choose how many characters they want in the simulation. They could choose 2, 3, or 4 characters. Once the users in this group customize the grid and how many players they want, the simulation starts and runs until the players meet each other. The number of moves it takes for the players to meet are displayed and recorded in a separate text file. The users can run the simulation multiple times to determine the average time for characters to meet up, the shortest run for the characters to meet up, and the longest time for the characters to meet up. The 6-8 grade group is set up in the same way as the 3-5 grade group, but we added the option for users to control players. This allows users to determine how quickly they can get the characters to meet up. It gives this grade group some more freedom compared to the other groups so that they can experiment with different scenarios.

### **Section 2:**

To achieve the above result for the game, we did have to create a rather elaborate program. The program has many different features and functions that allow us to achieve the above results. We had to test the different parts carefully and one at a time so that we could make sure each part ran the way

it was supposed to. The following shows the parts that we tested that were vital to our completion of the program:

#### 2.1. Basic Setup of Wandering in the Woods Game (04/09/2021, Yihong and Jeff)

The first test was performed when the base gameplay for the game was developed. At this point, we had a screen open with an animated background, two characters at the top left and bottom right start positions, music playing, and the ability for users to control the two characters with the wasd and arrow keys. This test was to ensure we had audio, visual, and character movement working. It involved us moving each character around the screen and making sure the border blocked the characters from leaving the screen. We ran this portion of the program about 5 times. We achieved the result we were looking for and users were able to move characters and hear sound. We also wrote up a user-acceptance test draft which guides users through the game. The end of the game structure will be based on this test.

#### 2.2. Border Collision Sound and Character meetup (04/10/2021, Yihong, Jeff, Omon)

This test was performed check is sound plays when characters hit a wall and if the game stops when characters run into each other. We performed this test by manually moving the characters around the screen until they meet. This test also was used to make sure the step counter worked, and the total number of steps did appear when the characters met each other. We checked this function about 10 times to ensure the proper steps were tracked. We had no issues with this test, and the correct number of steps did appear.

#### 2.3. Base Testing and Debugging (04/15/2021, Yihong and Jeff)

This test was to check what we had completed so far. It involved us moving the characters and making sure the sound and visual components worked properly. We wanted to make sure these components were up and running before we started to add the rest of the features. We each ran the program multiple times, and a few small bugs were found like boarder collision not occurring and characters sometimes starting off the screen. We corrected these issues and were able to get the functions we had created to work properly.

#### 2.4. Different Map Sizes and Celebration Sound (04/17/2021, Yihong and Jeff)

More map sizes were created and different backgrounds to the maps were added. We also added celebration noises when characters met up. We wanted to make sure the different map sizes worked with the code we already had, so testing consisted of moving the characters around on all the different map sizes. We also tested the celebration sound on these different sized maps. We tested each map size around 5-6 times. They seemed to work as expected as implementing the new map sizes was not that different then our first map created.

#### 2.5. Menu (04/18/2021, Yihong)

This test was to make sure the menu worked properly, and the buttons led users to the appropriate game modes. Testing was done by clicking each of the menu buttons and checking that the buttons led to the right map size. Yihong tested each button on the menu only twice since there really should not be issues with a button leading users to another window. Each button worked accordingly.

## 2.6 Random Simulation (04/21/2021, Yihong and Jeff)

At this point, we had player movement, audio and visual components, different map sizes, and a counter added to the program. We added the random simulation for the different players and map sizes. This test consisted of running through each of the game modes with different player counts to make sure the random simulation worked properly. This test was time consuming as the larger maps took a rather long time to finish. We tested each map three times, and the time for the characters to meet was long. At first, we were worried, but realized that with the larger maps, the steps it takes for the characters to meet should be rather large. Eventually characters did meet up and the total steps were displayed.

## 2.7 Loop to Take User Back to Main Menu (04/22/2021, Yihong and Jeff)

We added the feature for users to loop back to the main menu at any point during the game. If a user accidentally clicked the wrong button, we wanted them to be able to go back a screen. Also, when the game finished, we wanted users to be able to loop back to the main menu and play again. We tested this by going through each button, map size, and player count and making sure users could go back to the previous screen. We also made sure the user could return to the menu once the simulation ended. We had no issues with this feature, and it worked flawlessly.

## 2.8 Base Testing (04/26/2021, Yihong and Jeff)

This was an overall testing day. We wanted to run through everything and make sure the functions and features we added worked correctly. We ran about two dozen tests on the whole code. We tried every option that a user could select and ran into no issues. We changed a few small details, but the game seemed to be working properly.

## 2.9 Statistics Test (04/27/2021, Yihong and Tesean)

Yihong added the ability for user to see the shortest run, longest run, and average run of each game mode and each player mode. He tested the feature a few times on each of the game modes. The statistics were exported to a .txt file and could be viewed by the user.

## 2.10 Final Testing (04/28/2021, Jeff, Jonathan, Omon, Aaron, Tesean, Yihong)

The test was to see how the whole code ran and how the data was outputted. The whole group ran tests on their own to check for issues or bugs in the code. We found a few issues with the statistics and the .txt files. Yihong adjusted some code, and we ran more tests.

## 2.11 Final Testing (04/29/2021, Jeff, Jonathan, Omon, Aaron, Tesean, Yihong)

We continued to test the code as a group. The code seemed to deliver what we wanted it to deliver. We followed the user acceptance test file on the GitHub that walked us through each separate game mode. We ran tests on every option a user picked, and everything seemed in order.



### Section 3: Signatures of the Testing Team

Jeff Amorose

DecSigned by:  
  
CE3B5B3F70484D0...

Omon Carter

DecSigned by:  
  
2DFDF03B8A8D48A...

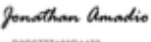
Yihong Ye

DecSigned by:  
  
D4D22C02985D48D...

Aaron Czulanda

DecSigned by:  
  
181CF1F5A12F44A...

Jonathan Amadio

DecSigned by:  
  
D305777180D1459...

Tesean Ferguson

DecSigned by:  
  
01A8C299388E4CA...