Kasha APCS1 pd04
Karina Ionkina, Dasha(Daria) Shifrina 2017-1-5

Tardy **Terminal**

As **Savvy Susan**, you've grown used to being able to solve every challenge thrown your way. The moment you get accepted at your dream job as the train conductor of the 2 train for the MTA, you are assigned the task of making sure your train is *never* delayed. As impossible as that may seem to you, you accept this daunting endeavor. In order to achieve that kind of perfection, you calculate that you must find the best path through all the stations to the Chambers Street Terminal. Will you accept this responsibility and make sure that a Stuy kid is never late to their first-period class again?

- The player will be a train operator with the goal of picking up all of the waiting passengers (number proportional to the level; as level number increases, amount of passengers to pick up increases) and reach the Terminal in a certain amount of steps.
- The operator will have to create his own path which he must follow to collect all of the people and then reach the terminal.
- This game's objective is also to improve the player's **typing skills**. In order to move in a particular direction, Savvy Susan will have to type a word assigned to the direction. For example:

  Left: Banana
  Right: Apple
  Up: Down
  Down: Up

If the player types Down, they will move Up. Mistyping a word will result in a number of steps deduction, and going in the wrong direction will derail Savvy Susan. As the levels increase, the word difficulty will increase.

Coding Progression

- MVP:
  - The game will run without any sort of algorithm, but will have a set size, set path, a set location for the passenger stations, and a set location for the terminal and train. The player will have a set amount of moves to clear the level by, requiring them to find the quickest path to the passengers and then the terminal. The player will be able to move around on the board by typing in the WASD keys. What is printed in the terminal, or the actual game, will be refreshed with every move made by the player.
- Once that is accomplished, the following will be attempted in the given order and hopefully finished:
  - Instead of typing in WASD keys to move, the player will now have to write set words to move around on the screen. Every move, the words will be refreshed and drawn from a list of vocabulary words.
  - One Passenger, Terminal, and Train will be randomly generated and assigned a location on the game board. An algorithm will be written to calculate the quickest

Path to the Passenger and then to the Terminal. The amount of moves it will take to complete the quickest path would be the amount of moves the player must finish the level by. The difficulty will always remain the same in that there will only be one Passenger on the map and the size of the map won't change. So if the player succeeds, the next level would be just as simple for them to do.

- Level component: level difficulty will increase every time the player passes the level. There will be more Passengers added with every level, the difficulty of the words to type in order to make a move will increase, and the size of the game board will increase.
- Multiplayer mode: two players will connect to each other's computer terminals in order to play. The game will be comprised of everything up to this bullet point. Every time this mode is initiated, the two players enter the exact same board game with random difficulty. The player who finds the quicker path through the Passengers to the Terminal will win.

Tools and concepts

- We will use 2D arrays to create the game board. Each spot in the 2D array would either be a Train, Passenger, Terminal, or possible Path.
- Random number generation will be used to place Passenger, Terminal, Train at random x and y coordinates. In addition to that, a random number will be used to pick the words the player must type to move from the vocabulary list.
- ArrayList: we will be storing passenger information in it and will use its expansion when the difficulty level increases
- private/public: private instance variables, such as xcor and ycor will be assigned to the classes (passenger, train, terminal).
- Working with algorithms: using the skills we learned from writing sorting algorithms, we will come up with an algorithm which will determine the fastest path the player can take to pick up all of the passengers and arrive at the terminal by using the xcor and ycor of each passenger.