CS 440 MP 1: Maze Search

Authors: Sanil Pruthi (**pruthi2**), Andrew Yang (**ayang14**)

Three-Unit Report

Introduction

We started by creating a `Maze` object. This was accomplished by passing in the maze from a text file, removing any newlines, and appending each character to a new array. When we come across the **`P`** or `.` characters, we set the coordinates of the current position or goal position respectively as a tuple of row number and column number. We then append the entire array to the maze object.

Also we implemented three helper functions, `getChar`, `adjacent`, and `debug`. `getChar` will get the character at the coordinate we pass in, and `adjacent` will return a vector containing four tuples (coordinate, direction) which are the potential path options for the current coordinates. We're also keeping track of the number of nodes we expand and the running total of the path cost for each search, incremented every time we visit a new node. The last helper function is `debug`, which prints out a maze given a certain path. We use that function to output the solutions.

Part 1.1 Overview

BFS, or **Breadth-First-Search**, expands nodes starting with the ones closest to the starting point. We start by implementing a queue and adding the current position to the queue. We also have a counter for the number of nodes expanded, and keep track of a set of visited coordinates. While the queue is not empty, we pop the first element in the queue, and check if it's a wall or the goal and respond accordingly. Walls are ignored by the algorithm as we cannot move into them. If we reach the goal we return with the path. In any other case, we fetch the adjacent coordinates, and add every valid move to our queue. This algorithm guarantees that the first complete path encountered is a path of the shortest length, but perhaps too many nodes will be expanded.

DFS, or **Depth-First-Search**, is a different approach to graph traversal than BFS. DFS uses a stack rather than a queue, so you "drill down" into paths rather than exploring several paths at once in a sense. Therefore, the order that you add adjacent coordinates onto the stack is very important and can lead to unpredictable performance. Our DFS algorithm is implemented by creating a stack where each element is a tuple containing the current position, the path so far, and the elements visited. BFS expands nodes so they're the same distance away from the starting point. This is in contrast to DFS which drills down into one direction depending on which direction you put first (East in this case). Therefore, we can't have a shared visited set for DFS like we do for BFS because the DFS algorithm could hit a dead end and we would need to backtrack and find a path that works. Also the first path DFS finds isn't necessarily a shortest path. Hence, every element on the stack needs its own set of visited nodes. While the stack is not empty, we pop off the stack, add the current coordinate to our visited set specific to this path, and check if the element is a wall. If it’s not a wall, we call the same adjacent function that we did in BFS, and append the stack with each new coordinate, and repeat the process.

Greedy BFS, or **Greedy Best-First-Search**, uses a heuristic to decide which nodes to expand first. Our Greedy BFS is similar to DFS except it uses a Priority Queue, abbreviated PQ. We use this so that the coordinates closest to the goal at the start of the queue based on a heuristic (in this case the Manhattan, or ‘taxi cab’, distance from the node to the goal). The first thing we put on the Priority Queue is a tuple with the Manhattan distance between the current position and goal position, as well as another tuple containing the current position and an empty list, representing an empty path. We get the element at the start of the PQ while the PQ isn’t empty and check to see if it’s a wall (and pass if it is). We also add the point to the common visited set, from which we can infer the frontier nodes. We then check to see if it’s the goal. If it is, we call our debug function with the current path on the stack. Else, we call our adjacent function and add the resulting moves to our Priority Queue. This algorithm returns the first path found, but this path is usually closer to the shortest path than DFS.

The **A*** search algorithm uses a more informed heuristic than Greedy BFS: we use the sum of the Manhattan distance and the path length. We now sort nodes in the Priority Queue using this heuristic. If there’s already a path found, it compares the accumulated heuristic sum with the heuristic sum of the best path found so far. If the accumulated heuristic sum exceeds the best path’s heuristic sum, then the coordinate is ignored. This algorithm is not guaranteed to have its first path found as the shortest path, so we need to cycle through the algorithm until the PQ is empty, and then return the shortest path found then.

Part 1.1 Results

Medium Maze

DFS, *Nodes Expanded*: 100, *Path Length*: 64

```
%%%%%%%%%%%%%%%%%%%%%%%
%.%   %   %   %   % % %
%.  %   %% % %%%% %%  %
%.%   % % % %  .... % %
%.%% %%%     %%.%%.%%%%
%.% %...% % % %.%..   %
%.%%%.%.%%%  % . .%% %%
%.% %. ...  % %.%.  % %
%.....% %.%%%...%.%   %
% % % %  ...%.% %.  % %
%   %%% %%%.%.%  .% %%%
% %     % %...  %.    %
%% % %%    %%%% %.%% %%
% % %   % %      .  % %
%   %% %% %% %%%%.%%  %
% % %     %     %.% % %
%%%% %% %%  %% % . % %%
```

```
% % %     %   % %.% % %
% %% % %%%%% %   ...  %
% %       %   % % %.% %
%   %% %% %%% %   %.%%%
% % %   %       % %..P%
%%%%%%%%%%%%%%%%%%%%%%%
```

BFS, *Nodes Expanded*: 350, *Path Length*: 42

```
%%%%%%%%%%%%%%%%%%%%%%%
%.%   %   %   %   % % %
%.  %   %% % %%%% %%  %
%.%   % % % %       % %
%.%% %%%     %% %% %%%%
%.% %   % % % % %     %
%.%%% % %%%  %    %% %%
%.% %...    % % %   % %
%.....%.% %%%   % %   %
% % % %.....% % %   % %
%   %%% %%%.% %   % %%%
% %     % %.....%     %
%% % %%    %%%%.% %% %%
% % %   % %    ...  % %
%   %% %% %% %%%%.%%  %
% % %     %     %.% % %
%%%% %% %%  %% % . % %%
% % %     %   % %.% % %
% %% % %%%%% %   ...  %
% %       %   % % %.% %
%   %% %% %%% %   %.%%%
% % %   %       % %..P%
%%%%%%%%%%%%%%%%%%%%%%%
```

Greedy BFS, *Nodes Expanded*: 104, *Path Length*: 56

```
%%%%%%%%%%%%%%%%%%%%%%%
%.%   %   %   %   % % %
%.  %   %% % %%%% %%  %
%.%   % % % %       % %
%.%% %%%     %% %% %%%%
%.% %...% % % % %     %
%.%%%.%.%%%  % ...%% %%
%.% %. ...  % %.%.  % %
%.....% %.%%%...%.%   %
% % % %  ...%.% %.  % %
%   %%% %%%.%.%  .% %%%
% %     % %...  %.    %
%% % %%    %%%% %.%% %%
% % %   % %      .  % %
%   %% %% %% %%%%.%%  %
% % %     %     %.% % %
%%%% %% %%  %% % . % %%
% % %     %   % %.% % %
% %% % %%%%% %   ...  %
% %       %   % % %.% %
%   %% %% %%% %   %.%%%
% % %   %       % %..P%
%%%%%%%%%%%%%%%%%%%%%%%
```

A**, *Nodes Expanded*: 109, *Path Length*: 42

```
%%%%%%%%%%%%%%%%%%%%%%%
%.%   %   %   %   % % %
%.  %   %% % %%%% %%  %
%.%   % % % %       % %
%.%% %%%     %% %% %%%%
%.% %   % % % % %     %
%.%%% % %%%  %    %% %%
%.% %.....  % % %   % %
%.....% %.%%%   % %   %
% % % %  ...% % %   % %
%   %%% %%%.% %   % %%%
% %     % %.....%     %
%% % %%    %%%%.% %% %%
% % %   % %    ...  % %
%   %% %% %% %%%%.%%  %
% % %     %     %.% % %
%%%% %% %%  %% % . % %%
% % %     %   % %.% % %
% %% % %%%%% %   ...  %
% %       %   % % %.% %
%   %% %% %%% %   %.%%%
% % %   %       % %..P%
%%%%%%%%%%%%%%%%%%%%%%%
```

Big Maze

DFS, *Nodes Expanded*: 574, *Path Length*: 154

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   %   % % %...% %   % % ........% %
%% % % %% % %.%.%  % %   %.%% % %.  %
% %       ....%.. %   % % ...    .% %
%% % %% %%.% % %.%  %    % %.% %%.%%%
%       % ..%   ..% % % % %..   %.  %
% % % %% %%.% % %.% %% %  %.% %% .% %
% %   %.....% %  .......%...  % %.. %
%  % %%.% %  % % %%%% %...%% %  %%.%%
% % %  ...  % %         %     %   ..%
%% % %% %.% % % % %% %%% % % % % %%.%
% %   % %..       %         %...%...%
% % %    %.% % %%  % %% %% %%.%...% %
%     % % ..    % %          .. % % %
% % %% % %%.% %   % %% % % % %.%%%% %
%   %   % %.% % %   % % %     .     %
% %  %%  %%.      %    % % %%%.%%% %%
%   %   %  .% % %   % %   % ...     %
% %  % % %%.  % % %%% % %% %.% %% % %
% % % ......% %   %     % ... % % % %
% %% %.% % %%  %%  %% % %%.% % %    %
% %....   % % %   %   %   .     % % %
%% .% %%%    %%% %% %  %%%.%%% %%   %
% %.      % %   % %   %   ...     % %
% %.% % %% %% % %   % % %% %.% %% %%%
%  .%         %   % %   %   ..  %   %
%%%. % % %% %   % % % %%  % %.%   %%%
% %.% %       %   %   % % %  .. %   %
%% .   %%% % %  %   %   %   %%.% % %%
% %.% %       %   %   %   %   ...   %
% %. %  % %% %  % % %%%%% % % %%.%%%%
% ..% %   %   %   %   % %   %   ..% %
%%.% %  %  % %  %   %     % % % %...%
% ..% %   %   %   % % % %   %     %.%
%%%.   % % %% %% %  % %%% %   % %  .%
%...% % %     %   %       % %   % %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

BFS, *Nodes Expanded*: 8236, *Path Length*: 62

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   %   % % %   % %   % %         % %
%% % % %% % % % %  % %   % %% % %   %
% %           %   %   % %         % %
%% % %% %% % % % %  %    % % % %% %%%
%       %   %     % % % % %     %   %
% % % %% %% % % % % %% %  % % %%  % %
% %   %     % %         %     % %   %
%  % %% % %  % % %%%% %   %% %  %% %%
% % %       % %         %     %     %
%% % %% % % % % % %% %%% % % % % %% %
% %   % %         %         %   %   %
% % %    % % % %%  % %% %% %% %   % %
%     % %       % %             % % %
% % %% % %% % %   % %% % % % % %%%% %
%   %   % % % % %   % % %           %
% %  %%  %%       %    % % %%% %%% %%
%   %   %   % % %   % %   %         %
% %  % % %%   % % %%% % %% % % %% % %
% % %       % %   %     %     % % % %
% %% % % % %%  %%  %% % %% % % %    %
% %       % % %   %   %         % % %
%%  % %%%    %%% %% %  %%% %%% %%   %
% %...    % %   % %   %           % %
% %.%.% %% %% % %   % % %% % % %% %%%
%  .%.........%   % %   %       %   %
%%%. % % %% %...% % % %%  % % %   %%%
% %.% %       %...%   % % %     %   %
%% .   %%% % %  %...%   %   %% % % %%
% %.% %       %   %.  %   %         %
% %. %  % %% %  % %.%%%%% % % %% %%%%
% ..% %   %   %   %...% %   %     % %
%%.% %  %  % %  %   %.....% % % %   %
% ..% %   %   %   % % % %...%.....% %
%%%.   % % %% %% %  % %%% %...% %...%
%...% % %     %   %       % %   % %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Greedy BFS, *Nodes Expanded*: 154, *Path Length*: 70

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   %   % % %   % %   % %         % %
%% % % %% % % % %  % %   % %% % %   %
% %           %   %   % %         % %
%% % %% %% % % % %  %    % % % %% %%%
%       %   %     % % % % %     %   %
% % % %% %% % % % % %% %  % % %%  % %
% %   %     % %         %     % %   %
%  % %% % %  % % %%%% %   %% %  %% %%
% % %       % %         %     %     %
%% % %% % % % % % %% %%% % % % % %% %
% %   % %         %         %   %   %
% % %    % % % %%  % %% %% %% %   % %
%     % %       % %             % % %
% % %% % %% % %   % %% % % % % %%%% %
%   %   % % % % %   % % %           %
% %  %%  %%       %    % % %%% %%% %%
%   %   %   % % %   % %   %         %
% %  % % %%   % % %%% % %% % % %% % %
% % %       % %   %     %     % % % %
% %% % % % %%  %%  %% % %% % % %    %
% %       % % %   %   %         % % %
%%  % %%%    %%% %% %  %%% %%% %%   %
% %...    % %   % %   %           % %
% %.%.% %% %% % %   % % %% % % %% %%%
%  .%....     %   % %   %       %   %
%%%. % %.%% %...% % % %%  % % %   %%%
% %.% % ......%.  %   % % %     %   %
%% .   %%% % % .%   %   %   %% % % %%
% %.% %       %.  %   %   %         %
% %. %  % %% % .% % %%%%% % % %% %%%%
% ..% %   %   %.  %   % %   %     % %
%%.% %  %  % % .%...%     % % % %   %
% ..% %   %   %...%.% % %...%.....% %
%%%.   % % %% %% % .% %%%.%...% %...%
%...% % %     %   %.......% %   % %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

_A_**, *_Nodes Expanded_: 343, _Path Length_: 62

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   %   % % %   % %   % %         % %
%% % % %% % % % %  % %   % %% % %   %
```

```
% %           %   %   % %         % %
%% % %% %% % % % %  %    % % % %% %%%
%       %   %     % % % % %     %   %
% % % %% %% % % % % %% %  % % %%  % %
% %   %     % %         %     % %   %
%  % %% % %  % % %%%% %   %% %  %% %%
% % %       % %         %     %     %
%% % %% % % % % % %% %%% % % % % %% %
% %   % %         %         %   %   %
% % %    % % % %%  % %% %% %% %   % %
%     % %       % %             % % %
% % %% % %% % %   % %% % % % % %%%% %
%   %   %   % % %   % %   %         %
% %  %%  %%       %    % % %%% %%% %%
%   %   %   % % %   % %   %         %
% %  % % %%   % % %%% % %% % % %% % %
% % %       % %   %     %     % % % %
% %% % % % %%  %%  %% % %% % % %    %
% %       % % %   %   %         % % %
%%  % %%%    %%% %% %  %%% %%% %%   %
% %...    % %   % %   %           % %
% %.%.% %% %% % %   % % %% % % %% %%%
%  .%.........%   % %   %       %   %
%%%. % % %% %...% % % %%  % % %   %%%
% %.% %       %...%   % % %     %   %
%% .   %%% % %  %...%   %   %% % % %%
% %.% %       %   %.  %   %         %
% %. %  % %% %  % %.%%%%% % % %% %%%%
% ..% %   %   %   %...% %   %     % %
%%.% %  %  % %  %   %.....% % % %   %
% ..% %   %   %   % % % %...%.....% %
%%%.   % % %% %% %  % %%% %...% %...%
%...% % %     %   %       % %   % %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Open Maze

DFS, *Nodes Expanded*: 674, *Path Length*: 194

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%..........                         %
%.....    .                         %
%.....    .                         %
%.....    .%%%%%%%%%%%%%%%%%%%%%    %
%.....    .%.................  %    %
%.....    .%.     %%%%%%%%%%.  %    %
%.....    .%.     %........%.  %    %
%.....    .%.     %...    .%.  %    %
%.....    .%.     %...    .%.  %    %
%.....    .%.     %...    P%.  %    %
%.....    .%.     %... %%%%%.  %    %
%.....    .%.     %..........  %    %
%.....    .%.     %..........  %    %
```

```
%.....    ...     %..........  %    %
%.....            %%%%%%%%%%%%%%    %
%.....                              %
%.....                              %
%......                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

BFS ran into MemoryErrors for open maze. That was expected since BFS expands a lot of nodes.

Greedy BFS, *Nodes Expanded*: 6297, *Path Length*: 60

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
%                                   %
%                                   %
%          %%%%%%%%%%%%%%%%%%%%%    %
%          %.................  %    %
%          %.     %%%%%%%%%%.  %    %
%          %.     %        %.  %    %
%          %.     %        %.  %    %
%          %.     %        %.  %    %
%          %.     %.......P%.  %    %
%          %.     %.   %%%%%.  %    %
%          %.     %..........  %    %
%          %.     %            %    %
%     .......     %            %    %
%     .           %%%%%%%%%%%%%%    %
%     .                             %
%     .                             %
%     .                             %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

A**, *Nodes Expanded*: 660, *Path Length*: 54

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
%                                   %
%                                   %
%          %%%%%%%%%%%%%%%%%%%%%    %
%          %.................  %    %
%          %.     %%%%%%%%%%.  %    %
%          %.     %        %.  %    %
%          %.     %        %.  %    %
%          %.     %        %.  %    %
%          %.     %   ....P%.  %    %
%          %.     %   .%%%%%.  %    %
%          %.     %   .......  %    %
%          %.     %            %    %
%     .......     %            %    %
%     .           %%%%%%%%%%%%%%    %
%     .                             %
%     .                             %
%     .                             %
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Part 1.2 Overview

In this task, we modify the A* algorithm so that it penalizes forward movement and turns differently. Penalizing forward movement more than turns results in a better performing algorithm with the given test cases. This is due to the long straightaways in the two mazes. The algorithm will first explore paths that have a lot of turns due to their lower path cost, as determined by our heuristic. Our modified path cost for A* results in better performance for the given test mazes.

Part 1.2 Results

Small Turns

Forward: 2, Turn: 1, *Nodes Expanded*: 73, *Path Length*: 52

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                             P%
% %  %%%%%%%%%%%%%%%%%%%%%%%%%.%
% %                %.......%...%
% %    %%%%%%%%%%% %.%%%%%.%.%%%
% %%%% %           %...%...%...%
%   %    %%%%%%%%%%%%%.%.%%%%%.%
%               % .....%.......%
% %%%%%%%%%%%%%%%%.%%%%%%%%%%%%%
%                 .............%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Forward: 1, Turn: 2, *Nodes Expanded*: 171, *Path Length*: 52

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                             P%
% %  %%%%%%%%%%%%%%%%%%%%%%%%%.%
% %                %.......%...%
% %    %%%%%%%%%%% %.%%%%%.%.%%%
% %%%% %           %...%...%...%
%   %    %%%%%%%%%%%%%.%.%%%%%.%
%               % .....%.......%
% %%%%%%%%%%%%%%%%.%%%%%%%%%%%%%
%                 .............%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Forward: 1, Turn: 0*, *Nodes Expanded*: 74, *Path Length*: 52

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                             P%
% %  %%%%%%%%%%%%%%%%%%%%%%%%%.%
% %                %.......%...%
% %    %%%%%%%%%%% %.%%%%%.%.%%%
% %%%% %           %...%...%...%
%   %    %%%%%%%%%%%%%.%.%%%%%.%
%               % .....%.......%
% %%%%%%%%%%%%%%%%.%%%%%%%%%%%%%
%                 .............%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Big Turns

Forward: 2, Turn: 1, *Nodes Expanded*: 284, *Path Length*: 58

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
% %%  %% %% %% %%%%  %%   % %% % %  %
%   %               %   %         % %
% %%% %% % %% % % %   %   %%%%% %%  %
%     % % % % %     %   %     % %   %
%  %% %  % %%   %% %% % %%%% % % %% %
%   %   %     % %   %   %         % %
% % % % %% %%   % %   % % % %%% %   %
% %           %     %     %   %   % %
% %%% %%% %% % % %%% % %%  % %%% %  %
%     %   %       %   % % %   %   % %
% %%% %% % % %%%% %%%%%%    % % % % %
% %     % % %     %   % % % %       %
%  %% %%    %% % % % %% %  % %% %%  %
%       % % %             %   %     %
% %%%% %     % % %% % % %   %  % %% %
%.% % % % % % % % %   %   % % %     %
%. % %  %% %%% %  %% %% %   %% %% % %
%.% % %   %   % %         % %   % % %
%.     %% % %    % % % % %  % %% % %
%.% % %   %   % %           % %     %
%.%% %% %  % %%%%% %%%% % %%%   %%% %
%.... %   %           % % %   %     %
%  %.% %%%  %% % %% % %%%% % %  % % %
% % .   % % %           %     % % % %
% %%.%%% %  %%% % %% % %% % %% %%   %
%   ....% % % % %   %     %       % %
% % %%%.% %.....% % %% % %  % %% %% %
% %    .....% %.% % % %   %   % %   %
%  %%% % % %%  ...% %%% %%  % % %%  %
%   %   % % % % %.....    %   %   % %
% %  % %   %%    %% %.%%%% %%  %%%  %
% % %   % %   % %    .......% %.....%
% %% % %  % %   % %% % % %%.% %.% %.%
%       %     %   % % % %  .....% %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Forward: 1, Turn: 2, *Nodes Expanded*: 434, *Path Length*: 58

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
% %%  %% %% %% %%%%  %%   % %% % %  %
%   %               %   %         % %
% %%% %% % %% % % %   %   %%%%% %%  %
%     % % % % %     %   %     % %   %
%  %% %  % %%   %% %% % %%%% % % %% %
%   %   %     % %   %   %         % %
% % % % %% %%   % %   % % % %%% %   %
% %           %     %     %   %   % %
% %%% %%% %% % % %%% % %%  % %%% %  %
%     %   %       %   % % %   %   % %
% %%% %% % % %%%% %%%%%%    % % % % %
% %     % % %     %   % % % %       %
%  %% %%    %% % % % %% %  % %% %%  %
%       % % %             %   %     %
% %%%% %     % % %% % % %   %  % %% %
%.% % % % % % % % %   %   % % %     %
%. % %  %% %%% %  %% %% %   %% %% % %
%.% % %   %   % %         % %   % % %
%.     %%  % %    % % % % %  % %% % %
%.% % %   %   % %           % %     %
%.%% %% %  % %%%%% %%%% % %%%   %%% %
%.    %   %           % % %   %     %
%. % % %%%  %% % %% % %%%% % %  % % %
%.%     % % %           %     % % % %
%.%% %%% %  %%% % %% % %% % %% %%   %
%...    % % % % %   %     %       % %
% %.%%% % %...  % % %% % %  % %% %% %
% %.........%.% % % % %   %   % %   %
%  %%% % % %%.....% %%% %%  % % %%  %
%   %   % % % % %...      %   %   % %
% %  % %   %%    %%.% %%%% %%  %%%  %
% % %   % %   % %  .........% %.....%
% %% % %  % %   % %% % % %%.% %.% %.%
%       %     %   % % % %  .....% %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Forward: 1, Turn: 0*, *Nodes Expanded*: 235, *Path Length*: 52

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                                   %
% %%  %% %% %% %%%%  %%   % %% % %  %
%   %               %   %         % %
% %%% %% % %% % % %   %   %%%%% %%  %
%     % % % % %     %   %     % %   %
%  %% %  % %%   %% %% % %%%% % % %% %
%   %   %     % %   %   %         % %
% % % % %% %%   % %   % % % %%% %   %
% %           %     %     %   %   % %
% %%% %%% %% % % %%% % %%  % %%% %  %
%     %   %       %   % % %   %   % %
% %%% %% % % %%%% %%%%%%    % % % % %
% %     % % %     %   % % % %       %
%  %% %%    %% % % % %% %  % %% %%  %
%       % % %             %   %     %
% %%%% %     % % %% % % %   %  % %% %
%.% % % % % % % % %   %   % % %     %
%. % %  %% %%% %  %% %% %   %% %% % %
%.% % %   %   % %         % %   % % %
%.     %%  % %    % % % % %  % %% % %
%.% % %   %   % %           % %     %
%.%% %% %  % %%%%% %%%% % %%%   %%% %
%.    %   %           % % %   %     %
%. % % %%%  %% % %% % %%%% % %  % % %
%.%     % % %           %     % % % %
%.%% %%% %  %%% % %% % %% % %% %%   %
%...    % % % % %   %     %       % %
% %.%%% % %...  % % %% % %  % %% %% %
% %.........%.% % % % %   %   % %   %
%  %%% % % %%.....% %%% %%  % % %%  %
%   %   % % % % %...      %   %   % %
% %  % %   %%    %%.% %%%% %% %%%   %
% % %   % %   % %  .........% %.....%
% %% % %  % %   % %% % % %%.% %.% %.%
%       %     %   % % % %  .....% %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Part 1.3 Overview

There were two primary difficulties in the implementation of Pac-Man with a ghost. The first was simulating the ghost's movement. We were eventually able to find a pattern in the ghosts movement, which greatly simplified our simulation.

The second difficulty was handling the difficulty of getting past the ghost. In the "smallGhost" maze, we weren't able to get past the ghost if we went straight for the goal. To solve this issue, Pac-Man loiters at the beginning. Loitering is visiting a previously visited coordinate for the purposes of stalling. While loitering, the ghost's position changes, possibly allowing Pac-Man to pass. We handled the loitering case by assigning a penalty equal to the number of squares in the maze. We allowed the algorithm to expand into a coordinate that we already visited, but when the algorithm does so, we attach a penalty to the heuristic to ensure that we only use that coordinate as an absolute last resort if Pac-Man cannot get past the ghost.

Part 1.3 Results

Small Ghost

Nodes Expanded: 5474, *Path Length*: 21

```
%%%%%%%%%%%%%%%%%%%%%%
% %%        % %      %
%    %%%%%% %.%%%%%% %
%%%%%%     P..%      %
%    % %%%%%%.%% %%%%%
% %%%% %gGgg..ggg%   %
%        %%%.%%%   % %
%%%%%%%%%%... %%%%%% %
%..........%%        %
%%%%%%%%%%%%%%%%%%%%%%
```

Medium Ghost

Nodes Expanded: 107, *Path Length*: 26

```
%%%%%%%%%%%%%%%
%.      %     %
%.%%%% %%% %  %
%.%   % %   % %
%...%% %% %%  %
% %.....  % % %
%% % %%.%%%%  %
% %Gggg.gggg% %
%   %%%.% %%% %
% %   %.%.... %
% %%%  ...%%.%%
%   % % % % ..%
%%%%%  %%   %.%
%     %   % %P%
%%%%%%%%%%%%%%%
```

Big Ghost

Nodes Expanded: 318, *Path Length*: 70

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%.%     % %   % %   %               %
%...%% %%  % %  % %  % % % %% % %% %%
% %.% %   %   %   % %     %   %     %
%  .    %% % %%%%   %% % %%% %  %% %%
% %.% % %         %           %   % %
% %.     % % %% %%%% %% % %% % % %  %
%  .% % %         %       %       % %
%%%.%% % % % % %%%  % % %%  %% % % %%
%  .....      %   % % %   % % %     %
% %% %%.%%% %   %       %%      %% %%
%      .      %   % % %   % % % %   %
% %%% %.%%%% % % %% % % %%%%    % % %
%   %  .%   %   %       % % % % %   %
%% % %%...% % %   % %% %% %%  %  % %%
% %   % %.      %   %       %   %   %
% % % %  .%% %% %%%%  % % %%% %   % %
%       %..       % %     % %   %   %
%% % %%  %.%% % %  % % %%    %%% % %%
```

```
%   %     ..      % % % % % %       %
% %  % % %%.% % % %         %% %%% %%
%   % %   %..ggggG% % % % % %   %   %
% %%  %%%  %.%% %%    % %% %  %% %% %
%   %   % % . %   % %       %   %   %
%% % % %% %%.% %%.....% %% % %%  % %%
% %   %     ......% %.... %     %   %
%   %   %% %%% %% %   %%.%% % % % % %
% %   %     % % %   %   ...     % % %
%   %  % %% %  %% % % % %%.%%% %%  %%
% % % % % % % %       % % ..% % % % %
% %   %      % % % %%     %.....% %%%
%   %   % % %   %   % % % % % %.....%
%%% %% % %    %   %  % %  %% % %%%%.%
% %       % %   % % %   % %       %.%
% % % % %    % %% % % %  % %% % %  .%
%         % %   %       %     %   %P%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Contributions

Sanil Pruthi (pruthi2)

Wrote BFS and Greedy BFS algorithms, and worked with Andrew to develop modified heuristics for part 1.2, as well as wrote the majority of the report.

Andrew Yang (ayang14)

Wrote DFS and A* algorithms, developed heuristics for part 1.2, and modified A* algorithm for parts 1.2 and 1.3.