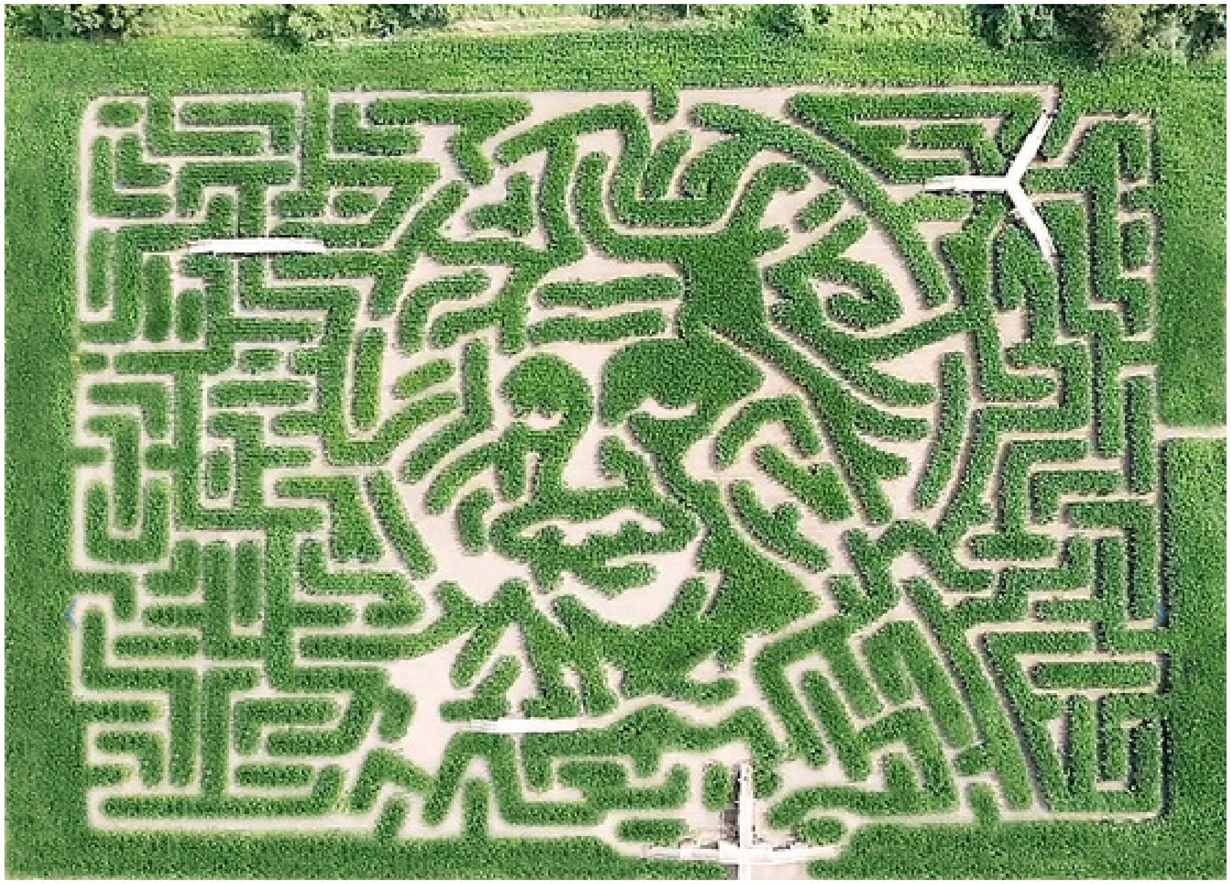
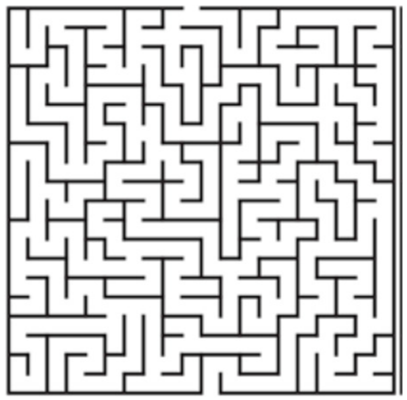
Davis Mega Maze



Objective

Given a maze with a unique start and a unique end, find a generic algorithm that will successfully get your character through the maze. Try “coding” your solution using the provided programming concepts (in the form of code puzzle pieces). There are green function cards, black loop cards, and blue conditional statement cards available.



Program Development Process

1. Understand the Problem

2. Develop and Describe an Algorithm

3. Test the algorithm with different inputs (in this case mazes)

4. Translate the algorithm into code

5. Test the program – Does it work for all given mazes?

Algorithm Development

Using Functions

The green command cards are available functions:

- **Go Forward One Space**
- **Turn Right 90°**
- **Turn Left 90°**

Using Loops

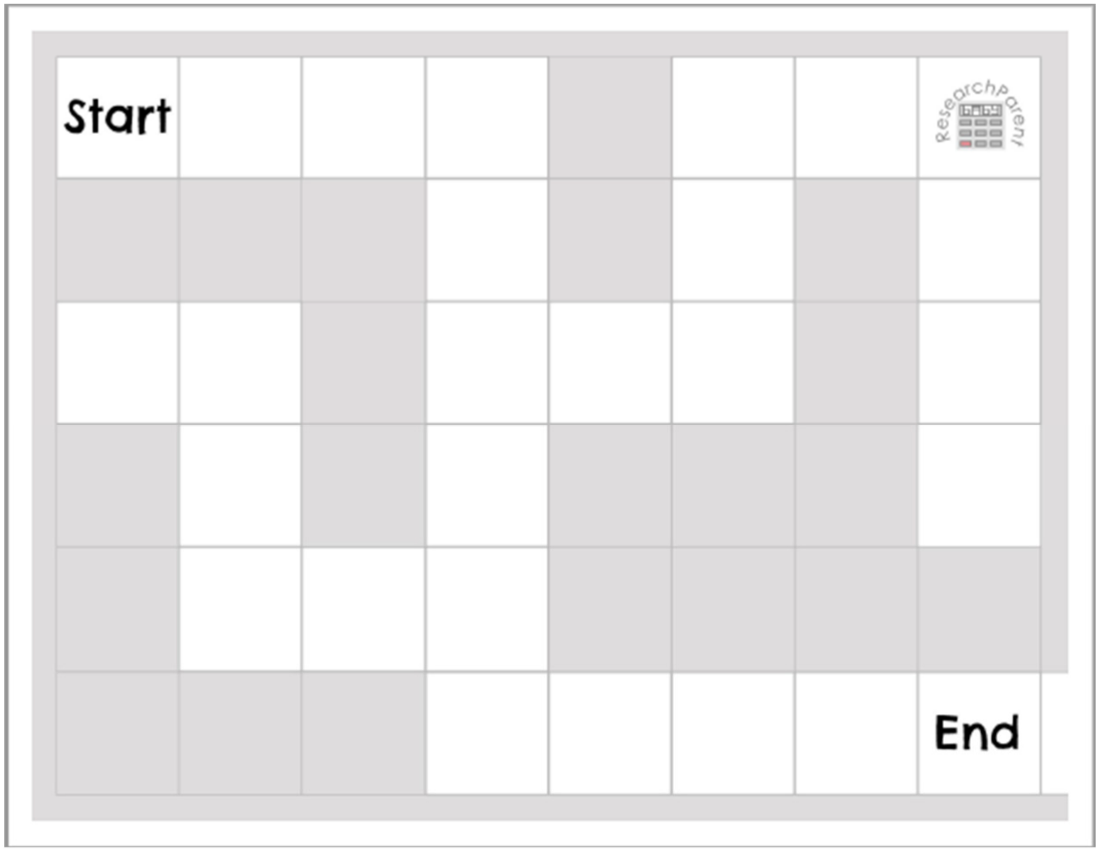
Using the **black** cards that say "**For ___ Steps**", write a solution that performs the green command card(s) repeatedly rather than duplicating the same card over and over.

Using Conditional Logic and Abstraction

Using the blue cards (if … else if … else), find a solution by thinking more abstractly.

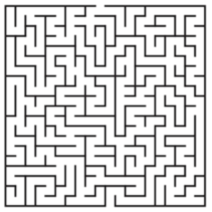
- **IF** means definitely do this if the specified condition is true.
- **ELSE IF** means do this if the specified condition is true *AND* the **IF** and **ELSE IF**'s that came before were not true.
- **ELSE** means do this if everything above has failed.

Test Algorithm



WHILE
still in maze

A-Mazing Algorithms



Objective: Given a maze with a unique start and a unique end, find a generic algorithm that will successfully get your character through the maze. Try "coding" your solution using the provided programming concepts (in the form of code puzzle pieces). There are green function cards, black loop cards, and blue conditional statement cards available.

Program Development Process:

1. Understand the Problem
2. Develop and Describe an Algorithm
3. Test the algorithm with different inputs (in this case mazes)
4. Translate the algorithm into code
5. Test the program – Does it work for all given mazes?

Possible Implementation Levels:

Level 1: Using Functions

Using only the green command cards (**Go Forward (one space)**, **Turn Right (90°)**, and **Turn Left (90°)**), the first step is to write a solution from the point of view (reference frame) of the character in the maze. To do this, place the figure at the start of the maze and find a solution.

Level 2: Using Loops

Using the **black** cards that say "**For ___ Steps**", write a solution that performs the green command card(s) repeatedly rather than duplicating the same card over and over.

Level 3: Using Conditional Logic and Abstraction

Using the blue cards (if … else if … else), find a solution by thinking more abstractly.

Think about all the possibilities and evaluate what you should do in each case and in what order (or with what priority).

IF means definitely do this if the specified condition is true.

ELSE IF means do this if the specified condition is true *AND* the **IF** and **ELSE IF**'s that came before were not true.

ELSE means do this if everything above has failed.

END means this is the end of the **IF** statement.

For example, our character in a maze only faces a limited number of options. He/she may or may not have walls in front of him/her or to his/her left or to his/her right.

Level 4: Finding a "Generic" Solution for all mazes

Do you have a solution that will work for every maze?