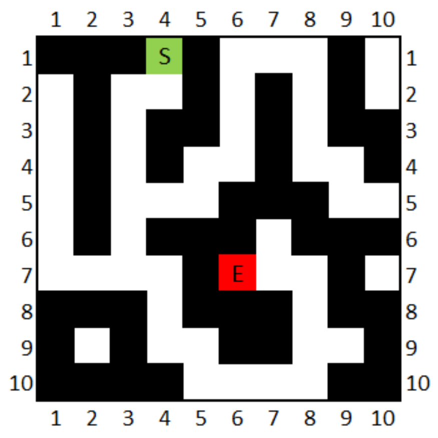

24-780 B—ENGINEERING COMPUTATION

Assigned: Wed. Oct. 12, 2022
Due: Tues. Oct. 25, 2022, 11:59pm
but bulk of work to be done in lecture on Oct. 12

Problem Set 6: Maze Navigation

For this assignment, you are asked to add some more functionality to the maze application you developed for PS05. As much as possible we will try to avoid making too many changes from what we already have. Thus, the maze input file format will not change (see figure at right).

```
10
1 4
7 6
1 1 1 0 1 0 0 0 1 0
0 1 0 0 1 0 1 0 1 0
0 1 0 1 1 0 1 0 1 1
0 1 0 1 0 0 1 0 0 1
0 1 0 0 0 1 1 1 0 0
0 1 0 1 1 1 0 1 1 1
0 0 0 0 1 0 0 0 1 0
1 1 1 0 1 1 1 0 1 1
1 0 1 0 0 1 1 0 0 1
1 1 1 1 0 0 0 0 1 1
```

Once again, I've organized the assignment as a series of tasks, and I recommend that you accomplish them in order (although you know yourselves better than I ever could).

When creating assignments that build on previous work, I do not want you to start the current assignment at a disadvantage. For example, if you are not happy with your solution for PS05, it would be awful if I asked you to build on that program. Thus, I am officially making clear that you have the following choice for Task 4:

- Take your PS05 solution
  (advantage is that you carry through with your assumptions and data models rather than trying to figure out mine)

OR

- Use my *ps05_maze_ng27_fa22(SolutionB).zip* solution now on Piazza
  (advantage is that you "cut your losses" and have few extra features already in there)

Task 4 (Relationship from Entity to Maze)

As discussed in Lecture 12 (Oct. 10, 2022), it may be easier to work this maze game if we had a more formal relationship from Entity to Maze. Some of you have already done this. Add a pointer in Entity.h:

```
Maze *myMaze = nullptr;
```

These three functions:	Now become:

`void move(Maze &aMaze, int direction);`
`bool reset(Maze &aMaze);`
`bool reachedGoal(Maze &aMaze);`
	`void move(int direction);`
	`bool reset();`
	`bool reachedGoal();`

The body of the functions need the change slightly, but you can probably do most of the work with a simple find and replace ("aMaze." is replaced by "myMaze->")

You'll also need to add a function in Entity.h and call it in main() somewhere (before loop, probably):

```
void setMaze(Maze &aMaze) { myMaze = &aMaze; };
```

Task 5 (Graphical editing of maze)

Give the user the ability to edit the maze graphically by providing a function in *Maze* class like this:

```
// mouseX and mouseY are the pixel coordinates of a left click
// key is either FSMOUSEEVENT_LBUTTONDOWN, FSKEY_S, or FSKEY_E
//   mouse click toggles obstacle state of cell/block below mouse location
//       unless block is start or end (start and end MUST be navigable
//   if block is navigable, S and E will set it to either start or end
//       note, start or end may become undefined (-1, -1)
void changeMapState(int mouseX, int mouseY, int key);
```

You also need to add some keystroke detection and mouse event stuff in main() to make this work.

Task 6 (Save maze to file)

Add a feature to your program that will ask the user for a filename and will then save the maze data to that file. Note that you should be able to make use of your *operator*<< function. The function will be called when the user presses the 'W' key. Note: some of you have already done this as part of PS05 (even though you were only required to code the *operator*<< function), so great job.

Task 7 (Find shortest path)

Now we get to the engineering part of the assignment and I expect most of your effort will be devoted to this task. We will discuss this task in the lecture at length and even work on some of the code.

Add a function to Entity class as follows:

```
void findShortestPath();
```

The function will find the shortest path through the maze from the current location of the entity to the end square of the maze. The function is called when the user presses the 'G' key (as in "GO").

You can choose how to use the results of the function to display and/or use the shortest path.

Deliverables

1 zip file, containing everything that is need for your whole project

Upload the zip file to the class Canvas page before the deadline (Tuesday, Oct. 25, 11:59pm).

Learning Objectives

Implement breadth-first search algorithm to solve path navigation problem

Read/write data from/to permanent storage.

Use OpenGL primitives and allow user graphical interaction (through mouse click).

Searching references (online or textbook) for C++ library functions.