

CSC 680: Artificial Intelligence

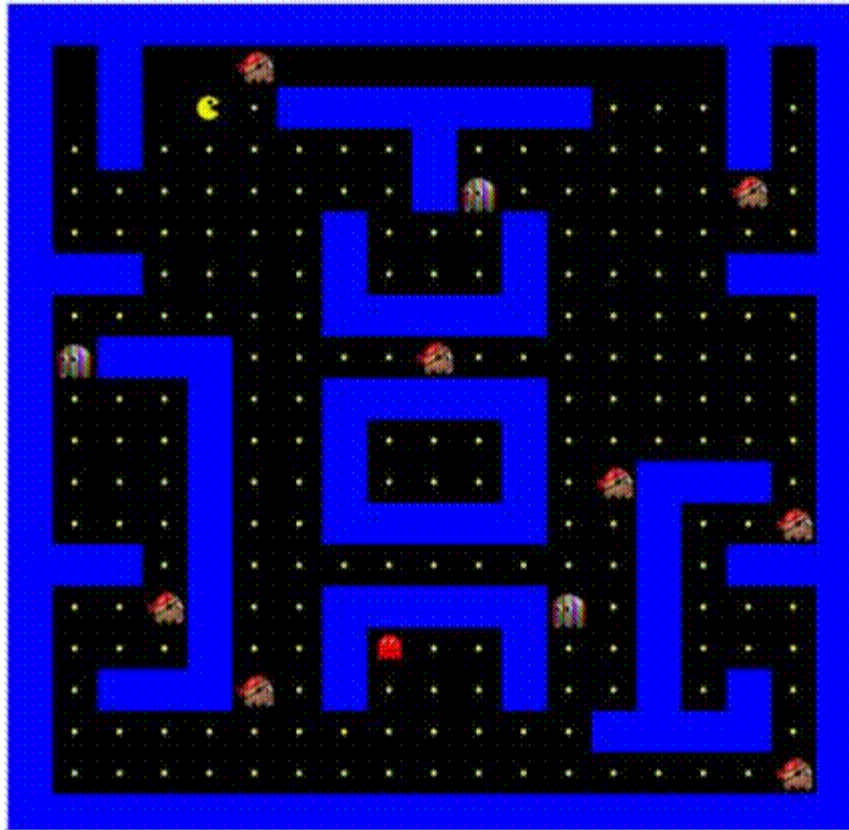
Fall 2012

Assignment #2 (30 Points)




Due date: Wednesday October 3, 2012 (in class)

Objective: To gain programming experience using a heuristic-based search method.

In this assignment you may work in groups of two to implement a game application of the A* algorithm. The main character, *Pacman*, sets out to find the red ghost which is hiding somewhere in the maze.



The cast of characters/objects in this game includes:

Pacman		Pacman's main goal is to find and destroy the red ghost with shifty eyes.
Pirate		Pacman's enemy #1 Pacman will incur the highest cost for visiting a maze cell occupied by a Pirate.
Stripes		Pacman's enemy #2 Pacman will incur the next highest cost for visiting a maze cell occupied by Stripes.
Ghost		The red ghost will run away for extra credit; otherwise, it stays in one spot.



Power Pellet



Pacman consumes power pellets as it moves around the maze.

Pacman will incur the lowest cost for visiting the cells with power pellets.

Programming Guidelines

1. The ghost does not move from its initial position.
2. Use a priority queue to implement the **OPEN** list. Here is a [sample program](#) that shows how *java.util.PriorityQueue* can be used.
3. Use *Manhattan* distance for h^* and, in general, experiment with the evaluation function of the form:

$$f(n) = \alpha \cdot g(n) + \beta \cdot h^*(n)$$

Use appropriate values for α (say, 2) and β (say, 10). Actual path cost $g(n)$ must be computed using the information provided in the table shown above.

4. The seed programs you need to get started (*Maze.java* and *MazePanel.java*) along with the image files can be found [here](#).

Input/Output Guidelines

1. To compile and run your application:

```
javac Maze.java
java Maze
```

2. The animated Java application must actually show Pacman taking the optimum path through the maze as the power pellets disappear along that path.

Grading Guidelines

- I do not accept multiple submissions
- Read [my policy on late submissions](#)

Technical Correctness	90%
<ul style="list-style-type: none"> • Program compiles and executes correctly • Proper error checking is performed on input file, if applicable, and during execution 	

<ul style="list-style-type: none"> • Assignment objectives are met • Specified I/O guidelines are followed 	
Coding Style & Documentation <ul style="list-style-type: none"> • Opening comments (author name, assignment number, date, purpose, etc.) are included • General comments appear in code • Code is properly indented and spaced • Meaningful variable names are used • Code is properly modularized 	10%

Coding Style Guidelines

1. Include the following opening comments in the beginning of each program you submit:

```

/* *****
Author's name(s):
Course Title:
Semester:
Assignment Number
Submission Date:
Purpose: This program ...
Input:
Output:
Help: Acknowledge any help you might have received or simply indicate that you worked alone.
***** */

```

2. Recall from our in-class discussion that 10% of your grade will be based on how well you develop and present your work:
 - a. Use suitable names for Java classes, variables, and constants.
 - b. Provide appropriate and non-trivial comments for your well-indented code.
 - c. Every method must have an opening **Purpose** comment with **Input** and **Output**, when appropriate.

```

//*****
//*** Purpose: This method adds a node to the head of a given list
//*** Input:   A DLL node
//*** Output:  None
//*****
public static void addFirst(DLL L){
...
}

```

- d. Modularize your programs.
3. The importance and benefits of a *consistent coding style* are well known. A consistent style:
- a. Improves the readability, and therefore, maintainability of code.
 - b. Facilitates sharing of code among programmers, especially teams of programmers working on the same project.
 - c. Allows easier development of automated tools to assist in program development, such as tools which automatically format or pretty-print source code.
 - d. Makes it easier to conduct code reviews, another software engineering process with well-known benefits. In turn, a practice of regular code reviews can help enforce a consistent style.

Extra Credit

- Modify A* in order to deal with the **moving target problem**, i.e., the ghost moves around randomly.
- **Hint:** Remember the ghost's last 5 moves and guess its next location using the cubic-spline method.

Submission Guidelines

1. Place your program files (**Maze.java**, **MazePanel.java**, etc.) and the supporting data files (if any) in a directory called **YourName-HW2** and archive it using Winzip or any other compression utility that you might have available on your PC. Include any special instructions to run your code in a **README** file.
2. Email me the zipped archive on or before the due date using the following subject line for your email message:

Your Name – AI – HW2

In addition, submit a **printout of all your program files** in class on the due date.