CS-361L Artificial Intelligence Lab 02

Type of Lab: Open Ended Weightage: 5%

CLO 1: Apply Informed and Uninformed Search Techniques and build the ability to theoretical and practical understanding of Blind and Informed machine search and machine learning techniques.

Student Understand the search project of PacMan and where to write the code to navigate the pacman.	**Cognitive/Understanding**	CLO1	Rubric A
Demonstrate ethical and professional responsibilities involved in completion of Tasks	**Affective/Valuing**	(CLO6)	Rubric B

Reference: The Lab contents are extracted from the BerkeleyX/CS188

Rubric A: Cognitive Domain

Evaluation Method: GA shall evaluate the students for Question 1-3 according to following rubrics.

CLO	**0**	**1**	**2**	**3**	**4**	**5**
CLO1	Student not able to run the pacman code	Student understand where to write the code for actions	The pacman is navigating but not reaching at goal for any single maze	The pacman is navigating and reaching at the goal for atleast one maze	The pacman is navigating and reaching at the goal for atleast two maze	The pacman is navigating and reaching at the goal for all mazes
Roll Number						

Reference: The Lab contents are extracted from the BerkeleyX/CS188

Reference: The Lab contents are extracted from the BerkeleyX/CS188

Rubric B: Affective Domain: Lab Staff shall help GA in evaluation of the students for their CLO 6

CLO 6	**0**	**1**	**2**	**3**
Demonstrate ethical and professional responsibilities involved in completion of Tasks.	Student was not on time in the lab	Student showed some unethical behavior and was late in lab	Student was on time and showed some unethical behavior	Student was obedient and showed ethical behavior
Roll Number				

Reference: The Lab contents are extracted from the BerkeleyX/CS188

Introduction

In this project, you shall help pacman to find path through his maze world by giving him a list of actions. These Actions shall be calculated by you and pacman only execute those action.

PacMan Project Details

Files you'll edit:	
search.py	Where all of your search algorithms will reside.
searchAgents.py	Where all of your search-based agents will reside.
Files you should read but NOT edit:	
pacman.py	The main file that runs Pacman games. This file describes a Pacman GameState type, which you use in this project.
game.py	The logic behind how the Pacman world works. This file describes several supporting types like AgentState, Agent, Direction, and Grid.
util.py	Useful data structures for implementing search algorithms.
Supporting files you can ignore:	
graphicsDisplay.py	Graphics for Pacman
graphicsUtils.py	Support for Pacman graphics
textDisplay.py	ASCII graphics for Pacman
ghostAgents.py	Agents to control ghosts
keyboardAgents.py	Keyboard interfaces to control Pacman
layout.py	Code for reading layout files and storing their contents
autograder.py	Project autograder
testParser.py	Parses autograder test and solution files
testClasses.py	General autograding test classes
test_cases/	Directory containing the test cases for each question
searchTestClasses.py	Project 1 specific autograding test classes

File and Function to Edit

In this lab, all your code shall be written in the search.py file.

Welcome to Pacman

After downloading the code (search.zip), unzipping it, and changing to the directory, you should be able to play a game of Pacman by typing the following at the command line:

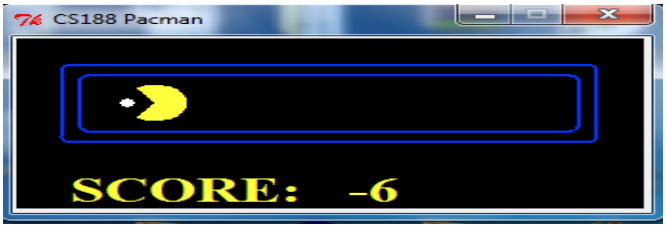
```
python pacman.py
```

Pacman lives in a shiny blue world of twisting corridors and tasty round treats. Navigating this world efficiently will be Pacman's first step in mastering his domain.

Reference: The Lab contents are extracted from the BerkeleyX/CS188

The simplest agent in "**`searchAgents.py`**" is called the "`GoWestAgent`", which always goes West (a trivial reflex agent). This agent can occasionally win:

```
python pacman.py --layout testMaze --pacman GoWestAgent
```




```
python pacman.py --layout tinyMaze --pacman GoWestAgent
```

If Pacman gets stuck, you can exit the game by typing CTRL-c into your terminal.

Note that `pacman.py` supports a number of options that can each be expressed in a long way (e.g., `--layout`) or a short way (e.g., `-l`). You can see the list of all options and their default values via:

```
python pacman.py -h
```

In `searchAgents.py`, you'll find a fully implemented `SearchAgent`, which plans out a path through Pacman's world and then executes that path step-by-step. The search algorithms for formulating a plan are not implemented -- that's your job.

First, test that the `SearchAgent` is working correctly by running:

```
python pacman.py -l tinyMaze -p SearchAgent -a fn=tinyMazeSearch
```

The command above tells the `SearchAgent` to use `tinyMazeSearch` function as its search algorithm, which is implemented in `search.py`. Pacman should navigate the maze successfully.

Reference: The Lab contents are extracted from the BerkeleyX/CS188

Question 1: Find the search pattern for mediumClassicMaze. Your task is two find a search pattern Pacman should eat at least one of his food successfully when you run following command

```
python pacman.py -l mediumClassic -p SearchAgent -a fn=mediumClassicSearch
```

you need to add code under function mediumClassicSearch inside the search.py

Question 2:

```
python pacman.py -l mediumMaze -p SearchAgent -a fn=mediumMazeSearch
```

Question 3:

```
python pacman.py -l bigMaze -z .5 -p SearchAgent -a fn=bigMazeSearch
```