CMSC 671 (Introduction to AI) – Fall 2019

Homework 1: Python, AI, Agents

Turnin: Blackboard.

Submit:
- Parts I and II together as a **single PDF file** named *yourlastname*_hw1_text.pdf.
- Part III as a **single .py file** named *yourlastname*_hw1_program.py, containing everything specified in the assignment.

Notes:
- These are individual assignments, not group work.
- All files must start with your last name.

PART I. WHAT IS AI? (15 PTS)

Reading: Read John McCarthy's paper, "What is AI?"
http://tiny.cc/mc-what-is-ai
Reading: Read the 100-year retrospective *through section II*:
http://ai100.stanford.edu/2016-report

Assignment: Answer all of the following questions in a short **essay** (500-800 words):
- Based on *both* papers:
  - What did you think "AI" meant before the reading?
  - Did anything you read change your mind? How?
- After reading these two papers, do you see the primary goal of AI as modeling human intelligence? Why or why not?
- After reading the Stanford report:
  - How much of AI *right now* is based on trying to model human intelligence?
- What do you think the primary goal of AI should be? Is it achievable?
- What current research trend do you think shows the most *promise*, that is, seems most likely to produce interesting and important results?
- Do you think an artificial agent can be 'intelligent'? (We talked about this in class.) Why?

PART II. AI NOW (9 PTS.)

Assignment: Answer the following **short-answer questions** (3-5 sentences per question)
1. What current area (research trend OR application area) of AI are you most excited about? Why?
2. Name three AI problems that you think someone should be working on right now, and why.
3. Name one AI problems that you think *nobody* should work on, and why.

PART III. INTRODUCTION TO PYTHON (40 PTS)

- If you are not familiar with Python, there are lots of online resources. Any resource where you look at code samples should be cited in the comments at the beginning of the program.
- Documentation and error checking are essential in this class, so although these problems are very simple, your code must be documented, and error cases must be handled. (For example, what if someone passes a number greater than 10,000?)
- We will test your code partly automatically, so it is important that you follow all naming conventions specified and use the correct parameters and types.
- Please be careful to *only* return something when a function calls for a return, and *only* print when it calls for printed output. You will lose points for, e.g., printing the result of problem 1(b).
- We are using **Python 3, not 2**, for this course.

Problem 1: Lists, Sets, Tuples, Strings, and Libraries (12 points)

(a) Import the `random` library.

(b) Write a short function called `lottery` that: *3*

1. Populates a list with 1000 random numbers under 10000 (235, 2, 75, 100, 7, 4...)
   a. You'll want to use a helper function to create this list.
2. Uses `random.shuffle` to randomly permute the elements of the list.
3. Randomly chooses four of those elements as winning lottery numbers.
4. Returns (not prints!) a list containing the winning numbers.

Here is an example of one possible return value:
`lottery()  ⇒ (75, 235, 7, 100)`

(c) Write a short function called `select_set` that: *4*

1. Creates an empty set.
2. Populates the set with 4 winners from `lottery`.
3. Concatenates the elements of the set into a comma-separated string; and
4. Returns the result.

Here is an example of one possible return value:
`lottery()  ⇒ "75, 235, 7, 100"`

(d) Write a short function called `print_winners` that: *5*

1. Creates an empty list.
2. Populates the list with 4 two-element tuples representing *pairs* of the winning numbers returned by `select_set`, *plus* their positions in the list.
3. *Prints* the resulting list, five items per line.

Here is an example of the expected output:

```
(75, 0) (235, 1) (7, 2) (100, 3)
```

Problem 2: Dictionaries and String Manipulations (18 points)

- For this part, you will need 16 food objects and their prices. Use this list:

apples	4	eggplants	13	ice cream	1	mozzarella	16
bananas	11	falafel	15	jicamas	9	nectarines	2
carrots	12	grapes	9	kale	6	oranges	8
daikon	5	horseradish	14	lemons	10	pineapples	3

(a) Write a function called product_dict that: *6*

1. Creates a dictionary containing 16 key/value pairs labeling the map spaces:
    a. Key: product name string, e.g., "apple"
    b. Value: the price of the product, in whole dollars
    c. Returns this dictionary

(b) Write a function called price_deltas that: *12*

1. Takes three product names as arguments.
2. Calculates the *delta of the cost* between the cheapest and middle, and middle and highest.
3. Returns a five-part tuple containing the labels and deltas, from cheapest to most expensive. Break ties randomly.

Here are some examples of possible test cases:

price_delta("apple", "banana", "grape") ⇒
("apple", 5, "grape", 2, "banana")

Why: apples are lowest; grapes are $5 more than apples; bananas are $2 more than grapes.

(c) Write a function called price_print that: *8*

1. Takes the return of price_delta as an argument.
2. Prints the output in exactly the following format:
   [*middle*] cost(s) [*delta1*] more than [*lowest*] but [*delta2*] dollars more than [*highest*].

Here is an example of one possible test case:

grapes cost(s) five dollars more than apples but two dollars less than bananas

Note that dollar values are spelled out!

Problem 3: Call and Test (10 points)

Write a main method that takes three food labels as command line arguments, then calls all of the functions from Problems 1 and 2, and prints out the results of print_winners and price_print. Do not add additional labels or formatting.

Here is an example of one possible output:

```
(75, 0) (235, 1) (7, 2) (100, 3)
grapes cost(s) five dollars more than apples but two dollars less than bananas
```

PART IV. AGENTS (20 PTS)

Problem 1: (10 points)

Fill out the following PEAS table for agents doing these tasks. This is a design question – how would you design this agent? What would you use from the environment? What would you consider a 'good' performance? (You do not have to type in boxes. You can give us an answer with four bullet points per agent.)

System	Performance Measure(s)	Environment	Actuators	Sensors
Example: *Robot Soccer Player*	*Winning games, scoring goals for team, blocking goals against team*	*Field, ball, teammates, other team, own body*	*Kickers (legs), movement (legs or wheels)*	*Camera, touch sensors, orientation sensors, wheel/joint encoders*
(a) Pancake-stacking robot *http://tiny.cc/pancake-robot*				
(b) Stock market agent				
(c) Virtual checkers player				
(d) Customer service chatbot				

Problem 2: (10 points)

Define each of the following for the stock market agent, as designed above.

(a) The world (a description of all possible states)
(b) Start state(s)
(c) Actions available to the agent
(d) All goal state(s)
(e) The solution criteria for this agent