CPSC 322 **Artificial Intelligence I** **Fall 2020**

Assignment One: Search

Due: 11:59pm, Monday 21 September 2020. Submit solution using Canvas

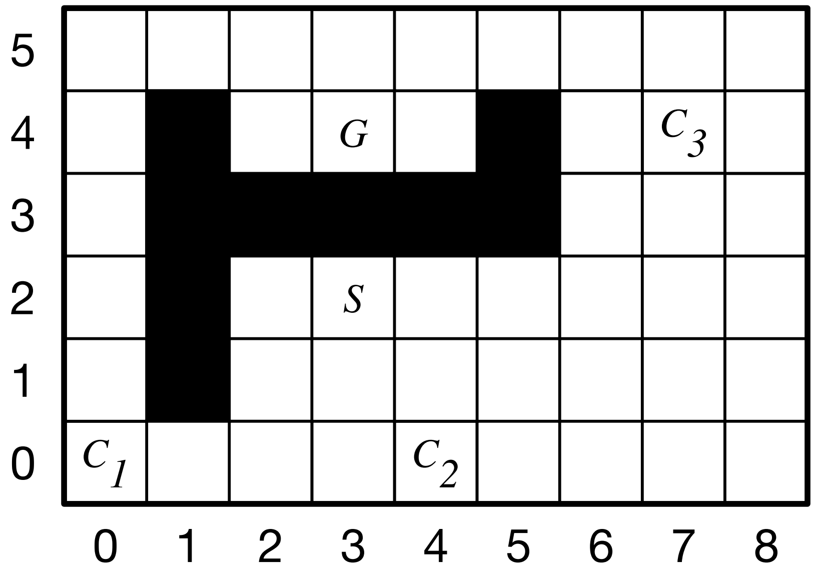
This can be done in groups of size 1, 2 or 3. Working alone is not recommended. All members of the group need to be able to explain the group's answer.

Submit your answers using Canvas. Use proper sentences (not note form) in your answer.

Ask questions on Canvas discussion board. Feel free to answer them too.

Question One

Consider the grid world in the following figure:



Assume that the robot can be in any of the white squares, and can do one step up, down, left, or right at each time. It cannot step into one of the black squares or outside of the boundary. The cost of a path is the number of steps in the path. At the squares marked with C_i are coffee shops where, if the robot goes to the square, it will be given coffee. Suppose the robot starts at the square marked as S, without coffee, and must end up at the square marked G carrying coffee.

Represent the grid world as a state-space search problem.

(a) [2 marks] What is a state for this problem?

(b) [2 marks] How many states are there?

(c) [2 marks] What is the start state?

(d) [4 marks] Define (in pseudo-code or Python) the neighbor function that, given a state, returns a list of the neighbors of that state.

(e) [2 marks] What is the goal function?

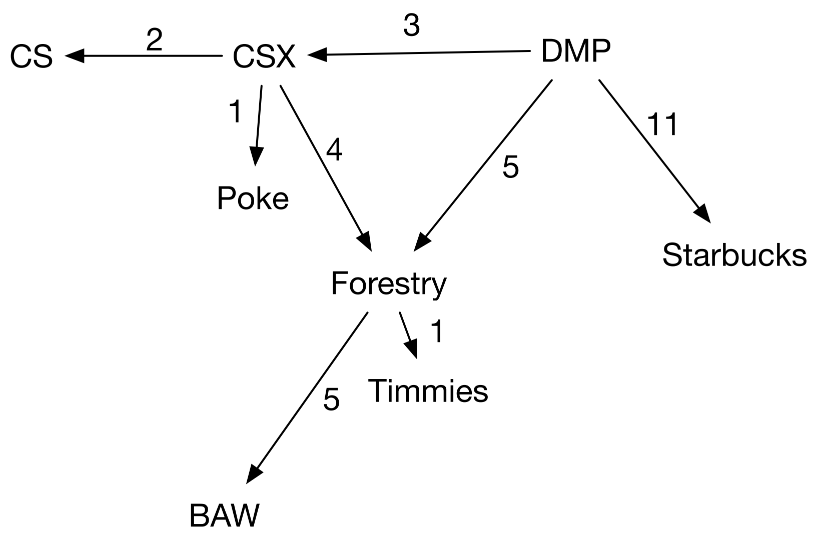
(f) [2 marks] What is a least-cost solution? (Don't run a program; just work it out yourself.)

Question Two

For this question you should use the Python code at `http://aipython.org/aipython_322.zip` optionally with the AISpace2 GUI interface at
`https://aispace2.github.io/AISpace2/install.html`

Note that the documentation is the file `aipython_322.pdf` and depth-first search is implemented by `Searcher` in `searchGeneric.py`. All of these are in the zip file. Hint: look at the example search problems, and modify one to fit this example; don't worry about positions.

(a) [10 marks] Using the representation of search graphs defined in Python, represent the following search graph (with costs on the arcs), which could, in the past and perhaps in the future, be used to find food at UBC (when augmented with more locations).



The start node is *DMP*, and the goal nodes are *Poke*, *BAW*, *Timmies*, *Starbucks*. Call the graph *ubcfood*.

(b) [4 marks] Give all solutions in the order found by depth-first search, assuming the neighbours of a node are ordered, starting with the leftmost neighbour, and then going counter-clockwise. Note: you just need *Searcher*; calling *search*() will find a (or another) path.

You should hand in both the code and a transcript of a session with Python showing all of the paths found.

Question Three

[2 marks] For each question, specify how long you spend on it, and what you learned. How was the work in the team allocated? Was the question reasonable? (This questions is worth marks, so please do it!)