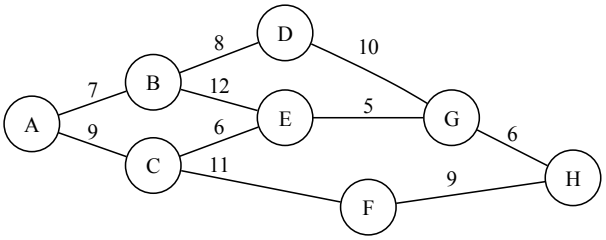
Name: _____ **CCID:** _____

CMPUT 355 Quiz #2 **15 marks** **50 minutes** **closed-book** **no devices**

1. [5 points] Consider the following search graph. Edges are labelled with costs, and the table beneath gives the heuristic value for each node. The goal is to find a path from A to H. Recall that a node is *expanded* when it is removed from the fringe. List the order in which nodes are expanded by A*, and contents of the fringe after each expansion. When two nodes have identical priorities, expand the one that comes alphabetically first (i.e., break ties in alphabetical order).



n	A	B	C	D	E	F	G	H
$h(n)$	20	17	15	12	10	9	6	0

Node	**Fringe**
Expand A	B:24, C:24

2. [4 points] Consider the following 6×6 sliding tile position:

	3	2	1	5	4
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23
24	25	26	27	28	29
30	31	32	33	35	34

(a) How many *inversions* does this position have?

(b) How many *misplaced tiles* does this position have?

(c) What is the *taxicab distance* for this position?

(d) Is this position *solvable*? Why or why not?

3. [4 points] Suppose that running breadth-first search on a 8×3 sliding tile puzzle takes 0.5s in the worst case.

(a) How many possible 8×3 positions are there? (You don't need to expand the expression to get an actual number)

(b) What is the average degree (i.e., number of neighbours) of a 8×3 position?

(c) What is the average degree (i.e., number of neighbours) of a 4×6 position?

(d) Give an estimate of how long it would take to run breadth-first search on an unsolvable 4×6 position. Justify your answer.

4. [2 points] Recall that `stile/15puzzle.py` has three different subgoal schedules:

(i) `[[1],[2],[3,4],[5],[6],[7,8],[9,13],[10,14],[11,12,15]]`

(ii) `[[1,2],[3,4],[5,6,7,8],[9,10,11,12,13,14,15]]`

(iii) `[[1,2,3,4],[5,9,13],[6,7,8,10,11,12,14,15]]`

schedule (i) places tile $\{1\}$ first, then tile $\{2\}$, etc. Schedule (ii) places tiles $\{1, 2\}$ first, then tiles $\{3, 4\}$; Suppose that we run `stile/15puzzle.py` on a solvable position.

(a) Which schedule is likely to produce the *shortest solution*? Why?

(b) Which schedule is likely to *search the fewest nodes*? Why?