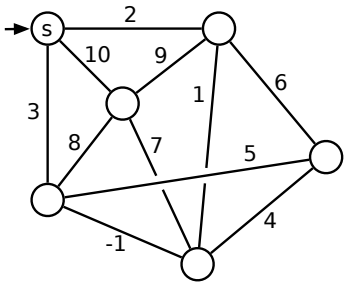
This exam lasts 120 minutes.
Write your answers in the separate answer booklet.
Please return this question sheet with your answers.

1. Find the following spanning trees for the weighted graph shown below.



   (a) A depth-first spanning tree rooted at s.
   (b) A breadth-first spanning tree rooted at s.
   (c) ~~A shortest-path tree rooted at s.~~ ***Oops!***
   (d) A minimum spanning tree.

   You do *not* need to justify your answers; just clearly indicate the edges of each spanning tree in your answer booklet. Yes, one of the edges has negative weight.

2. *[Taken directly from HBS 6.]* An *Euler tour* of a graph G is a walk that starts and ends at the same vertex and traverses every edge of G exactly once. ***Prove*** that a connected undirected graph G has an Euler tour if and only if every vertex in G has even degree.

3. You saw in class that the standard algorithm to INCREMENT a binary counter runs in $O(1)$ amortized time. Now suppose we also want to support a second function called RESET, which resets all bits in the counter to zero.

   Here are the INCREMENT and RESET algorithms. In addition to the array $B[\ldots]$ of bits, we now also maintain the index of the most significant bit, in an integer variable msb.

```
INCREMENT(B[0..∞], msb):
  i ← 0
  while B[i] = 1
      B[i] ← 0
      i ← i + 1
  B[i] ← 1
  if i > msb
      msb ← i
```

```
RESET(B[0..∞], msb):
  for i ← 0 to msb
      B[i] ← 0
  msb ← 0
```

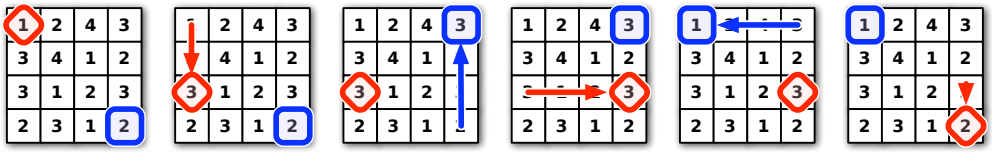
   In parts (a) and (b), let n denote the number currently stored in the counter.

   (a) What is the worst-case running time of INCREMENT, as a function of n?
   (b) What is the worst-case running time of RESET, as a function of n?
   (c) ***Prove*** that in an *arbitrary* intermixed sequence of INCREMENT and RESET operations, the amortized time for each operation is $O(1)$.

4. The following puzzle was invented by the infamous Mongolian puzzle-warrior Vidrach Itky Leda in the year 1473. The puzzle consists of an $n \times n$ grid of squares, where each square is labeled with a positive integer, and two tokens, one red and the other blue. The tokens always lie on distinct squares of the grid. The tokens start in the top left and bottom right corners of the grid; the goal of the puzzle is to swap the tokens.

   In a single turn, you may move either token up, right, down, or left *by a distance determined by the **other** token*. For example, if the red token is on a square labeled 3, then you may move the blue token 3 steps up, 3 steps left, 3 steps right, or 3 steps down. However, you may not move a token off the grid or to the same square as the other token.

   Describe and analyze an efficient algorithm that either returns the minimum number of moves required to solve a given Vidrach Itky Leda puzzle, or correctly reports that the puzzle has no solution. For example, given the puzzle below, your algorithm would return the number 5.



A five-move solution for a 4×4 Vidrach Itky Leda puzzle.

5. Suppose you are given an array $X[1..n]$ of real numbers chosen independently and uniformly at random from the interval $[0,1]$. An array entry $X[i]$ is called a *local maximum* if it is larger than its neighbors $X[i-1]$ and $X[i+1]$ (if they exist).

   What is the *exact* expected number of local maxima in X? ***Prove*** that your answer is correct. *[Hint: Consider the special case $n = 3$.]*

0.7	0.3	**1.0**	0.1	0.0	0.5	**0.6**	0.2	0.4	**0.9**	0.8

A randomly filled array with 4 local maxima.