1	2	3	4	5	Σ
24	16	20	20	20	100

Student number	2	0						
Name								

Midterm Exam — Introduction to Algorithms (CS 300A)

April 16, 2018, 13:00–15:40

Instructions:

- Before you start: **Write your name and student number** (one digit per square!) **clearly** on **all three pages** of this exam booklet.
- This booklet has **three pages** with **five problems** in total. Check that you have all!
- The space provided is sufficient for your answer. If you need scratch space, use the back side. If you need more space for your answer, you can also use the back side (but *indicate clearly on the front side* that your answer continues on the back).
- This is a **closed book exam**. You are not allowed to consult any book or notes.
- Your answers must be in **English**. Write clearly!
- To ensure a quiet exam environment, we will **not answer questions** during the exam. If you think there is a mistake in the question, explain so, and use common sense to answer the question.
- **Relax. Breathe.** This is just an easy, silly, and stupid midterm exam.

Problem 1: (24 pts) Each of these eight questions has one of the following six answers:

$$A : \Theta(1) \qquad B : \Theta(\log n) \qquad C : \Theta(n) \qquad D : \Theta(n \log n) \qquad E : \Theta(n^2) \qquad F : \Theta(n^3)$$

Write the letter (A–F) for the correct answer for each question in the box on the right. Do not justify your answer.

(a) What is $\sqrt{\sum_{i=1}^{n} i}$?

(b) What is $\sum_{i=1}^{n} c^i$, where $0 < c < 1$?

(c) What is $\log(n!)$?

(d) What is the solution to the recurrence $D(n) = D(\lfloor n/\sqrt{3} \rfloor) + \sqrt{3}$?

(e) What is the solution to the recurrence $E(n) = 7E(n/7) + 8n$?

(f) What is the solution to the recurrence $F(n) = 27F(n/3) + 19n^2 \log^3 n$?

(g) How many digits are required to write 13^n in decimal?

(h) What is $\sum_{i=1}^{n} \frac{n}{i}$?

Problem 2: (16 pts) An (undirected) graph is *bipartite* if we can color its vertices red and blue, so that every edge connects a red vertex to a blue vertex. It is known that a graph is bipartite if and only if the graph does not contain a cycle of odd length.

One can use DFS to color a given undirected graph G with two colors if that is possible, or to detect that it is impossible. Below is an implementation: the function `two_color` sets `color[v]` to 0 for red and to 1 for blue. It throws an exception when the graph is not bipartite.

(a) Fill in the two empty gaps in `explore`.

```
def two_color(G):
  for all vertices v of G:
    visited[v] = false
  for all vertices v of G:
    if not visited[v]:
      explore(G, v, 0) # 0 for red
```

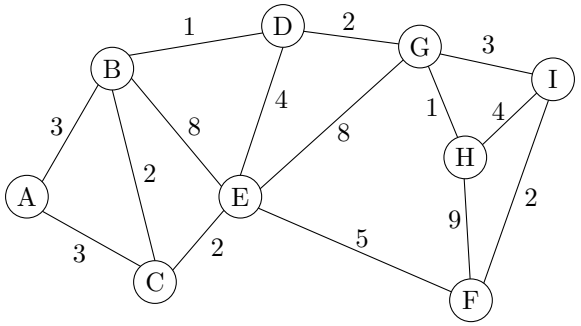
```
def explore(G, v, col):
  visited[v] = true
  color[v] = col
  for all edges {v,u} of G:
    if not visited[u]:
      explore(G, u, ____________________)

    else if ____________________________:
      throw NonBipartiteError
```

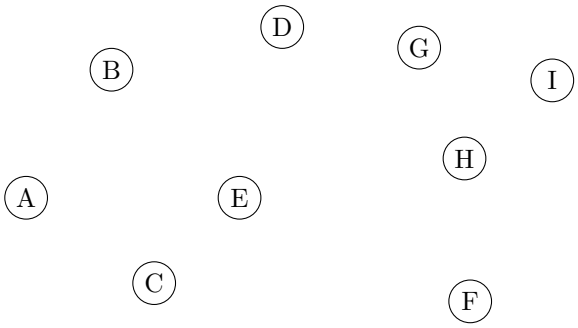
(b) Argue in two sentences why the graph cannot be bipartite when an exception is thrown. (You do not need to argue the other direction.)

Student number	2	0						
Name								

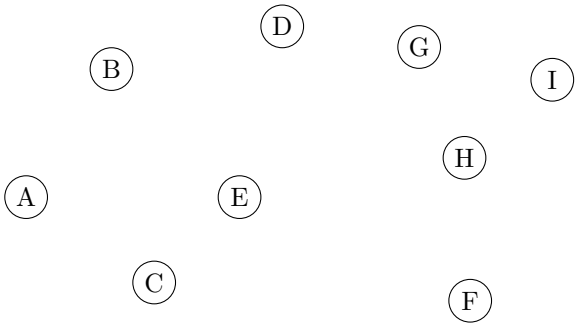
Problem 3: (20 pts) You are given the following undirected graph G with weighted edges.



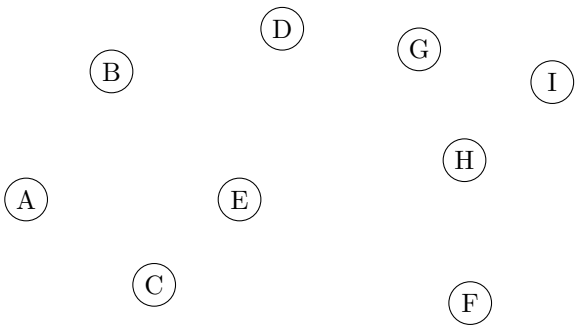
(a) Run Dijkstra's algorithm on G from source vertex E. When vertices have the same cost, choose them in alphabetical order. Draw below the final shortest-path tree for G. Label each vertex with the length of the shortest path from E. Also label the tree edges with the numbers 1 to 8 to indicate the order in which the edges are added by Dijkstra's algorithm.



(b) Run Prim's algorithm on G, starting from vertex H. When edges have the same weight, choose the edge with the alphabetically smallest endpoints first. Draw below the resulting minimum spanning tree for G, and label the tree edges with the numbers 1 to 8 in the order in which they are added by Prim's algorithm.

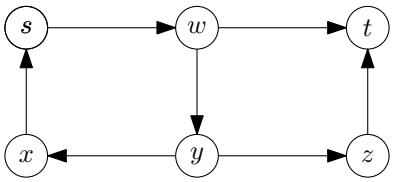


(c) Run Kruskal's algorithm on G. When edges have the same weight, choose again the edge with the alphabetically smallest endpoints first. Draw below the resulting minimum spanning tree for G, and label the tree edges with the numbers 1 to 8 in the order in which they are added by Kruskal's algorithm.



Student number	2	0						
Name								

Problem 4: (20 pts) Suppose you are given a directed graph $G = (V, E)$ and two vertices s and t. Describe and analyze an algorithm to determine if there is a walk in G from s to t (possibly repeating vertices and / or edges) whose length is divisible by 3. For example, given the graph below, with the indicated vertices s and t, your algorithm should return true, because the walk $s \to w \to y \to x \to s \to w \to t$ has length 6. (*Hint: Build a different graph.*)



You will receive 5 points if you write "I don't know" and nothing else.

Problem 5: (20 pts) Let $G = (V, E)$ be a DAG with n vertices, m edges, and *positive* edge weights. G is given in linearized order $v_1, v_2, v_3, \ldots, v_n$: for every edge (v_i, v_j) we have $i < j$.

We know that there is a *shortest path* from v_1 to v_n that goes through all n vertices of G.

Describe an algorithm with running time $O(m \log m)$ which computes, for each edge $e \in E$, the length of the shortest path from v_1 to v_n in $G \setminus e$. Here, $G \setminus e$ is the graph obtained from G by deleting the edge e. In other words, your algorithm must output a set of m shortest-path distances, one for each edge of G. Show why your algorithm is correct, and why it runs within the given running time. (You can use all algorithms we discussed in the class without explaining them.)

Writing pseudo-code without explanation or correctness proof is not considered an answer.

You will receive 5 points if you write "I don't know" and nothing else.