1. **True, False, or Maybe**

   Indicate whether each of the following statments is always true, sometimes true, always false, or unknown. Some of these questions are deliberately tricky, so read them carefully. Each correct choice is worth $+1$, and each incorrect choice is worth -1. **Guessing will hurt you!**

   (a) Suppose SMARTALGORITHM runs in $\Theta(n^2)$ time and DUMBALGORITHM runs in $\Theta(2^n)$ time for all inputs of size n. (Thus, for each algorithm, the best-case and worst-case running times are the same.) SMARTALGORITHM is faster than DUMBALGORITHM.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (b) QUICKSORT runs in $O(n^6)$ time.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (c) $\lfloor \log_2 n \rfloor \geq \lceil \log_2 n \rceil$

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (d) The recurrence $F(n) = n + 2\sqrt{n} \cdot F(\sqrt{n})$ has the solution $F(n) = \Theta(n \log n)$.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (e) A Fibonacci heap with n nodes has depth $\Omega(\log n)$.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (f) Suppose a graph G is represented by an adjacency matrix. It is possible to determine whether G is an independent set without looking at every entry of the adjacency matrix.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (g) NP $\neq$ co-NP

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (h) Finding the smallest clique in a graph is NP-hard.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (i) A polynomial-time reduction from X to 3SAT proves that X is NP-hard.

   ☐ True ☐ False ☐ Sometimes ☐ Nobody Knows

   (j) The correct answer for exactly three of these questions is “False”.

   ☐ True ☐ False

2. **Convex Hull**

   Suppose you are given the convex hull of a set of n points, and one additional point (x, y). The convex hull is represented by an array of vertices in counterclockwise order, starting from the leftmost vertex. Describe how to test in $O(\log n)$ time whether or not the additional point (x, y) is inside the convex hull.

3. **Finding the Largest Block**

   In your new job, you are working with screen images. These are represented using two dimensional arrays where each element is a 1 or a 0, indicating whether that position of the screen is illuminated. Design and analyze an efficient algorithm to find the largest rectangular block of ones in such an array. For example, the largest rectangular block of ones in the array shown below is in rows 2–4 and columns 2–3. *[Hint: Use dynamic programming.]*

1	0	1	0	0
1	1	1	0	1
0	1	1	1	1
1	1	1	0	0

4. **The Hogwarts Sorting Hat**

   Every year, upon their arrival at Hogwarts School of Witchcraft and Wizardry, new students are sorted into one of four houses (Gryffindor, Hufflepuff, Ravenclaw, or Slytherin) by the Hogwarts Sorting Hat. The student puts the Hat on their head, and the Hat tells the student which house they will join. This year, a failed experiment by Fred and George Weasley filled almost all of Hogwarts with sticky brown goo, mere moments before the annual Sorting. As a result, the Sorting had to take place in the basement hallways, where there was so little room to move that the students had to stand in a long line.

   After everyone learned what house they were in, the students tried to group together by house, but there was too little room in the hallway for more than one student to move at a time. Fortunately, the Sorting Hat took CS 373 many years ago, so it knew how to group the students as quickly as possible. What method did the Sorting Hat use?

   More formally, you are given an array of n items, where each item has one of four possible values, possibly with a pointer to some additional data. Design and analyze an algorithm that rearranges the items into four clusters in $O(n)$ time using only $O(1)$ extra space.

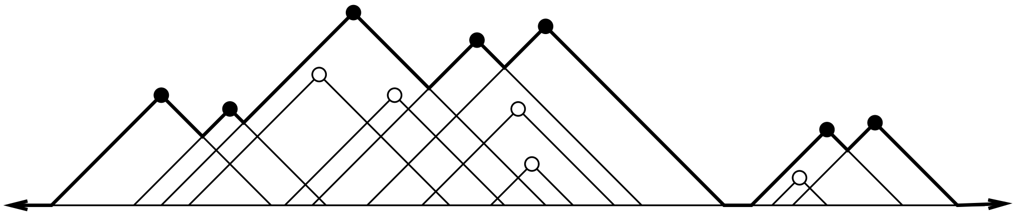
G	H	R	R	G	G	R	G	H	H	R	S	R	R	H	G	S	H	G	G
Harry	Ann	Bob	Tina	Chad	Bill	Lisa	Ekta	Bart	Jim	John	Jeff	Liz	Mary	Dawn	Nick	Kim	Fox	Dana	Mel

$$\Downarrow$$

G	G	G	G	G	G	G	H	H	H	H	H	R	R	R	R	R	R	S	S
Harry	Ekta	Bill	Chad	Nick	Mel	Dana	Fox	Ann	Jim	Dawn	Bart	Lisa	Tina	John	Bob	Liz	Mary	Kim	Jeff

5. **The Egyptian Skyline**

   Suppose you are given a set of n pyramids in the plane. Each pyramid is a isosceles triangle with two 45° edges and a horizontal edge on the x-axis. Each pyramid is represented by the x- and y-coordinates of its topmost point. Your task is to compute the "skyline" formed by these pyramids (the dark line shown below).



The skyline formed by these 12 pyramids has 16 vertices.

   (a) Describe and analyze an algorithm that determines which pyramids are visible on the skyline. These are the pyramids with black points in the figure above; the pyramids with white points are not visible. *[Hint: You've seen this problem before.]*

   (b) One you know which pyramids are visible, how would you compute the *shape* of the skyline? Describe and analyze an algorithm to compute the left-to-right sequence of skyline vertices, including the vertices between the pyramids and on the ground.

6. **DNF-SAT**

   A boolean formula is in *disjunctive normal form* (DNF) if it consists of clauses of conjunctions (ANDs) joined together by disjunctions (ORs). For example, the formula

$$(\overline{a} \wedge b \wedge \overline{c}) \vee (b \wedge c) \vee (a \wedge \overline{b} \wedge \overline{c})$$

   is in disjunctive normal form. DNF-SAT is the problem that asks, given a boolean formula in disjunctive normal form, whether that formula is satisfiable.

   (a) Show that DNF-SAT is in P.

   (b) What is wrong with the following argument that P=NP?

   > Suppose we are given a boolean formula in conjunctive normal form with at most three literals per clause, and we want to know if it is satisfiable. We can use the distributive law to construct an equivalent formula in disjunctive normal form. For example,
   >
   > $$(a \vee b \vee \overline{c}) \wedge (\overline{a} \vee \overline{b}) \iff (a \wedge \overline{b}) \vee (b \wedge \overline{a}) \vee (\overline{c} \wedge \overline{a}) \vee (\overline{c} \wedge \overline{b})$$
   >
   > Now we can use the answer to part (a) to determine, in polynomial time, whether the resulting DNF formula is satisfiable. We have just solved 3SAT in polynomial time! Since 3SAT is NP-hard, we must conclude that P=NP.

7. **Magic 3-Coloring** *[1-unit graduate students must answer this question.]*

   The recursion fairy's distant cousin, the reduction genie, shows up one day with a magical gift for you—a box that determines in constant time whether or not a graph is 3-colorable. (A graph is 3-colorable if you can color each of the vertices red, green, or blue, so that every edge has do different colors.) The magic box does not tell you *how* to color the graph, just whether or not it can be done. Devise and analyze an algorithm to 3-color any graph **in polynomial time** using this magic box.