CS 173: Discrete Structures, Spring 2014
Final Review Problems

These problems should not be turned in. They are to help you review for the final.

1. **Counting problems**

   (a) Let $M = \{a, b, c, d\}$. How many different partitions of M are there?

   (b) In the game Tic-tac-toe (played on the usual 3 by 3 grid), how many different board configurations are possible after four moves (i.e. two moves by each player)?

   (c) Suppose a car dealer is planning to buy a set of Civics, Accords, and Fits (three kinds of cars). The dealer will buy ten cars in total and can buy any number of each type. How many different choices does he have? The sets are unordered, so three Civics and seven Fits is the same as seven Fits and three Civics.

   (d) Suppose a set S has 11 elements. How many subsets of S have an even number of elements? Express your answer as a summation. (You do not need to simplify expressions involving permutations and/or combinations.)

   (e) If $x, y, z \in \mathbb{N}$, how many solutions are there to the equation $x + y + z = 25$?

   (f) Suppose that A is a set containing p elements and B is a set containing n elements. How many functions are there from A to $\mathbb{P}(B)$? How many of these functions are one-to-one?

2. **Big O**

   Sort the following 7 functions of n by asymptotic complexity. That is, write f to the left of g if and only if $f(n) \ll g(n)$.

   $$3n \log(n^3) \qquad 3^n + n \qquad 17^n + 25 \qquad n^2 \qquad 20n^2 \log n \qquad 173 \qquad 7n! + 2$$

3. **Binomial theorem**

   (a) How many terms are contained in $(x + y + z)^{30}$ after carrying out all multiplications, but before collecting like terms?

   (b) How many terms are contained in $(x + y + z)^{30}$ after carrying out all multiplications and collecting like terms?

   (c) What is the coefficient of the $x^{15}y^6z^9$ term? (You may leave you answer in terms of factorials!)

4. **Algorithms**

Consider the following procedure Gnarly, which returns true or false. Notice the indenting on lines 04-05: they execute only when the test on line 03 succeeds. You can assume that $n \geq 1$ and that extracting a subarray (e.g. in the recursive calls in line 06) requires only constant time.

01 Gnarly ($a_1, \ldots, a_n$: array of integers)
02   if ($n = 1$) return true
03   else if ($n = 2$)
04   if ($a_1 = a_2$) return true
05   else return false
06   else if (Gnarly($a_1, \ldots, a_{n-1}$) and Gnarly($a_2, \ldots, a_n$)) return true
07   else return false

(a) If Gnarly returns true, what must be true of the values in the input array? Briefly justify your answer.

(b) Give a recursive definition for $T(n)$, the running time of Gnarly on an input of length n. Be sure to include a base case.

(c) Give a big-theta bound on the running time of Gnarly. For full credit, you must show work or briefly explain your answer.

5. **Algorithms**

Here is pseudocode for functions `find_closest` and `find_closest_sorted`. Each function takes as input an array of n numbers and a target number and outputs the closest value in the input that is at least as large as the target (or NULL if no such number exists). `find_closest_sorted` assumes that the input array is sorted in ascending order.

function find_closest(input_array, target)
sorted_array = cocktailsort(input_array); // cocktailsort takes $\Theta(n^2)$ time
return find_closest_sorted(sorted_array, target);

function find_closest_sorted(sorted_array, target)
length = get_length(sorted_array) // takes constant time
if length = 0
    return NULL;
else if length = 1
    return sorted_array[0];
mid = $\lfloor$length/2)$\rfloor$;
if sorted_array[mid-1]<target
    half_array = sorted_array(mid ... length-1); // array containing second half of elements
else
    half_array = sorted_array(0 ... mid-1); // array containing first half of elements
return find_closest_sorted(half_array, target);

Answer the questions below using worst-case analysis (e.g., the target value is equal to the largest number in the input array). Assume that the input length n is a power of 2.

(a) Suppose $R(n)$ is the running time of `find_closest_sorted`. Give a recursive definition of $R(n)$. Assume that it takes constant time to extract the first or second half of the array in lines `12` and `14`.

(b) What is the height of the recursion tree for $R(n)$?

(c) How many leaves are in the recursion tree for $R(n)$?

(d) What is the big-Theta (aka tight big-O) running time of `find_closest_sorted`? Make your answer as simple as posible.

(e) What is the big-Theta running time of `find_closest`. Make your answer as simple as possible.

6. **Power sets**

Suppose you were given the following sets:

$$\begin{aligned} \mathbf{A} &= \{\text{Vine}, \text{Tree}, \text{Shrub}\} \\ \mathbf{B} &= \{\{\text{Tree}\}\} \\ \mathbf{C} &= \{\text{Vine}, \text{Moss}\} \\ \mathbf{D} &= \{\text{Red}, \text{Green}\} \\ \mathbf{E} &= \{\text{Red}\} \end{aligned}$$

For the following expressions, list the elements of the set or give the cardinality (as appropriate):

(a) $D \times C$

(b) $\mathbb{P}(B \cup E)$

(c) $|A \times \ \mathbb{P}(A \cup D)\ |$

(d) $\{S \in \mathbb{P}(A \cup D)\ :\ |S| \text{ is a multiple of } 4\}$

(e) $\mathbb{P}(\mathbb{P}(\mathbb{P}(\emptyset)))$

7. **Set-valued functions**

   Let $f : X \to Y$ be any function, and let A and B be subsets of X. For any subset S of X define its image $f(S)$ by $f(S) = \{f(s) \in Y \mid s \in S\}$.

   (a) Prove that $f(A \cap B) \subseteq f(A) \cap f(B)$. You **must** use the technique of choosing an element from the smaller set and showing that it is also a member of the larger set.

   (b) Prove that it's not necessarily the case that $f(A) \cap f(B) \subseteq f(A \cap B)$ by giving specific **finite** sets and a specific function for which this inclusion does not hold.

8. **Proof by contradiction**

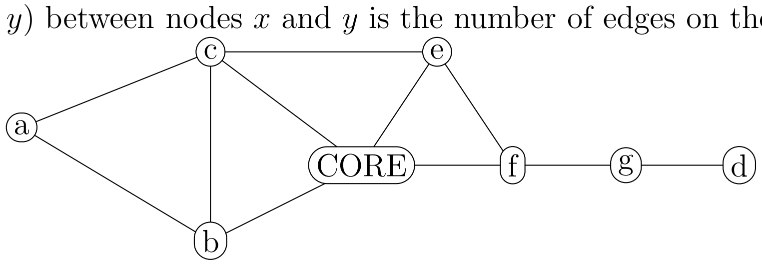
   (a) State the negation of the following claim. Your answer should be in words, with all negations (e.g. "not") on individual predicates.

   For any berry b, if birds like b, then b is edible or b is red.

   (b) Use proof by contradiction to show that $\sqrt{2} + \sqrt{3} \leq 4$.

9. **Partitions**

   Graph G is shown below with set of nodes V is shown below. Recall that the distance $d(x, y)$ between nodes x and y is the number of edges on the shortest path from x to y.



   Now let's define:

$$
\begin{aligned}
M &= \{1, 2, 3, 4\} \\
f(k) &= \{n \in V \ : \ d(n, CORE) = k\} \\
P &= \{f(k) \mid k \in M\}
\end{aligned}
$$

   (a) Fill in the following values:

   f(1) =

   f(3) =

   (b) Is P a partition of V? For each of the three conditions required to be a partition, explain why P does or doesn't satisfy that condition.

10. **State diagrams**

   (a) Suppose we have a state diagram with n states and k different actions. In how many different ways could we construct a transition function for this diagram?

   (b) Suppose we're modelling an RC crane which is receiving a sequence of input commands, each of which is UP, DOWN, or BEEP. This crane has only two vertical positions, and starts in the high position. It should go into an ERROR end state if it is asked to go UP when it is in the high position or DOWN when it is in the low position. BEEP commands are legal at any point. Give a state diagram for this system, in which each edge corresponds to receiving a single command. There should be no edges leading out of the ERROR end state. Explain any notation or decisions that might be unclear.

   (c) Recall that a phone lattice is a type of state diagram, i.e. a directed graph where each node represents a state of a system. A phone lattice has exactly one letter on each edge. Each path from the start node to a final/end node represents a word. Draw a phone lattice representing exactly the following set of words, using a single start node and no more than 9 nodes total.

```
bat, back, sat, sack
boo, booo, boooo, ...     [i.e. b followed by two or more o's]
moo, mooo, moooo, ...     [i.e. m followed by two or more o's]
```

11. **Countability**

   Let's define sets A and B as follows:

   $A = \{0, 2, 4, 6, 8, 10, 12, \ldots\}$, i.e. the even numbers starting with 0.
   $B = \{1, 4, 9, 16, 25, 36, 49, \ldots\}$, i.e. perfect squares starting with 1.

   Show that $|A| = |B|$ by giving a formula for a specific function $f : A \to B$ that is a bijection. (You do not need to prove that your function f is a bijection.)