Introduction to Computer Science I
Spring 2016
Final exam

Instructions: This exam is intended to take about 90 minutes. This exam is closed-book and closed-note. Write **all of your work in your blue book**. Please also **write your name only on the front page of your blue book**. *Good luck!*

1. (15 points) Provide short answers (one to three sentences) to each of the following questions:

   (a) What makes a method *recursive*?

   (b) Suppose we have an array that we want to *resize* to make space for new data that a user enters. Why is it a better idea to double the size of the array when we run out of space, rather than increasing the size of the array by 1?

   (c) What is the *scope* of a variable?

2. (20 points) Given an array of single-digit ints, (`0` to `9`), **write a method** that converts the array into a single integer value. That is, if passed a pointer to an array that contains `{ 5, 1, 3, 7 }`, the method should return the integer `5137`.

3. (20 points) Consider some program that needs to work with *sets* of values. Specifically, a *set* would be a *collection of unique values (no duplicates) for which order is irrelevant.* Assume that the following methods are already written—you do **not** need to write them—for managing and manipulating sets of integers that are stored in arrays:

   - `public static boolean contains (int[] set, int value)`:
     Return whether the given `value` is contained in the given `set`.

   - `public static int[] copy (int[] set)`:
     Creates and returns a duplicate of the given set.

   - `public static int[] insert (int[] set, int value)`:
     If the given `value` is not already contained in the given `set`, then return a new set with that `value` added to it. If the `value` is already present, then return the existing `set`, unmodified.

   - `public static int[] remove (int[] set, int value)`:
     If the given `value` is contained in the given `set`, then return a new set with that `value` omitted from it. If the `value` is not present, then return the existing `set` unmodified.

   To complete this collection of methods for use on sets, **write the following two methods**. Note that you may (and likely should) use the methods listed above in writing these two.

   - `public static int[] intersection (int[] setA, int[] setB)`:
     Return a new set that is the *intersection* of `setA` and `setB`. That is, return a set that contains only the values that `setA` and `setB` have in common. If there are no elements in common, it should return an *empty set*—an array of length zero.

   - `public static int[] union (int[] setA, int[] setB)`:
     Return a new set that is the *union* of `setA` and `setB`. That is, return a set that contains the values contained in either or both of `setA` and `setB`. (Remember that the union is itself a set, and that sets do not contain duplicates.)

4. (20 points) Consider *Pascal's Triangle*, show here in a usefully lopsided form:

```
col | 0  1  2  3  4  5  6
----+--------------------
row |
0   | 1
1   | 1  1
2   | 1  2  1
3   | 1  3  3  1
4   | 1  4  6  4  1
5   | 1  5  10 10 5  1
6   | 1  6  15 20 15 6  1
```

Specifically, we can define the value of any position in the triangle at row r and column c as:

$$T(r,c) = \begin{cases} 1 & \text{if } c = 0 \text{ or } c = r \\ T(r-1,c-1) + T(r-1,c) & \text{if } 0 < c < r \end{cases}$$

That is, the edges (the left- and right-most columns) are always `1`, while the interior values are the sum of the values above-and-to-the-left and immediately-above. Further consider the following method header:

```
public static int[][] pascal (int r)
```

Write this method such that it creates a two-dimensional matrix of integers that are the first `r` rows of Pascal's Triangle. Note that each row is one element longer than the previous one, and so too should the second dimension arrays of this matrix be.

5. (20 points) You are trying to break into ACData, specifically the Registrar's account, so that you can change your grades before your transcript gets sent to a bunch of potential employers.[1] You haven't a clue what her password is, but you can write a bit of code to search for the right one.

   Knowing that her password cannot be more than 16 characters long, and that the characters must be *alphanumeric* (uppercase letters, lowercase letters, or decimal digits),[2] **write a method**, named `crack`, that attempts *every possible 16-character password composed of alphanumeric characters*. To determine whether a given password is correct, your code can call the following, already written method that tests and returns whether the given `password` is the correct one. Your `crack()` method should return the correct password when it finds it.

```
public static boolean correctPassword (char[] password)
```

   You may also assume that the following array, which contains all 62 alphanumeric characters, is defined and available for your method:

```
char[] alphanum = { 'A', 'B', 'C', ... , 'Z',
                    'a', 'b', 'c', ... , 'z',
                    '0', '1', '2', ... , '9' };
```

   Finally, you may assume the existence of *helper methods* that perform simple operations on arrays. For example, you may assume and use a `copy()` method that create a duplicate of a given character array; or, you can assume an `append()` method that duplicates a given array but increases it length by one to store an additional value. You may also write helper methods of your own if you wish to split the tasks that `crack()` must perform across more than one method.

   Hint: Recursion is likely helpful here.

[1] Officially, this is a truly bad idea.

[2] These are artificial limits on the password that wouldn't exist in the real world, but are useful for constraining this exam question. Just go with it.