**Spring 2015 COT 4210 Final Exam**

**Date: April 30, 2015**

**First Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Last Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**1) (10 pts) Recursive Coding**

In programming assignment number 2, you created a DFA class for the purposes of DFA minimization. In this question, you'll add a ***recursive*** method that determines whether or not a String is accepted by a dfa object. Assume that the alphabet size is 10 or less and that the characters are the digits, starting at '0'. (For example, if the alphabet size was 4, the alphabet would be '0', '1', '2' and '3'.) The transition table delta stores the integer 0 to represent the character ‘0’, and so on. The start state is 0 and delta[i][j] stores the transition moving from state i on character version of j.You should be able to determine any other necessary information by looking at the code given. For this question, simply fill in the method inLanguageRec.

public class dfa {

private int numStates;

private int numLetters;

private int numAccept;

private boolean[] acceptStates;

private int[][] delta;

public dfa(int[] accept, int[][] transFunc) {

numStates = transFunc.length;

numLetters = transFunc[0].length;

delta = transFunc;

numAccept = accept.length;

acceptStates = new boolean[numStates];

for (int i=0; i<accept.length; i++)

acceptStates[accept[i]] = true;

}

public boolean inLanguage(String s) {

return inLanguageRec(s, 0);

}

private boolean inLanguageRec(String s, int curState) {

}

}

**2) (25 pts) Algorithms Question**

A radio station offers various cash prizes on various days. However, once you win a prize, you have to wait a certain number of days before you can win another prize. Assuming that you have the ability to win any prize they offer and you know the complete schedule of prizes (the day each prize will be offered, each prize’s value and the number of days you would have to wait until being allowed to win another prize), determine the maximum amount of money you can win. For example, if the prizes were as follows:

|  |  |  |
| --- | --- | --- |
| **Day** | **Value** | **Wait** |
| 2 | 80 | 9 |
| 8 | 50 | 2 |
| 10 | 40 | 2 |
| 13 | 20 | 5 |

You could take the prize on days 8, 10 and 13 for a total cash value of $110. Notice that if you take the prize on day 2, you can't claim another prize until day 11, at which point only the $20 prize on day 13 is left, leaving you with a total cash value of $100. Thus, for this problem instance, the correct answer (maximal possible cash winnings) is $110.

Write a method to solve this problem. Your method will take in an array of RadioPrize objects that are already sorted by day, from smallest to largest, and it is guaranteed that no two RadioPrizes are for the same day. You may assume that the sum of the values of all the RadioPrize objects does not exceed one million and that the array passed to the method doesn't exceed 1000 elements. The RadioPrize class is below:

class RadioPrize implements Comparable<RadioPrize> {

public int day;

public int value;

public int wait;

public RadioPrize(int myDay, int myValue, int myWait) {

day = myDay;

value = myValue;

wait = myWait;

}

public int compareTo(RadioPrize other) {

return this.day - other.day;

}

}

Write your solution on the next page. All correct solutions will be given more credit than any incorrect solutions. Between correct solutions, faster solutions will receive more credit than slower solutions.

public static int getMaxCash(RadioPrize[] prizes) {

}

**3) (20 pts) Undecidability**

Let L = { <M1, M2, k> | M1 and M2 are Turing Machines and there exists at least one string w, with |w| = k such that M1 and M2 reject w.}

Prove that the language L is undecidable.

**4) (20 pts) Polynomial Time Reducibility**

In class we proved that SUBSET-SUM is NP-Complete via two separate proofs. One of those proofs showed that VC(Vertex Cover) is polynomial time reducible to SUBSET-SUM. For this problem, take the graph input described below and the integer k = 2 and use the function shown in class to output an instance of subset sum - a collection of integers and a target value. ***Please clearly label your set and your target value.*** You may just assume that both are expressed in base 10 already. (Namely, in class we mentioned that "base 4" would suffice, but to avoid any real arithmetic, we can just interpret the numbers in base 10 for the purposes of this question.)

Graph vertices : {0, 1, 2, 3, 4}

Graph (undirected) edges: { (0, 1), (0, 2), (0, 3), (0, 4), (1, 3), (2, 3), (3, 4)}

**5) (20 pts) Proving a Problem is NP-Complete**

The bookshelf problem is as follows:

Given the widths of a set of books, and the widths of a set of shelves upon which to place the books (we assume that the heights of all of the books are less than the height of each shelf, so that no book is prevented from being placed on a shelf due to height), determine whether or not the books can all be placed on the shelves.

For example, if the books have widths {3, 2, 9, 4, 5, 6, 7, 12, 2} and the shelves have widths {18, 18, 15} then we can place the books as follows {3, 4, 5, 6} on the first shelf, {2, 9, 7} on the second shelf and {12, 2} on the third shelf. (Note that it's not necessary to fill a shelf.)

Formally we can define a language as follows:

BOOKSHELF = {<A, B> | A and B are lists of positive integers such that A can be partitioned into k = |B| sets such that the sum of the elements of the ith partition is less than or equal to bi, the ith value of the list B, for all i, 1 ≤ i ≤ k}

**Prove that BOOKSHELF is NP-Complete.**

**6) (5 pts) For Fun**

Who was King Henry VIII's father? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Note: Here is a Java method (in the String class) that might be useful:**

public [String](http://docs.oracle.com/javase/7/docs/api/java/lang/String.html) substring(int beginIndex)

Returns a new string that is a substring of this string. The substring begins with the character at the specified index and extends to the end of this string.

Example: "unhappy".substring(2) returns "happy"

**Scratch Page - Please clearly mark any work below that you would like graded.**