COP 3503 Spring 2022 Section 1 Exam #1

Sheet 1: Java API, Disjoint Sets

Last Name: _____ , First Name: _____

Recitation Time (Circle One): 12:30 1:30 2:30 3:30

Java API

1) (10 pts) Complete the program below so that it reads in an integer, ***n*** from standard input, and then reads in ***n*** strings, representing people, coming in order to the DMV. When a person comes for the first time, they should be assigned the first unassigned positive integer ID, and the program should print out, "Giving ***s*** id number ***d***.", where ***s*** is the name of the person. If a person comes for a repeat visit, the program should print out, "Person with id number ***d*** visits again." Use a **HashMap** to complete this program efficiently. (Nearly all the credit will be given for the appropriate use of a HashMap.)

```
import java.util.*;

public class personlist {

    public static void main(String[] args) {

        Scanner stdin = new Scanner(System.in);
        int n = stdin.nextInt();
        HashMap<String,Integer> map = new HashMap<String,Integer>();
        int id = 1;

        for (int i=0; i<n; i++) {

            String name = stdin.next();

            if ( _____ ) {

                System.out.print("Person with id number ");

                System.out.println( _____ +" visits again.");
            }

            else {

                System.out.println("Giving "+name+" id number "+id);

                _____ ;

                _____ ;

            }
        }
    }
}
```

Disjoint Sets

2) (15 pts) Consider a disjoint set of 10 elements, numbered 0 through 9, where the root of larger value always gets attached to the root of smaller value in union operations, and there is no path compression. Draw the trees, clearly indicating the root (to be at the top), after each of the following operations has been executed on a newly created disjoint set of 10 separate sets. For grading purposes, show the picture after each of the labeled points in the steps (A, B, C)

union(3, 7)
union(2, 5)
union(4, 5)
---------- Point A
union(1, 9)
union(4, 9)
---------- Point B
union(8, 0)
union(5, 6)
union(2, 8)
---------- Point C

Picture after Point A

Picture after Point B

Picture after Point C

Sheet 2: Backtracking

Last Name: _____ , First Name: _____

Recitation Time (Circle One): 12:30 1:30 2:30 3:30

3) (25 pts) Consider the problem of being given a permutation of the integers 1 through n, with the commas in between terms removed. For example, for n = 13, the permutation 3,11,8,1,2,13,7,12,10,6,9,5,4 would appear as 31181213712106954. For this problem, complete a program that reads in the value of n and the String of a permutation (with the commas removed), and prints out a valid way of inserting the commas to produce a permutation. (Notice that the answer isn't always unique. For the above example, we can exchange where 1 and 2 are with 12 to create a second valid answer.) n is limited to be in between 1 and 50, inclusive.

The strategy to solve the problem is as follows: the recursive function that does backtracking takes in two important integers: ***k***, the number of terms in the permutation that have been fixed and ***strIdx***, the current index into the string where the start of the k^{th} term will be selected. (You'll recognize and understand what the other two parameters are…) The goal of the function is to return true if there is a solution with the first ***k*** items fixed. To do this, there are only 2 possible items that could be the $(k+1)^{st}$ item: either the one digit integer at index ***strIdx***, OR the two digit integer using the digits at both ***strIdx*** AND ***strIdx***+1. Complete code on the back of this page to solve the problem. All of the set up code is on this side and the method you need to complete is on the back side of this page.

```
import java.util.*;

public class whichperm {

      public static int n;
      public static String input;

      public static void main(String[] args) {

            Scanner stdin = new Scanner(System.in);
            n = stdin.nextInt();
            input = stdin.next();

            int[] perm = new int[n];
            boolean[] used = new boolean[n+1];
            boolean res = go(perm, used, 0, 0);

            for (int i=0; i<n-1; i++)
                  System.out.print(perm[i]+", ");
            System.out.println(perm[n-1]);
      }
```

```
    public static boolean go(int[] perm, boolean[] used,
                             int k, int strIdx) {

        if (k == n) return true;

        // Calculate the one digit option here.

        int oneD = ____________________________________;

        if (oneD == 0) return _________________;

        if (oneD <= n && !used[oneD]) {

            used[oneD] = __________;

            perm[k] = ___________;
            boolean tmp = go(perm, used, __________, _____________);

            if (tmp) return ____________;

            used[oneD] = _____________;
        }

        if (strIdx+1 == input.length()) return false;

        // Calculate the two digit option here.

        int twoD = ____________________________________________;

        if (twoD <= n && !used[twoD]) {

            used[twoD] = ___________;

            perm[k] = _____________;

            boolean tmp = go(perm, used, __________, ____________);

            if (tmp) return ___________________ ;

            used[twoD] = _____________;
        }

        return ____________________;
    }
}
```

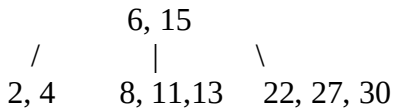
Sheet 3: 2-4 Trees, Red-Black Trees

Last Name: _____ , First Name: _____

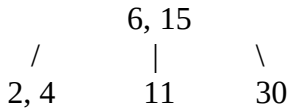
Recitation Time (Circle One): 12:30 1:30 2:30 3:30

2-4 Trees

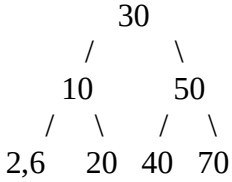
4) (5 pts) Show the result of inserting 24 into the 2-4 Tree shown below:



5) (5 pts) Show the result of deleting 11 from the 2-4 Tree shown below:



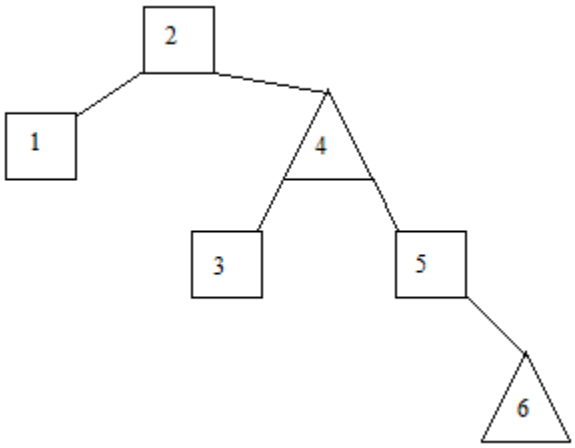
6) (5 pts) Show the result of deleting 70 from the 2-4 Tree shown below:



Red-Black Trees

7) (5 pts) Show the end picture of inserting the following nodes into an initially empty Red-Black Tree, in this order: 10, 5, 2, 20, 7, 9. To show black nodes, draw a square the number in the node and to show red nodes, draw a triangle around the number in the node.

8) (5 pts) Show the result of deleting the value of 1 from the Red-Black Tree shown below. Black nodes have squares around them and red nodes have triangles around them.



Sheet 4: Programs, Odds and Ends

Last Name: _____ , First Name: _____

Recitation Time (Circle One): 12:30 1:30 2:30 3:30

9) (5 pts) One clever solution to the CD problem (Kattis Problem #1) involved just storing one HashSet of both of Jack and Jill's CDs and then subtracting the size of that set from the sum of the total number of CDs that Jack had and the total number of CDs that Jill had. An expression for the answer printed was:

```
numJill+numJack-myset.size()
```

where numJill was the # of CDs Jill had, numJack was the number of CDs Jack had and myset was the HashSet for which each of Jack and Jill's CDs were added.

Explain ***why*** this solution works.

10) (5 pts) In one of the politics solutions and in both of the Tentaizu solutions posted, a technique was used to store an ordered pair of ints into a single int (or long). Consider using this same system to store an ordered pair (x, y) where $0 \leq x < 250$ and $0 \leq y < 500$. Give the single integer that would correspond to the ordered pair (40, 327). Let x represent the "row" and y represent the "column" in the conversion and make sure that all ordered pairs map to integers in the range [0, 124999]. (Note: The values are chosen so that the arithmetic by hand is relatively easy.)

_____

11) (5 pts) Recall that $n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \cdots = 2n$. How does this mathematical statement relate to the expected number of physical nodes in a Skip List (original version shown in class before we started tinkering with it) that contains n different values?

12) (5 pts) In the posted solution, tentaizu_alt.java, DX/DY arrays were used. Explain why using these arrays inside of a loop is better than using an if statement with many branches.

13) (5 pts) One of the featured events in the Winter Olympics is snowboarding. On top of what form of precipitation does a person typically ride a snowboard?

_____