COS 126 **General Computer Science** **Fall 2004**

Exam 2 Solutions

1. **Boolean circuits.**

(a)

REQ_0	REQ_1	REQ_2	GRA_0	GRA_1	GRA_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	1	0	0
1	1	0	1	0	0
1	1	1	1	0	0

(b)
$$GRA_0 = REQ_0$$
$$GRA_1 = REQ_1 \; REQ_0'$$
$$GRA_2 = REQ_2 \; REQ_1' \; REQ_0'$$

2. **Analysis of algorithms.**

(a) 775
$78^{123} \approx 64^{123} = (2^6)^{123} = 2^{738}$.

(b) 33
It's an exponential algorithm (like the inefficient Fibonacci function we saw in class).

(c) 499,500
It's quadratic, but the inner loop only goes halfway to N on average.

(d) 137,775,671
Quicksort is $N \log N$.

3. **Data types.**

```
public class ChargedParticle {
   private double x, y;  // position
   private double q;     // charge

   public ChargedParticle(double x, double y, double q) {
      this.x = x;
      this.y = y;
      this.q = q;
   }

   public double distanceTo(double x, double y) {
      double dx = this.x - x;
      double dy = this.y - y;
      return Math.sqrt(dx*dx + dy*dy);
   }

   public double potential(double x, double y) {
      double k = 8.99E9;
      return k * q / distanceTo(x, y);
   }
}
```

4. **Strings and regular expressions.**

   (a) `CAACAAAACA`

```
String s = "CAAGAATTGA";
s = s.replaceAll("A", "T");     CTTGTTTTGT
s = s.replaceAll("C", "G");     GTTGTTTTGT
s = s.replaceAll("G", "C");     CTTCTTTTCT
s = s.replaceAll("T", "A");     CAACAAAACA
System.out.println(s);
```

   (b) `([1-9][0-9]*,)*[1-9][0-9]* | 1*`
   The last piece is used to match the empty string.

5. **Turing machines.**

(a) `# # # # # # # # # 1 0 x x # # # # # # # # # # # # #`

(b) `# # # # # # # # 1 0 0 x x x x # # # # # # # # # # #`

(c) Overwrites N with x's and writes the binary representation of N to the left of the x's.

(d) N^2

6. **Cryptography.**

For each problem on the left, put the letter of the *best* matching *guarantee* on the right. You may use an answer more than once.

B or D Determine Bob's private RSA key (d, N), given Bob's public RSA key (e, N), an RSA encrypted message from Alice, and the original unencrypted message.

A Determine Bob's private RSA key (d, N), given Bob's public key (e, N) and a factorization of $N = p \times q$.

B or D Determine Alice's original message, given Bob's public RSA key (e, N) and Alice's RSA encrypted message to Bob.

E Decrypt a message sent with a one-time pad without knowing the one-time pad key.

A. Solvable in a polynomial time.

B. Solvable in polynomial time if factoring can be solved in polynomial time.

C. Solvable in polynomial time if P = NP.

D. Solvable in exponential time.

E. Unsolvable: there is no algorithm to solve this problem.

7. **Intractability.**

All four statements are true.

8. **Symbol tables.**

```
while (!StdIn.isEmpty()) {
    String s = StdIn.readString();
    String corrected = (String) st.get(s);
    if (corrected == null) System.out.print(s + " ");
    else                   System.out.print(corrected + " ");
}
```

9. **Linked structures.**

```
public void insert(String s) {
    Node x = new Node();
    x.value = s;
    x.next  = first;
    first   = x;
}

public int size() {
   int N = 0;
   for (Node x = first; x != null; x = x.next)
      N++;
   return N;
}

public String delete() {
   if (first == null) return null;
   int r = (int) (Math.random() * size());
   if (r == 0) {
      String s = first.value;
      first = first.next;
      return s;
   }

   Node x = first;
   for (int i = 0; i < r - 1; i++)
      x = x.next;
   String s = x.next.value;
   x.next = x.next.next;
   return s;
}
```

10. **References.**

It prints `a == c` and then goes into an infinite loop.

The expression `(a == b)` is false because `a` and `b` reference different randomized queues (even though they happen to have the same contents). The expression `(a == c)` is true since by this point, `a`, `b`, and `c` all reference the same randomized queue. As a result, the `while` loop repeatedly deletes an element and re-inserts it into the same queue, leading to an infinite loop. In a cruel twist of fate, the program never prints `goodbye`.