COS 126 **General Computer Science** **Fall 2003**

Exam 2 Solutions

1. **Combinational circuits.**

   $a = AND(x, y, z), b = MAJ(x, y, z), c = OR(x, y, z)$

2. **Sequential circuits.**

   (a) 000

   (b) iv

   This is called a *Gray-code counter*. It cycles through all 8 binary numbers in such a way that adjacent numbers differ in exactly one bit.

3. **TOY architecture.**

   b, d, e, f

4. **Debugging.**

```
public EditDistance(String a, String b) {
    x = a;
    y = b;
    M = x.length();
    N = y.length();
    opt = new int[M+1][N+1];
    for (int i = 0; i <= M; i++) {
        for (int j = 0; j <= N; j++) {
             opt[i][j] = -1;
        }
    }
}
```

   We must initialize `x` and `y`, and do so before executing `x.length()` and `y.length()`; otherwise we get a `NullPointerException`. We must allocate memory for `opt` with `new`, but we should not redeclare it with `int[][]`; otherwise we will have two independent variables named `opt` and the data member named `opt` will not be initialized.

5. **References.**

```
x = 5, y = 7
x = 11, y = 13
x = 30, y = 42
x = 30, y = 42
```

   The statement `q = p;` makes `q` reference the same memory location as `p`.

6. **Abstract data types.**

```
public class Particle {
    private double mass;
    private double x, y;

    public Particle(double mass, double x, double y) {
        this.mass = mass;
        this.x = x;
        this.y = y;
    }


    public static Particle add(Particle a, Particle b) {
        double mass = a.mass + b.mass;
        double x = (a.x * a.mass + b.x * b.mass) / mass;
        double y = (a.y * a.mass + b.y * b.mass) / mass;
        return new Particle(mass, x, y);
    }

}
```

7. **Linked structures.**

   A, D

8. **Fundamental ADTs.**

   B, E, C, A

9. **Hash tables.**

   b

   Hashing enables fast lookups and inserts at the expense of using more memory. Compared with other efficient symbol table implementations, it is relatively easy to implement (fits on one lecture slide). Lookups are slower than inserts. Hashing does not store elements in sorted order, so it doesn't enable you to sort. You shouldn't expect the library implementation to work any The library implementation might be more robust at dealing with errors and it might have additional functionality, but don't expect it to be faster.

10. **Regular expressions.**

    `(1*01*01*01*)* | 1* | (1|0)*0` The `1*` term is needed to match patterns that have zero 0's since zero is a multiple of three.

11. **Analysis of algorithms.**

N^2 Sort N randomly ordered items using insertion sort.

$N \log N$ Sort N randomly ordered items using quicksort.

N Shuffle an array of N items, using the shuffling algorithm from lecture.

N^2 Compute the edit distance of two strings of length N using dynamic programming, as in Assignment 7.

1 Insert or remove one item in a queue, where the queue is implemented as in lecture.

N^2 Insert N cities into a TSP tour using the nearest insertion heuristic, as in Assignment 6.

12. **Computational complexity.**

For each problem on the left, put the letter of the *best* matching *guarantee* on the right. Use each answer *exactly once*.

A Sort N real numbers.

D Factor an N-digit integer.

B Test whether an N-digit integer is prime.

E Determine whether there is a winning position for black in an N-by-N game of checkers.

C Solve a CIRCUIT-SAT problem with N gates.

F Solve the Halting problem on programs with N characters.

A. Can be solved in $N \log N$ steps but not N.

B. Can be solved in a polynomial number of steps.

C. Cannot be solved in a polynomial number of steps unless P = NP.

D. Cannot be solved in a polynomial number if the RSA cryptosystem is secure.

E. Can be solved in an exponential number of steps, but not a polynomial number.

F. Cannot be solved in an exponential number of steps.