Midterm Exam

[after release: fixed typo in cover page table]

Instructions. This exam is closed: no book, no notes, no gadgets, but you are allowed **one sheet of notes**. You will have the full class period (50 minutes). If you put answers on the back of a sheet, mark this clearly.

Name (Print): ______________________________

This exam is my own work. I understand that this exam is governed by the **Emory Honor Code**.

Signature: ______________________________

Please leave this blank.

Problem	Topic	Score	Max
1	Short Answer		12
2	Trees		12
3	Strings		18
4	Potential Functions		10
5	Fill in the Blank		26
6	Extra Credit		+3
Raw Total			78(+3)
Curved			100
Grade			

Problem 1. (12 points) Short Answer. Answer each question with at most a few sentences.

1(a). For each of these ordered data structures, list an advantage it has over the other two:

skip list:

red-black tree:

splay tree:

1(b). Describe two situations needing a large family of hash functions, rather than just one or two.

Problem 2. (12 points) Trees. Consider the binary search tree on the blackboard.

2(a). Treat it as a red-black tree. Draw the red-black tree that results from inserting 17. (Do ordinary two-pass insertion as described in the book and code, not one-pass insertion).

2(b). Treating it as a splay tree (ignore colors). Draw the splay tree that results from inserting 17.

Problem 3. (18 points) Strings.

3(a). Draw the suffix trie for "kaaan!". (The character ordering is `!` $<$ `a` $<$ `k` $<$ `n`.)

3(b). Find the Burrows-Wheeler transform of "kaaan", using '!' as the marker character (like `hw3`).

3(c). Finish this table, the Knuth-Morris-Pratt failure function for pattern "babbabaa"

j	0	1	2	3	4	5	6	7
$P[j]$	b	a	b	b	a	b	a	a
$f(j)$	0	0						

Problem 4. (10 points) Potential functions.

4(a). In the splay tree analysis, what was our bound on the number of rotations necessary to splay a node x to the root position? (It should involve the potential function Φ).

4(b). What was the potential function we used to bound the space usage of persistent red-black trees? (If you use the word "free", define it.)

Problem 5. (26 points) Fill in the Blank.

Recall (2,4)-trees. If a node is about to overflow (during insertion), we may be able to repair it with a(n) ______________________ operation. If a node is about to underflow (during deletion), we may be able to repair it with a(n) ______________________ operation.

In cuckoo hashing, an insert runs in expected time O(1), but worst case time ______________________.

Suppose we carefully implement MSD radix sort, and we use it to sort N strings of total length L, with alphabet size V (that is, each character code is in the range 0 to $V - 1$). Then the total time is $O($______________________$)$, and the extra space (the space needed beyond that for storing the input) is $O($______________________$)$.

In "Latency Lags Bandwidth", Paterson proposes three general techniques to cope with the growing gap. One is caching, another is ______________________.

A B-tree (with large B) may perform much better than a binary search tree, when the nodes are stored in ______________________ memory.

Crosby and Wallach identified several software systems vulnerable to known-hash-function attacks. One example is ______________________.

The fast union-find data structure uses two heuristics: union-by-size and and ______________________.

If we flip a fair coin until we see heads, the expected number of flips is ______________________(a number). In the skip-list analysis, we use this to bound the expected ________________________________.

In the Knuth-Morris-Pratt `match` method (next page), the quantity ______________________ increases with each iteration of the loop.

"bb%aa" is the Burrows-Wheeler transform of ______________________ ('%' is the marker, as in hw3).

Problem 6. (+3 points) Extra Credit. Describe the modified potential function Φ, used to argue that splay trees perform better on keys with a non-uniform probability distribution (a "hot spot").

```java
public class KMP
{
    static int match(String text, String pattern) {
        int n = text.length();
        int m = pattern.length();
        int[] fail = failFunction(pattern);
        int i = 0, j = 0;
        while (i < n) {
            if (pattern.charAt(j) == text.charAt(i)) {
                if (j == m-1) return i-m+1;
                i++; j++;
            } else if (j > 0) {
                j = fail[j - 1];
            } else {
                i++;
            }
        }
        return -1;
    }
    static int[] failFunction(String pattern) {
        int[] fail = new int[pattern.length()];
        fail[0] = 0;
        int m = pattern.length();
        int i = 1, j = 0;
        while (i < m) {
            if (pattern.charAt(j) == pattern.charAt(i)) {
                fail[i] = j+1;
                i++; j++;
            } else if (j > 0) {
                j = fail[j - 1];
            } else {
                fail[i] = 0;
                i++;
            }
        }
        return fail;
    }
}
```