

COS 226 – Data Structures and Algorithms
Fall 2014 – Flipped Lecture Section
Individual Worksheet – 09.18.14
25 minutes

Instructions: Answer as many questions as you can. You can discuss this with anyone in the class. These worksheets are not graded.

1. Consider the sum: $1 + 2 + 4 + 8 + 16 \dots + X$, assume X is a power of 2. Give a closed form expression for this sum.

Assume $X = 2^k$ without loss of generality.

Then $1 + 2 + \dots + 2^k = 2^{(k+1)} - 1 = 2X - 1 \sim 2X$

For the following problems 2-7, estimate the runtime in tilde notation.

2. `for (int p = 1; p < N*N*N; p = p*2)`

The loop runs approximately: $\lg(N^3)$ times $\rightarrow 3 \lg N$

3. `for (int p = 1; p < N*N*N; p = p*2)`

`for (int q = 1; q < p; q++)`

`sum++;`

The sum of the inner loop runs: $0 + 1 + 3 + 7 + 15 + \dots + (2^k - 1) = 2^k - 1$

assume $2^k - 1 \sim N^3 - 1$ (the last index of p) $\rightarrow 2^k = N^3$ so this is n^3

4. `for (int p = 1; p < N/2; p = p*2)`

`sum++;`

$\lg(n/2) \sim \frac{1}{2} \lg n$

5. `for (int p = 1; p*p <= N; p = p*2)`

`sum++;`

$\lg(\sqrt{n}) \sim \frac{1}{2} \lg n$

6. `for (int p = 1; p*p <= N; p = p*2)`

`for (int q = 1; q < p; q++)`

`sum++;`

$0 + 1 + 3 + 7 + 15 + \dots + (2^k - 1) = 2^k - 1$ where $2^k - 1 = \sqrt{n}$

7. Given an unbounded stack (or resizing stack), give a series of operations that would argue against reducing the array to half its size when it is one less than half full (instead of making it half the size when it is quarter full)

An alternate series of pop and push operations will make this process not constant amortized time.

8. Write a java method to determine if a given word (given as an array of chars) is a palindrome or not. For example, “dad” is a palindrome as it reads the same from left to right to right to left. You may use the class Stack.

```
boolean isPalindrome(char[] str) {  
    int i = 0;  
    while (i < str.length && str[i] == str[str.length-i-1]) i++;  
    if (i < str.length) return false; else return true;  
}
```

9. How much memory is allocated for an object of type Node in the code below.

```
public class TwoThreeTree<Key extends Comparable<Key>, Value> {  
    private Node root;  
  
    private class Node {  
        private int count;           // subtree count  
        private Key key1, key2;      // the one or two keys  
        private Value value1, value2; // the one or two values  
        private Node left, middle, right; // the two or three subtrees  
    }  
    ...  
}
```

Count = 4 , key1 and key 2 – 8 each (just references), value1 and value2 – 8 each, left, right, middle – 8 each as Node is a reference here. So a total of :

10. What does the following code fragment print when N is 50? Give a high-level description of what it does when presented with a positive integer N.

```
Stack<Integer> stack = new Stack<Integer>;  
while (N > 0)  
{  
    stack.push(N%2);  
    N = N / 2;  
}  
for (int d : stack) StdOut.print(d);  
StdOut.println();
```

The binary representation of N