University of Nevada, Las Vegas Computer Science 477/677 Fall 2015

Assignment 1: Due September 3, 2015

Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. But the handwriting on this document must be your own. You may attach extra sheets, using a stapler.

1. Solve the following recurrences, giving asymptotic answers, using O, Ω, or Θ, whichever is appropriate.

   (a) $G(n) = 2G(n/2) + 5n$

   (b) $H(n) = 4H(n/2) + n$

   (c) $F(n) \geq F(n/3) + 9$

   (d) $G(n) \leq 8G(n/2) + n^3$

   (e) $H(n) = H(n/2) + n^2$

   (f) $G(n) \geq 2G(n/4) + \sqrt{n}$

(g) $F(n) \leq F\left(\frac{7}{10}n\right) + F\left(\frac{1}{5}n\right) + n$

Hint: use the generalized master theorem.

(h) $T(n) \leq T(\sqrt{n}\,) + 1$

Hint: make a substitution. Let $n = 2^m$. Let $T(n) = T(2^m) = G(m)$. Then write a recurrence for G, solve in terms of m, then back-substitute.

(i) $F(n) = F(n-2) + 5n^2$

(j) $G(n) \leq G(n - \log n) + \log n$

(k) $H(n) = H\left(\frac{3}{5}n\right) + H\left(\frac{4}{5}n\right) + n^2$

Hint: use the generalized master theorem.

(l) $F(n) = \sqrt{n}\,F(\sqrt{n}\,) + n$

Hint: This one is hard.

2. The asymptotic solution to the recurrence $H(n) = H(\log n) + 1$ is $H(n) = \Theta(\log^* n)$. In fact, this recurrence is the definition of $\log^*$.

   More generally, for any increasing function $f(n)$ such that $0 \le f(n) < n$, we can define f^* by the recurrence:

$$f^*(n) = f^*(f(n)) + 1$$

   (a) Suppose that $f(n) = n - 2$. What is $f^*(n)$, that is, the solution to the recurrence

$$F(n) = F(n-2) + 1$$

   The asymptotic answer is $F(n) = \Theta(n)$. However, I don't want an asymptotic answer; instead, I want an exact answer. You may assume that $F(0) = 0$ and $F(1) = \frac{1}{2}$.

   (b) Find a function $f(n)$ such that $f^*(n) = \Theta(\log n)$. (Hint: This is amazingly easy, if you use the master theorem.)

3. Determine the asymptotic time complexity, in terms of n, of each of these code fragments.

(a)
```
for(int i = n; i > 1; i = i/2)
    for(int j = 1; j < i; j++)
        cout << "Hello world" << endl;
```

(b)
```
for(int i = n; i > 1; i = i/2)
    for(int j = i; j < n; j++)
        cout << "Hello world" << endl;
```

(c)
```
for(int i = 1; i < n; i++)
    for(int j = 1; j < n; j = 2*j)
        cout << "Hello world" << endl;
```

(d)
```
for(int i = 1; i < n; i++)
    for(int j = 1; j*j < i; j++)
        cout << "Hello world" << endl;
```
Hint: This one is not as difficult as it looks.

(e)
```
for(int i = n; i > 1; i = sqrt(i))
    cout << "Hello world" << endl;
```

(f)
```
int i = n;
while (i > 1)
 {
    cout << "Hello world" << endl;
    i = i/2;
 }
```

(g)
```
for(int i = n; i > 1; i = log(i))
    cout << "Hello world" << endl;
```
Hint: Look Problem 2.