EECS 477. Homework 2.

Due on Thursday 9/19/2002 before noon
in mailbox labeled 477 in room 2420 EECS

You must show all work to receive credit!

Please read the statement below and sign your name; otherwise, your homework will not be graded. The text of the College of Engineering's Honor Code can be found at http://honor.personal.engin.umich.edu/

I hereby acknowledge that I understand the College of Engineering's Honor Code and have pledged to uphold it and abide by it.

Signature: ____

1 Search in 2D array (55 points)

Let $a_{i,j}, i = 1 \dots m, j = 1 \dots n$ be a two-dimensional array that is ordered in every row and every column so that

- $a_{i,j} \leq a_{i+1,j}$ for $1 \leq i \leq m-1$ and $1 \leq j \leq n$,
- $a_{i,j} \leq a_{i,j+1}$ for $1 \leq i \leq m$ and $1 \leq j \leq n-1$.

You are presented with two algorithms A_1 and A_2 that search for an element x within the array a_{ij} (see the next page). Assume that $m \leq n$ for convenience.

(a: 20pts) Prove that both algorithms return the location of x within the array or return `not_found` if a does not contain x.

(b: 15pts) Let $\phi_k^{[a,x]}(m,n)$ denote the number of `(a[i,j]<x)` comparisons performed in the algorithm $A_k, k = 1,2$ for input array a (of the size $m \times n$) and element x. Find $\Phi_k(m,n) = \max_{a,x} \phi_k^{[a,x]}(m,n)$ that is the number of comparisons in the worst case for $k = 1,2$.

(c: 10pts) Taking $\Phi_k(m,n)$ as the measure of performance, which algorithm is better to use when $m = n$ for large values of n?

(d: 10pts) Taking $\Phi_k(m,n)$ as the measure of performance, which algorithm is better to use when $m = 5$ for large values of n?

A_1:

```
procedure search_A1(array a[1..m,1..n], element x) {
    i = 1;
    j = n;
    while(a[i,j]!=x) {
        if(a[i,j]<x) {
            ++i;
            if(i>m)
                return not_found;
        } else {
            --j;
            if(j<1)
                return not_found;
        }
    }
    return (i,j);
}
```

A_2:

```
procedure search_A2(array a[1..m,1..n], element x) {
    for i=1..m {
        jmin = 1;
        jmax = n;
        do {
            j = (jmin+jmax)/2;
            if ( a[i,j] < x ) {
                jmin = j+1;
            } else if ( a[i,j] > x ) {
                jmax = j-1;
            } else {
                // a[i,j]==x
                return (i,j);
            }
        } while(jmin<=jmax);
    }
    return not_found;
}
```

2 Limits (45 points)

Find the following limits:

(a:15pts)

$$\lim_{n\to\infty} \frac{2^{n+1} + \log n}{n^3}$$

(b:15pts)

$$\lim_{n\to\infty} \frac{3^{n+1}}{3^n + n^3}$$

(c:15pts)

$$\lim_{n\to\infty} \sum_{i=0}^{n} 2^{-n+4}$$