

CIS 121 - Fall 2009

Homework 1 - Due Wednesday, September 23, 10a

Turn in your assignment to the course administrator, Charity Payne, in Levine 459.

Remark: When you analyze code, if you find a running time of, e.g. $3n^2 + 5$, you can write directly that it is $\Theta(n^2)$ no need for proof with c and N .

Problem 1: 30 Points

Method `hasTwoTrueValues` returns true if at least two values in the array of Booleans are true. Provide Big-Theta running time for all three implementations proposed.

(a)

```
public boolean hasTwoTrueValues(boolean[] arr){
    int count = 0;
    for(int i = 0; i < arr.length; i++)
        if (arr[i])
            count++;
    return count >= 2;
}
```

(b)

```
public boolean hasTwoTrueValues(boolean[] arr){
    for(int i = 0; i < arr.length; i++)
        for(int j = i+1; j < arr.length; j++)
            if(arr[i] && arr[j])
                return true;
    return false;
}
```

(c)

```
public boolean hasTwoTrueValues(boolean[] arr){
    for(int i = 0; i < arr.length; i++)
        if(arr[i])
            for(int j = i+1; j < arr.length; j++)
                if(arr[j])
                    return true;
    return false;
}
```

Problem 2: 20 Points

Consider the following code fragment. Analyze the worst-case running time of bar (b) as a function of $n = b.length$ and give a Big-Theta bound. Explain your analysis.

```
static int foo(char[] a) {
    int r1 = a.length / 2;
    int r2 = a.length / 3;
    for (int i = 0; i < a.length; i++) {
        if (a[i] == '@') {
            return r1;
        }
    }
    return r2;
}
```

```
static void bar(char[] b) {
    int max = 0;
    for (int i = 0; i < foo(b); i++) {
        for (int j = 0; j < 101; j++) {
            if (i+j > max) {
                max = i + j;
            }
        }
    }
}
```

Problem 3: 20 Points

Compute a big-Theta-bound in terms of n for the following code fragment :

```
for (int i=0; i < n; i++)
    for (j=0; j<i*i; j++)
        sum++;
```

Problem 4: 20 Points

- a) Give an algorithm that takes as input an array A storing n reals and returns the average of those elements of A that are strictly bigger than any other element that occurs before them in the array. (Eg., if the array is [30, 10, 50, 40, 20, 60] then the algorithm would return 30+50+60 divided by 3.)
- b) Analyze your algorithm and give a big-Theta characterization of its running time.

Problem 5: 10 Points Order the following functions by growth rate: N , \sqrt{N} , $N^{1.5}$, N^2 , $N \log N$, $N \log \log N$, $N \log^2 N$, $N \log(N^2)$, $2/N$, 2^N , $2^{N/2}$, 42 , N^3 , $N^2 \log N$. No Proof necessary.