

## CMSC 132 Homework #2, Due Date Wed Feb 14 in Lecture

Complete each of the problems below and hand in your answers on Wed Feb 14 in lecture (NOT LAB). You must consider this homework a closed assignment as described in

<http://www.cs.umd.edu/class/Spring2007/cmsc132/openClosedPolicy.shtml>

There is no late deadline for this homework. Any homework submitted in lab will not be accepted.

1. List the following big-O expressions in order of asymptotic complexity (with the lowest complexity first).

$O(n \log(n))$   $O(n!)$   $O(\log(n))$   $O(2^n)$   $O(n^2)$

2. For each of the following code snippets, identify the critical section(s), compute the time ( $T(n)$ ) each one takes, and specify the asymptotic complexity using Big-O. **Circle the area(s) you consider critical section(s).**

- ----- Example -----

```
for(int k=0; k < n; k += 3) {
    for (int p=n; p > 6; p--) {
        System.out.println(p % 2); // critical section
    }
}
```

$T(n) = (n/3) * (n-6) = n^2/3 - 2n$   
Big-O =  $O(n^2)$

- a. -----

```
for(int k=0; k <=n/8; k++) {
    System.out.println(k);
}
System.out.println("Next");
for (int p=n; p >= 1; p--) {
    System.out.println(p % 2);
}
```

$T(n) =$

Big-O =  $O(\quad)$

- b. -----

```
for(int k=0; k < n-1; k++) {
    for (int m=k+1; m < n; m++) {
        System.out.println(k*m);
    }
}
```

$T(n) =$

Big-O =  $O(\quad)$

c. -----

```
for (int i=n-3; i<=n-1; i++) {
    System.out.println(i);
    for (int k=1; k<=n; k++) {
        System.out.println(i + k);
    }
}
```

T(n) =

Big-O = O ( )

d. -----

```
for(int a=1; a <= n/3; a++) {
    for (int b=1; b <= 2*n; b++) {
        System.out.println(a * b);
    }
}
```

T(n) =

Big-O = O ( )

e. -----

```
for (int i=0; i<n-3; i++) {
    for (int k=1; k<n; k*=10) {
        System.out.println(i + k);
    }
}
```

T(n) =

Big-O = O ( )

f. -----

```
for(int k=3; k <=n; k++) {
    System.out.println(k);
}
System.out.println("Next");
for (int p=n; p>= 1; p--) {
    for (int a=n-3; a<=n-2; a++) {
        System.out.println(p*a);
    }
}
```

T(n) =

Big-O = O ( )