

Computer Science 245
Homework 1
Algorithm Analysis
Due Monday, February 6th, 2012

1. Give $\Theta()$ running times for each of the following code fragments. HINT – if you are having trouble with finding $\Theta()$, first find $O()$, then $\Omega()$. For partial credit, just find $O()$.

- (a)

```
for(i=0; i<n; i++)
  for(j=n/2; j>0; j -= 2)
    for(k=0; k<n; k++)
      A[i] = A[j] * A[k]
```
- (b)

```
tmp = 0;
while (tmp < n * n) {
  for (tmp2=0; tmp2<n; tmp2++)
    sum++;
  tmp++;
}
```
- (c)

```
for (i=1; i<n*n i=i+2)
  for (j=1; j<n*n; j=j+2)
    sum++;
```
- (d)

```
for (i=1; i<n*n i=i*2)
  for (j=1; j<n*n; j=j+2)
    sum++;
```
- (e)

```
for (j=0; j < n*n; j++)
  for(k=0; k<j*j; k++)
    sum++;
```
- (f)

```
tmp1 = n;
while (tmp1 > 1) {
  tmp2 = 0;
  while (tmp2 < n/2) {
    tmp2++;
  }
  tmp1 = tmp1/2
}
```
- (g)

```
for (tmp1=0; tmp1<n; tmp1++)
  for(tmp2=0; tmp2 < tmp1; tmp2++)
    for(tmp3=1; tmp3<n; tmp3 = tmp3*3)
      sum++;
```

2. For each of the following recursive functions:

- Describe what the function computes (careful, some of these are tricky!)
- Give a recurrence relation that describes the running time of the function (Give both base and recursive cases)
- Solve the recurrence to get a Θ running time for the function. Use either the repeated substitution method, the recursion tree method (which is essentially the same as the repeated substitution method, just a little more graphical), or the Master Method. Note that the Master Method does not apply in all cases!

(a) `int recursive1(int n)`

```
{
    sum = 0;
    for (int i = 1; i <= n; i++)
        sum++;
    if (n > 1)
        return sum + recursive1 (n-1);
    else
        return n;
}
```

(b) `int recursive2(int n)`

```
{
    if (n > 1)
        return recursive2(n-1) + recursive2(n-1) + recursive2(n-1);
    else
        return n;
}
```

(c) `int recursive3(int n)`

```
{
    if (n > 1)
        return 3 * recursive3(n-1);
    else
        return n;
}
```

(d) `int recursive4(int n)`

```
{
    int no_op;
    if (n > 1)
    {
        for (i=1; i<=n; i++)
        {
            no_op++;
        }
        return recursive4(n/2) * recursive4(n/2);
    } else {
        return 1;
    }
}
```

```
(e) int recursive5(int n)
    {
        int no_op;
        if (n > 1)
        {
            for (i=1; i<=n; i=i*2)
            {
                no_op++;
            }
            return recursive5(n/4) * recursive5(n/4) *
                recursive5(n/4) + recursive5(n/4);
        } else {
            return 1;
        }
    }
}
```