

CIS121 - Fall 2009

Lab 1 – Monday, September 13

Recall that when we analyze running time we analyze the *worst* case.

Count the exact number of steps for the following snips of pseudo-code and give a *tight* big-oh analysis

1. **Algorithm** *printSquare*(n)

Prints a square of '*' of size n

```
1  for  $i \leftarrow 0$  to  $n$             $n$ 
2      do for  $j \leftarrow 0$  to  $n$     $n^2$ 
3          do print *              $n^2$ 
4      end for
5      print lineBreak            $n$ 
6  end for
```

$$n + n^2 + n^2 + n = 2n^2 + 2n \quad O(n^2)$$

2. **Algorithm** *printTriangle*(n)

Prints a triangle of '*' of height n

```
1  for  $i \leftarrow 0$  to  $n$             $n$ 
2      do for  $j \leftarrow 0$  to  $i$     $1 + 2 + \dots + n = n(n + 1)/2$ 
3          do print *              $n(n + 1)/2$ 
4      end for
5      print lineBreak            $n$ 
6  end for
```

$$n + n(n + 1)/2 + n(n + 1)/2 + n = n^2 + 3n \quad O(n^2)$$

3. **Algorithm** *countPositive*(A, n)

Count the number of positive integers in array A .

```
1  count  $\leftarrow 0$                 1
2  for  $i \leftarrow 0$  to  $n$            $n$ 
3      do if  $A[i] > 0$                $n$ 
4          then count  $\leftarrow$  count + 1   $n$ 
5      end if
6  end for
7  print count                       1
```

$$1 + n + n + n + 1 = 3n + 2 \quad O(n)$$

4. **Algorithm** *weird*(A, n) Assume that the function $\min(a, b)$ is $O(1)$

```

1  count ← 0                                1
2  for i ← 0 to n                            n
3      do if A[i] ≥ 0                        n
4          then count ← count + 1           Step 4 + 5 = n
5          else for j ← 0 to min(A[i], n)
6              do print *                    0 never executed
7                  j ← j + 1                0 never executed
8              end for
9          end if
10 end for

```

$$1 + n + n + n = 3n + 1 \quad O(n)$$

5. **Algorithm** *weird2*(n) Assume that the function $\text{mod}(a, b)$ is $O(1)$

```

1  i ← 0                                    1
2  while i < n                              n
3      do if mod(i, 2) = 0                  n
4          then for j ← 0 to n             n · n/2
5              do print *                  n · n/2
6          end for
7      end if
8          i ← i + 1                        n
9  end while

```

$$1 + n + n + n \cdot n/2 + n \cdot n/2 + n = n^2 + 3n + 1 \quad O(n^2)$$

6. **Algorithm** *weird3*(n) Assume that the function $\text{mod}(a, b)$ is $O(1)$

```

1  i ← 1                                    1
2  while i < n                              n/2
3      do if mod(i, 2) = 0                  n/2
4          then for j ← 0 to n             0 never executed
5              do print *                  0 never executed
6          end for
7      end if
8          i ← i + 2                        n/2
9  end while

```

$$1 + n/2 + n/2 + n/2 = 3n/2 + 1 \quad O(n)$$