CSE 2331 Homework 3
Spring, 2016
10 Problems

Write a recurrence relation describing the WORST case running time of each of the following algorithms and determine the asymptotic complexity of the function defined by the recurrence relation. Justify your solution using either substitution, a recursion tree or induction. You may NOT use the Master theorem. Assume that all arithmetic operations take constant time.

Simplify and express your answer as $\Theta(n^k)$ or $\Theta(n^k(\log n))$ wherever possible. If the algorithm takes exponential time, then just give exponential lower bounds.

1.

```
func1($A$,$n$)
/* $A$ = array of $n$ integers                                   */
if ($n \leq 20$) then return $A[1]$;
for $i \leftarrow 1$ to $\lfloor n/2 \rfloor$ do
    for $j \leftarrow 1$ to $\lfloor \sqrt{n} \rfloor$ do
        $A[i] \leftarrow A[i] + A[i+j]$;
    end
end
$x \leftarrow$ func1($A$, $n-6$);
return ($x$);
```

2.

```
func2($A$,$n$)
/* $A$ = array of $n$ integers                                   */
if ($n \leq 15$) then return $A[1]$;
for $i \leftarrow 1$ to $\lfloor n/2 \rfloor$ do
    for $j \leftarrow 1$ to $\lfloor n/2 \rfloor$ do
        $A[j] \leftarrow A[i] + A[i+j]$;
    end
end
$x \leftarrow$ func2($A$, $\lfloor 5n/7 \rfloor$);
return ($x$);
```

3.

```
func3($A$,$n$)
/* $A$ = array of $n$ integers                                   */
if ($n \leq 15$) then return $A[1]$;
if ($A[1] \leq A[n]$) then
    $x \leftarrow A[n]+$ func3($A$, $\lfloor 4n/5 \rfloor$);
else
    $x \leftarrow A[2]+$ func3($A$, $\lfloor 3n/5 \rfloor$);
end
return ($x$);
```

4.

```
func4($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 30$) then return $A[1]$;
$x \leftarrow 0$;
for $i \leftarrow 1$ to 7 do
    for $j \leftarrow 1$ to $\lfloor 2n/3 \rfloor$ do
        $A[j] \leftarrow A[j] - A[i+j]$;
    end
    $x \leftarrow x+$ func4($A$, $\lfloor n/7 \rfloor$);
end
return ($x$);
```

5.

```
func5($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 15$) then return $A[1]$;
for $i \leftarrow 1$ to 5 do
    for $j \leftarrow 1$ to $n - 8$ do
        $A[j] \leftarrow A[j] + A[j+i]$;
    end
    $x \leftarrow x+$ func5($A$, $n - 2 * i$);
end
return ($x$);
```

6.

```
func6($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 37$) then return $A[1]$;
for $i \leftarrow 1$ to $\lfloor n/2 \rfloor$ do
    for $j \leftarrow 1$ to $\lfloor \sqrt{n} \rfloor$ do
        $A[j] \leftarrow A[i] + A[i+j]$;
    end
end
$x \leftarrow$ func6($A$, $\lfloor 3n/4 \rfloor$);
return ($x$);
```

7.

```
func7($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 14$) then return $A[1]$;
$i \leftarrow n - 3$;
while ($i > 14$) do
    $A[i] \leftarrow A[i] + A[i-2]$;
    $i \leftarrow \lfloor i/4 \rfloor$;
end
$x \leftarrow A[n]$ + func7($A$, $n - 9$);
return ($x$);
```

8.

```
   func8($A$,$n$)
   /* $A$ = array of $n$ integers */
1  if ($n \le 32$) then return $A[1]$;
2  $x \leftarrow 0$;
3  $i \leftarrow n - 3$;
4  while ($i \ge 25$) do
5      $A[i] \leftarrow A[i] + A[i-4]$;
6      $x \leftarrow$ x+func8($A$, $i$);
7      $i \leftarrow i - 7$;
8  end
9  return ($x$);
```

(Note: Step 7 in `func8` uses subtraction, NOT division.)

9.

```
func9($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 20$) then return $A[1]$;
$x \leftarrow$ func9($A$, $\lfloor n/5 \rfloor$);
for $i \leftarrow 1$ to $\lfloor n/5 \rfloor$ do
    $A[i] \leftarrow A[i] - A[3 * i]$;
end
$x \leftarrow x+$ func9($A$, $\lfloor n/5 \rfloor$);
for $i \leftarrow 1$ to $\lfloor 2n/5 \rfloor$ do
    $A[i] \leftarrow A[i] + A[2 * i]$;
end
$x \leftarrow x+$ func9($A$, $\lfloor n/5 \rfloor$);
return ($x$);
```

(Note three function calls to `func9`.)

10.

```
func10($A$,$n$)
/* $A$ = array of $n$ integers */
if ($n \le 20$) then return $A[1]$;
$x \leftarrow$ func10($A$, $\lfloor 2n/5 \rfloor$);
for $i \leftarrow 1$ to $\lfloor n/4 \rfloor$ do
    $A[i] \leftarrow A[i] + A[3 * i]$;
end
$x \leftarrow x+$ func10($A$, $\lfloor 3n/5 \rfloor$);
return ($x$);
```

(Note two function calls to `func10`.)