

(45 points)

1. (5)

for $i \leftarrow 1$ **to** n **do**
 for $j \leftarrow 0$ **to** i **do**
 cost: i

The inner loop has cost

$$\sum_{j=0}^i i = i \sum_{j=0}^i 1 = i(i+1)$$

so the entire cost is

$$\begin{aligned} \sum_{i=1}^n i(i+1) &= \sum_{i=1}^n i^2 + \sum_{i=1}^n i \\ &= \frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2} \\ &= \frac{n(n+1)(2n+1)}{6} + \frac{3n(n+1)}{6} \\ &= \frac{n(n+1)(2n+4)}{6} \end{aligned}$$

2. (5)

for $i \leftarrow 0$ **to** n **do**
 for $j \leftarrow 0$ **to** i **do**
 cost: $2j$

The inner loop has cost $\sum_{j=0}^i 2j = 2 \sum_{j=0}^i j = 2 \frac{i(i+1)}{2} = i(i+1)$ so the entire cost is

$\sum_{i=0}^n i(i+1)$ which can be simplified to $\frac{n(n+1)(2n+4)}{6}$ just as in Problem 1.

3. (5)

for $i \leftarrow 0$ **to** n **do**
 for $j \leftarrow 1$ **to** n **do**
 cost: $i+j$

The inner loop has cost $\sum_{j=1}^n (i+j) = \sum_{j=1}^n i + \sum_{j=1}^n j = ni + \frac{n(n+1)}{2}$

so the entire cost is

$$\begin{aligned} \sum_{i=0}^n \left[ni + \frac{n(n+1)}{2} \right] &= n \sum_{i=0}^n i + \frac{n(n+1)}{2} \sum_{i=0}^n 1 \\ &= n \frac{n(n+1)}{2} + \frac{n(n+1)}{2} (n+1) \\ &\dots(\text{some algebra omitted})\dots \\ &= \frac{2n^3 + 3n^2 + n}{2} \end{aligned}$$

4. (10)

```

j ← 0
while n > 0 do
  i ← n
  while i > 0 do
    i ← i - 1
    j ← j + 1
  n ← n div 2

```

(a) The inner **while** statement is the barometer. On each pass through the outer loop the barometer is executed with $i = n, n-1, \dots, 0$, so the barometer is executed $n+1$ times.

The outer loop is executed with $n = 2^k, 2^{k-1}, \dots, 2^0$. So the barometer is executed

$$\sum_{i=0}^k (2^i + 1) = \sum_{i=0}^k 2^i + \sum_{i=0}^k 1 = 2^{k+1} - 1 + (k+1) = 2n + k = 2n + \lg n$$

(Note that the $2^i + 1$ terms in the sum correspond to the $n+1$ times the barometer is executed on each pass through the outer loop. Also, the summation runs in the “opposite direction” than the outer loop.)

(b) j is incremented n times on each pass through the outer loop so the final value of j is

$$\sum_{i=0}^k 2^i = 2^{k+1} - 1 = 2n - 1$$

5. (15)

```
for  $i \leftarrow 1$  to  $n$  do  
   $j \leftarrow i$   
  while  $j > 0$  do  
    if  $(j \bmod 2) == 1$   
      cost: 1  
       $j \leftarrow j \text{ div } 2$ 
```

This code counts the number of 1-bits in the binary representations of the numbers from 1 to $n = 2^k$. We can illustrate how many 1-bits there are by making the following table related to the case for $n = 4$.

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 |

This is the table of binary numbers from 0 through $2^k - 1$, for $k = 4$. (Note that leading zeros are explicitly shown in the table.) The two main observations are that this table has 2^k rows and k columns, so it contains $k2^k$ bits, and exactly half of those bits are ones, that is, $k2^{k-1}$ bits in the table are ones. Clearly, there is only one more 1-bit in the binary numbers 1 through 2^k (contributed by 2^k itself), so this code segment has “cost” $k2^{k-1} + 1$.