Sample Questions: Code Run Time Analysis

August 2015 Computer Science A Question 2 (Iterative Code Segment)

Consider the following segment of code, assuming that n has been previously declared and initialized to some positive value:

```
int i, j, k;
for (i = 1; i <= n; i++){
    for(k =1; k <= i; k++){
        j = k;
        while(j > 0)
            j--;
    }
}
```

(a) (3 pts) Write a summation (3 nested sums) equal to the number of times the statement `j--;` executes, in terms of n.

$$\sum_{i=1}^{n}\left(\sum_{k=1}^{i}\left(\sum_{j=1}^{k} 1\right)\right)$$

(b) (7 pts) Determine a closed form solution for the summation above in terms of n.

$$= \sum_{i=1}^{n}\left(\sum_{k=1}^{i} k\right)$$

$$= \sum_{i=1}^{n}\left(\frac{i(i+1)}{2}\right)$$

$$= \frac{1}{2}\sum_{i=1}^{n} i^2 + \frac{1}{2}\sum_{i=1}^{n} i$$

$$= \frac{1}{2}\cdot\frac{n(n+1)(2n+1)}{3} + \frac{n(n+1)}{4}$$

$$= \frac{n(n+1)}{12}\left(2(2n+1)+3\right)$$

$$= \boxed{\frac{n(n+1)(4n+5)}{12}}$$

FND REV W4 11/18/2015
P2

Dec 14 CS A Q2a

Write a summation, **but do NOT solve it,** that represents the value of the variable sum at the end of the following code segment, in terms of the variable *n*, entered by the user. (Note: your answer should have two summation signs in it and appropriate parentheses that clearly dictate the meaning of the expression you've written.)

```
int i, j, n, sum = 0;
printf("Please enter a positive integer.\n");
scanf("%d", &n);

for (i=n; i<2*n; i++) {
    sum += i;
    for (j=1; j<=i; j++)
        sum += (j*j);
}
```

$$\sum_{i=n}^{2n-1}\left(i + \sum_{j=1}^{i} j^2\right)$$

August 14 CSA Question 2b

Determine the run time of the code segment shown below, in terms of n. Provide your answer as a Big-Theta bound.

```
int n;
scanf("%d", &n);
int i, step = 1, total = 1;

for (i=0; i<n*n; i+= step) {
    total++;
    step += 2;
}
```

i $\cancel{0}$ $\cancel{3}$ $\cancel{8}$ 15 24

step $\cancel{1}$ $\cancel{3}$ $\cancel{5}$ $\cancel{7}$ 9

In n steps, i becomes n^2-1, at which point the loop will stop.

$\boxed{O(n)}$

$3+5+7+9+\dots$

$$\sum_{i=2}^{n}(2i-1)$$

$$=\left(\sum_{i=1}^{n}(2i-1)\right)-1$$

$$=2\cdot\frac{n(n+1)}{2}-n-1$$

$$=n^2+n-n-1$$

$$=\boxed{n^2-1}$$

FND REV W4
11/18/2015
P3

Run-Time for
Dec 14 CS A Q2A

$$\sum_{i=n}^{2n-1}\left(1+\sum_{j=1}^{i}1\right)$$

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

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

$$= n + \frac{(2n-1)2n}{2} - \frac{(n-1)n}{2}$$

$$= \frac{n}{2}\left(2 + 2(2n-1) - (n-1)\right)$$

$$= \frac{n}{2}\left(2 + 4n - 2 - n + 1\right)$$

$$= \frac{n}{2}(3n+1) = O(n^2)$$

FNDREV W4
11/18/2015
P4

DEC 2013 CSA Q2

(a) (3 pts) Write a summation that represents the number of times the statement `p++` is executed in the following function:

```
int foo(int n)
{
   int i, j, p = 0;

   for (i = 1; i < n; i++)
      for (j = i; j <= i + 10; j++)
         p++;

   return p;
}
```

(b) (5 pts) Determine a simplified, closed-form solution for your summation from part (a), in terms of *n*. **You MUST show your work.**

Aug 12 CSB Q1

(a) (4 pts) Determine, **with proof**, the run-time of the following function in terms of the formal parameters a and b:

```
int f(int a, int b) {

    int i,j, sum = 0;

    for (i=0; i<a; i++) {
        j = b;
        while (j > 0) {
            j = j/2;
            sum++;
        }
    }
    return sum;
}
```

CS A May 14 Q2a

Write a recurrence relation that represents the runtime of the following function, then solve it (i.e., derive its closed form) using iterative substitution:

```
int foo(int n)
{
   if (n == 0 || n == 1)
      return 18;

   else
      return foo(n-2) + foo(n-2);}
```

FNDREV W4
11/18/2015
PS

August 2015 Computer Science B Question 1 (Recursive Code Segment)

Consider the recursive function `diminish` shown below:

```
double diminish(int m, int n){
    if (n == 0)
        return m;
    return  1.0/2*diminish(m,n-1)
}
```

perm(n,k)

for(i=0;i<n;i++) {
  if (!used[i]) {
    perm(~~...~~ n,k+1)

(a) (3 pts) Let T(n) represent the run time of the function `diminish`. Write a recurrence relation that T(n) satisfies.

$$T(n) = \boxed{T(n-1)} + O(1)$$

$$\boxed{T(n) = n \cdot T(n-1) + n}$$

(b) (6 pts) Using the iteration method, determine a closed-form solution (Big-~~Oh bound~~) ~~for~~ T(n).

$$T(n) = \boxed{T(n-2) + 1} + 1$$

$$= \boxed{T(n-2)} + 2$$

$$= \boxed{T(n-3) + 1} + 2$$

$$= \boxed{T(n-3) + 3}$$

After k iterations we have

$$= T(n-k) + k$$

Let $k = n-1$,

$$= T(n-(n-1)) + (n-1)$$

$$= T(1) + (n-1)$$

$$= 1 + (n-1)$$

$$= n$$

$$= O(n)$$

FNDREV W4
11/18/2015
P6

Recurrence Relations to Solve

1) $T(n) = 2T\left(\frac{n}{2}\right) + 1, T(1) = 1$

2) $T(n) = T(n-1) + n, T(1) = 1$

3) $T(n) = T\left(\frac{n}{2}\right) + n, T(1) = 1$

4) $T(n) = 2T\left(\frac{n}{2}\right) + n, T(1) = 1$

Solution to #1 using iteration technique

Original equation: $T(n) = 2T\left(\frac{n}{2}\right) + 1$

Plugging in for $\frac{n}{2}$, we get $T\left(\frac{n}{2}\right) = 2T\left(\frac{\frac{n}{2}}{2}\right) + 1 = 2T\left(\frac{n}{4}\right) + 1$

Similarly, we find:

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2}\right) + 1 \\ &= 2\left(2T\left(\frac{n}{4}\right) + 1\right) + 1 \\ &= 4T\left(\frac{n}{4}\right) + 2 + 1 \\ &= 4T\left(\frac{n}{4}\right) + 3 \end{aligned}$$

Repeat, plugging in $T\left(\frac{n}{4}\right)$:

$$\begin{aligned} &= 4\left(2T\left(\frac{n}{8}\right) + 1\right) + 3 \\ &= 8T\left(\frac{n}{8}\right) + 4 + 3 \\ &= 8T\left(\frac{n}{8}\right) + 7 \end{aligned}$$

In general, after k steps, we get:

$$T(n) = 2^k T\left(\frac{n}{2^k}\right) + (2^k - 1)$$

If we let $2^k = n$ (so that $k = log_2 n$), we get

$$T(n) = nT\left(\frac{n}{n}\right) + (n-1) = n(1) + (n-1) = 2n - 1 = O(n)$$

Yielding the Big-Oh bound of the recurrence relation.