1.8 Ben Pass

1.
```
for(i=0; i < n)          n
    for(k=0; k < n)      n
        for(j=0; j < n)  n
```

So each time the j loop runs it does n.
Each time k runs it does n·(n of j) = n^2
Each time i runs it does n·(n of k) = $n(n^2)$
Overall f(T) = $O(n^3)$ = n^3

2. (same as above but analyze)

```
for(i=0, i<n)                n
    for (k=i, k<n)           n - i
        for (j=k, k<n)       (n - k)
```

Starting from the outer most we have n iterations. moving to our middle loop we have: $0, 1, 2, \ldots, n-1 = (n)\left(\frac{n}{2}\right)$ total. Moving inner loop we once again have $0, 1, 2, \ldots, n-1$ for a total of $(n)\left(\frac{n}{2}\right)\left(\frac{n}{3}\right) = \frac{n^3}{6}$ this is $O(n^3)$

```
for (i<n)
    for (k=i+1; k<n)
        for (j=k+2; j<n, k++)
```
(you gave it)

Assuming this code is correct, the code is an infinite loop because the innermost loop will never resolve.

Assuming you meant j++ in the middle, then it would be $\left(\frac{n(n-1)}{2}\right)\left(\frac{(n-2)}{2}\right)$
(instead of k++)

The outermost runs n times. The middle runs n-1 but you can think of the overall formula like a sum. Here the innermost runs similar to middle but it is at most n-3 when k=1 & i=0. As k increases the formula still holds so overall $O(n^3)$