cmput396 fall 2018 **asn4 solutions**

2. (a)
```
def foo(a,b,c):
    if a==0 and b==0 and c==0: return 1
    ttl = 1
    for aa in range(0,a): ttl += foo(aa,b,c)
    for bb in range(0,b): ttl += foo(a,bb,c)
    for cc in range(0,c): ttl += foo(a,b,cc)
    return ttl

for j in range(5): print(foo(j,j,j))
```

j	0	1	2	3	4
calls(j,j,j)	1	16	550	24136	1176106

(b)

j	0	1	2	3	4	5	6	7	8	9
mod'd foo(j,j,j) calls	1	6	24	106	480	2186	9968	45466	207392	946026

(c) Always equal. There are two possible scores, and we initialize at -1, so if we cutoff it means we can reach score 1. This is what happens in the current code.

(d)

j	0	1	2	3	4	5	6	7	8	9
solveall nim(j,j,j) updates	0	10	31	88	151	262	439	736	955	1246

3. Proof. Let $f(n) = n$. For all integers $n \geq 1$, $1/n \leq n$, so $r(n) = 3n+1/n \leq 3n+n = 4n$. So, we have found an integer k (1) and a constant c (4) so that $r(n) \leq cf(n)$, so $r(n)$ is in $O(f(n))$.

4. $c10^3 = 10$ so $c = .01$ so $r(20) = .01(20^3) = .01(8000) = 80$.

5. False. Here is a proof.

   The runtime grows at least as fast as the number of win updates, which as you can see from the chart below is growing faster than the number of positions. But this is just circumstantial evidence, it's not a proof.

   By looking at the algorithm, we see that the number of win updates is equal to the number of edges in the state space from losing positions. So, how many losing positions are there?

   Recall the nim formula: a position loses if and only if the exclusive-or sum of the binary representation of the pile sizes is identically 0. Using this, I wrote a program to compute the number of losing positions, as shown in the table below. Then I looked up the sequence in the on-line encyclopedia of integer sequences, and there is a closed form: $f(n) = (2 + (-1)^n + 3^n)/4$, or about $3^n/4$ for large n.

   (As expected, this number is close to $3^n/4$. Each pile has either 0, 1, or 2 stones, so a position loses if and only if the number of piles with exactly 1 stone is even and the number of piles with exactly 2 stones is also even. Over all possible positions, about half the positions will satisfy each property, and these events are (almost) independent, and 1/2 times 1/2 equals 1/4.)

   Finally, the number of updates from each losing position is roughly n, because from each losing position we can add a stone to each of the n piles that does not already have 2 stones.

   So the number of updates will be roughly $n * 3^n/4$. So, is there some constant c such that the number of updates will be (eventually) at most $c * 3^n$? No.

piles of 2	1	2	3	4	5	6	7	8	9
positions	3	9	27	81	243	729	2187	6561	19683
win updates	2	6	24	84	310	1098	3836	13128	44298
losing positions	1	3	7	21	61	183	547	1641	4921