Fundamentals of Computer Algorithms
Homework 1 Additional Problems

Prof. Matthew Moore

Due: 2018-08-28

1. Prove that $\sum_{k=0}^{n} k = (1/2)n(n+1)$.

2. Prove that $\sum_{k=0}^{n} a^k = \dfrac{a^{n+1}-1}{a-1}$.

3. For each of the sums below, (1) write a python function to evaluate it and (2) evaluate it by hand, showing your work (you need not simplify).

   (i) $\sum_{k=0}^{100} k$.

   (ii) $\sum_{k=0}^{100} k^2$.

   (iii) $\sum_{k=12}^{123} k^2 + k + 1$.

4. Evaluate $\sum_{0 \le k < n^2} \sum_{j=1}^{\lfloor \sqrt{k} \rfloor} 1$.

5. Consider the algorithm below.

```
def S(N):
  A = []
  for n in range(2,N+1):
    do_add_n = True
    for a in A:
      if n % a == 0:
        do_add_n = False
        break
    if do_add_n:
      A.append(n)
  return A
```

   (i) Implement this code in python.

   (ii) What is $S(10)$?

   (iii) What is $S(100)$?

   (iv) What does S do?

   (v) It is a famous fact that, in the outer loop, `len(A)` $= O(n/\ln(n))$. Use this to estimate the O-complexity of $S(N)$. It may help in your analysis to ignore the `break` statement.

$O(n^2/\ln(n))$

1

HW 1 Additional probs

Luke Dercher

1. Proof $\sum_{k=0}^{n} k = \frac{n(n+1)}{2}$

let $S = 1 + 2 + \cdots + n$

$2S = (n+1) + (n+1) + \cdots + (n+1) = n(n+1)$

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

2. Proof $\sum_{k=0}^{n} a^k = \frac{a^{n+1} - 1}{a-1}$

let $S = 1 + a + a^2 + \cdots + a^n$

$a \cdot S = a + a^2 + a^3 + \cdots + a^{n+1}$

$a \cdot S - S = (a + a^2 + \cdots + a^{n+1}) - (1 + \cdots + a^n)$

$\frac{S(a-1)}{a-1} = S = \frac{a^{n+1} - 1}{a-1}$

3(i) eval by hand

$\sum_{k=0}^{100} k = \overset{\text{(by prob 1)}}{\frac{100(100+1)}{2}} = 5050$

(ii) $\sum_{k=0}^{100} k^2 = \frac{n(n+1)(2n+1)}{6} = \frac{100(100+1)(2(100)+1)}{6} = 338350$

3.(iii) $\displaystyle\sum_{k=12}^{123} k^2+k+1 = \frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2} + n-k$

$$= \frac{123(123+1)(2(123)+1)}{6} + \frac{123(123+1)}{2} + 123-12$$

$$= 635040$$

4. Eval

$$\cancel{\sum_{k=12}} \sum_{0\le k<n^2} \sum_{j=1}^{\lfloor\sqrt{k}\rfloor} 1$$

k	0	1 2 3	4..8	9..15	16..24	25..
$\lfloor\sqrt{k}\rfloor$	0	1..1	2	3	4	5
l = block num	0	1	2	3	4	5
s = block size	1	3	5	7	9	11

$$= \sum_{0\le k<n^2} \lfloor k \rfloor$$

$$= \sum_{l=0}^{n-1} ls$$

$$= \sum_{l=0}^{n-1} l(2l+1)$$

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

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

$$= 2\left(\frac{n-1((n-1)+1)(2(n-1)+1)}{6}\right) + \frac{n-1((n-1)+1)}{2}$$

5. (i) see HW1.py
(ii) . .
(iii)

(iv.) S returns a list of all of the primes $\leq N$

(V.) The inner loop is ran n times.
Therefore $S(N) = O(\frac{n^2}{\ln(n)})$