Homework #4

1. I. Let $n=1$. Then

$$\sum_{k=1}^{1} 1 \cdot H_1 = 1 = \frac{1}{2}(1)(1+1)H_1 - \frac{1}{4}(1)(1-1)$$

$$= \frac{1}{2} \cdot 2(1) - 0$$

$$= 1 = \text{True.}$$

II. Let $n \geq 1$ and assume that $\sum_{k=1}^{n-1} kH_k = \frac{1}{2}(n-1)((n-1)+1)H_{n-1} - \frac{1}{4}(n-1)((n-1)-1)$

We must show that $\sum_{k=1}^{n} kH_k = \frac{1}{2}n(n+1)H_n - \frac{1}{4}n(n-1)$

We have

$$\sum_{k=1}^{n} kH_k = \sum_{k=1}^{n-1} kH_k + nH_n$$

$$= \frac{1}{2}(n-1)(n)H_{n-1} - \frac{1}{4}(n-1)(n-2) + nH_n$$

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

$$= \frac{1}{2}n(n-1)H_{n-1} + 1 + nH_{n-1} - \frac{1}{4}(n-1)(n-2)$$

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

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

$$= \frac{1}{2}n(n+1)H_{n-1} + 1 - \frac{n^2-3n+2}{4}$$

$$= \frac{1}{2}n(n+1)H_{n-1} + 1 - \frac{n^2}{4} - \frac{n}{4} - \frac{2n}{4} + \frac{1}{2}$$

$$= \frac{1}{2}n(n+1)H_{n-1} + \frac{3}{2} + \frac{n}{2} - \frac{n^2-n}{4}$$

$$= \frac{1}{2}n(n+1)\left(H_n - \frac{1}{n}\right) + \frac{3-n}{2} - \frac{1}{4}n(n-1)$$

2. I. Let $n=1$. Then

$$(1-1)+\frac{2}{1}\cdot\sum_{k=1}^{0} t(k) = 2(1+1)\tfrac{1}{1}-4\cdot 1$$

$$0+0 = 4-4$$

$$0=0 \Rightarrow \text{True}$$

II. Let n be chosen arbitrarily such that $t(n)=(n-1)+\frac{2}{n}\cdot\sum_{k=1}^{n-1} t(k)$

$$=2(n+1)H_n-4n$$

We must show for $n+1$ such that $t(n+1)=n+\frac{2}{n+1}\cdot\sum_{k=1}^{n} t(k)$

$$=2(n+2)H_{n+1}-4(n+1)$$

We have $t(n+1)=n+\frac{2}{n+1}\cdot\sum_{k=1}^{n} t(k)$

$$=n+\frac{2}{n+1}\cdot\sum_{k=1}^{n}(2(k+1)H_k-4k)$$

$$=n+\frac{4}{n+1}\cdot\sum_{k=1}^{n}(kH_k+H_k-2k)$$

$$=n+\frac{4}{n+1}\cdot\sum_{k=1}^{n} kH_k+\frac{4}{n+1}\cdot\sum_{k=1}^{n} H_k-\frac{4}{n+1}\cdot\sum_{k=1}^{n} 2k$$

$$=n+\frac{4}{n+1}\cdot\left(\frac{1}{2}\cdot n(n-1)H_n-\frac{1}{4}\cdot n(n-1)\right)+\frac{4}{n+1}\cdot((n+1)H_n-n)-\frac{8}{n+1}\left(\frac{n(n+1)}{2}\right)$$

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

$$=-3n+2nH_{n+1}-\frac{2n}{n+1}-\frac{n^2-n}{n+1}+4H_{n+1}-\frac{4}{n+1}-\frac{4n}{n+1}$$

$$=-3n+2nH_{n+1}-\frac{n^2-n}{n+1}+4H_{n+1}+\frac{-4-6n}{n+1}$$

$$=-3n+2(n+2)H_{n+1}+\frac{-n^2+n}{n+1}+\frac{-4-6n}{n+1}$$

$$=-3n+2(n+2)H_{n+1}+\frac{-n^2-5n-4}{n+1}$$

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

$$=2(n+2)H_{n+1}-3n-(n+4)$$

$$=2(n+2)H_{n+1}-4n-4$$

$$=2(n+2)H_{n+1}-4(n+1)$$ ▨

3. a) min(x,y) max(x,y) sort(x,y)

return (x<y?x:y) return (x>y?x:y) return(x<y? (x,y):(y,x)

```
Extrema(A, p, r)
  if p=r
      return (A[p], A[p])
  else if p=r-1
      return sort(A[p], A[p+1])
  else
      q = ⌊(p+r)/2⌋
      (m1, M1) = Extrema(A, p, q)
      (m2, M2) = Extrema(A, q+1, r)
      return(min(m1, m2), max(M1, M2))
```

b) I. Let n=1. Then we have p=r and line 2 is executed, which is true as A[p] is the min and max.

Let n=2. Then we have p=r-1 and line 4 is executed, which is true as it returns the 2 elements in (min, max) format.

II. Let n>2 and assume Extrema is correct on length less than n.

We must show the same case holds true for length n.

Since n>2, the algorithm is directed to line 6, and line 7 is executed and 2 subarrays are created. Note that:

length(A[p...q]) = q-p+1 < n

length(A[q+1...r]) = r-q < n

Since the two subarrays have length smaller than n, by the IH, lines 7 and 8 correctly returns the correct output, and line 9 determines the min and max of A[p...r], which is of length n.

Thus the algorithm is correct for size n. ▨

c) $T(n) = \begin{cases} 0 & n=1 \\ 1 & n=2 \\ T(\lceil \frac{n}{2} \rceil - 1) + T(\lfloor \frac{n}{2} \rfloor + 1) + 3 & n>2 \end{cases}$

Note: $T(\lceil \frac{n}{2} \rceil - 1) + T(\lfloor \frac{n}{2} \rfloor - 1) = T(n-2)$

$T(1) = \lceil \frac{3}{2} \rceil - 2 = 0 = True$

$T(2) = \lceil \frac{6}{2} \rceil - 2 = 1 = True$

$RHS = T(n-2) + 3$

$= \left(\lceil \frac{3(n-2)}{2} \rceil - 2 \right) + 3$

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

$= \frac{3n}{2} - 5 + 3$

$= \frac{3n}{2} - 2$

$= T(n) = LHS$ ■

4. The recurrence for dot product matrix multiplication of a $n \times n$ is:

$$T(n) = \begin{cases} 1 & n=1 \\ 8T(\frac{n}{2}) + 1 & n>2 \end{cases}$$

By applying master theorem, we get $n^{\lg 8}$ and n^0.

Thus the run time is $O(n^3)$.

By using Strassen's, the recurrence is

$$T(n) = \begin{cases} 1 & n=1 \\ 7T(\frac{n}{2}) + 1 & n>2 \end{cases}$$

By applying master theorem, we get $n^{\lg 7}$ and n^0.

By comparing growth rates, we go by Strassen's.

Thus $T(n) = \Theta(n^{\lg 7})$. ■

5. We can use the dot product to find a solution for k multiplications.
For instance, we can iterate through $k \cdot i \cdot m$ where $k,i,m \leq n$ by doing
the dot product k times. This can be written as $k \cdot i \cdot m \leq n \cdot n \cdot n \Rightarrow O(n^3)$.

However, we can use recursion to save time, which is shown in problem 4
as $O(n^3)$ became $\Theta(n^{\lg 7})$. Thus the same can be said for here where $O(n^3)$
can be changed to less than $O(n^3)$ by recursion. There is also an implication that
a function bounded below $O(n^3)$ cannot be $O(n^3)$, hence the recursive
solution must have a maximum speed of $o(n^3)$.

6. a) To be 1-sorted means the array is in increasing order after 1 swap

b) Let $A=[1,2,3,4,6,5]$. Then the average of 2 consecutive
numbers are increasing, but the array is not sorted

c) $\sum_{j=1}^{i+k-1} A[j] \leq \sum_{j=i+1}^{i+k} A[j]$

$A[i] + \sum_{j=1}^{i+k-1} A[j] \leq \sum_{j=i+1}^{i+k-1} A[j] + A[i+k]$

$A[i] \leq A[i+k]$ ∎

d) The algorithm would return if $k >$ length(A).
Else it would divide the array into $\lceil k \rceil$ subarrays.
Then it will sort the subarrays into increasing order and pick the first
element of each subarray to be in the array. This step is repeated until
no elements are left in the subarrays. Since the subarrays are sorted,
the array would contain elements in an increasing order, thus is sorted.