284 Assignment 2

Zane Larking

1.

(A) Solve the following recurrence relation. Show all working. You can leave the n-th harmonic function as H_n in your answer. Show all working. **[10 marks]**

$$\begin{cases} T(n) = \frac{1}{n}(T(0) + T(1) + \ldots + T(n-1)) + 5n, \\ T(0) = 0. \end{cases}$$

$T(0) = 0$

$T(1) = \frac{1}{1}(T(0)) + 5(1) = 5$

$T(2) = \frac{1}{2}\left(\sum_{i=0}^{2-1} T(i)\right) + 5(2) = \frac{5}{2} + 10$

$T(3) = \frac{1}{3}\left(\sum_{i=0}^{3-1} T(i)\right) + 5(3) = \frac{\frac{5}{2}+10}{3} + 15$

$T(4) = \frac{1}{4}\left(\sum^{4-1} T(i)\right) + 5(4) = \frac{\frac{\frac{5}{2}+10}{3}+15}{4} + 20$

$T(5) = \frac{1}{5}\left(\sum_{i=0}^{5-1} T(i)\right) + 5(5) = \frac{\frac{\frac{\frac{5}{2}+10}{3}+15}{4}+20}{5} + 25$

$T(n) = \frac{1}{n}\left(\sum_{i=0}^{5-1} T(i)\right) + 5(n) = \frac{\frac{\frac{\frac{\frac{5}{2}+10}{3}+15}{4}+20}{5} + 25 + \ldots}{n} + 5n$

$T(n) = 5n + \frac{5(n-1)}{n} + \frac{5(n-2)}{(n)(n-1)} + \frac{5(n-3)}{(n)(n-1)(n-2)} + \ldots + \frac{5(2)}{\left(\frac{n!}{1!}\right)} + \frac{5(1)}{\left(\frac{n!}{0!}\right)}$

$T(n) = \sum_{j=1}^{n} \frac{5(j)}{\left(\frac{n!}{j!}\right)} = \sum_{j=1}^{n} \frac{5 \cdot j \cdot j!}{n!}$

284 Assignment 2

Zane Larking

1.

(A) Solve the following recurrence relation. Show all working. You can leave the n-th harmonic function as H_n in your answer. Show all working. **[10 marks]**

$$\begin{cases} T(n) = \frac{1}{n}(T(0) + T(1) + \ldots + T(n-1)) + 5n, \\ T(0) = 0. \end{cases}$$

1

$$nT(n) = (T(0) + T(1) + \ldots + T(n-2) + T(n-1)) + 5n^2$$

2

$$(n-1)T(n-1) = (T(0) + T(1) + \ldots + T(n-2)) + 5(n-1)^2$$

1 − 2

$$nT(n) - (n-1)T(n-1) = T(n-1) + 5n^2 - 5(n-1)^2$$

$$nT(n) - (n)T(n-1) = 5n^2 - 5n^2 + 10n - 5$$

telescoping

$$T(n) = \cancel{T(n-1)} + \frac{5}{n} + 10$$

$$\cancel{T(n-1)} = \cancel{T(n-2)} + \frac{5}{n-1} + 10$$

$$\cancel{T(n-2)} = \cancel{T(n-3)} + \frac{5}{n-2} + 10$$

$$\cancel{T(n-3)} = \cancel{T(n-4)} + \frac{5}{n-3} + 10$$

$$\vdots$$

$$\cancel{T(n-i)} = T(n-(i+1)) + \frac{5}{n-i} + 10$$

$i = 0$ to $i = n-1$: n

$$\Rightarrow T(n-(n-1)) = T(n-(n-1)+1) + \frac{5}{n-(n-1)} + 10$$

$$\Rightarrow T(1) = T(0) + \frac{5}{1} + 10 = \frac{5}{1} + 10$$

Summing the above telescoped equations

$$T(n) = \cancel{T(0)}^{0} + 5\left(\frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \ldots \frac{1}{3} + \frac{1}{2} + \frac{1}{1}\right) + 10n$$

$$T(n) = 5\sum_{i=0}^{n}\left[\frac{1}{i}\right] + 10n$$

$$T(n) = 5H_n + 10n$$

(B) Solve the following recurrence relation. You can assume that n is a power of 7. Show all working. **[10 marks]**

$$\begin{cases} T(n) = T\left(\left\lfloor \frac{n}{7} \right\rfloor\right) + \log_3(n), \\ T(1) = 0. \end{cases}$$

$n = 7^k$, $k \in \mathbb{N}$

$$T(n) = T\left(\left\lfloor \frac{7^k}{7} \right\rfloor\right) + \log_3(7^k)$$

telescoping

$$T(7^k) = T(7^{k-1}) + \log_3(7^k)$$
$$T(7^{k-1}) = T(7^{k-2}) + \log_3(7^{k-1})$$
$$T(7^{k-2}) = T(7^{k-3}) + \log_3(7^{k-1})$$
$$T(7^{k-3}) = T(7^{k-4}) + \log_3(7^{k-1})$$
$$\vdots$$
$$T(7^{k-(k-1)}) = T(7^{k-k}) + \log_3(7^{k-(k-1)})$$

k

$$\Rightarrow T(7) = \overset{0}{\cancel{T(0)}} + \log_3(7) = 0 + \log_3(7)$$

Summing the above telescoped equations

$$T(7^k) = 0 + \sum_{i=1}^{k} \log_3(7^i)$$
$$= \log_3\left(\prod_{i=1}^{k} 7^i\right)$$
$$= \log_3\left(7^{\sum_{i=1}^{k} i}\right)$$

$$\sum_{i=1}^{k} i = \frac{k(k+1)}{2}$$

$$= \log_3\left(7^{\frac{k(k+1)}{2}}\right)$$
$$= \log_3\left((7^k)^{\frac{(k+1)}{2}}\right)$$
$$= \frac{k+1}{2} \log_3(7^k)$$

$$k = \log_7(n)$$

$$T(n) = \frac{\log_7(n) + 1}{2} \log_3(n)$$

2 Algorithm Analysis [20 marks]

You are presented with a brute force string-search algorithm given below.

Input: text t of length n and word p of length 3.

Output: a position at which we have p in the text. If p can be found in the text several times then we take the first occurrence.

Algorithm:

i) We start from the beginning of the text index $i = 0$

ii) If $i > n - 3$ then terminate and return FALSE.

iii) Query if the sub-string $t[i, ..., i + 2]$ is the same as p.

v) If yes, return i and terminate. Otherwise set i to $i + 1$ and go to step ii)

Note that we are looking only for the first match.
The pseudocode is given below:

Algorithm 1 String-search algorithm

```
function FIND(string t[0..n − 1], word p of length 3)
    i ← 0                                  0    1
    for i ← 0 to n − 3 do                  0    n−2
        if [t[i, ..., i + 2]] = p then     3    n−2
            return i                       0    ?
    return FALSE
```

I am regarding only array operations as elementary operations

(A) Find the asymptotic running time in the best case.

Best possible case is where p accounts for the first 3 elements of the array. Querying the sub-string is only array operation and takes 3 operations each time the loop runs. The loop runs at least once.

$T_B(n) = 3 \cdot 1 = 3$

let $n > n_0$, $n_0 = 1$, $c_1 = 2$, $c_2 = 4$

$2 \cdot 1 \leq 3 \leq 4 \cdot 1$

$\therefore \forall n \geq n_0 \; \exists c_1, c_2$ s.t $c_1 \leq T_B(n) \leq c_2$

$\therefore T_B(n) \in \Theta(1)$

(B) Find the asymptotic running time in the worst case.

Worst possible case is where p either accounts for the last 3 elements of the array or does not exist within t at all. Querying the sub-string is only array operation and takes 3 operations each time the loop runs. The loop runs at most n-2 times.

$$T_w(n) = 3(n-2) = 3n-6$$

let $n \geq n_0$, $n_0 = 1$, $c_1 = -3$, $c_2 = 3$

$$-3n = 3n - 6n \leq 3n-6 \leq 3n$$

$$\therefore \forall n \geq n_0 \; \exists c_1, c_2 \text{ s.t } c_1 n \leq T_w(n) \leq c_2 n$$

$$\therefore T_w(n) \in \Theta(n)$$

(C) Find the asymptotic running time in the average case. You may assume that there is at most one occurrence of p in the text, and any such occurrence (or lack thereof) is equally likely. [**10 marks**]

The run time of the average case should be the average of the best and worst case.

$$T_A(n) = \frac{(3n-6)+3}{2} = \frac{3}{2}n - \frac{3}{2} = \frac{3}{2}(n-1)$$

let $n \geq n_0$, $n_0 = 1$, $c_1 = 0$, $c_2 = 3/2$

$$0n = \frac{3}{2}n - \frac{3}{2}n \leq \frac{3}{2}n - \frac{3}{2} \leq \frac{3}{2}n$$

$$\therefore \forall n \geq n_0 \; \exists c_1, c_2 \text{ s.t } c_1 n \leq T_A(n) \leq c_2 n$$

$$\therefore T_A(n) \in \Theta(n)$$

3 Binary Search Trees [20 marks]

(A) Perform the linear time algorithm to construct a maximum heap on the following array. Please show the heap structure after each iteration. [**5 marks**]

$$A = [0, 54, 93, 2, 12, 32, 40, 16, 25, 51].$$

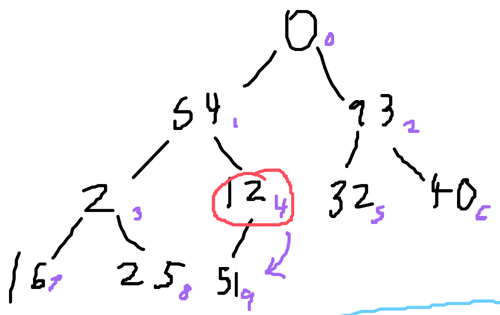
The bottom up approach/array implementation is the linear algorithm.

Start by sinking the first non-leaf node

$i = \left\lfloor \frac{n-1}{2} \right\rfloor, \quad n = 10$

$i = \left\lfloor \frac{9}{2} \right\rfloor = 4$

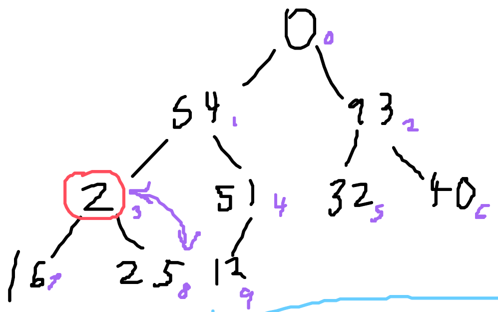
∴ sink 12 first



0, 54, 93, 2, 12, 32, 40, 16, 25, 51

$i = 3$

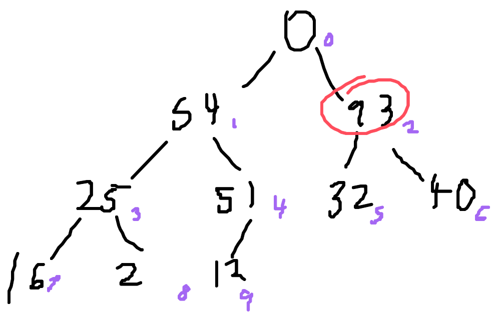
sink 2



0, 54, 93, 2, 51, 32, 40, 16, 25, 12

$i = 2$

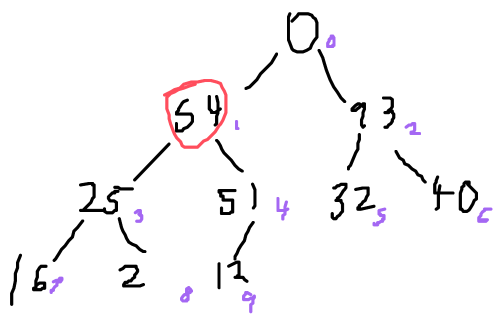
93 is fully sunken already



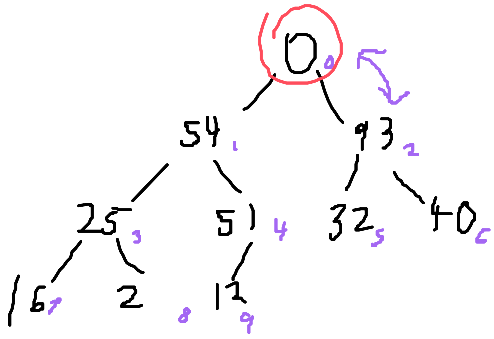
0, 54, 93, 25, 51, 32, 40, 16, 2, 12

$i = 1$

54 is fully sunken already



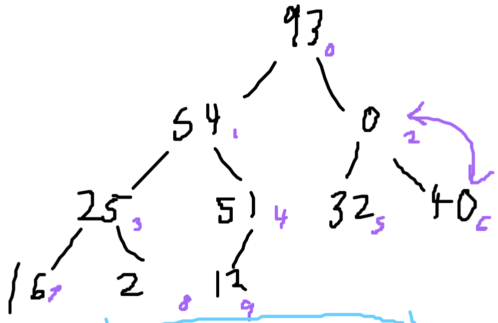
0, 54, 93, 25, 51, 32, 40, 16, 2, 12



$i=0$

Sink 0

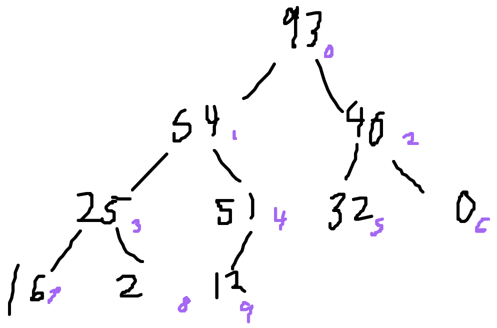
0, 54, 93, 25, 51, 32, 40, 16, 2, 12



$i=i\cdot 2+2$

Sink 0 again

93, 54, 0, 25, 51, 32, 40, 16, 2, 12

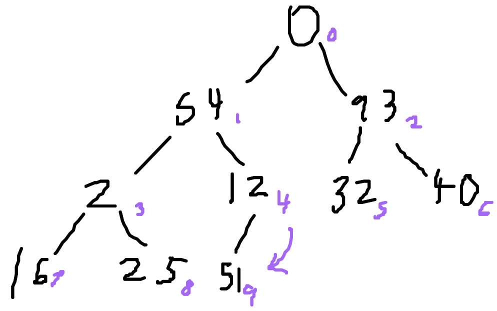


fully heapified

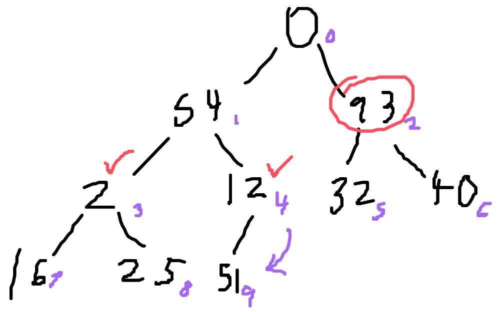
93, 54, 40, 25, 51, 32, 0, 16, 2, 12

93, 54, 40, 25, 51, 32, 0, 16, 2, 12

(B) Consider the same array A above. Is A a minimum heap? Justify your answer by briefly explaining the min-heap property. If A is not a min-heap, then restore the min-heap property and show the restored array A'. **[5 marks]**



0, 54, 93, 2, 12, 32, 40, 16, 25, 51



0, 54, 93, 2, 12, 32, 40, 16, 25, 51

For a heap to be a min-heap each parent node must be less than both of their children nodes. In this heap nodes 1 and 2 are greater than it's children.

start sinking at

$i = \lfloor \frac{n}{2} \rfloor - 1 = 4$

12 is fully sunk

$i = 3$

2 is fully sunk

$i = 2$

$a[2] < a[2 \cdot 2 + 1] < a[2 \cdot 2 +]$

sink index 2 with right child (40)