Please fill in the following information.

THEN PUT YOUR EMAIL ADDRESS ON EVERY PAGE OF THE EXAM !

NAME: **SOLUTION**

Email (please put complete address):

Neatness counts. We will not grade what we cannot read.

Exam is worth 100 points. Please phrase your answer succinctly. Write your answer on the same page, on which the question is given. There are six questions and a total of twelve pages. Make sure you turn in all these pages.

Do not attempt to look at other students' work. Keep your answers to yourself.
Any sort of cheating will result in a zero grade.

Read and sign the statement below. Wait for instructions to start the examination before continuing to the next page.

"I signify that the work shown in this examination booklet is my own and that I have not received any assistance from other students nor given any assistance to other students."

_____(Signature)

Q(1) Rank the following functions by increasing growth rates. If two functions differ by only a constant, group them together. **(10 points total)**

$$n^{4\lg n}, \sqrt{\lg n}, \lg\left(\sqrt{n}\right), \frac{(n+1)^2}{n^2+n-2}, 10n^2, n!, (n/\lg n)^3, n2^n, \lg n, n^3$$

Answer:

$$\frac{(n+1)^2}{n^2+n-2}, \sqrt{\lg n}, \ \{\lg\sqrt{n}, \ \lg n\}, \ 10n^2, \ (n/\lg n)^3, n^3, \ n^{4\lg n}, \ n2^n, \ n!$$

Q (2) Give a closed form for the functions given in parts (a), (b), (c). **(20 points total)**

(a) $T(0) = T(1) = 0$ and $T(n) = T(n-2) + n + 3$, for $n \geq 2$. (5 points)

Answer:

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

$$= (n+3) + (n-2+3) + T(n-4)$$

$$= (n+3) + (n-2+3) + (n-4+3) + T(n-6)$$

$$\bullet$$

$$\bullet$$

$$\bullet$$

$$= \sum_{i=0}^{\lfloor n/2 \rfloor - 1} (n + 3 - 2i)$$

$$= n\lfloor n/2 \rfloor + 3\lfloor n/2 \rfloor - 2\sum_{i=0}^{\lfloor n/2 \rfloor - 1} i$$

$$= (n+3)\lfloor n/2 \rfloor - \lfloor n/2 \rfloor(\lfloor n/2 \rfloor - 1)$$

$$= (n + 3 - \lfloor n/2 \rfloor + 1)\ \lfloor n/2 \rfloor$$

$$= (\lceil n/2 \rceil + 4)\lfloor n/2 \rfloor$$

$$= \Theta(n^2)$$

(b) $T(n) = \sum_{i-1}^{n^2} (n+i)$. (5 points)

Answer:

$$T(n) = \sum_{i=1}^{n^2} (n+i)$$

$$= \sum_{i=1}^{n^2} n + \sum_{i=1}^{n^2} i$$

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

$$= \Theta(n^4)$$

(c) $T(n) = \sum_{i=1}^{n} \left[c_1 + \sum_{j=1}^{n} c_2 \right]$ (5 points)

Answer:

$$T(n) = \sum_{i=1}^{n} [c_1 + \sum_{j=i}^{n} c_2]$$

$$= \sum_{i=1}^{n} [c_1 + c_2(n-i+1)]$$

$$= \sum_{i=1}^{n} c_1 + c_2 \sum_{i=1}^{n} (n-i+1)$$

$$= nc_1 + c_2 n^2 + c_2 n - c_2 \frac{n(n+1)}{2}$$

$$= nc_1 + c_2(\frac{n^2+n}{2})$$

$$= \Theta(n^2)$$

(d) Prove or disprove: $n^{\frac{1}{2}} = \Theta\left(n^{\frac{1}{4}} \right)$ (5 points)

Answer: False.

$$\lim_{n \to \infty} \frac{n^{\frac{1}{2}}}{n^{\frac{1}{4}}} = \lim_{n \to \infty} n^{\frac{1}{4}} = \infty.$$

Q(3) **(20 points total)**

(a) Analyze the time complexity of **fn_nested(n)** and solve it in asymptotic form (5 points).

fn+nested(n)

1. $k \leftarrow 0$
2. **for** $i \leftarrow 1$ **to** n
3. **for** $j \leftarrow i$ **to** n
4. **do** $k \leftarrow k + j$

Answer:

$$T(n) = \sum_{i=1}^{n} \sum_{j=i}^{n} C_1 + C_2 = C_1 \sum_{i=1}^{n} (n - i + 1) + C_2 = C_1 n^2 - C_1 \frac{n(n+1)}{2} + C_1 n + C_2$$

$$= C_1 \frac{n(n+1)}{2} + C_2 = \Theta(n^2)$$

Q 3 (cont'd)

(b) Analyze the time complexity of **Do_it(n)** and solve it in asymptotic form. (5 points)

```
Do_in(n)
1. j ← 2
2. if n > 10
3.     then Do_it(n/3)
4.         for i ← 1 to n
5.             do j ← 3
6.         Do_it(n/3)
```

Answer:

$T(n) = 2T(n/3) + n,\ n > 10$

$T(n) = \Theta(1),\ n \le 10$

By applying Master Theorem, $\gamma = \log_3 2 < 1$ and $f(n) = n$, hence $T(n) = \Theta(n)$.

(f(n) dominates; Case 3 of Master Theorem applies).

Q3 (cont'd)

(c) (10 points) Given number **a** and positive integer **n**, write a recursive expression for computing the value of a^{2^n} by repeated squaring (example of repeated squaring is $x^4 = x^2 * x^2$). Assuming that multiplying two numbers takes constant time, determine the tight asymptotic complexity of your recursive expression using Θ notation.

Solution:

$$a^{2^n} = a^{2^{n-1}} a^{2^{n-1}}$$

Letting $T(n) = a^{2^n}$, we have:

$$\begin{aligned} T(n) &= 2T(n-1) + \Theta(1) \\ &= \Theta(n) \\ &= O(n \lg n) \end{aligned}$$

Note: it is important to understand that $\Theta(n) = O(n \lg n)$.

Q(4) (15 points)

Suppose **m** balls are thrown into **n** bins. However, the bins are not equally likely to be chosen. Each time a ball is thrown, it goes into bin 1 with probability p1, bin 2 with probability p2, and so on. The numbers p1; p2; ...., pn are nonnegative and sum to 1.

(a) Let Xi be the number of balls that fall into bin i. What is the probability that Xi is exactly k? (3 points)

$$\binom{m}{k} p_i^k (1-p_i)^{m-k}$$

(b) Give an upper bound on the probability that there is an empty bin. The larger m is, the smaller this probability. Assuming the uniform case, where all pi's are equal, roughly how large should **m** be for this probability to be less than 1/2 ? (6 points)

Let A_i = event bin i is empty

$P[A_i] = (1-p_i)^m$

Using Union Bound

$$P[A_1 \cup A_2 \cdots \cup A_n] \le \sum_{i=1}^{n} P[A_i] = \sum_{i=1}^{n} (1-p_i)^m$$

We need

$$\binom{m}{2} \sum_{i=1}^{n} p_i^m < \frac{1}{2}$$

which gives

$m = \Theta(n \log n)$

(c) Give an upper bound on the probability that there is a bin with at least 2 elements. The larger **m** is, the larger this probability. Assuming the uniform case, where all pi's are equal, roughly how small must **m** be for this probability to be less than 1/2? (6 points)

Let A_i = Event bin i has at least two balls

$P[A_i] = \binom{m}{2} p_i^2$

Using Union Bound $P[A_1 \cup A_2 \cdots \cup A_n] \le \sum_{i=1}^{n} P[A_i] \le \binom{m}{2} \sum_{i=1}^{n} p_i^2$

Solution for finding smallest m is the same as solution to the birthday paradox.

Q(5) **(20 points total)**

(a) What is wrong with the following version of HEAP_EXTRACT_MAX() on a heap (note this is different than the version we studied)? **(6 points)**

Remove the root, leaving a hole and swap the hole with the largest child, moving the hole down one level.

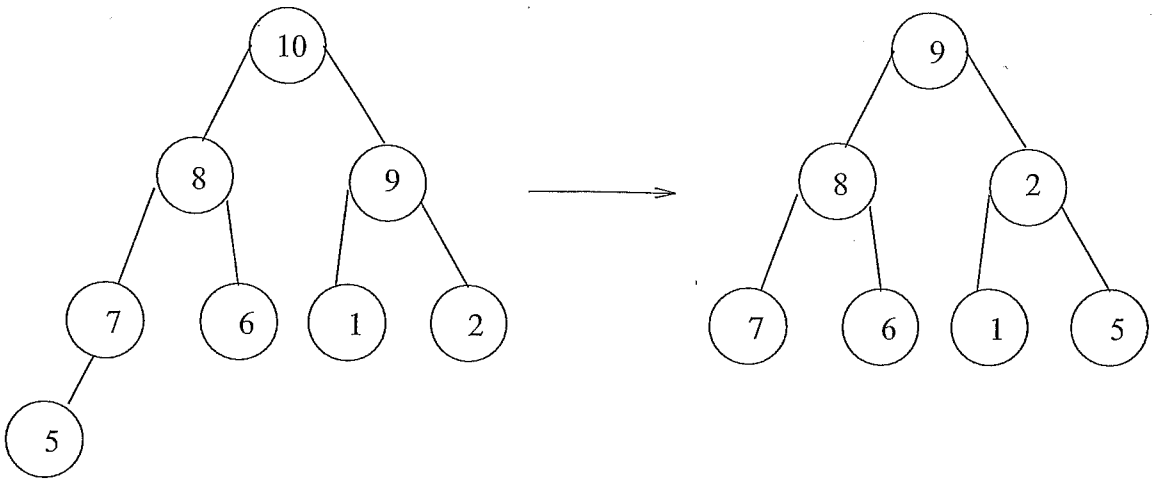
Continue sending the hole down the heap until it reaches a leaf position.

Move the last key into that hole.

Show an example of 8-nodes heap to illustrate your argument.

Answer:

If we move the hole in this way, the heap property may be violated as the following Figure shows. We would need to check the path from the last inserted key in the hole up to the root to ensure that the heap property is maintained. If the hole moves into the position of the last key in the heap, there would be nothing to swap into the hole. It is necessary to decrement the heapsize after the swap.



Q(5, cont'd)

(b) Suppose that you are given a maximum heap containing $n = 2^k - 1$ keys and that the heap is complete (i.e., the bottom level is full). Which of the nodes in the heap could contain the specified key and how many nodes are there that could contain that key? Assume that all elements in the heap are distinct (i.e., no duplicate keys). **You must explain answers to get a grade. (6 points)**

i. The largest key (1 point)

Answer: Only one, the root, because of the heap property.

ii. The second largest key (1 point)

Answer: Nodes that are direct children of the rot nodes; there are two of them.

iii. The third largest key (2 points)

Answer: Nodes that are direct children or the grand children of the rot nodes; there are $2 + 4 = 6$ possible locations. The second largest can either be a sibling of the second largest or a child of the second largest. Since we do not know which subtree the second largest is stored in, there are 6 possible locations.

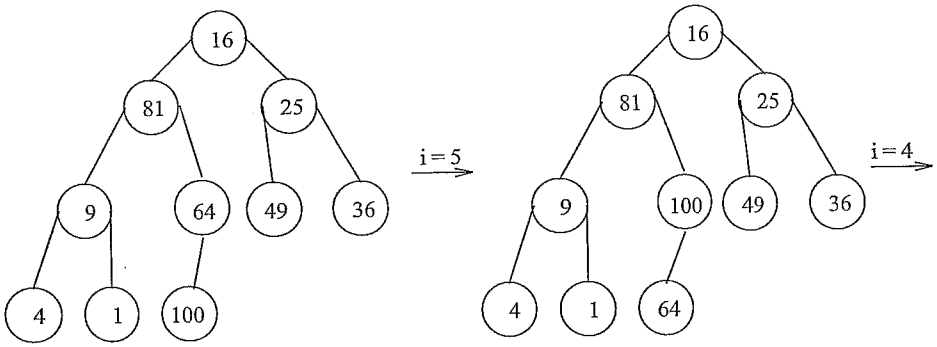
iv. The smallest key (2 points)

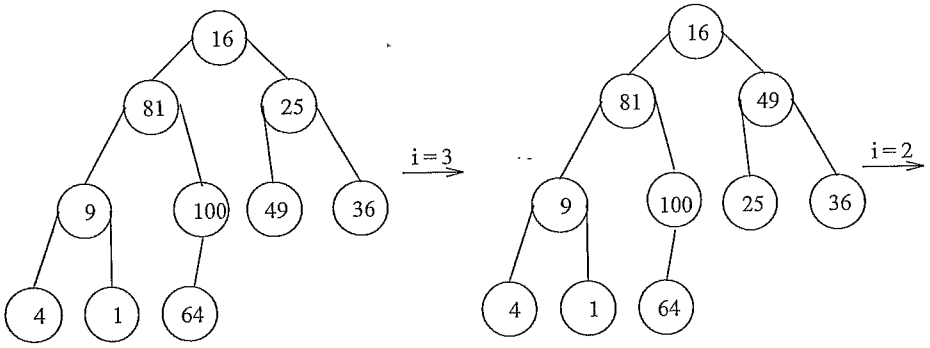
Answer: Any of the 2^{k-1} leaves can contain the smallest element.

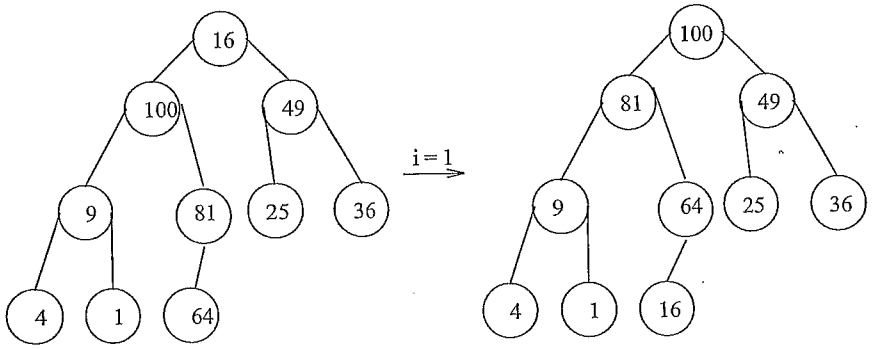
Q(5, cont'd)

(c) Show the final state of the following input array after using the Build-Heap algorithm for generating a valid heap within A. You do not need to show all the steps of the algorithm. Remember, there is only one unique solution for the given input **(8 points)**

$A[1..10] = \{16, 81, 25, 9, 64, 49, 36, 4, 1, 100\}$







Final Heap:

100	81	49	9	64	25	36	4	1	16
1	2	3	4	5	6	7	8	9	10

Q (6) (15 points) The recurrence equation for Quicksort can be given as:

$T(n) = T(\alpha n) + T((1-\alpha)n) + n$, where α is a constant ranging $0 < \alpha < 1$.

Using a recursion tree obtain an asymptotical form of T(n) in Θ notation (Hint: you need to find bound for both the longest and the shortest paths).

Answer:

Using a recursion tree shown in Figure , assume that $0 < \alpha \leq ½$ and $½ \leq (1 - \alpha) < 1$.

The length of the shortest path goes to 1 quicker, i.e.:

$$\alpha^k n = 1 \Rightarrow n = \frac{1}{a^k} \Rightarrow \log\frac{1}{\alpha} n = k. \quad \text{So } T(n) \geq n \log\frac{1}{\alpha} n = \Omega(n \lg n).$$

On the other hand, $(1-\alpha)^k n = 1$ is the longest path length. Thus,

$$n = \frac{1}{(1-\alpha)^k} \Rightarrow \log\frac{1}{(1-\alpha)} n = k$$

Hence $T(n) \leq n \log\frac{1}{(1-\alpha)} n = 0(n \lg n)$.

Thus, $T(n) = \Theta(n \lg n)$. We can show a similar proof when we assume that $1/2 \leq \alpha < 1$ and $0 < (1-\alpha) \leq 1/2$.