Department of Computer Science
Data Structures (CSC212)
Final Exam (2nd Semester 1428-29)
Date: 8/6/1429H

Time: 3 hours **Marks: 100**

Question 1 (30 marks)

n+2 n+1

(a) What is the step count of the 'for' statement: `for (int i = 0; i <= n; i++) x++;`.

(b) What is the step count of the statement S1 in the following code segment:

```
for (int i = 1; i < n; i++) for (int j = i; j < n ; j++) S1;
```

Note : $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$.

(c) What is the growth rate of the following functions in terms of the big-O notation:

(i) $f(n) = 10$ (ii) $f(n) = (2n^3 + n^2 + 1)\log(n^2 + 1)$.

(d) Why is remove operation in singly linked list O(n)?

(e) Give one advantage of having sentinel nodes in lists?

(f) What type of structure do trees have? التركيب البنائي

(g) Give a formula for the number of nodes in a full binary tree in terms of its height, *h*.

(h) Give a formula for the minimum number of nodes in a complete binary tree in terms of its height, *h*.

(i) What is the *worst-case* and the *best-case* time complexity of the FindKey operation in BSTs?

(j) Draw an expression tree for the expression: (a*b/2+3*(c-1)).

(k) What is the minimum number of keys a non-leaf node (except the root) can have in a B+-tree of order M?

(l) What is the minimum and maximum number of data elements a root can have in a B+-tree?

(m) What is the time complexity of HeapSort algorithm?

(n) What is $(\log_2 100)$ equal to?

Question 2. (15 marks)

(a) Draw a graphical representation of a doubly-linked list.

(b) Implement **insert** and **~~remove~~** operations for a doubly-linked list, according to the specification of ADT List.

Question 3. (15 marks)

(a) Convert the following array of integers into a min-heap and show the resulting heap as an array and as a tree. The array is {12, 14, 3, 16, 8, 7, 10, 17, 5, 11, 9, 6, 13, 15, 4}. Use SiftDown operation.

(b) From the heap you obtained in part (a) delete the minimum element (deleteMin) twice and show the result after each deletion as an array.

0

Final Exam Sem2 28/29 CSC212

Q1-

(a) What is the step count of the 'fot' statement :

```
for(int i = 0 ; i <= n ; i++)
    x++;
```

Answer:

f(n) = n+2+n+1
2n + 3

(b) What is the step count of the statement S1 in the following code segment:

```
for(int i = 1 ; i < n ; i++)
    for(int j = i ; j < n ; j++)
        S1;
```

Answer:

Suppose n = 5.

first 'for' has steps : n
second 'for' :

i	j	j < n	S1
1	1	1	0
2	2	2	1
3	3	3	2
4	4	4	3
5			

step count for S1 : 0 + 1 + 2 + 3 + ...+ n-2

$$\sum_{i=1}^{N} i = 1+2+3+\ldots+(n-2)+(n-1)+(n) = \frac{n(n+1)}{2}$$

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

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

$$= 2n^2 - 2n - 2$$

(c)

(i) f(n) = 10

O(1)

(ii)

$$f(n) = (2n^3 + n^2 + 1) \log(n^2 + 1)$$

$$2n^3 + n^2 + 1 < 2n^3 + n^3 + n^3 \quad where \quad n_0 \geq 1$$

$$< 4n^3 \quad where \quad n_0 \geq 1$$

$$o(n^3) \quad where \quad c=4 \text{ and } n_0 \geq 1$$

$$\log(n^2 + 1) \leq \log(n^2 + n^2) \quad where \quad n_0 \geq 1$$

$$\leq \log(2n^2)$$

$$\leq \log(2) + \log(n^2)$$

$$\leq \log(n) + 2 \log(n)$$

$$\leq 3 \log(n)$$

$$o(\log(n)) \quad where \quad c=3 \text{ and } n_0 \geq 1$$

$$O(n^3)O(\log(n))$$

$$O(n^3 \log(n)) \quad where \quad C=12 \quad \text{and } n_0 \geq 1$$

(d) Because you to do ~~shift~~ go through for all nodes [redacted] before current, you do the shift with loop.

(e) Advantage of Sentinel nodes Linked list : It simplifies code because all nodes will have previous and next node, you will not have any special case.

(f) Hierarchical.

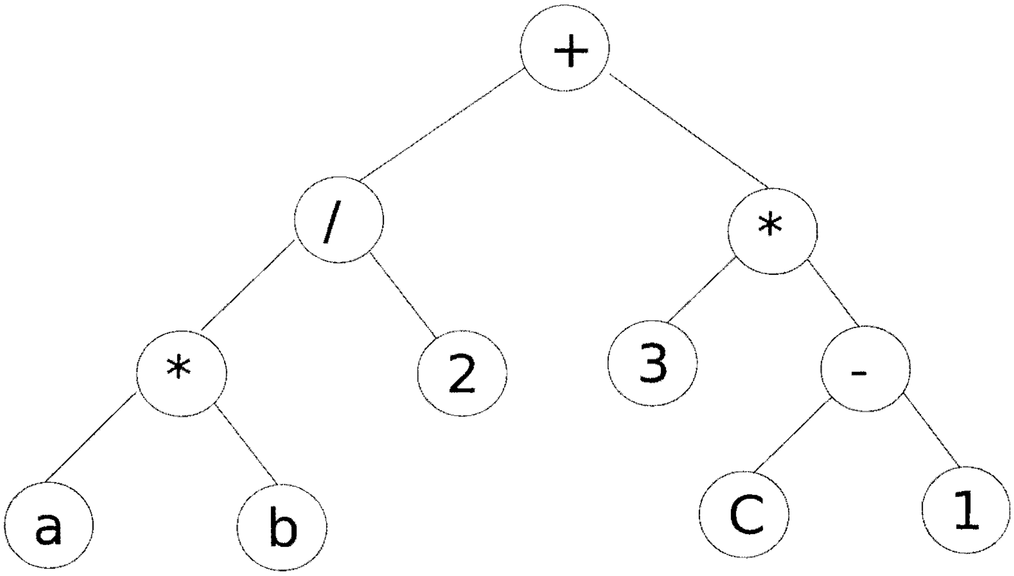
(g) Max. Number of nodes in a full binary tree $= 2^h - 1$

(h) [redacted]

Best case O(log(n)) Worst case O(n).

(i) Min Number of nodes in a complete binary tree
$= 2^{h-1}$.

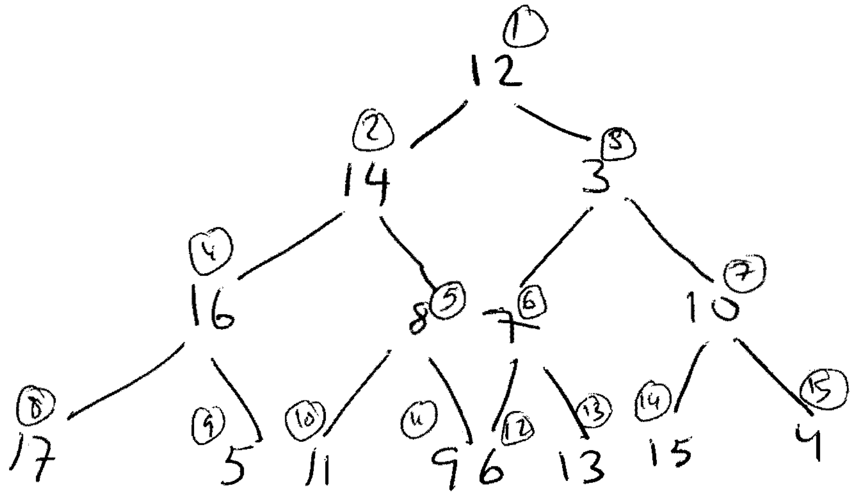
(j) (a*b/2+3*(c-1))
((a*b)/2) + (3*(c-1))



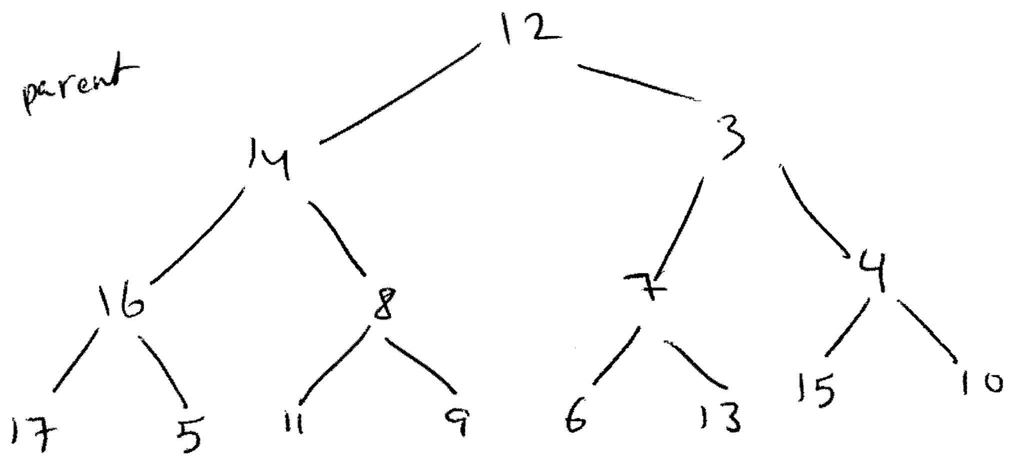
(k) Minimum number of keys in a leaf node except root with order m = (m/2 - 1).
(l) Minimum = 1 Maximum = order m.
(m) Time complixity of heapsort = $O(n \log_2(n))$.
(n) $\log_2 (100) = \log_2(10^2) = 2 \log_2(10)$

Q3

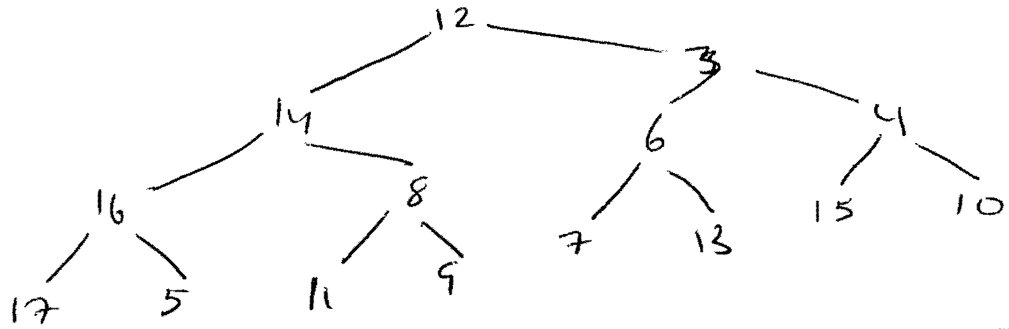
(a)



Start with last parent
index 7

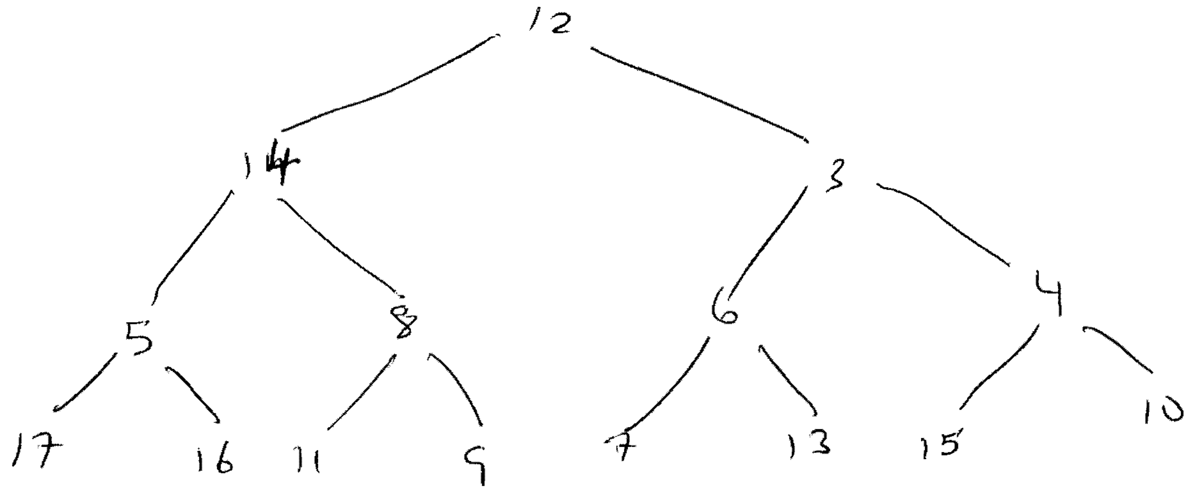


index 6



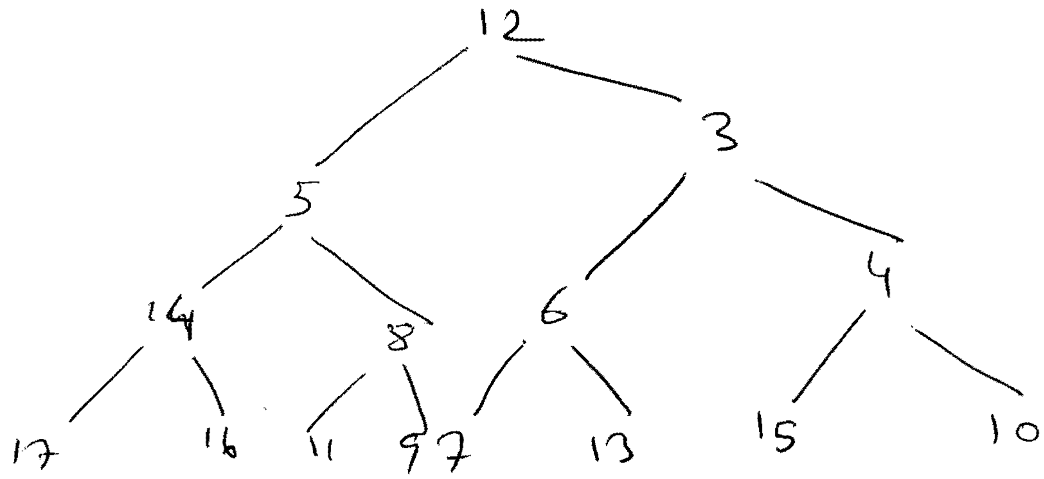
index 5 (ok)

index 4

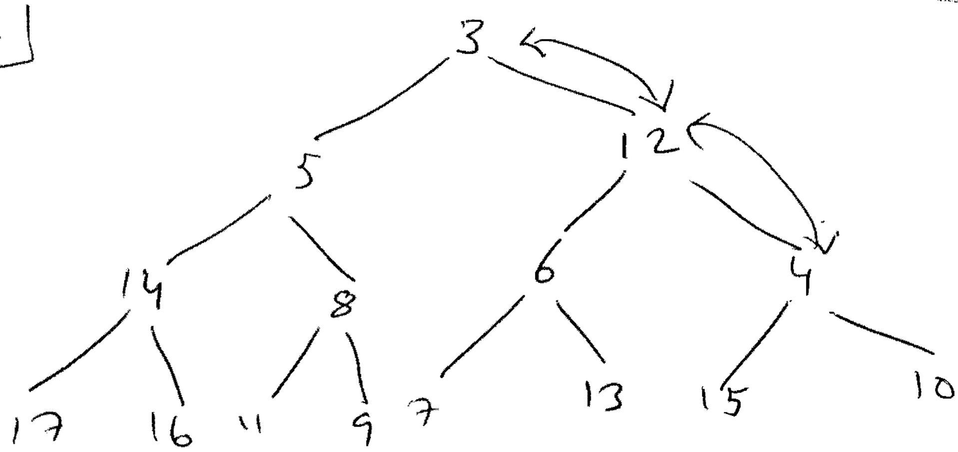


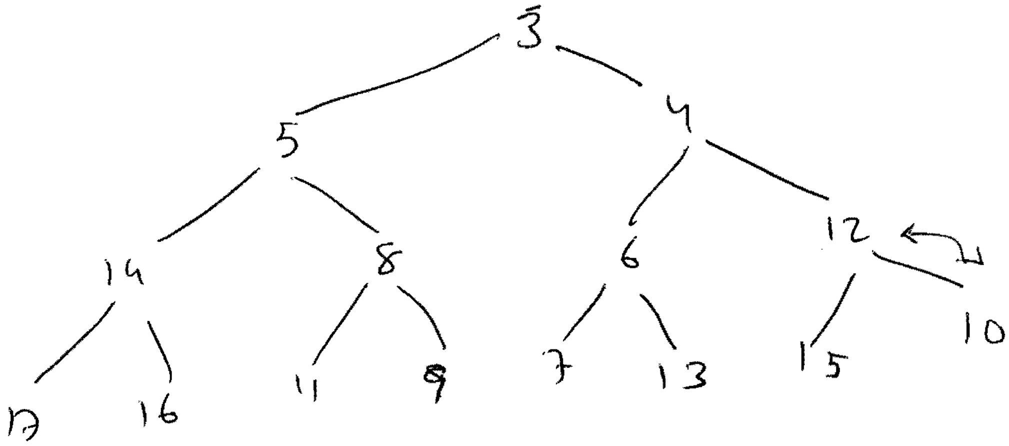
index 3 ok

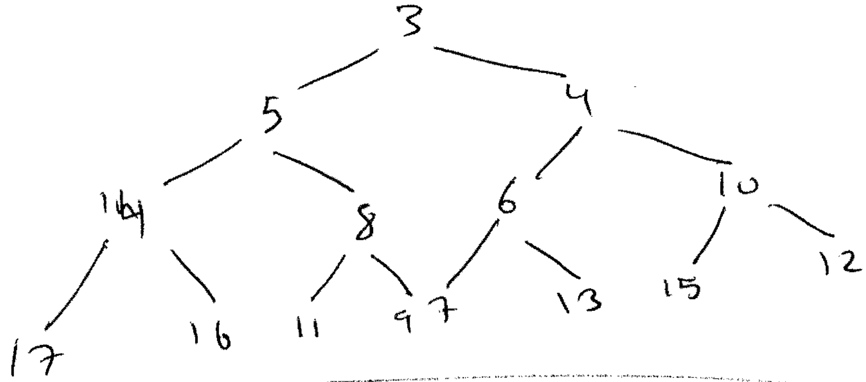
index 2



index 1







as an array:

	3	5	4	14	8	6	10	17	16	11	9	7	13	15	12
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

(6) delete first time:

	~~3~~	5	4	14	8	6	10	17	16	11	9	7	13	15	12
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

	12	5	4	14	8	6	10	17	16	11	9	7	13	15	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	

Siftdown

Index = 1 Left child in = 2 Right child in = 3

swap 12 with 4.

	4	5	12	14	8	6	10	17	16	11	9	7	13	15
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Index = 3 Left child = 6 Right child = 7

swap 12 with 6.

	4	5	6	14	8	12	10	17	16	11	9	7	13	15
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Index = 6 Left child = 12 Right child = 13

swap 12 with ~~3~~ 7

	4	5	6	14	8	7	10	17	16	11	9	12	13	15
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

delete second

	15 ~~4~~	5	6	14	8	7	10	17	16	11	9	12	13	~~15~~
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

Index = 1 Left child = 2 Right child = 3

Swap 15 with 5.

	5	15	6	14	8	7	10	17	16	11	9	12	13
	1	2	3	4	5	6	7	8	9	10	11	12	13

Index 2 Left child = 4 Right child = 5

Swap 15 with 8.

	5	8	6	14	15	7	10	17	16	11	9	12	13
	1	2	3	4	5	6	7	8	9	10	11	12	13

Index = 5 Left child = 10 Right child 11

Swap 15 with 9,

	5	8	6	14	9	7	10	17	16	11	15	12	13
	1	2	3	4	5	6	7	8	9	10	11	12	13