King Fahd University of Petroleum and Minerals
College of Computer Science and Engineering
Information and Computer Science Department

ICS 353: Design and Analysis of Algorithms

Second semester 2015-2016
Major Exam #1, Friday, October 9, 2015.

Name:

ID#:

Instructions:

1. The exam consists of 7 pages, including this page, containing 6 questions. You have to answer all 6 questions.
2. The exam is closed book and closed notes. No calculators or any helping aides are allowed. Make sure you turn off your mobile phone and keep it in your pocket if you have one.
3. The maximum number of points for this exam is **100**.
4. You have exactly **90** minutes to finish the exam.
5. Make sure your answers are **readable**.
6. If there is no space on the front of the page, feel free to use the back of the page. Make sure you indicate this in order for me not to miss grading it.

Question Number	Learning outcome	Maximum # of Points	Earned Points
1	2, 1,1	20	
2	1	10	
3	1	15	
4	2	20	
5	2	20	
6	1	15	
Total		**100**	

***. Some Useful Formulas:**

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2} \qquad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \qquad \sum_{i=1}^{n} i^3 = \left(\frac{n(n+1)}{2}\right)^2$$

$$\sum_{i=0}^{n} x^i = \frac{x^{n+1}-1}{x-1} \qquad \sum_{i=1}^{n} i.c^i = \Theta(1) \text{ for } 0 < c < 1 \qquad 2^{\lg n} = n,$$

$$log_b a = \frac{log_c a}{log_c b} \text{ where } c, b \neq 1 \qquad \log a^b = b \log a \qquad \log ab = \log a + \log b$$

Q1. **(20 points; 14, 3, 3 points)** Consider the following array:

[11, 12, 1, 5, 15, 3, 4, 10, 7, 2]

a. Illustrate the operation of Algorithm BUBBLESORT on the array?
b. How many element comparisons are performed by the algorithm on the above array?
c. What is the space complexity of Algorithm BUBBLESORT? Briefly justify your answer?

Q1

11 12 1 5 15 3 4 10 7 2
11 1 5 12 3 4 10 7 2 15 9
1 5 11 3 4 10 7 2 12 15 8
1 5 3 4 10 7 2 11 12 15 7
1 3 4 5 7 2 10 11 12 15 6
1 3 4 5 2 7 10 11 12 15 5
1 3 4 2 5 4
1 3 2 4 3
1 2 3 4 2
1

Q1. **(20 points; 14, 3, 3 points)** Consider the following array:

[11, 12, 1, 5, 15, 3, 4, 10, 7, 2]

a. Illustrate the operation of Algorithm BUBBLESORT on the array?
b. How many element comparisons are performed by the algorithm on the above array?
c. What is the space complexity of Algorithm BUBBLESORT? Briefly justify your answer?

a.

11		1		1		1		1		1
12		11		2		2		2		2
1		12		11		3		3		3
5		2		12		11		4		4
15		5		3		12		11		5
3	$\rightarrow$	15	$\xrightarrow{9}$	5	$\xrightarrow{8}$	4	$\xrightarrow{7}$	12	$\xrightarrow{6}$	11
4		3		15		5		5		12
10		4		4		15		7		7
7		10		7		7		15		10
2		7		10		10		10		15

	1		1		1		
	2		2		2		
	3		3		3		
	4		4		4		
	5		5		5		
	7		7		7		
$\xrightarrow{5}$	11	$\xrightarrow{4}$	10	$\xrightarrow{3}$	10	$\xrightarrow{2}$	Only eight iterations
	12		11		11		
	10		12		12		
	15		15		15		

b. $\sum_{i=2}^{9} i = \frac{9(10)}{2} - 1 = 44.$

c. $\Theta(1)$.

The additional storage required consists of a temporary element to carry out the interchange of elts. (in addition to the iterators). These are independent of input size & hence the constant storage requirement.

Q2. **(10 points)** Express the following function

$$f(n) = 4^{n^2} + 2^{n^2}$$

in terms of the Θ-notation in the simplest form. Make sure you prove your answer.

Using the limits to find the maximum term,

$$\lim_{n \to \infty} \frac{2^{n^2}}{4^{n^2}} = \lim_{n \to \infty} \left(\frac{2}{4}\right)^{n^2} = 0$$

Hence, $f(n) = \Theta(4^{n^2})$.

Q3. **(15 points; 8, 5, 2 points)** For the following code segments:

```
Algorithm SomeCount
count = 0;
for (k=1; k<=n; k=k*2)
  for (j=1; j<=k; j=j*2)
     count++;
```

a. Formulate the cost of running the above code in summation form (note that it will be equal to the value of `count`). You may assume that *n* is a power of 2.
b. Evaluate the summation in part "a".
c. Express the cost of this code segment in terms of Θ() notation.

a. Assume that $n = 2^a$ where $a \in \mathbb{Z}^+$ ($a = \lg n$)

r: $\quad 0, 1, 2, \ldots, a$

k: $1, 2^1, 2^2, \ldots, 2^a = n$

Let $k = 2^r$ ($r = \lg k$)

Then $r: 0, 1, \ldots, a$.

s: $\quad 0, 1, \ldots, l$

j: $1, 2^1, \ldots, k = 2^l$ where $l = \lg k$.

Similarly, let $j = 2^s$, i.e. $s = \lg j$.

Then count $= \sum_{r=0}^{a} \sum_{s=0}^{\lg k} 1$

b. count $= \sum_{r=0}^{a} (\lg k + 1) = \sum_{r=0}^{a} (r+1) = \sum_{r=1}^{a+1} r = \frac{(a+1)(a+2)}{2}$

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

c. The cost is $\Theta(\lg^2 n)$.

Q4. **(20 points; 10 points, each)** Consider the twelve - singleton disjoint sets:

{1} {2} {3} {4} {5} {6} {7} {8} {9} {10}

The following describes an execution of Union and Find operations on the above set. Assume each element is initially assigned a rank of zero. Apply the following union and find operations as stated in parts (a) and (b).

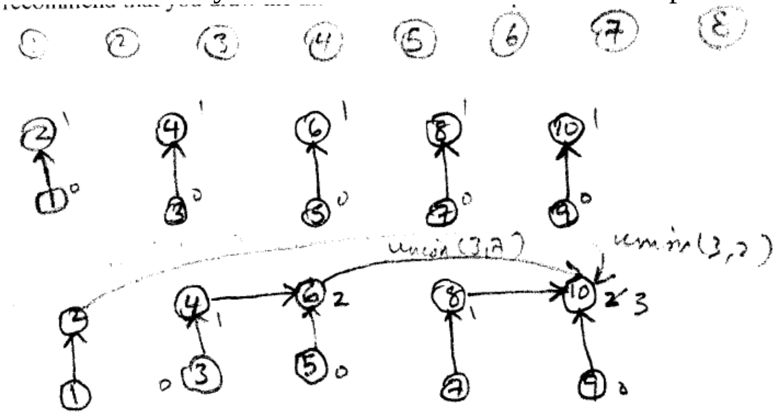
Union (1, 2), Union (3, 4), Union (5, 6), Union (7, 8), Union (9, 10),
Union (3, 5), Union (7, 9), Union (3, 7), Union (3, 2), Find (5).

(a) Give the tree (parent-pointer data structure) from executing the above steps using union by rank with no path compression. Be sure to label the nodes in the final tree including their final ranks.

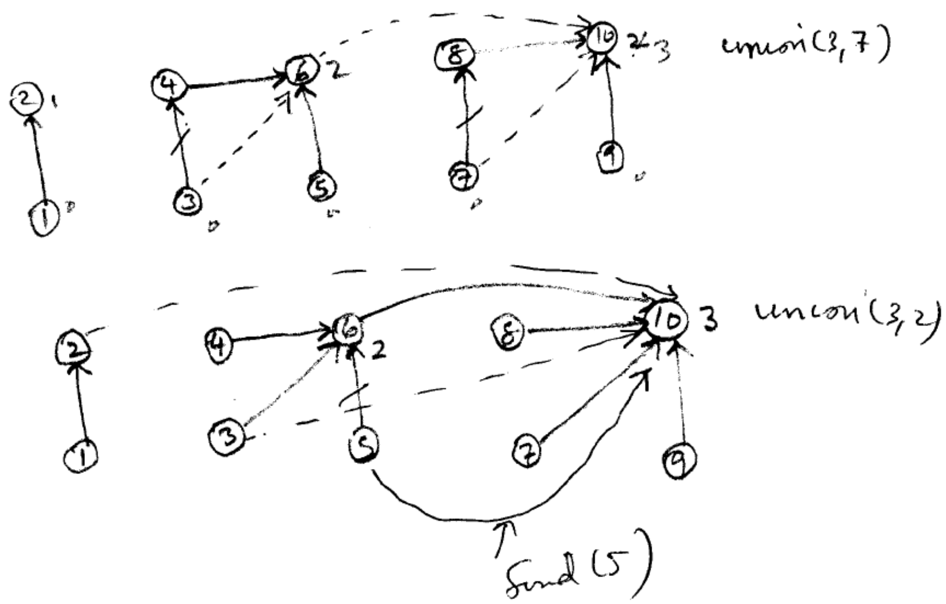
(b) Give the tree (parent-pointer data structure) from executing the above steps using union by rank with path compression. Be sure to label the nodes in the final tree including their final ranks.

We recommend that you draw the intermediate trees for partial credit.

(a)



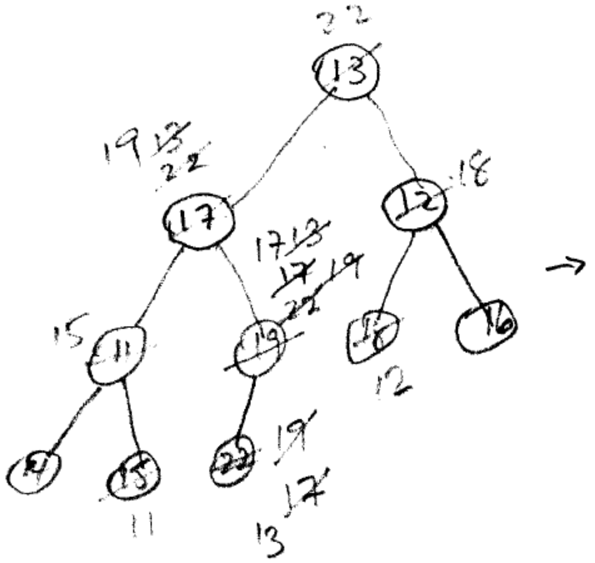
(b)

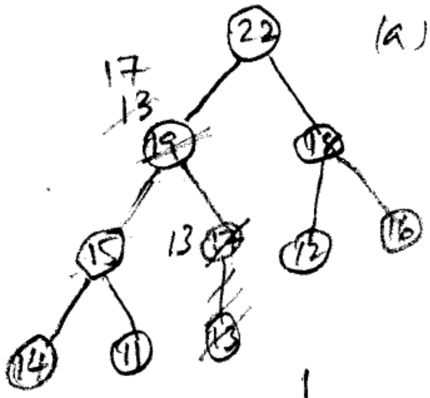


Q5. **(20 points; 10, 5, 5 points)** Consider the following array:

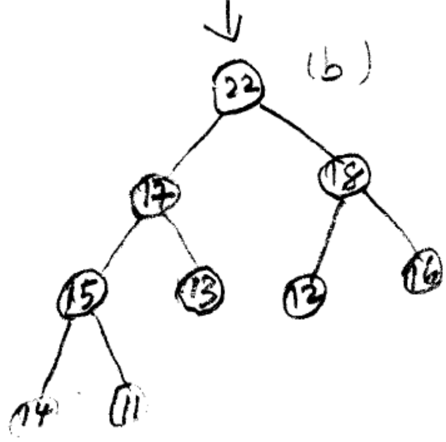
13, 17, 12, 11, 19, 18, 16, 14, 15, 22

a. Consider the max-heap data structure. Illustrate the operation of Algorithm MAKEHEAP on the following array to build a max-heap. (Show the intermediate steps).
b. Show the max-heap after deleting the element with key value 19 from the heap H. (Show the intermediate steps).
c. Show the max-heap after adding an element of value 25 to the heap H of (a). (Show the intermediate steps).



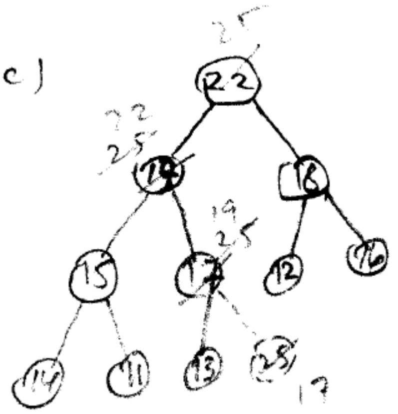


(a)



(b)

(c)



Q6. **(15 points)** Consider the linear search problem. Assume that a key x is to be found in an array A of size n, where the probability that the key is found in the i^{th} position in A is $\left(\frac{1}{3}\right)^i$. Note that it is possible that the element is not found in the array. Analyze the average case complexity of the above algorithm, finding it in terms of $\Theta()$ notation.

Probability of the element not in $A = 1 - \sum_{i=1}^{n} \left(\frac{1}{3}\right)^i$

$$= 1 - \left(\frac{\left(\frac{1}{3}\right)^{n+1} - 1}{\frac{1}{3} - 1} - 1 \right)$$

$$= 2 + \frac{3}{2}\left(\left(\frac{1}{3}\right)^{n+1} - 1\right) = 0.5\left(1 + \left(\frac{1}{3}\right)^n\right)$$

$\therefore$ Average time complexity $= \sum_{i=1}^{n} i \left(\frac{1}{3}\right)^i + n\left(0.5\left(1 + \left(\frac{1}{3}\right)^n\right)\right)$

$$= c + \frac{n}{2} + \cancel{\frac{n}{2 \cdot 3^n}} \; 0 \quad \left(\text{since } \sum_{i=1}^{n} i \left(\frac{1}{3}\right)^i = \Theta(1)\right)$$

$$= \Theta(n).$$