UNIVERSITY OF SOUTHAMPTON COMP1201W1

SEMESTER 2 EXAMINATION 2015/2016

ALGORITHMICS

Duration: 120 mins

You must enter your Student ID and your ISS login ID (as a cross-check) on this page. You must not write your name anywhere on the paper.

Student ID:	
ISS ID:	

Question	Marks
1	
2	
3	
4	
Total	

Answer all parts of the question in section A (30 marks) and TWO *questions from section B (35 marks each).*

The examination is worth 85% of the course. The tutorials are worth 15%.

University approved calculators MAY be used.

A foreign language translation dictionary (paper version) is permitted provided it contains no notes, additions or annotations.

Each answer must be completely contained within the box under the corresponding question. No credit will be given for answers presented elsewhere.

You are advised to write using a soft pencil so that you may readily correct mistakes with an eraser.

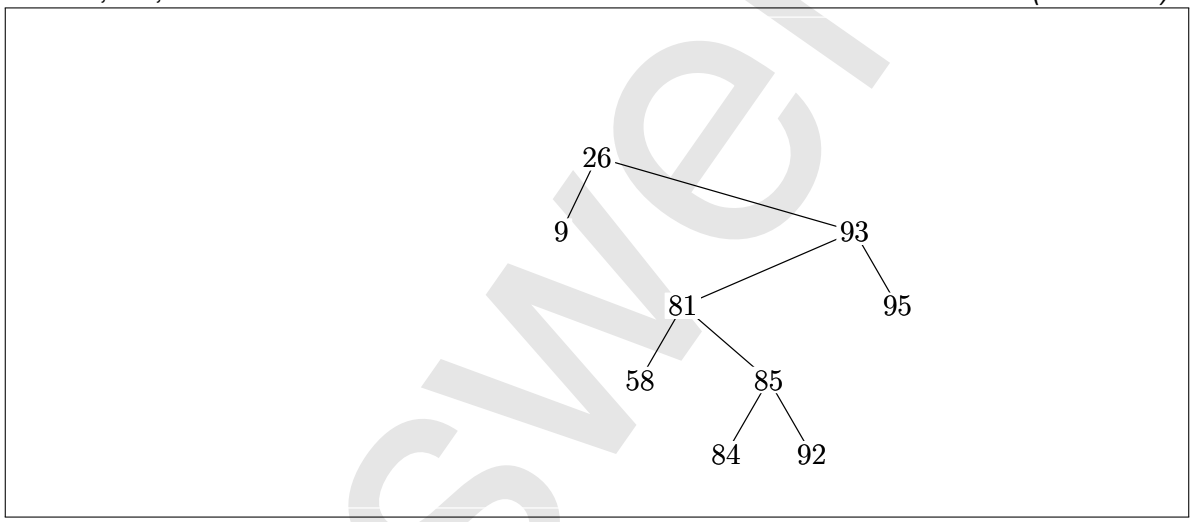
You may use a blue book for scratch—it will be discarded without being looked at.

 Number of pages 18

Section A

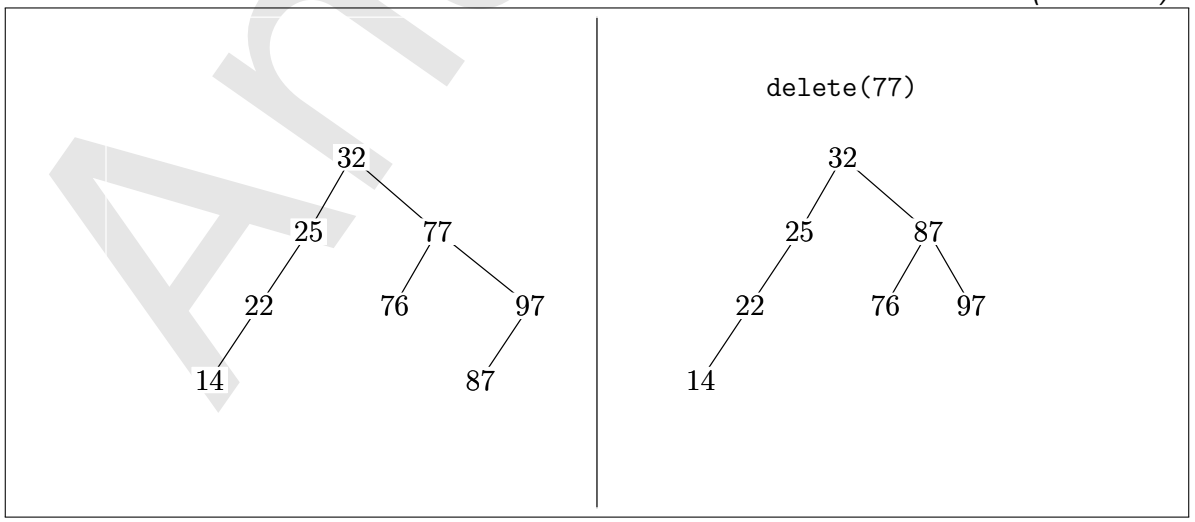
Question A 1

(a) Draw the binary search tree produced when inserting 26, 93, 81, 85, 58, 92, 9, 84, 95. *(2 marks)*



$\frac{1}{2}$

(b) Draw the tree obtained by deleting 77 from the binary search tree shown. *(2 marks)*



$\frac{1}{2}$

(c) What is the worst case time complexity of an unbalanced binary search tree and what is the worst case time complexity for a red-black tree? *(2 marks)*

---

unbalanced binary search tree $\Theta(n)$

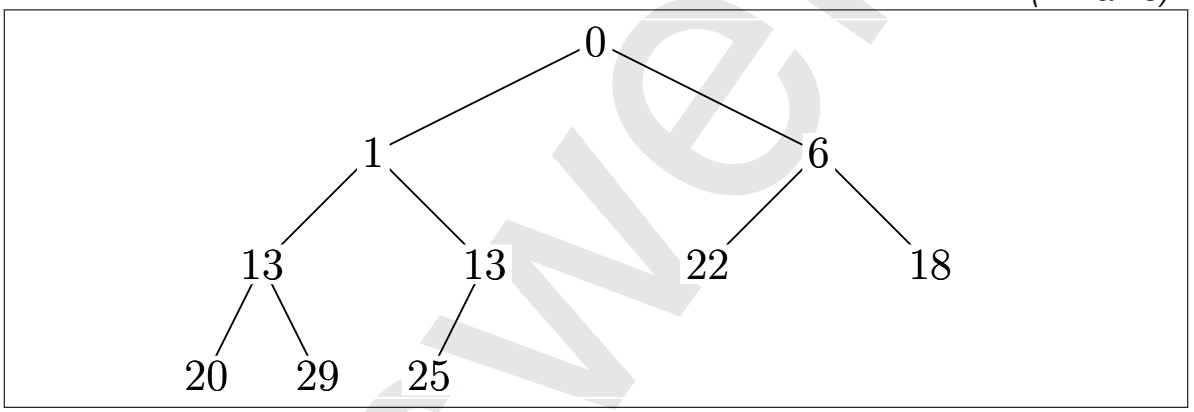
red-black tree $\Theta(\log(n))$

$\frac{1}{2}$

(d) Heaps use a binary tree encoded into an array. Show the binary tree represented by the following array.

0	1	6	13	13	22	18	20	29	25

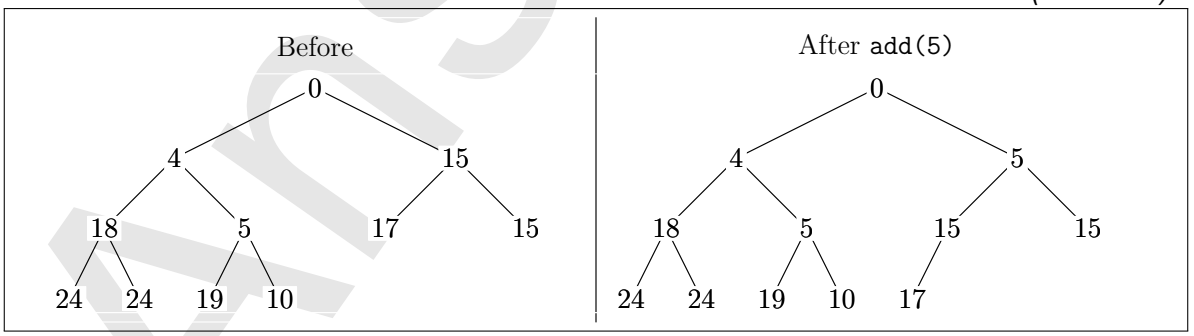
(2 marks)



2

(e) Show what happens to the heap shown on the left when you add 5.

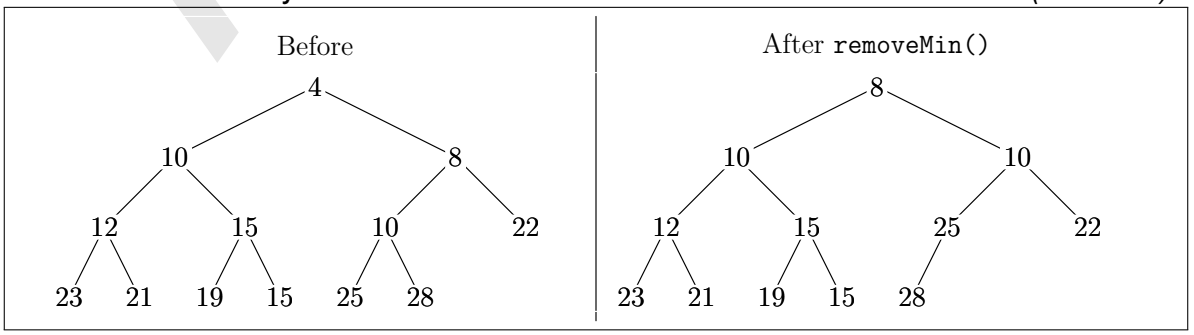
(2 marks)



2

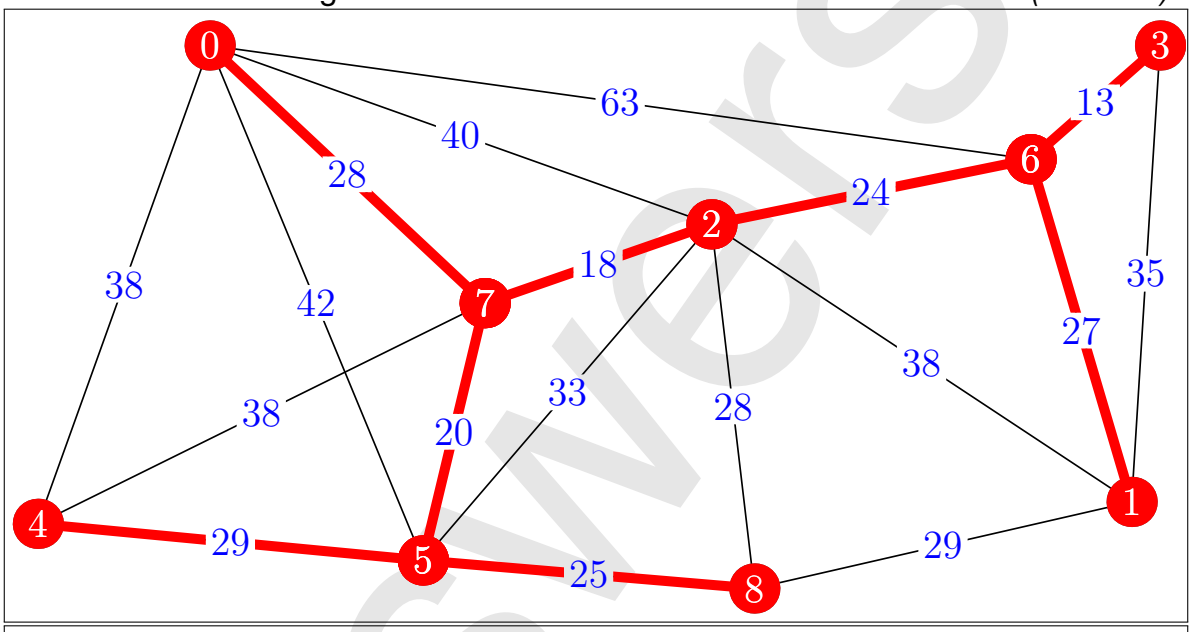
(f) Show what happens to the heap shown on the left when you remove the minimum entry.

(3 marks)



3

(g) Highlight the edges of the minimum spanning tree found by Prim's algorithm for the graph below and write down the edges (i.e. $v_1 - v_2$) in the order they are found starting from node 0. *(5 marks)*



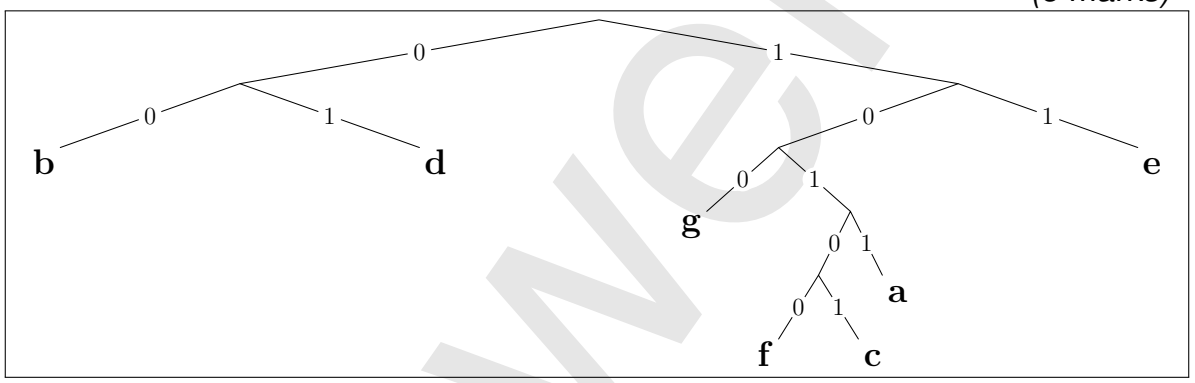
1. 0–7	**2.** 7–2
3. 7–5	**4.** 2–6
5. 6–3	**6.** 5–8
7. 6–1	**8.** 5–4

5

(h) Compute a Huffman tree for the following alphabet.

Letter	a	b	c	d	e	f	g
Frequency	5	9	2	12	19	1	7

(5 marks)



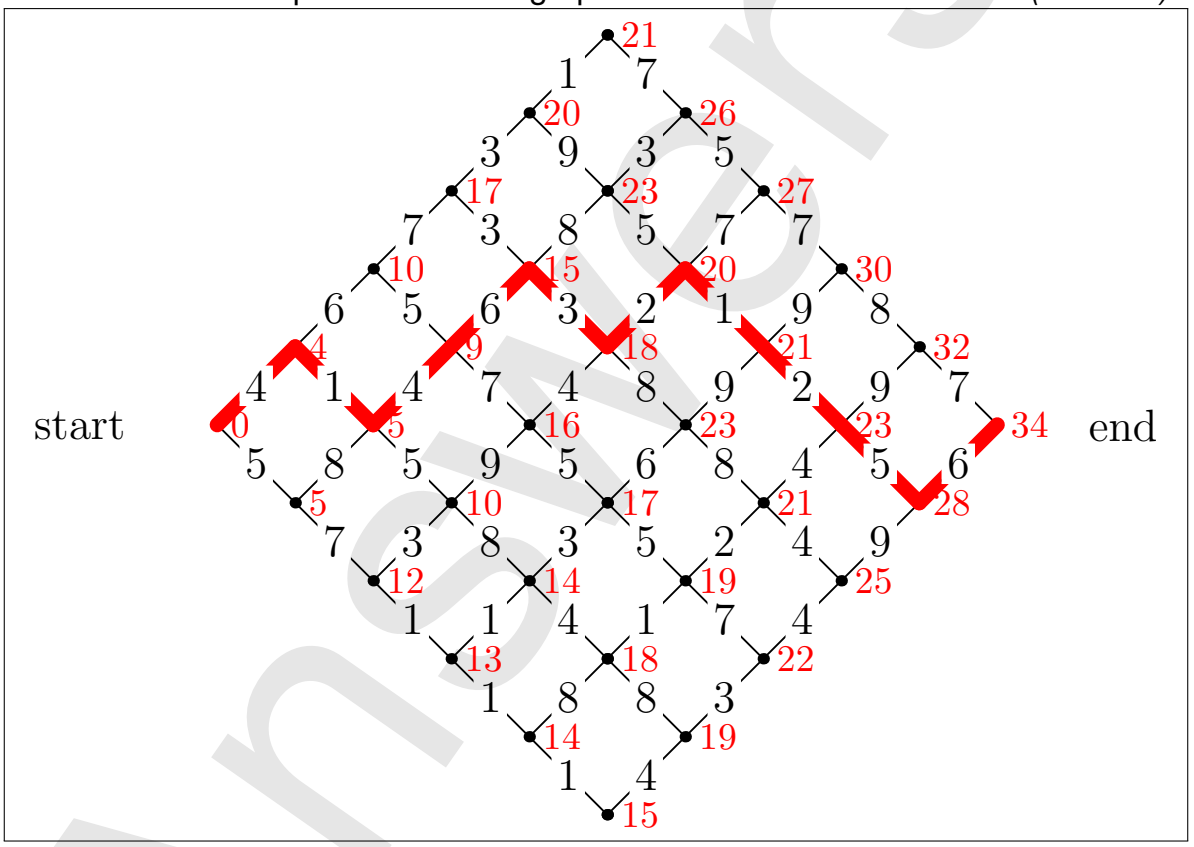
5

(i) How would the word “gaffe” be coded in your Huffman tree (show the letter break with a hyphen). *(2 marks)*

100-1011-10100-10100-11

2

(j) Use the dynamic programming forward algorithm to compute the minimum cost of each path from the left most node to each other node, where the cost of moving along an edge is equal to the number shown. An edge can only be traversed from left to right. Use the backwards algorithm to find the minimum cost path across the graph. *(5 marks)*



5

End of question 1

Q1: (a) 2	(b) 2	(c) 2	(d) 2	(e) 2	(f) 3	(g) 5	(h) 5	(i) 2	(j) 5	Total 30

Section B

Question B 2 Merge sort has the form

```
MERGESORT(a[1:n]) {
  if (n>1) {
    b ←a[1:n/2]
    c ←a[n/2+1:n]
    MERGESORT(b)
    MERGESORT(c)
    MERGE(b,c,a)
    }
  }
```

The number of comparison operations to merge two arrays of length $n/2$ is in the worst case $n-1$.

(a) Let $T(n)$ be the worst case number of comparison operations used by MergeSort to sort an array of size n. Write down a recurrence relation for $T(n)$ valid if $n = 2^m$ *(5 marks)*

$$T(n) = 2\,T(n/2) + n - 1$$

5

(b) Write down the boundary condition $T(1)$ and use the recurrence relation to compute $T(2)$, $T(4)$, and $T(8)$ *(4 marks)*

$$T(1) = 0$$

$$T(2) = 2 \times 0 - 2 + 1 = 1$$

$$T(4) = 2 \times 1 - 4 + 1 = 5$$

$$T(8) = 2 \times 5 - 8 + 1 = 17$$

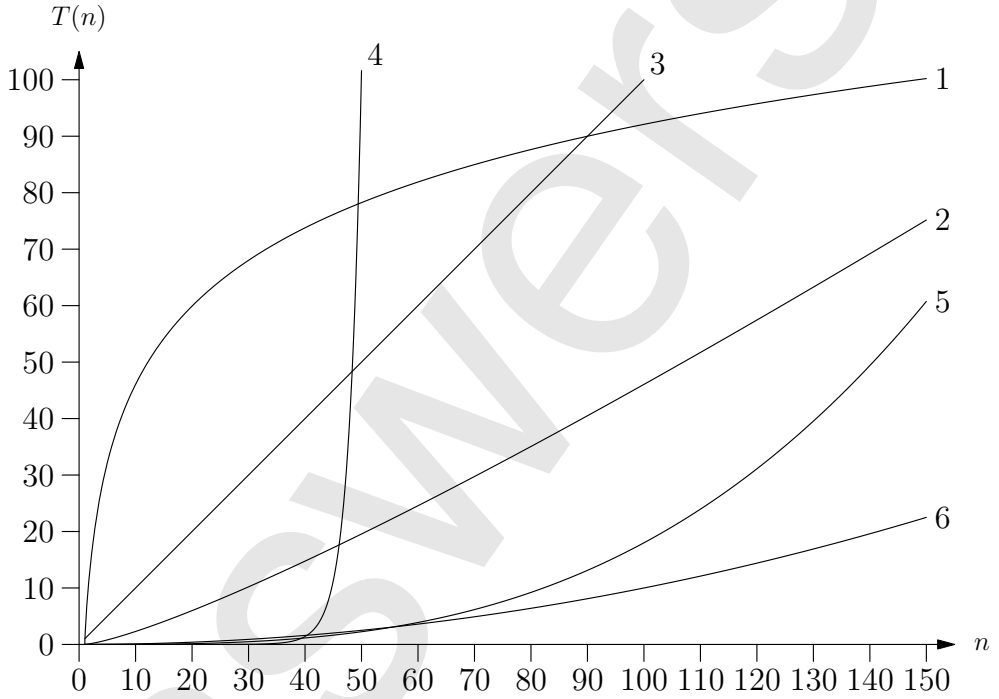
4

(c) Demonstrate, for $n = 2^m$, that $f(n) = n\log_2(n) - n + 1$ satisfies the recurrence relation in part (a). *(9 marks)*

$$
\begin{aligned}
T(n) &= 2\,T(n/2) + n - 1 \\
T(n) &= 2\,f(n/2) + n - 1 \\
&= 2\,\frac{n}{2}\log_2\left(\frac{n}{2}\right) - 2n + 2 + n - 1 \\
&= n\left(\log_2(n) - \log_2(2)\right) - n + 1 \\
&= n\left(\log_2(n) - 1\right) - n + 1 = n\,\log_2(n) + n - 1 = f(n)
\end{aligned}
$$

9

(d) The graph below shows the time complexity for the following algorithms (a) $\Theta((n/a)!)$, (b) $\Theta(n^2)$, (c) $\Theta(n\ \log(n))$, (d) $\Theta(n)$, (e) $\Theta(n^3)$, and (f) $\Theta(\log(n))$. Match the time complexity classes with the curves on the graph.



(6 marks)

1.	(f) $\Theta(\log(n))$	**2.**	(c) $\Theta(n\ \log(n))$
3.	(d) $\Theta(n)$	**4.**	(a) $\Theta((n/a)!)$
5.	(e) $\Theta(n^3)$	**6.**	(b) $\Theta(n^2)$

(e) Which of the following statements are true? Give reasons why (marks will only be awarded if correct reasons are given). *(6 marks)*

(i) All $\Theta(n^2)$ algorithms are faster than all $\Theta(n^3)$ algorithms

False, this is only true when n is large

(ii) An $O(n)$ algorithm will run faster than a $\Omega(n\ \log(n))$ algorithm for sufficiently large n

True, the worst case $O(n)$ algorithm runs in n which is faster than the best case $\Omega(n\ \log(n))$ algorithm which runs in $n\ \log(n)$ (i.e. $\log(n)$ times longer).

$\overline{6}$

(iii) All $O(n^3)$ algorithms run slower than all $O(n^2)$ asymptotically

---

False, $O(n^3)$ is an upper bound on the time complexity and, for example, includes algorithms of $\Theta(n)$ which can be faster than some $O(n^2)$ algorithms

---

(f) Why is it widely believed that $NP \neq P$? *(5 marks)*

There is a large class of problems, the NP-complete problems with the property that if any could be solved in polynomial time then all problems in NP can be solved in polynomial time, thus NP would equal P. But, so far no one has found a polynomial algorithm for a single NP-complete problem.

End of question 2

Q2:	(a) $\frac{}{5}$	(b) $\frac{}{4}$	(c) $\frac{}{9}$	(d) $\frac{}{6}$	(e) $\frac{}{6}$	(f) $\frac{}{5}$	Total $\frac{}{35}$

5

Question B 3

(a) What is a trie? *(3 marks)*

A trie or digital tree is a multiway tree that works on strings composed from some alphabet. The split is done on each string element based on the alphabet.

3

(b) What is its disadvantage? *(2 marks)*

The disadvantage of a trie is that uses very large amounts of memory since most of the table is empty.

2

(c) Show how the following words would be inserted into a trie based on tables *(10 marks)*

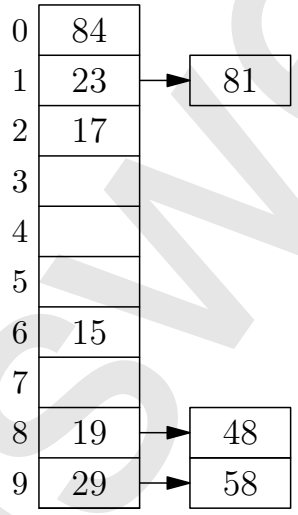
Add words: {ADD, THESE, WORDS, INTO, A, TRIE, IN, THEIR, RIGHT, PLACES}

	0	1	2	3	4
$		A$		IN$	
A	A* (1)				
B					
C					
D		ADD$			
E					
H			THE* (4)		
I	IN* (3)				THEIR$
O					
P	PLACES$				
R	RIGHT$		TRIE$		
S					THESE$
T	T* (2)			INTO$	
W	WORDS$				

10

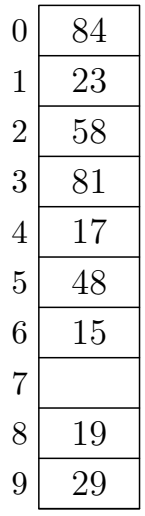
(d) Show how the numbers 23, 29, 84, 15, 58, 19, 81, 17, 48 would be hashed using a hash function $d_2 + 3d_1$ where d_1 is the first (least significant) digit and d_2 the second digit. Show how these would be stored in a hash table of size 10 using separate chaining. *(10 marks)*

d_2d_1	23	29	84	15	58	19	81	17	48
$d_2 + 3d_1$	11	29	20	16	29	28	11	22	28
$(d_2 + 3d_1)\%10$	1	9	0	6	9	8	1	2	8



10

(e) Show how the numbers 23, 29, 84, 15, 58, 19, 81, 17, 48 would be stored in a hash table using linear probing assuming the same hash codes. *(6 marks)*



6

(f) What is the disadvantage of linear probing and how can open addressing be modified to overcome this disadvantage? *(4 marks)*

(i) Linear probing is susceptible to developing primary clusters where large contiguous regions of the hash table are used.

(ii) Either quadratic probing or double hashing is used so if there is no space a larger jump is made.

(I would also accept a discussion of the problem of deleting elements.)

End of question 3

$\frac{}{4}$

Q3: (a) $\frac{}{3}$ (b) $\frac{}{2}$ (c) $\frac{}{10}$ (d) $\frac{}{10}$ (e) $\frac{}{6}$ (f) $\frac{}{4}$ Total $\frac{}{35}$

Question B 4

(a) We can implement a fast set for a fixed number of integers using two arrays. Below we show the representation of the set $\{2, 7, 3, 1\}$.

	0	1	2	3	4	5	6	7	8	9
indexArray	-1	3	0	2	-1	-1	-1	1	-1	-1
memberArray	2	7	3	1						

Show the state of the arrays when we add 8 to the set. *(5 marks)*

	0	1	2	3	4	5	6	7	8	9
indexArray	-1	3	0	2	-1	-1	-1	1	4	-1
memberArray	2	7	3	1	8					

5

(b) Show the state of the arrays when you remove 3 from the original set shown in question 4a. *(5 marks)*

	0	1	2	3	4	5	6	7	8	9
indexArray	-1	2	0	-1	-1	-1	-1	1	-1	-1
memberArray	2	7	1							

5

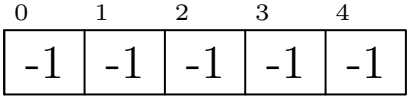
The disjoint set class is described by the following program

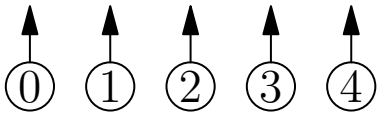
```
public class DisjSets
{
    private int[] s;

    public DisjSets(int numElements) {
        s = new int[numElements];
        for(int i=0; i<s.length; i++)
            s[i] = -1;
    }

    public void union(int root1, int root2) {
        if (s[root2]<s[root1]) {
            s[root1] = root2;
        } else {
            if (s[root1]==s[root2])
                s[root1]--;
            s[root2] = root1;
        }
    }

    public int find(int x) {
        if (s[x]<0)
            return x;
        else
            return s[x] = find(s[x]);
    }
}
```

We assume that we have created an instance of the disjoint sets class

```
DisjSets disjset = new DisjSets(5);
```

Below we show the initial settings of the array `s` and a graphical representation of the forest (set of trees) representing the array.





• Do not write in this space •

(c) Show the state of array and the forest after performing the following operation

```
disjset.union(disjset.find(2), disjset.find(3));
```

(5 marks)

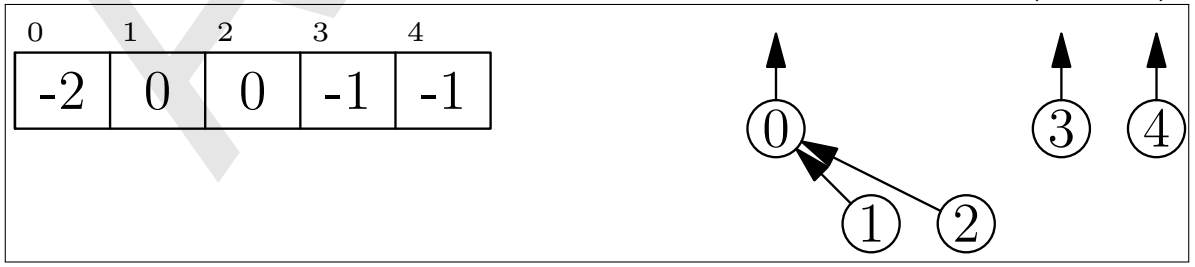
5

(d) Given an array shown below

Show the state of array and the forest after performing the following operation

```
disjset.union(disjset.find(1), disjset.find(2));
```

(5 marks)

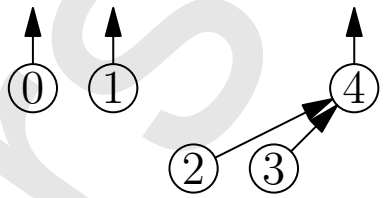


5

• Do not write in this space •

(e) Given an array shown below

0	1	2	3	4
-1	-1	4	4	-2

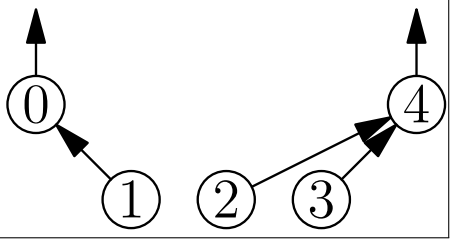


Show the state of array and the forest after performing the following operation

```
disjset.union(disjset.find(0), disjset.find(1));
```

(5 marks)

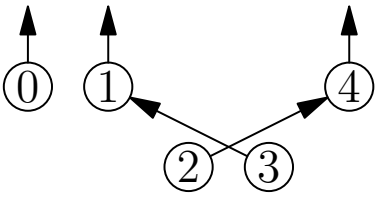
0	1	2	3	4
-2	0	4	4	-2



5

(f) Given an array shown below

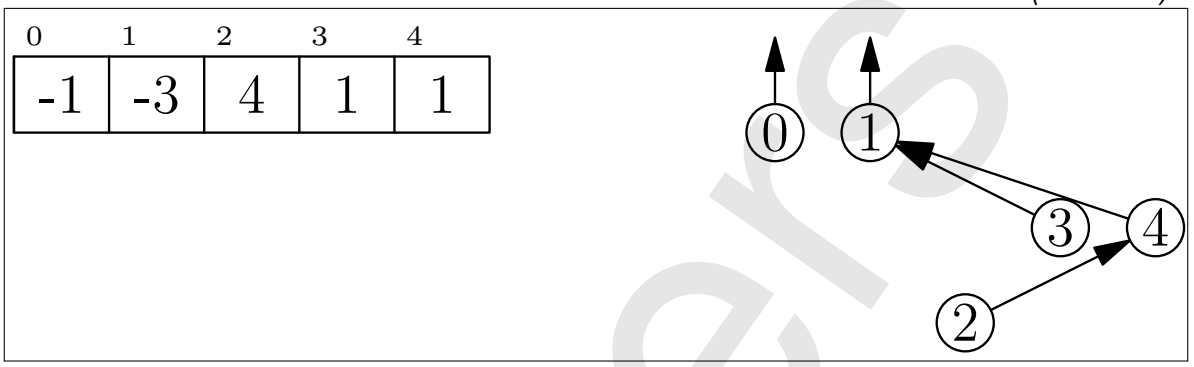
0	1	2	3	4
-1	-2	4	1	-2



Show the state of array and the forest after performing the following operation

```
disjset.union(disjset.find(1), disjset.find(2));
```

(5 marks)



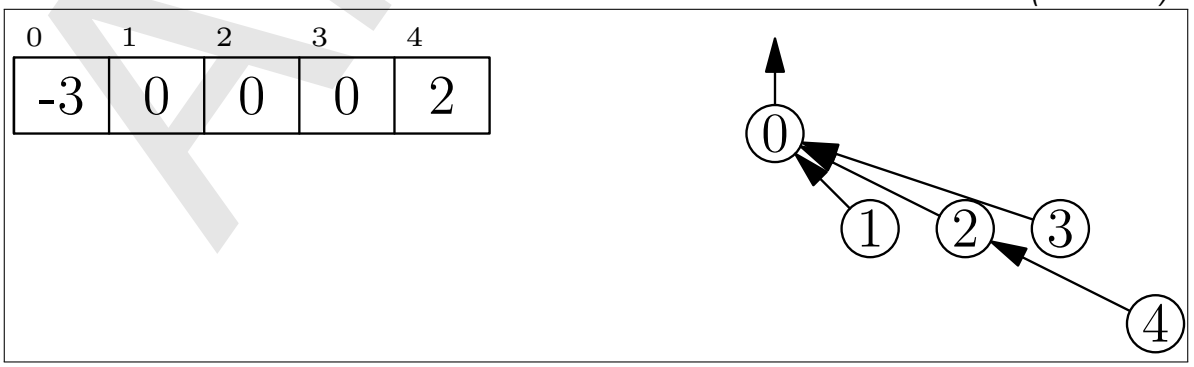
5

(g) Given an array shown below

Show the state of array and the forest after performing the following operation

```
disjset.find(3);
```

(5 marks)



5

End of question 4

Q4: (a) $\frac{}{5}$ (b) $\frac{}{5}$ (c) $\frac{}{5}$ (d) $\frac{}{5}$ (e) $\frac{}{5}$ (f) $\frac{}{5}$ (g) $\frac{}{5}$ Total $\frac{}{35}$

END OF PAPER