Student number:	2467273
Course title:	COMPSCI4009 Algorithmics I
Questions answered:	ALL

1.

1. a) $$d_{i,j} = \begin{cases} d_{i-1,j-1} & \text{if } a[i] = b[j] \\ \min(1 + d_{i,j-1},\ 2 + d_{i-1,j},\ 3 + d_{i-1,j-1}) & \text{otherwise} \end{cases}$$

b)

a\b		0	1	2	3	4	5	6
			s	u	n	d	a	y
0		0	1	2	3	4	5	6
1	s	1	0	1	2	3	4	5
2	a	2	2	3	4	5	3	4
3	t	3	4	5	6	7	5	6
4	u	4	5	4	5	6	7	8
5	r	5	6	6	7	8	9	10
6	d	6	7	8	9	7		
7	a	7					7	
8	y	8						7

→ +1
↓ +2
↘ +3

m - match
s - substitution
d - deletion

c) d d d d d v v d

m m m s m d d m (backwards)

✓ × × ✓ ✓ ✓ ✓ ✓

a = s a t u r d a y
b = s - - u n d a y

2.

a)

2. a)

q	d(q) +v	+z	+w	+y	+x	pred(q) +v	+z	+w	+y	+x
u	0 →	0 →	0 →	0 →	0	u →	u →	u →	u →	u
v	5 →	5 →	5 →	5 →	5	u →	u →	u →	u →	u
z	8 →	7 →	7 →	7 →	7	u →	v →	v →	v →	v
w	10 →	8 →	8 →	8 →	8	u →	v →	v →	v →	v
y	∞ →	14 →	13 →	13 →	13	u →	v →	z →	z →	z
x	∞ →	∞ →	18 →	14 →	14	u →	v →	z →	w →	w

b)

Both the heap and the array take O(n) time to build for all vertices.

In the worst case, the graph will be complete; thus, m will be much bigger than n.

Advantage: can access minimum distance in O(1) time.

Disadvantage: heap needs to be *heapified* every time a new vertex is added to the set of found vertices, which takes O(log(n)) time (standard binary search tree time).

Ambiguous: array takes O(n^2) time because, for each vertex, it goes through all other vertices to check for minimal distance; however, minimum heap needs to search vertices for each edge, making it O(m * log(n)), where $m = \frac{v(v-1)}{2} \approx v^2$

3.

a)

1. False because it shows that P3 is at least as hard as P1, but might map problems of P1 to easy problems of P3.

2. True because, to solve an instance of P3, it’s possible to reduce it to an instance of P1.

3. True. If P2 is NP-complete, then we can reduce any NP problem to P2 in polynomial time. By the hypothesis, we can reduce P2 to P3 in polynomial time. Therefore, we can reduce any NP problem to P3 in polynomial time. Since P3 is in NP => P3 is NP-complete.

4. False.

b)

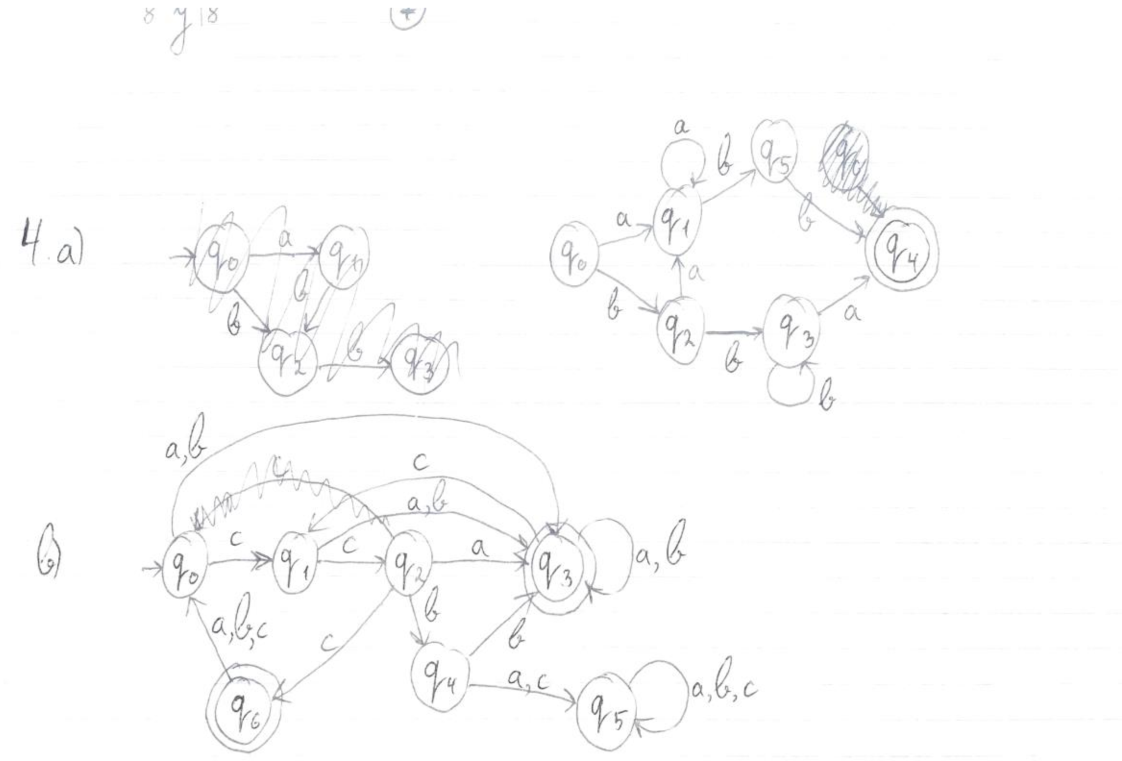
1. True.

2. False.

3. True. Proof by contradiction. If P2 is undecidable and we can perform such a reduction, then P3 is undecidable. Suppose for a contradiction that P3 is decidable. Then, we can use the reduction to decide P2. We can take an instance of P2, and convert it into an instance of P3. Since P3 is decidable, we can solve this problem. Therefore, the instance of P2 can also be solved since it has the same answer. Since 1 is undecidable, this is a contradiction. So, P3 must be undecidable.

4.

a), b)



c)

(a*|b*)*cc(a|bb|c*)(a*|b*)*