Name: V.N. Anirudh Oruganti
Class: CSCE 350
Assignment: Homework #1: Questions 1-4.

1. Answer the following questions:

• (5 points) Assume you have an empty stack, show the stack after a sequence of operations: ▪ push(p), push(q), pop, push(r), pop, push(s), push(t), pop

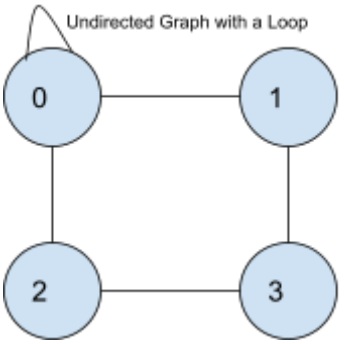
Operation: push(p)	Operation: push(q)	Operation: pop	Operation: push(r)	Operation: pop	Operation: push(s)	Operation: push(t)	Operation: pop
p	p	p	p	p	p	p	p
	q		r		s	s	s
						t	

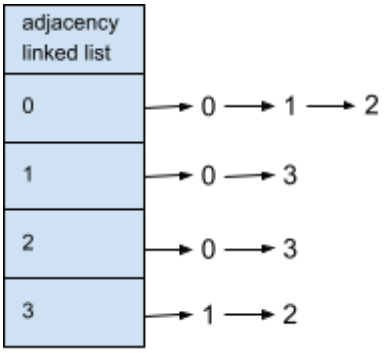
• (5 points) Assume you have an empty queue, show the queue after a sequence of operations: ▪ enqueue(p), enqueue(q), dequeue, enqueue(r), dequeue, enqueue(s), enqueue(t), dequeue

Operation: enqueue(p)	p		
Operation: enqueue(q)	p	q	
Operation: dequeue	q		
Operation: enqueue(r)	q	r	
Operation: dequeue	r		
Operation: enqueue(s)	r	s	
Operation: enqueue(t)	r	s	t
Operation: dequeue	s	t	

• (5 points) Give an example of an undirected graph with a loop. Use an adjacency matrix and an adjacency linked list to represent your graph.

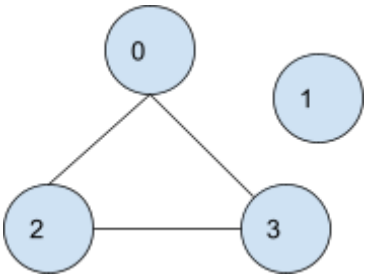
X to Y ***adjacency matrix***	0	1	2	3
0	**1**	**1**	**1**	**0**
1	**1**	**0**	**0**	**1**
2	**1**	**0**	**0**	**1**
3	**0**	**1**	**1**	**0**

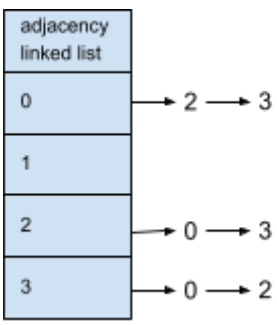




• (5 points) Give an example of an undirected graph with an isolated vertex, i.e., a vertex with no edges connected to it. Use an adjacency matrix and an adjacency linked list to represent your graph.

X to Y ***adjacency matrix***	0	1	2	3
0	**0**	**0**	**1**	**1**
1	**0**	**0**	**0**	**0**
2	**1**	**0**	**0**	**1**
3	**1**	**0**	**1**	**0**





2. (a) (2 X 10 = 20 points) Write the adjacency matrix and the adjacency linked lists for both graphs shown in

• Figure-1A and • Figure-1B

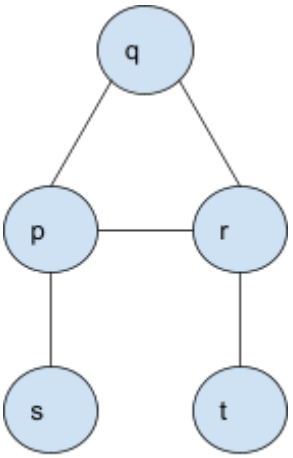


Figure 1A

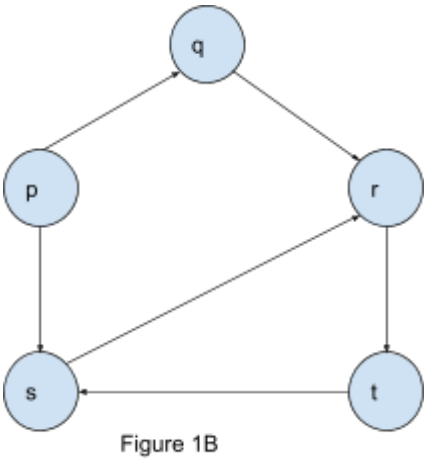
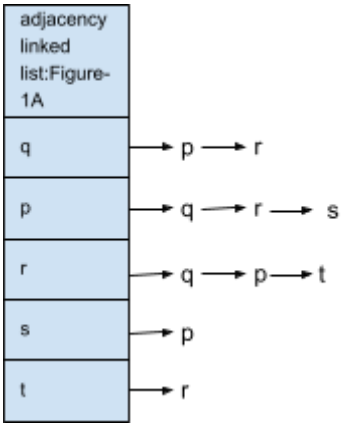
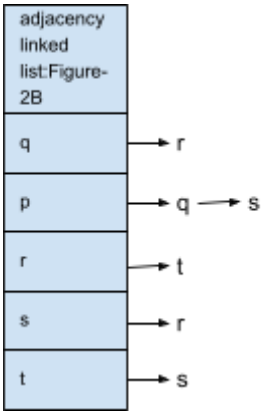


Figure 1B

X to Y ***adjacency matrix:*** Figure-1A	q	p	r	s	t
q	**0**	**1**	**1**	**0**	**0**
p	**1**	**0**	**1**	**1**	**0**
r	**1**	**1**	**0**	**0**	**1**
s	**0**	**1**	**0**	**0**	**0**
t	**0**	**0**	**1**	**0**	**0**



X to Y ***adjacency matrix:*** Figure-2B	q	p	r	s	t
q	**0**	**0**	**1**	**0**	**0**
p	**1**	**0**	**0**	**1**	**0**
r	**0**	**0**	**0**	**0**	**1**
s	**0**	**0**	**1**	**0**	**0**
t	**0**	**0**	**0**	**1**	**0**



3. For each of the graphs (in Figures - 1A and 1B),

• (5 points) Indicate if each graph is complete?

-Figure 1A does not seem complete since every pair of its vertices is not connected by an edge.

-Figure 2B does not seem complete since every pair of its vertices is not connected by an edge.

• (5 points) Identify any loops in each graph? If a loop exists, write down the corresponding edges.

` There are no loops in both Figures - 1A and 1B. Since both graphs(Figures - 1A and 1B) does not contain a edge which connects to vertices itself.

• (5 points) Are there any cycles in each graph? If so, write down the corresponding paths.

Figures - 1A: There is a cycle for Figure -1A in the upper part of the graph where it contains an equilateral triangle with the following vertices q,r,p.

Paths: Q-r-p-q
R-p-q-r
P-q-r-p
Q-p-r-q
P-r-q-p
R-q-p-r

Figure -2B: There is a cycle for Figure -2B in the bottom right part of the graph where it contains a right triangle with the following vertices r,s,t.

Paths:
r->t->s->r
s->r->t->s
t->s->r->t

4. Consider the following algorithm for finding the distance between the two closest elements in an array of numbers.

```
ALGORITHM MinDistance(A[0..n − 1])
    //Input: Array A[0..n − 1] of numbers
    //Output: Minimum distance between two of its elements
    dmin ←∞
        for i ← 0 to n − 1 do
            for j ← 0 to n − 1 do
                if |A[i] − A[j]| < dmin
                dmin ← |A[i] − A[j]|
    return dmin
```

(a) (10 points) What is the basic operation of this algorithm? How many times is it performed as a function of the array size n?

What is the basic operation of this algorithm?

Element Comparison to find minimum distance between two of its elements.

How many times is it performed as a function of the array size n?

$C_{worst}(n) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} 1 = n^2$

(b) (10 points) Make at least 2 improvements to the algorithm, MinDistance. You must write down the pseudo code for your new algorithm. How many times is the basic operation performed as a function of the array size n ? (If you need to, you may change the algorithm altogether; if not, improve the implementation given).

```
Modified ALGORITHM MinDistance(A[0..n − 1])
    //Input: Array A[0..n − 1] of numbers
    //Output: Minimum distance between two of
    its elements
    dmin ← A[0] - A[1]
        for i ← 1 to n − 2 do
            for j ← i+1 to n − 1 do
                if |A[i] − A[j]| < dmin
                dmin ← |A[i] − A[j]|
    return dmin
```

How many times is the basic operation performed as a function of the array size n ?

$C_{worst}(n) = \sum_{i=1}^{n-2} \sum_{j=i}^{n-1} 1 = (n^2)/2 - (3*n/2) + 1$