Name = Owais Afridi

ID = 13686

Q:2:-

```
#Include <iostream>
Using namespace std;
Void main()
{
 int arr1[100], L, K, P, X;
Cout << "\n\k insert new values:\n"
cout << "endl";
cout << "Input the size of array"<<endl;
Scanf("%d", &k);
Printf("Input %d elements in
        ascending order:\n", n);
for(i=0; i<K; i++)
{
Printf("element-%d:", i);
Scanf("%d", &arr1[i]);
}
Printf("Input insertion value:");
Scanf("%d", &x);
Printf("Enter the value position:");
Scanf("%d", &P);
```

```
Printf("Current list of array:\n");
for(i=0;i<K;i++)
Printf("%5d",arr1[i]);
/* Move all data at right side */
for(i=K;i>=P;i--)
arr1[i] = arr1[i-1];
/* Insert value at given position */
arr1[P-1] = x;
Printf("\n\n Now new list is:\n");
for(i=0;i<=K;i++)
Printf("%5d", arr1[i];
Printf("\n\n");
}
```

Question=(3):-

Quick Sort:-

It is a sorting algorithm. It creates two empty arrays to hold elements less then the pivot value and elements greater then the pivot value and then recursively sort the sub arrays.

Quick sort is a divide and conquer algorithm. There are two basic operations in the algorithm, swapping items in a place and partitioning a section of the array.

1). - Always Pick first element as a pivot.

2). - Always Pick last element as a pivot.

3). - Start a pointer at the first or last item in array.

4). While the value at the right pointer in the array is less or greater then the pivot value.
while value is less (add 1)
while value is greater (Subtract 1).

5). If the left pointer is less than or equal to the right pointer then swap the values at these location in the array.

6). Move left pointer to the right by one and the right to the left.

7). - If the left pointer and right pointer don't meet. go to step 1.

Example:-

```
var array=[9,2,5,6,4,3,7,10,1,12,8,11];
function quick sort(array){
if(array.length==0).return[];
var left=[], right=[], Pivot=array[0];
for(var i=1; i<array.length; i++)
{
if(array[i]<Pivot)
left.Push(array[i])
else
right.Push(array[i];
}
return quicksort(left).Concat(Pivot,
quicksort(right));
}
Console.log(quicksort(array.slice()));
=>[1,2,3,4,5,6,7,8,9,10,11,12]
```

Algorithm for insertion:-

1). If the head node is null then insert data in the head node.

2). If the input data is less then the start node then insert the node at start

3). If the input data is greater then the start node, till you get the right position to insert move the pointer.

4). If the element lies b/w any two values then connect the node to the previous node and the next node ie. t->next = temp->next and temp->next = t.

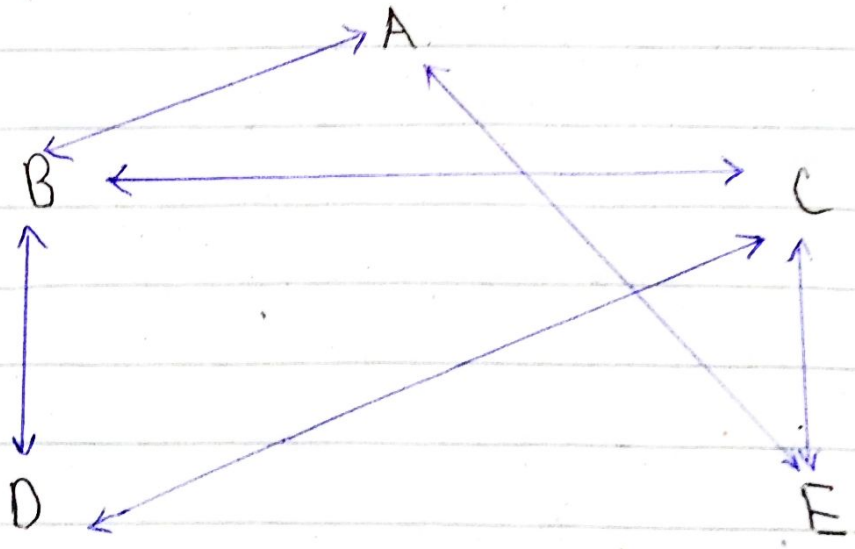
(6)

Question 4 - (a)

	A	B	C	D	E
A	0	1	0	0	1
B	1	0	1	1	0
C	0	1	0	1	1
D	0	1	1	0	0
E	1	0	1	0	0

Part :- (b) :-

Graph :-



Question = 1 -

Answer:-

Insertion of an element:-

Steps:-

1- We will create a newNode with given values.
2- Check to find wether it is Empty (head == Null)
3- If Empty then set Newnode → next = Null and head = newnode.
4- If not empty the define a node pointer temp and initialize with head.
5):- Keep moving untill it reaches to our desires point where we want to insert new node (until temp1 → data is equal to location.
6):- Check whether temp reaches to last node or not.
7):- Finally, Set 'newnode → next = temp → next' and 'temp → next = newnode

Delition from list:-

1). Check for empty list (head==NULL)
2). If it is empty Display list is empty.
3). If its not empty define two nodes pointer. 'temp1' and 'temp2'
4). Keep moving until reached to our desired pointer.
5:- when we reach to that point which we want to delete then check wether list is having only one node or not.
6. If it is only one then set head = Null and delete temp1 (free(temp1)).
7:- If it contains multiple node then check whether temp1 is first node in list (temp1==head)
8:- If temp1 is first node move to next node (head = head->next) and delete temp1.

9.- If temp1 is not first node then check wether it is last node in list (temp1 -> next == NULL)

10.- If temp1 is last node then set temp2 -> next = NULL and delete temp1 (free(temp1)).

11).- If temp1 is not first and last node the set temp2 -> next = temp1 -> next and delete temp1 (free(temp1)).