Department of Electrical Engineering
Assignment
Date: 21/08/2020
Course Details

Course Title:	Data Structure and Algorithm	Module:	
Instructor:		Total Marks:	30

Student Details

13746

Name:	mohsin ali	Student ID:	

Note: Plagiarism of more than 20% will result in negative marking.
Similar answers of students will result in cancellation of the answer for all parties.

Q1. The following is your sorted array and let assume that you need to search the location of value 31 using binary search. (CLO 1, Marks 10)

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

Q2. Let LA be a Linear Array (Unordered) with N elements and K is a positive integer such that K<=N. Following is the algorithm where ITEM is inserted into the K^{th} position of LA- (CLO 2, Marks 10)

1. Start
2. Set J=N
3. Set N= N+1
4. Repeat steps 5 and 6 while J>=K
5. Set LA [J+1]= LA[J]
6. Set J=J-1
7. Set LA[K]=ITEM
8. Stop

Write the implementation of the above algorithm

Q3. Find a given target number (x=61) using linear Search from a list of number using C++. (CLO 1, Marks 10)

[18, 36,56,61,73,87,93]

Scanned with CamScanner

Name : Mohsin Ali
ID = 13746
Paper: Data structure.

Date: 21/8/2020

Q1

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

First we shall determine Half of The array by using This farmula

$$mid = low + \frac{(high - low)}{2}$$

Here it is, $0 + (9-0)/2 = 4$ (Integer Value of 4.5)
So 4 is The mid of The array.

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

Now we compare The value stored at location 4, with

p.t.o

being Searched i.e 31
we find that the value at
location 4 is 27 which is
not match As The value is
greater than 27 and we have
a Sorted array so we also
know that The integer value
must be in The upper portion
of The array

10	14	19	16	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

we chang our low to mid+1
and find The new mid value
again

Low = mid+1

mid = low + (high − low)/2

our new mid is 7 now
we Compare The value stored
at location 7 with our
Target value 31

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	35	42	44
0	1	2	3	4	5	6	7	8	9

10	14	19	16	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

p.t.o

Scanned with CamScanner

The value stored at location 7 is not match rather it is more then what we are looking for so, the value must be in the lower part from this location

10	14	19	26	27	31	33	35	42	44
0	1	2	3	4	5	6	7	8	9

we compere the value stored at location 5 with our target value we find that it is a match.

10	14	19	26	27	31	33	35	42	44	
0	1	2	3	4	5	6	7	8	9	

We conclude that the target value is 31 is stored at location 5

Binary Search halves the Searcheble items and thus reduces the count of comparisions to be made to very less numbers.

Scanned with CamScanner

(4)

Q 3

```
# include <iostream>
Using namespace std;
Void linner Search (int a[], int n)
int temp = -1;
For ( int i=0; i<7; i++

{
    if (a[i] == n)
{
Cout << " Element found at location: "
    " << endl;
}
}

}

int main()
{
    int arr[7] = { 18 36 56 61 73 87 93 };
    Cout << " please enter an element -
        to Search" << endl;
    int num;
    Cin >> num;
    Linear search ( arr, num);
    return 0;
}
```

Scanned with CamScanner

(5)

Q2

```
#include <stdio.h>
main(){
int LA[] {1, 3, 5, 7, 8};
 int  item = 10, k = 3, n = 5;
 int  i = 0, J = n;
Print f (" the orignal array elements
 are: \n");
 for (i = 0; i<n; i++
 Printf ("LA[%d] = %d \n", i,
    LA (i]);} n = n+1;
while (j >= k)
 {

    LA[j+i] = LA[j];
      j = J-1;
 }

   LA[k] = item
 Printf (" The array elements after
   insertions : \n");
for (i=0; i<n; i++){
```

Scanned with CamScanner

```
Print f ("LA[%d] = %d/n", i,
  LA[i]
}
}
```

::out put::

LA [0] = 1
LA [1] = 3
LA [2] = 5
LA [3] = 7
LA [4] = 8

the array element after insertion

LA [0] = 1
LA [1] = 3
LA [2] = 5
LA [3] = 10
LA [4] = 7
LA [5] = 8

Scanned with CamScanner