R. Ramya Manasa
AP19110010472
CSE-F

Assignment-6

1. Take the elements from the user and sort them in descending order and do the following.....

Code:-

```
#include <stdio.h>
void sort (int a[], int n)
{
int i, j, temp;
for(i=0; i<n; i++)
{
for(j+1; j<n; j++)
{
if (a[i]<a[j])
{
temp = a[i];
a[i] = a[j];
a[j] = temp;
}
}
}
}
int binary(int a[], int e, int n)
{
int i=0, j= n-1, mid;
while (i<=j)
{
mid = (i+j)/2;
if (a[mid]==e)
return mid+1;
```

```
else
{
if (e<a[mid])
j = mid-1;
else
i = mid+1;
}
}
if(i>j)
{
return 0;
}
}
int main()
{
int n, i, a[20], f, e, t1, t2;
Printf ("enter the no of elements of array: ");
Scanf ("%d", &n);
printf ("enter the elements of array:\n");
for(i=0; i<n; i++)
Scanf ("%d", &a[i]);
Sort (a, n);
printf("Desending order: ");
for(i=0; i<n; i++)
printf("%d\n", a[i]);
printf("\nenter the element to find in array: ");
scanf ("%d", &e);
f = binary(a, e, n);
if (f!=0)
{
```

```
Printf ("elements is found at %d positions : ", f);
}
else
{
printf ("element not found \n");
}
printf("enter the position of array to find sum and product:\n");
Scanf ("%d %d", &t1, &t2);
t1--;
t2--;
printf(" the sum is %d ", a[t1] + a[t2]);
printf(" the product is %d", a[t1] * a[t2]);
}
```

Input-Output:-

```
enter the no of elements of array : 3
enter the elements of array:
10
20
30
Descending order: 30
20
10
enter the elements to find in array: 40
element not found
enter the position of array to find Sum and product:
2
3
the Sum is 30 the product is 200.
```

2. Sort the array using Merge Sord where elements are taken from the user.....

```
#include <stdio.h>
#include <stdlib.h>
Void merge(int arr[], int l, int m, int r).
{
int i, j, k;
int n1 = m-l+1;
int n2 = r-m;
int L[n1], R[n2];
for (i=0; i<n1; i++)
L[i] = arr[l+i];
for (j=0; j<n2; j++)
R[j] = arr[m+1+j];
i = 0;
j = 0;
k = l;
while (i<n1 && j<n2)
{
if( L[i] <= R[j])
{
arr[k] = L[i];
i++;
}
else
{
arr[k] = R[j];
j++;
k++;
}
```

```
}
Void mergsort(int arr[], int l, int r)
{
if(l<r)
{
int m= l+(r-1)/2;
mergsort (arr, l, m);
mergsort (arr, m+1, r);
merge (arr, l, m, r);
}
}
Void printArray(int A[], int size)
{
int i;
for (i=0; i< size; i++)
Print f("%d", A[i]);
Printf("\n");
}
intmain()
{
int arr[6];
int i;
int arr_size = size of (arr)/size of (arr[0]);
printf(" Enter the 6 elements in an array \n");
print f ("Enter the elements ");
for(i=0; i<arr_size; i++){
Scanf ("%d", &arr[i]);
}
```

```
Printf(" Given array is \n");
PrintArray(arr, arr_size);
mergesort(arr, 0, arr_size-1);
Printf("\n Sorted array is \n");
PrintArray(arr, arr_size);
int k;
Printf("enter the Value of k");
Scanf("%d", &k);
int fromfirst = arr[k-1];
int fromlast = arr[6-(k)];
Printf("%d", fromlast*fromfirst);
return 0;
}
```

Input-output:-

```
Enter the 6 elements in an array.
Enter the elements: 10
20
30
40
50
60
Given array is
10 20 30 40 50 60
Sorted array is
10 20 30 40 5060
enter the value of k: 4
1200.
```

3- Discuss insertion Sort and Selection Sort with examples.

Insertion Sort:-

Insertion sort is a simple Sorting algorithm that works the way we sort playing cards in our hands.

Algorithm:

// Sort an arr[] of size n

insertion sort(arr, n)

Loop from i=1 to n-1.

a.) pick element arr[i] and insert it into sorted sequence arr[0.... i-1]

Example: 13, 09, 18, 05, 07

Let us loop for i=1 (second element of array) to 4 (last element of array)

i=1 Since 09 is smaller than 13, move 13 and intert 09 before 13

i=2 Since 18 will remain at its position as all elements in A [0.... i-1] are smaller than 18

09, 13, 18, 5, 07

i=3 5 will move to the beginning and other elements from 09 to 18 will move one position ahead of their current position

05, 09, 13, 18, 07

i=4 7 will move to position after 5, ∴

05, 07, 09, 13, 18

Selection Sort:-

The Selection Sort algorithm Sorts by an array by repeatedly find the minimum element from unsorted part and putting in into the beginning The alogorithm maintains two Subarrays in given array.

1.) The Subarray which is already Sorted.

2.) Remaining Subarray which is unsorted.

In every Iteration of Seclion Sort, the minimum elements from the unsorted subarray is picked and moved to the Sorted subarray.

Examples:-

arr[]: 68 35 22 27 13

// find minimum element in array and place it an begining

13 35 22 27 68

// Find minimum element in array [1---4]

// and place it at beginning of array[1---4]

13 22 35 27 68

// Find the minimum element in array [2---4].

// and place it at begining of array[2--4]

13 22 27 35 68

// Find the minimum element in array [3--4]

// and place it at begining of array[3---4]

13 22 27 35 68.

4.) Sort the array using bubble sort where elements are taken from the user and display the elements.----

```
#include <stdio.h>
Void main()
{
int a[100], n, i, j, temp, sum=0, prod=1, m;
Printf("Enter number of elements:\n");
Scanf("%d", &n);
Printf("Enter %d integers \n", n);
for(i=0; i<n; i++)
{
Scanf("%d", &a[i]);
}
for(i=0; i<n-1; i++)
{
for(j=0; j<n-i-1; j++)
{
if(a[j]>a[j+1])
{
temp = a[j];
a[j] = a[j+1];
a[j+1] = temp;
}
}
}
printf("\n Sorted list in ascending order:\n");
for(i=0; i<n; i++)
{
printf("the alternate order is:");
```

```
for(i=0; i<n; i++)
{
if(i%2==0)
{
printf("%d ",a[i]);
}
}
for(i=0; i<n;i++)
{
if(i%2!=0)
{
sum=sum+a[i];
}
}
printf("\nsumof odd Index is %d", sum);
for(i=0; i<n; i++)
{
if (i%2=0)
{
prod= prod * a[i];
}
}
printf("\n product of odd Index is %d", prod);
printf ("\n enter the value of m \n");
Scanf("%d",&m);
for (i=0; i<n; i++)
{
if(a[i]%m=0)
Prinf("%d", a[i]);
}
}
}
```

Input-Output:

```
Enter number of elements:
4
Enter 4 integers:
12
14
16
18
Sorted list in ascending order
12
14
16
18
the alternate order is 12 16
Sum of odd Index is 32
product of odd Index is 192
Enter the value of m: 3
12 18
```

5. Write a recursive program to implement binary search.

```
#include<stdio.h>
int recursiveBinarysearch(int array[],intstart_index,int end_
    index , int element {
if (end_index.>=start index){
 int middle = start_index + (endindex) /2;
if (array [middle] == element)
return middle;
 if (array[middle]' >element)
return recursive Binary search(array, start_index,middle-1,element);
return recursive Binarysearch(array, middle+1 ,end index, element);
}
return -1;
}
int main(void){
int array[]={1,4,7,9,16,86,90};
int n=7;
int element =9;
 int found_index =recursive Binary search(array, 0, n-1,element);
if (found_index ==-1){
Printf(" Element not found in the array"); }
else{
printf( "Element found at index: %d ", found_index);}
return 0;
}
```

Output:

Element found at Integer: 3