K. Darshini Reddy
AP19110010403
CSE-F

1) Take the element from the user and sort them in descending order & do the following.

a) using binary search find the element and the location in the array where the element is asked from user.

b) Ask the user to enter any two where elements are taken from the user any two locations print the sum & product of values at those locations in the sorted array.

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, m1, m2;
     Printf("enter the no. of elements of array");
      Scanf("%d", &n);
     Printf("enter the element of array \n");
      for (i=0; i<n; i++)
```

a) 'C' Program for merge sort */

```
#include <stdio.h>
#include <stdio.h>

// merge two subarrays of arr[].
// first subarray is arr[l...m]
// second subarray is arr[m+1...r]
void merge(int arr[], int l, int m, int r)
{
    int i, j, k;
    int n1 = m-l+1;
    int n2 = r-m;

    /* Create temp array */
    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[i];
            j++;
        }
```

```
    Scanf("%d", &a[i]);
Sort(a, n);
for(i=0; i<n; i++)
        Printf("%d", a[i]);
        Printf("enter the elements to find in array");
        Scanf("%d", &e);
        f = binary(a, e, n);
        if(f!=0)
        {
            Printf("Elements is found at %d position", f);
        }
    else
      {
        Printf("element not found);
      }
    Printf("enter the position of array to find sum and
                                    Product \n");
     Scanf("%d %d", &m1 & m2);
          m1 ---;
          m2 ----;
     Printf("the Sum is %d", a[m1] + a[m2]);
     Printf("the Product is %d", a[m1] * a[m2]);
}
```

```
    k++;
}
while (i<n1)
 {
  arr[k] = L[i];
   i++;
   k++;
 }
 while (i<n2)
  {
    arr[k] = R[j];
     j++;
     k++;
  }
}
void mergeSort(int arr[], int l, int r)
  {
    if (l<r)
     {
      int m = l+(r-1)/2;
      mergeSort(arr, l, m);
      mergeSort(arr, m+1, r);
      merge(arr, l, m, r);
     }
  }

    void PrintArray(int A[], int size)
      {
        int i;
       for (i=0; i<size; i++)
            Printf("%d", A[i]);
            Printf("\n");
       }
```

```
int main()
{
  int arr[5];
  int i;
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  for(i=0; i<arr_size; i++){
        Printf("enter the elements");
        Scanf("%d", &arr[i]);
      }
        Printf("Given array is \n");
        Print array(arr, arr_size);
        Merge sort(arr, 0, arr_size-1);
        Printf("\n Sorted array is \n");
        int k;
        Printf("enter the value of k");
        Scanf("%d", &k);
        int from first = arr[k-1];
        int from last = arr[5-(k)];
        Printf("%d", from last * from first);
            return 0;
      }
```

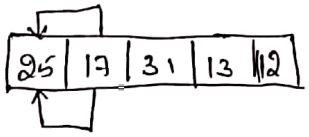
3) Insertion Sort:

Insertion sort is implemented by inserting a particular element at the appropriate position. In this method, the first iteration starts with comparison of 1st element with 0th element. In the 2nd iteration, 2nd element is compared with the 0th and 1st element. In general, in every iteration an element is compared with other elements. During comparison, it is found that the element in the given element can be inserted in the suitable position. This is repeated for all elements of the array.

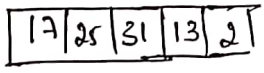
Example:

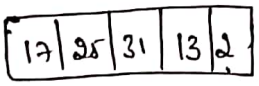
Insertion Sort:

1st iteration:

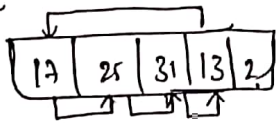


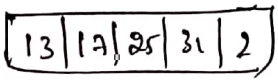
2nd iteration:

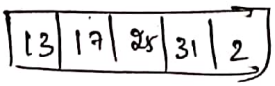




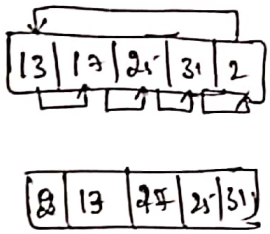
3rd iteration:







4th iteration:

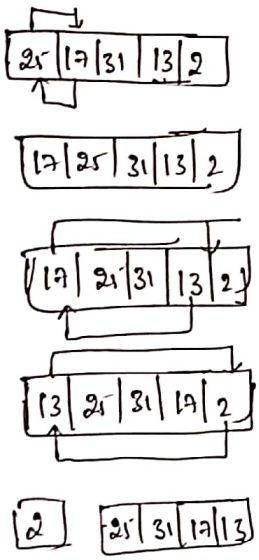


Selection Sort:

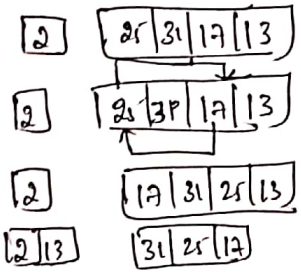
This is the simplest method in the method of sorting. In this method to sort the data in ascending order the 0th element is compared with all the other elements. If 0th element is found to be greater than the compared element, then it is exchanged. So after the overall iteration, the smallest element is placed in 0th position.

Example:

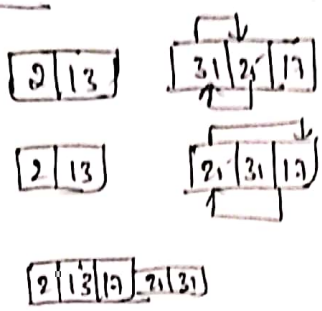
1st iteration:



2nd iteration:



3rd iteration:



4)

```
#include <stdio.h>
void main()
{
int a[100], n, r, i, temp, sum=0, prod=1, m;
Printf("Enter number of elements\n");
Scanf("%d", &n);
Printf("Enter %d integers\n", n);
for(r=0; r<n; r++)
{
Scanf("%d", &a[r]);
}
for(r=0; r<n-1; r++)
{
  for(j=0; j<n-r-1; j++)
  {
    if(a[j]>a[j+1])
    {
      temp=a[j];
      a[j]=a[j+1];
      a[j+1]=temp;
    }
}}
```

```
Printf("\n sated list in ascending Order:\n");
  for(i=0; i<n; i++)
   {
   Printf("%d\n", a[i]);
   }
   Printf("the alternate orden is");
    for(i=0; i<n; i++)
     {
       if(i%2==0)
        {
          Printf("%d", a[i]);
        }
     }

   Sumo = Sumo + a[i];
  }
}
 Printf("\n Sum of Odd index is %d", Sumo);
    for(i=0; i<n; i++)
     {
       if(i%2==0)
        {
          Prod = Prod * a[i];
       }
    }
    Printf("\n Product of odd inden 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)
            {
```

```
        Printf ("%d", a[i]);
    }
  }
}
```

5)

```
#include <stdio.h>
void binary_search (int[], int, int, int);
int main()
{
  int key, size, i;
  int list[25];
  Printf ("Enter the size of a list : ");
  Scanf ("%d", &size);
  Printf("Enter the elements \n");
  for (i=0; i< size; i++)
  {
    Scanf("%d", &list[i]);
  }
  bubble_sort (list, size);
    Printf("\n");
  Printf("Enter key to Search \n");
  Scanf("%d", &key);
  binary_search (list, 0, size, key);
}
void bubble_sort (int list[], int, size)
{
    int temp, i, j;
```

```
for (i=0; i< size ; i++)
  {
    for (j=0; j< size ; j++)
      {
        if (list[i] > list[j])
          {
            temp = list[i];
            list[i] = list[j];
            list[j] = temp;
          }
      }
  }
}

Void binary_search (int list[ ], int lo, int p, int )
{
  int mid ;
  if (lo> p)
  {
    Printf(" ice-cream not found \n");
      return ;
  }
  mid = b + p / 2
    if (list[mid] == key)
    {
      Printf (" ice-cream found \n")
    }
      else if ( list[mid] > key)
      {
        binary-search ( list, lo, mid -1 , icecream);
    }
      else if ( list[mid] < ice cream)
    {
      binary- search ( list, mid +1 , p , ice-cream)
  }
```