①

```
#include <stdio.h>
int binarySearch(int arr[], int a, int b, int x)
    {
        if (b>=a)
        {
          int mid = a+(b-a)/2;
           if (arr[mid] == x)
                return mid;
             if (arr[mid]>x)
                return binarysearch (arr, a, mid-1, x);
             return binarysearch(arr, mid+1, b, x);
          }
           return -1;
      }
    int main()
      {
         int num;
          printf ("Enter the size of array: ");
           scanf("%d", &num);
           int i, j, a, val[num], op, val, p, p2, sum, pro;
           for (a=0; a<num; a++)
      {   printf ("Enter Value: ");
             scanf ("%d", &val[a]);
           }
          for (i=0; i<num; ++i)
            {
```

```
    for(j=i+1;j<num;++j)
  {  if(val[i] < val[j])
     {
        a=val[i];
        val[i]=val[j];
         val[j]=a;
     }
   }
}
 printf("Array in descending order:   ");
 for(i=0;i<num;i++)
 {
   printf(" %d",val[i]);
   }
  printf("\n**OPERATION_LIST**/n");
  printf("1. Find value at entered position\n 2. Find the position
                                    of element\n3. Printing
                                    sum & multiplication of values
                                       at entered positions");

   printf("\nEnterchoice :\n");
    scanf("%d", &op);
    switch(op)
      {
        case 1:
    printf("Enter the position to obtain value: ");
    scanf("%d",&var);
    printf("the value at %d position is %d",var,val[var]);
     break;
```

```
case 2:
 printf("Enter element to find position: ");
 scanf("%d ",&var);
 int result = binarySearch(val,0,num-1, var);
 (result == -1) ? printf("Element is not present in array")
     :printf("Element is present at index %d",result);
     return 0;

  case 3;
   printf("\n Enter two positions to find sum and product of values\n");
   scanf("%d %d",&p1,&P2);
   sum = val[p1] + val[p2];
    pro = val {p1] * val[p2];
    printf ("sum = %d\n", sum);
    printf("MULTIPLICATION = %d", pro);
    break;
    }
  }
```

②
```
   # include <stdlib.h>
   # include <stdio.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+1)
        R(j]=arr[m+1+j]);

i=0;
j=0;
k=1;
while(i<n1 && j<n2)
      {
      if(L[i]<=R(j])
       {
        arr[k]=L[i]
         i++;
        }
        else
       {
       arr[k]=R[j];
         j++;
       }
      k++;
     }
    while(i<n1)
     {
      arr[k]=L[i];
        i++j
       k++;
     }
    while(j>n2)
    {
    arr[k]=R(j);
     j++;
  }} k++;
```

```
void mergeSort(int arr[], int l, int r)
 {
  if(l<r)
   { int m= l+(r-l)/2;

  mergeSort(arr,l,m);
  mergeSort(arr,m+1,r);
   mergeSort(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 size,v;
    printf("Enter arraysize :");
     scanf("%d"&size);
     int val[size];
    for (v=0;v<size;v++)
    {
      printf("Enter Value: ");
      scanf("%d",&Val[v]);
    }
      printf(" Given array is \n");
     printArray(val,size);
     mergeSort(val,0,size-1);
```

```
printf ("\n Sorted array is \n");
 printArray(val, size);
 int k, f, l, P1, P2, temp;
printf("Enter the val of k to find the product of elements from
                                          first and last :   ");

 scanf(" %d ", &k);
    P1 = P2 = 1;
   for (f =0; f<k; f++)
   {
     temp=val [f];
     P1* =temp;
   }
   for( l=size -1 ; l>= k ; l--)
     {
     temp =val [l];
       P2* = temp;
     }
 printf("product of kth elements from first and last are : %d %d",
                                                    P1, P2);

   }


(4)  #include <stdio.h>
     void bubblesort(int ar[], int n)
        {
        int i, j, temp;
         for(i=0; i<n-1; i++)
         for(j=0; j<n-i-1 ;j++)
       if (ar[j] >ar[j+1]) /*Exchanging values using condition and
                                                    temp variable*/
        {
```

```
temp = ar[j];
  ar[j] =ar[j+1];
  ar[j+1] = temp;
 }
}
  int main()
 {
  int siz, i;
 printf("Enter size of required array: ");
 scanf("%d", &arr[i]);
 }
  bubble sort (arr, siz);
 printf("Sorted array: \n");
  for (i=0; i<siz; i++)
  {
   printf( "%d", arr[i]);
   printf ("\t");
  }
    printf ("\n/**MENU**/\n");
  printf("1.Display elements in alternate order\n");
  printf("2. sum of elements in odd positions and product of elements
                                              in even positions\n");

  printf("3.Divisible by m\n");
  int op, sum=0, product =1, m;
   printf("Enter choice: ");
   scanf ("%d", &op);
   switch (op)
    {
     case 1:
    for (i=0; i<siz; i+=2)
  { printf ("%d\t", arr[i]); }
```

case 2:

```
for (i=0;i<siz;i+=2
 {
   sum=sum+arr[i];
 }
  for (i=1;i<siz;i+=2)
   {
   product = product*arr[i];
    }
     printf("sum: %d\n",sum);
    printf("Product :%.d\n",product);
```

case 3:

```
 printf("Entervalue m: ");
  scanf("%d",&m);
  printf("Numbers divisible by %d are :\n",m);
  for(i=0;i<size;i++)
   {    if (arr[i]%m==0)
         {
           printf(" %d \t",arr[i]);
           }
         }
       }
     }
```

⑤

```
   # include <stdio.h>
 int binarysearch(int a[], int l,int h,int x)
      {
```

```
int mid = (l+h)/2;
if(l>h) return -1;
  if (a[mid] = =x)
     return mid;
  if (a [mid] <x)
  return binary search(a, mid+1, h, x);

else
    return binarysearch(a, l, mid-1, x);
}
  int main(void)
  {
   int a[100];
   int siz,pos,val;
  printf("Enter length of the array");
   scanf ("%d", &siz);
  printf("\nEnter array elements\n");
      for (int i=0;i<siz;i++)
          scanf ("%d", &a[i]);
   printf ("Enter elements to search \n");
    scanf ("%d", &val[E]);

  pos = binary search(a, 0, siz-1, val);

  if(pos<0)
 printf("Cannot find the element %d in the array.\n", val);

   else
printf("the position of %d in the array is %d.\n", val,
                                          pos+1);
```

```
return 0;
}
```

— x —

① Insertion sort: One element from the array is selected and is compared to the one side of the array and inserted to the proper position while shifting the rest of the elements accordingly

eg:
```
# include <stdio.h>
int array [5] = {23,17,20,12,30};
void print_array(int elements [], int count)
{
  ~~printf(~~  for (int index = 0; index count; index ++)
   {
      printf(" %d ", elements[index]);
   }
   printf("\n");
}
 void insertion_sort (int elements[], int count)
   {
    int selection, index;
  for (selection = 1; selection < count; selection++)
{ int temp = elements [selection];
   printf (" position #%d value %d \n" selection, elements[selection]);
   print_array (elements, count);
for (index = selection; index > 0 && temp < elements[index -1]; index--);
      {
        printf("move %d → %d \n", elements [index -1], elements
                                                     [index]);
        elements [index] = elements [index-1];
        print_array (elements, count);
      }
    printf("insert @%d = %d \n", index, temp);
    elements[index] = temp;
    print_array(elements, count);
    printf("\n");
      }
  }
  int main(int fargs, char *argv[])
   {
     printf(" Insertion sort \n");
         print_array(array, 5);
         insertion_sort(array, 5);
   }
```

selection sort : selection sort in C is to sort numbers of an array in ascending order with a little modification it arranges numbers in descending order.

eg:-
```
    # include <stdio.h>
    int main()
    {  int array[100], n, c, d, position, t;
       printf(" Enter number of elements \n");
        scanf("%d", &n);
        printf(" Enter %d integers \n", n);
        for (c=0; c<n; c++)
        scanf("%d", &array[c]);
       for (c=0; c<(n-1); c++)
        {
          position = c;
```

```
for (d = c+1; d<n; d++)
  {
 if (array[position] > array(d])
     position = d;
 }
 ~~if (position =d)~~;
 }
if (position != c)
  { t = array [c];
   array [c) = array(position];
      array(position] = t;
   }
  }
 printf ("sorted list in ascending order : \n");
      for (c = 0; c<n; c++)
      printf ("%d\n", array [c]);
          return0,
        }
```

- x -

Scanned with CamScanner