ASSIGNMENT

M. SIRI ANJANI KEERTHI | CSE-F
AP19110010439

1. Take the elements from the user and sort them in descending order & do the following
   (a) using Binary search find element and location in array
   (b) Ask the user to enter any two locations print sum and product of values.

```
#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, var, P1, 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 in.
        2. Find the position of element in
        3. Printing sum & multiplication of
           values at entered positions");
printf("\n Enter choice");
scanf("%d", &op);
switch(op)
{
  case 1:
```

```
printf("Enter position to obtain value :");
scanf("%d", &var);
printf("The value at %d position is %d, var,
        val[var]);
break;
case2:
    printf("Enter element to find position:");
    scanf("%d", &var);
    printf("The value at %d position is %d");
    int result = binary search (val, 0, num-1, var);
(result == -1) ? printf("Element is not present
                        in array");
printf("Element is placed at index %d", result);
return 0;
case 3:
    printf("Enter two positions to find sum &
            products of values in");
    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. Sort the array using Merge sort where elements are taken from the user and find the product of kth elements from first and last where k is taken from the user

Solution:

```
#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++)
    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++;
}
while (i<n1)
{
arr[k] = L[i];
i++;
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);
    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 size, v;
  printf("Enter array size: ");
  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("Sorted Array is \n"); printArray(val, size);
int k, f, l, p1, p2, temp;
printf("Enter value of k: ");
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", k1, k2);
}
```

3 Discuss Insertion sort and selection sort with examples.

* 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.

Example:

```
#include<stdio.h>
int array[5] = {23,17,20,12,30};
void print_array(int elements[], int count)
{
  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 tmp = elements[selection];
    printf("position #%d value %d \n", selection,
           elements[selection]);
    print_array(elements,count);
    for(index=selection; index>0 && tmp<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 argc, char *argv[])
{
  printf("Insertion sort \n");
  print_array(array, 5);
  insertion_sort(array, 5);
}
```

* Selection Sort: Selection sort in C is to sort number of an array in ascending order with a little modification it arranges numbers in descending order. selection sort example :-

```
#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 != 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]);
    return 0;
}
```

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

(1) In alternate order

(2) Sum of elements in odd position and product of elements in even positions

(3) Elements which are divisible by m where m where m is taken from the user.

```
#include<stdio.h>
Void bubblesort (int arr[], 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])
   {
    temp = ar[j];
     ar[j]= ar[j+1] ;
     ar[j+1] = temp;
   }
}
  int main()
  {
    int size, i ;
   printf(" Enter size of required array: ");
```

```
scanf("%d", &size);
int arr[size];
for (i=0; i<size; i++)
{
printf(" enter element: ");
 scanf("%d", &arr[i]);
}
bubble sort (arr, size);
 printf(" sorted array: \n");
 for (i=0; i<size; i++)
 {printf("%d", arr[i]); printf("\t");
 }
printf("\n" MENU "\n");
printf(" 1. Display elements in alternate order");
printf(" 2. Sum of elements in odd & Product in even");
 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", 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: \n product: \n", sum, product);
case 3. printf(" enter m value", m); pf(" they are " divisible:, m);
 for (i=0<=i<siz; i++)
{ if(arr[i]%m==0) { printf("%d\t", arr[i]);
}}}
```

5. Write a recursive program to implement binary search?

```
#include <stdio.h>
int main() {
    int a[10], i, n, m, c, l, u;
printf("Enter the size of an array:");
scanf("%d", &n);
printf("Enter the elements of the array:");
for (i=0; i<n; i++)
{
  scanf("%d", &a[i]);
}
  printf("Enter the number to be search:");
   scanf("%d", &m);
   l=0, u=n-1;
   c = binary (a,n,m,l,u);
   if (c==0)
   printf("Number is not found.");
  else   printf ("Number is found");
return 0; }
int binary (int a[], int n, int m, int l, int u) {
    int mid, c=0;
if (l<=u) { mid = (l+u)/2;
    if (m == a[mid]) { c=1;
  } else if (m<a[mid]) { return binary (a,n,m,l,mid-1);
   } else return binary (a,n,m, mid+1,u); } else
                                    return c;
}
```