ASSIGNMENT-6

B. Deepika
AP19110010245
CSE-G

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

int comparator ( const void* s1 , const void* s2)

  {   return(* int*)s2 - *(int*) s1);

  }
 int  binary  Search (int arr[ ],int size, int search)

   {  int beg =0, end= size-1, mid;
      while (beg <= end)

      {  mid = (beg+ end)/2
         if (arr[mid] == search)

         {     return mid;

         }
    else if (arr[mid] < search)

         {  end = mid-1;

         }
         else beg = mid+1;

      }
      return -1;
 }
 int main ( )

   {  int arr[100], size, search, i, pos=-1, loc1, loc 2;
        printf("\nEnter the size of the array(max 100)
                    scanf (" %d", &size);
printf ("\nEnter elements in array \n");
    for (i=0;, i<size;, i++)
      {
         scanf (" %d", & arr[i]);
      }

qsort (arr, size, size of (int), comparator);
     printf(" \n The sorted array is : \n");
     for (i=0;, i< size ;, i++)
    { printf("%d", arr [i]);
        }

  printf (" \n Enter search element");
   scanf ("%d", &search);
   ~~if (pos ==-1)~~ pos = binarysearch (arr, size, search);
   if ( pos ==-1 )
        printf(" Not found \n");
     else
       printf (" \n the %d search element is found at
                              index %d\n",search, pos);
      printf(" Enter two indexes\n");
      scanf ("%d%d , &loc1, & loc 2);
  printf (" sum is %d\n",arr[loc1] + arr[loc 2]);
  printf (" product is %d \n", arr[loc1]* arr[loc 2]);
    }
```

```
(2.) #include <stdio.h>
# define ms 100
int x[ms];
Void merge (intMl, int n1, int m2, int n2)
{ int i, j, K, temp [ms];
    k= 0;
    i= m1;
    j= m2;
while ( i<= n1) && ( j <= n2))
{   if [x[i] < x[j])
  { temp [k] = x [i]., i++, k++
  }
  else
   {temp [k] = x [j] ; j++; K++
   }
  }
 while ( i<= n1 )
{
    temp[k] = x [i]., i++; K++
   }
  while ( j<= n2)
  { temp[K]= x[j]; j++; K++
   }
  for ( i= m1, k= 0., i <= n2; i++, k++)
     { x[i]= temp [K];
   }
    }
```

```
Void merge sort (int mb, int nb) {
{
    if (mb<nb)
  {
    int mid= (nb+mb)/2
    merge sort (mb,mid);
    merge sort (mid+1, nb);
    merge (mb,mid, mid+1, nb);
  }
 }
int main( )
{
   int i, n, product=1, k;
printf (" \n Entel the size of array max (100)");
   scanf ( "%d", &p);
   for(i=0; i<p ; i++)
  { printf(" x [%d] \t = ",i);
     scanf( "%d", & x[i]);
    }
merge sort (0, p-1);
   printf(" Entel k\p");
   scanf ("%d", & k);
   for (i=0 ; i<k; i++)
  {  product *= x[i];
   }
   printf(" \n the product till the kth element is %d\n",
                                                product);
      return 0;
   }
```

③ ~~Insertion Sort~~:

Output:

```
Enter the size of array. 5
x [0] = 1
x [1] = 6
x [2] = 1
x [3] = 54
x [4] = 2

Enter K = 3
The product till the kth element is 2.
```

③ Insertion sort:

- If the elements in first one is already sorted move to next element.
- Compare the current element with all elements in sorted array.
- If the element in sorted array is smaller than current element, iterate to the next element, otherwise shift all the greatest elements in array by one position towards right.
- Insert the value at the correct position.
- Repeat until the complete list is sorted.

123	18	94	4	37
123	18	94	4	37.
18	122	94	4	37
18	122	94	4	37
18	94	122	4	37
18	94	122	4	37

123, 18, 94, 4, 37.

for i = 1 (2nd element) to 37

i = 1, Since 18 is smaller than 123, move 123 & insert 18 before 123.

→ 18, 123, 94, 4, 36

i = 2, since 94 is smaller than 123, move and insert 93 before 123.

9	18	99	122	37

9	18	99	122	37

4	18	37	99	122

18, 94, 123, 9, 37

i=3, 9 will move to beginning and all other element from 18 to 123 will move one portion ahead of their current position.

9, 18, 94, 123, 37.

i=4, 37 will move to position after 18, and elements from 94 to 123 will move one position ahead of their current position.

9, 18, 37, 99, 123.

Selection sort:

Consider an array [12, 3, 5, 7], ~~10~~

The first element is 12, the next part we need to do is we need to find the smallest number from the array. The number from array is 3.

So, we replace 12 by 3.

The new array would be [3, 5, 7, 12].

Again the process would be repeated.

Finally, we get the sorted array as [3, 5, 7, 12].

- Set min to first location.
- Search minimum element in array.
- Swap the first location with min value in array.
- Assign the second element as min
- Repeat the process until we get a sorted array.

```
(9.)  #include <stdio.h>
      #include <conio.h>
  int main ( )

  {
    int arr[50], i, j, n, temp, sum=0, product = 1;
    printf ("enter total number of elements:"),
    scanf (" %d", &n);,

     printf(" Enter %d elements: ", n );
     for (i=0, i<n; i++)
     Scanf (" %d", & arr[i]);
   printf(" \n sorting array using bubble sort \n");

   for (i=0; i<(n-1) ;i++ );
     { for (j=0, j<(n-i-1); j++)
     { if   (arr[j] > arr[j+1]
        {temp = arr[j];
         arr[j] = arr[j+1]
         arr[j+1] = temp;
         }
         }
    }
  printf("All  array  elements  sorted \n");
    printf(" Array elements in ascending order:\n\n");
       for (i=0, i< n; i++)
            { printf(" %d \n", arr[i]);,
            }
      printf(" Array elements in alternate order \n");
for (i=0; i<=n; i=i+2)
{
    printf("%d\n", arr[i]);
}
for (i=1; i<=n; i=i+2)
{ sum = sum+ arr [i];
}
printf(" The sum of odd position elements
                    are : %d \n", sum);

for ( i=0; i<n; i=i+2 )
{ product = arr [i];
}
printf(" The product of even position elements.
                    are= %d \n", product );

    return 0;

}
```

Output!

```
Enter total  number of elements = 5
Enter 5 elements.
 4
         Sorting array using bubblesort.
 6

 2          sum of odd position is ,7
 10
         Product of even elements  : 6,4
 8
Ascending  order.
  2  4  6 8 10
Array  element in  alternate
        2 6  10.
```

```
(5.)  # include <stdio.h>
      #include <conio.h>
Void  binary search (int arr[ ], int num, int first, int last ).

{ int mid;
    if (first > last)
  { printf (" Number is not found");
    }
   else
   { mid = (first + last)/2;
    }
   if (arr[mid] = = num).
  { printf(" elements is found at index %.d ", mid);
    exit(0);
    }
  else if ( arr[mid] > num)
 { primary search (arr, num, first mid -1);
     }
    else
    { Binary search (arr, num, mid+1, last);
      };
       }
        }
    Void main( )
  {   Int arr[100], beg, mid, end, i, n, num;
      printf(" Enter size of array")
       Scanf(" %.d", &n);
       printf(" Enter the value in sorted sequence \n");
```

```
for ( i = 0 ; i < n ; i++ )
  {
     scanf ("%d", &arr[i]);
  }
   beg = 0,
   end = n-1 ;
  printf (" Enter value to be searched");
   scanf (" %d", &num);
  Binary search (arr, num, beg, end);
 }
```

Output :

```
Enter size of array - 6
Enter value in sorted sequence.
  3
   9
   8
   4
   2
   7
 Enter value to be searched: 4
 Element is found at Index : 3
```