Assignment-6.

P. Jayanth Kumar
AP19110010332.

1)

```
#include <stdio.h>
Void main( )
{
 int array[10],sompla,propla;
int a, b, c, d, num, temp, keynum;
int small, cen, rise;
Printf("Enter value of sort \n");
Scanf ("%.d ", &num);
Printf("Enter the elements \n");
for(a=0; a<num, a++)
{
 scanf("%d",&Array[a]);
 }
 printf("Array elements \n");
 for(a=0; a<num, n++)
 {
 for(b=0; b<(num-a-1); b++)
 {
  if(array[b]<array[b+1])
  {
   temp= array[b];
   array[b]= array[b+1];
   array[b+1]= temp;
```

```
}
}
}
printf("The sorted array \n");
for(a=0;a<num,a++)
{
printf("%d \n", array[a]);
}
printf("Enter the element that is needed to be search \n")
scanf("%d", &keynum);
small = 1;
rise = num;
do
{
cen = (small + rise)/2
if (key num > array[cen])
rise = cen - 1;
else:
(key num > array[cen])
small = cen + 1;
}
while (key num != array[cen] && small <= rise);
if (key num <> array [cen]).
{
printf("search success and %d found location %d\n"
keynum, %d");
}
```

```
else
{
 printf("search failed \n");
}
 printf("Enter the location in sorted array \n");
 scanf("%d %d", &c, &d);
  c--;
  d--;
  for(a=0, a<sum, a++)
  {sumloc = array[z]+array[k];
   proloc = array[z] + array[k];
  }
  printf("\n sum of the location is %d", sumloc);
  scanf("\n product of location %d", propla);
  }
```

Output :

```
Enter the value of sort
 3
Enter the elements
2
4
5
the sorted array
 5
 4
 2
Enter the the element that need to be search
 4
search success 4 is found at location 2
```

Enter the location in sorted array

4

2

Sum of location 6

Product of location 8

2)

```
# include <stdio.h>
void mergsort(int array[], int i, int j);
void merg(int array[], int i, int j, int i2, int j2);
void main()
{
int array[40], n, i, k;
printf("Enter the value of sort");
scanf("%d", &n);
printf("Enter the value in array");
for(i=0, i<n, i++)
scanf("%d", &array[i]);
mergesort(array, 0, n-1);
  printf("In sorted arrays is");
  for(i=0; i<n; i++)
  printf("%d", array[i]);
  int prodb = 1, prod1 = 1;
  printf("Enter the value of k");
  scanf("%d", &k);
  k = k-1;
  for(i=0, i<=k; i++)
```

```
{
 prodb = prod * array[i];
}
 for(i=n-1; i>=k; i--)
 {
  prod1 = prod1 * array[i];
 }
  printf("\n the product from start is equal %d", prodb);
  printf ("\n the product from list is equal %d", prod1);
 }
  void mergesort(int array[ ],int i, int j)
  {
   int mid;
   if (i<j)
   {
    mid=(i+j)/2;
    mergesort (array, i, mid);
    mergesort (array, mid+1, j);
    merge (array, i, mid, mid+1, j);
   }
  }
  voidmerge(int array [ ], int i1, int j1, int i2, int j2).
   {
    int temp[50];
    int i,j,k;
    i = i1;
    j = j2;
    k = 0
```

```
while(i<=j)
 {
  mid=(i+j)/2;
  if(array[i]<array[j])
  temp[k++]=array[i++];
 else
 temp[k++]=array[j++];
  }
   while(i<=j)
 temp[k++]=array[i++];
  while(j<=j2)
 temp[k++]=array[i++];
  while(j<=j2)
   temp[k++]=array[j++];
   for(i=i,j=0,j<=j2 ,i++,j++)
   array[i]=temp[i];
  }
```

Output:

```
Enter the value of sort 5
Enter the value of array 3
 4
 3
 2
Sorted array is 1 2 3 5
Enter the value of k 2
product from start is equal to 2
 product from last is equal to 24.
```

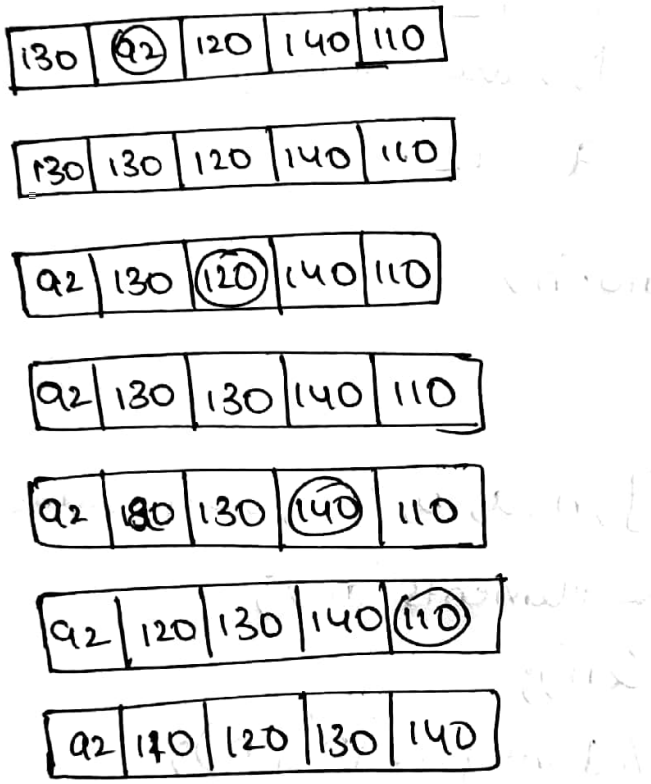
3) Insertion sort : Insertion sort in C is a sample & efficient algorithm, that creates the final sorted array one element at a time.

Insertion sort works in a similar manner as we arrange a deck of cards.

Avg & worst-case complexity of this algorithm is $O(n_2)$.

Insertion sort is not good for large data sets.

Eg : Initial array



Sorted array → 92 110 120 130 140

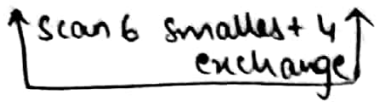
Selection sort :

In selection sort, the smallest element is exchange with the first element of the unsorted list of elements. Then the second smallest element is exchanged with the second element of the unsorted list of elements & so on

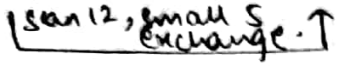
until all the elements are sorted.

Average worst case complexity of the algorithm is $O(n^2)$

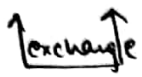
Eg: 6 12 9 4 5



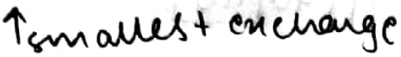
4 12 9 6 5



4 5 9 6 12



4 5 6 9 12



4 5 6 9 12

4)
```
# include <stdio.h>
int main ( )
{
 int array[100], n, a, b, i, m, swap, sum=0, prod=1;
 printf("Enter the elements \n");
 scanf ("%d", &n);
 printf ("Enter %d integers \n", n);
 for (a=0; a<n ; a++)
  scanf("%d", &array[a]);
 for(a=0; a<n-1; a++)
 {
  for (b=0; b<n-a-1; b++)
  {
   if (array[b] > array[b+1])
   {
```

```
swap = array[b];
array[b] = array[b+1];
array[b+1] = swap;
}
}
}
printf("sorted array in AC \n");
for(a=0;a<n,a++)
printf("%d \n",array[a]);
printf("Alternative series is");
for(i=0;i<n;i++)
{if(i%2 = 0)
{printf("%d",array[i]);
}
}
for(i=0, i<n; i++)
{if(i%2 != 0)
{sum = sum + array[i];
}else
{proo = proo + array[i];
}
}printf("\n sum in odd position is %d",sum0);
```

```
printf("\n product in even %d", pro0);
printf("\n Enter the value");
 scanf ("%d", &m);
  for (i=0; i<n ; i++)
  {
   if (array[i] %. m == 0)
   {
    printf("%d", array[i]);
   }
  }
  }
```

Output :

```
Enter the element 5
Enter 5 integers
5
4
3
2
1
Sorted array in AD
1
2
3
4
5
The alternative series 1 3 5
sum is odd i.e 9
product is even is 8
Enter the value
2
2 4.
```

5)
```
#include <stdio.h>
#include <stlib.h>
int Binary search(int arr[],int num, int first, int last)
{
 if (first>last)
 Printf("number you have entered is not found");
 }
  else
   {
    int mid;
    mid = (first+last)/2;
    if (arr[mid] == num)
    {
    Printf("Element you have asked for is found at index
                                          %d", mid);
    exit(0);
    }
    else if(arr[mid]>num)
    {
     Binary search(arr, num, first, mid -1);
     }else
       {Binary search(arr,num,mid+1,last);
     }
     }
     }
     int main()
     {
```

```
int arr[] = {110,140,160,180,120};
int num = 140;
int first = 0, last = (size of (arr)/size of (arr[0])) - 1;
 Binary search (arr, num, first, last);
}
```

Output :

Element you have asked for is found at index 1.