G. Harsha Vardhan
CSE-G
AP19110010554

1) Recursive program to implement binary search.

Code:

```
#include <stdio.h>
#define SIZE 100
int binary_search (int a[], int ele, int first, int last)
{
   int mid;
   mid = (first + last)/2;
   if (ele == a[mid]){
        return mid;
    }
   else if (ele < a[mid])
   {
       binary_search (a, ele, first, mid-1);
   }
   else
   {
       binary_search (a, ele, mid+1, last);
   }
}
void main()
{
    int a[size], ele, n, i, x=0;
    Printf("\n number of elements in array:");
```

```
scanf("%d", &n);
printf("\n enter elements:");
for (i=0 ; i<n ; i++)
{
    scanf("%d", &a[i]);
}
printf("\n enter searching element:");
scanf("%d", &ele);
x = binary_search(a, ele, 0, n-1);
printf("\n element found at %d Position",
                                x+1);
}
```

Output:

```
number of elements in array: 3
enter elements: 1
2
3
enter searching element: 2
element found at 2 Position.
```

2) Discuss insertion and Selection sort.

Ans:

Basis	Insertion sort	Selection sort
Basic	The data is sorted by inserting the data into an existing sorted file.	The data is sorted by selecting and placing the consecutive elements in sorted location.
Nature	Stable	unstable.
Process to follow	Elements are known beforehand while loop to place them is searched	Location is previously known while elements are searched.
Immediate data	Insertion sort is fine sorting technique which can deal with immediate data	It can not deal with immediate data, it needs to be present at the begining.
worst case complex	$O(n^2)$	$O(n^2)$

Insertion sort: It works by inserting set of values in existing sorted file. It constructs the sorted array by inserting a single element at a time. This process continous until whole array is sorted in same order.

Ex: [25] 15 30 9 99 20 26

[15] [25] 30 9 99 20 26

15	25	30	9	99	20	26
9	15	25	30	99	20	26
9	15	25	30	99	20	26
9	15	29	25	30	99	26
9	15	29	25	26	30	99

Selection sort: It Perform Shorting by searching for the minimum value number and placing it into the first or last position according to order. The process of searching the mini. key and placing it in proper position is continued until the all elements are placed at right position.

Ex:

0	1	2	3	4
17	16	3	13	6

① | 17 | 16 | 3 | 13 | 6 |
min Loc

②

3	16	17	13	6

r min Loc

③

3	6	17	13	16

min Loc

④

3	6	13	17	16

min Loc

⑤

3	6	13	16	17

(3) Bubble Sort :

Code:

```
#include <stdio.h>
int Bubble_Sort(int size, int *array)
{
    int i, j, temp;
    for (i = size -2; i >= 0; i--)
    {
        for (j=0; j<=i ; j++)
        {
            if (array(j) > array(j+1)
            {
                temp = array[j];
                array(j]= array[j+1]
                array[j+1] = temp;
            }
        }
    }
    return 1;
}
int main(void)
{
    int size, i, array[50], s=0, m=1, x;
    Printf("enter no. of elements in array:");
    Scanf("%d", &size);
```

```
Printf ("enter the elements in array:", size);
for (i = 0; i<size ; i++)
{
    Scanf ("%d", &array[i]);
}
Bubble_Sort (size, array);
Printf ("After the Sorted:");
for (i=0 ; i<size ; i++)
{
    Printf ("%d", array[i]);
}
Printf ("\n");
printf ("alternate elements \n");
for (i =0 ; i<size ; i++)
{
    printf ("%d", array [i++]);
}
Printf ("\n");
printf ("Sum of elements in odd position and
        Product of elements in even position \n");
for (i=0; i<size; i++)
{   if (i %. 2 == 0)
    {
        m = m * array [i]
    }
```

④

```
    else
    {
        s = s + array[i]
    }
}
Printf("Sum of elements in odd position : %d", s);
Printf("Product of elements in even position : %d", m);
Printf("enter x value: ");
Scanf("%d", &x);
for(i=0; i<size; i++)
{
    if(array[i] % x == 0)
    {
        Printf("%d", array[i]);
    }
}
Printf("\n");
return 0;
}
```

Output:

```
enter no. of elements in array : 7
Enter the elements in array : 1
2
```

```
}
4
5
6
7
```

After the sorted; 1 2 3 4 5 6 7

Alternate elements

1 3 5 7

Sum of elements in odd position and Product of Elemet

In even position

Sum of elments in odd position : 16

Product of elements in even position: 48

enter x value: 2

```
2
4
6
```

---

4) Take elements from the user and sort them in descendo order and do following.

Code:

```
#include <stdio.h>
Void descending()
{
    int array[100], Value, L1, L2, swap, x;
    Printf("Enter value:");
    Scanf("%d", & value);
```

⑤

```
for (L1 = 0; L1 < value ; L1 ++)
{
    printf ( "Value - %d :", L1+1 )
    Scanf ("%d", &array [L1]);
}
for (L2 = 0 ; L2 < (value -1) ; L2 ++)
{
    for (L1=0 ; L1 < (value-1); L1 ++)
    {
        if (array [L1+1] < array [L1])
        {
            swap = array [L1];
            array [L1] = array [L1+1];
            array [L1+1] = swap;
        }
    }
}
printf ( "Descending order : \n");
for (L1 = value ; L1 >0; L1 --)
{
    printf ("%d", array [L1-1]);
}
return 0;
}
```

```
Void binary_search()
{
    int i, first, last, mid, n, search, arr[100];
    Printf ("enter no. of elements \n");
    Scanf ("%d", &n);
    Printf ("Enter Elements\n", n);
    for (i=0; i<n; i++)
    {
        Scanf ("%d", &arr[i]);
    }
    printf ("enter search element \n");
    Scanf ("%d", &search);
    first = 0;
    last = n-1;
    mid = (first + last)/2;
    While (first < last)
    {
        if (arr[mid] < search)
        {
            first = mid + 1;
        }
        else if (arr[mid] == search)
        {
            printf ("% found at location %d \n", search,
                                                mid+1);
            break;
        }
```

```
        else
        {
            last = mid - 1;
            mid = (first + last)/2;
        }
        if (first > last)
        {
            Printf ( " not found! %d is not present
                                    in list. \n", search);
        }
    }
}
Void SumProduct()
{
    Int a, b, Sum, Product;
    Printf ( " enter location :")
    Scanf ( "%d", &a)
    Scanf ( "%d", &b);
    Sum = arr[a] + arr[b];
    Product = arr[a]* arr[b];
    Printf ( " Sum = %d", sum);
    Printf ( " Product = %d", Product);
}
Void descending()
Void binary_search()
```

```
Void Sumproduct( )
main( )
{
   int choice;
   while (1)
   {
      Printf( "1. Decending order \n");
      Printf ("2. searching element in array");
      Printf ("3. Product & Sum \n");
      Printf ("4. quit \n");
     Printf ("Enter choice :");
     Scanf ( "%d ", &choice);
     Switch (choice)
     {
       case 1 :
       {
          decending ()
          break;
       }
       case 2;
       {
         binarySearch ()
         break;
       }
       case 3 ;
       {
          Sumproduct ()
          break;
       }
```

```
        case 4:
        {
          exit(1);
        }
        default:
        {
            Printf( " incorrect choice \n");
        }
      }
    }
  }
```

Output:

1. Decending order
2. Searching element in array
3. Product & Sum
4. Quit

Enter choice : 1

Enter value : 4

Value - 1 : 2

Value - 2 : 4

Value - 3 : 1

Value - 4 : 3

Descending order : 4 3 2 1

Enter Choice : 2

Enter no. of elements : 4

Enter elements:

4

3

2

1

Enter search element.

2

2 found at location 3.

Enter Choice : 3

Enter location : 2

3

Sum = 5

Product = 6

Enter Choice : 4

Exit.

5) Meargesort:

Code:

```
# include <stdio.h>
# define size 100
int arr1 [size]
int arr2 [size]
Void merge (int first, int mid, int last)
{
   int i, j, k, l;
   For (i = first ; j = mid+1 ; k = first ; i < mid ee
                                        j <= last ; k++)
   {
      if (arr1 [i] < arr2 [j]
      {
         arr2 [k] = arr1[i++];
      }
      else
      {
         arr2[k] = arr1 [j++];
      }
   }
   while (i <= mid)
   {
      arr2 [k++] = arr1 [i++];
   }
```

```
        while ( j <= last)
        {
            arr2[k++] = arr1[j++];
        }
        for (int l=0; l < last+1 ; ++l)
        {
            arr1[l] = arr2[l];
        }
    }
Void Sort (int first, int last)
{
    if (first < last)
    {
        int mid = (first + last)/2;
        Sort(first, mid);
        Sort (mid+1, last);
        mearge (first, mid, last);
    }
    else
    {
        return;
    }
}
int main(void)
{
    int n;
```

```
Printf ("\n enter no.ifelements");
Scanf ("%d", &n);
Printf ("\n enter the elements");
PrTntf ("\n enter the elements", n);
for (int i=0; i<n; i++)
{
    Scanf ("%d", & arr[i]);
}
Printf ("\n array after sorted.: ");
Sort (0, n-1);
for (int i=0; i<n; i++)
{
    Printf ("%d", arr [i]);
}
int k, x=1;
Printf ("enter k value \n");
Scanf ("%d", &k);
for (i=0; i<k; i++)
{
    x=x* i
}
Printf ("Product of k element is %d", x);
}
```

Output:

```
Enter no. of elmts. 4
Enter elements :
3
4
2
Array after Sorted : 2 3 4
enter k value : 2
Product of k element is 4
```