Ans 1)

Assignment – 6

N. Aasrith Varma
AP19110010528
CSE - H

```
#Include <Stdio.h>
Void binary-search (int[], int, int, int);
Void bubble-Sort (int[], int);

int main ()
{
   int Value, len, i, a, b; Sum, Prod;
   int list [80];

  Printf ("Enter length of the list: ");
   Scanf ("%d", &len);
   Printf ("Enter elements \n");
   for(i=0; i<len; i++)
   {
    Scanf ("%d", &list[i]);
   }
   bubble_Sort (list, len);
   Printf ("\n");
   Printf ("Enter Value to \n");
   Scanf ("%d", &Value);
   binary-search (list, 0, len, Value);
```

```
}
Void bubble_sort (int list[], int len)
{
 int temp, i, j, sum, Prod, a, b;
 for (i=0; i<len; i++)
 {
   for (j= i; j< len; j++)
   {
      if (list[i] > list[j])
      {
         temp = list[i];
         list[i) = list [j];
         list [j] = temp;
      }
   }
}
   Printf ("sorted array is : \n");
   for (i=0; i< len; i++)
        Printf ("%d\t", list[i]);
 Printf ("\n Enter the first Position : \n");
  Scanf ("%d", &a);
 Printf ("\n Enter the second Position : \n");
 Scanf ("%d", &b);
```

```
sum = list[a] + list[b]
Printf("\n Sum of two numbers is : %d", sum);

Prod = list[a] * list[b];
Printf("\n Product of two numbers is : %d", Prod);
}

Void binary_search (int list[], int x, int y, int Value)
{
  int mid;
  if (x>y)
  {
    Printf("Value not found \n");
    return;
  }
  mid = (x+y)/2;
  if (list[mid] == Value)
  {
      Print("Value found \n");
  }
else if (list(mid) > Value)
{
   binary_search (list, x, mid-1, Value);
}
else if (list[mid] < Value)
```

```
    {
        binary_search (list, mid + 1, y, value);
    }
}
```

2)

```
#include <stdio.h>
Void merge_sort (int a[], int i, int j);
Void merge (int a[], int i1, int j1, int i2, int j2);

int main ()
{
    int arr[70], n, i, Prod;
    Printf ("Enter number of elements in an array : \n");
Scanf ("%d", &n);
    Printf ("Enter elements in array ; ");
    for (i=0; i<n ; i++)
{
    Scanf ("%d", &arr[i]);
}
    merge_sort (arr, 0, n-1)
    Printf ("\n Sorted array is : ");
```

```
for (i=0; i<n; i++)
{
  Printf("%d\t", arr[i]);
}
Printf ("Enter the Value of k less than %d: ", n);
Scanf ("%d", &k);
Prod = arr[k] * arr[n-k];
Printf ["\n Product of two elements is %d", Prod);
}
Void merge-sort (int arr[], int i, int j)
{
  int mid;
   if (i<j)
  {
   mid = (i+j)/2;
  merge-sort (arr, i, mid);
  merge-sort (arr, mid+1, j);
  merge (arr, i, mid, mid+1, j);
  }
}
```

```
Void merge ( int arr [], int i1, int j1, int i2, int j2 )
{
  int temp [100]
    i = i1 , j = i2 , k = 0;
  while (i <= j1 && j <= j2
  {
     if ( arr [i] < arr [j] )
    {
       temp [k++] = arr [i++] ;
    }
     else
     {
       temp [k++] = arr [j++] ;
    }
  while (i <= j1)
   {
   temp [k++] = arr [j++]; }
  for (i = i1 , j = 0 , i <= j2 , i++ , j++)
{
  arr [i] = temp [j];
}
```

3) Insertion Sort:-

Insertion Sort is a sorting algorithm where the array is sorted by taking one element at a time. The principle behind insertion sort is to take one element iterate through the sorted array and find its correct position in sorted array.

Algorithm:-

1): If the element is first one it is already sorted
2) Move the next element
3) Compare the current element with all the elements in the sorted array
4) If the element in the sorted array is smaller than the current element, iterate the next element. otherwise, shift all the greater element in the array by one position towards right.
5) Insert the value at the current position
6) Repeat until the complete list is sorted

Selection Sort:-

Selection Sort is simple among all sorting techniques. It works by selecting the smallest element in the array and placing it at the head of the array. The next largest element is selected and put into the next slot, and so down the line. Because a

selection sort looks at progressively smaller parts of array each time, a selection sort is slightly faster than bubble sort.

Example

For sorting 63425 first, 3 is inserted before 6, resulting 36425 then, 4 is inserted between 3 and 6. which gives [redacted], 34625, 2 is inserted at begining ~~We get~~ ~~2 3 6 4~~, 5 we get 2 3 4 6 5, 5 is inserted between 4 and 6, We get finally 2 3 4 5 6

Time Complexity

$O(n^2)$ as there are two nested loops.

Ans 24)

```
# include <stdio.h>
int main ( )
{
int arr[60], n, i, j, temp, sum = 0, prod = 1, r;
Printf ("Enter number of elements in array \n");
Scanf ("%d", &n);
for (i=0; i<n; i++)
{
  Scanf ("%d", arr[i]);
}
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("Sorted array in ascending order is \n");
for(i=0; i<n; i++)
  {
    Printf("%d \n", arr[i]);
}
 Printf("Sorted array in alternate order is \n");
for(i=0; i<n; i=i+2)
{
  Printf("%d \n", arr[i]);
}
Printf("sum of all elements in odd Position are: \n");
for(i=0; i<n; i=+2)
{
  Sum = sum + arr[i];
   Printf("%d \n", Sum);
}
Printf("Product of all elements in even Position: \n");
for(i=1; i<n; i=i+2)
{
    Prod = Prod * arr[i];
    Printf("%d \n", Prod);
}
```

```
}
Printf("Enter a number: ");
Scanf("%d", &k);
Printf("Elements divisible by %d are: \n", k);
for(i=0; i<n; i++)
  {
    if arr[i]%k == 0)
  {
    Printf("%d\n", arr[i]);
  }
}
return 0;
}
```

~~Ans 5~~

5)
```
# include <stdio.h>
Void binary(int [], int, int, int);
Void sorting [int[], int);
int main () {
    int num, len, i;
```

```
int arr[200];
Printf("Enter length of array : ");
Scanf("%d", &len);
Printf("Enter elements for array \n");
for(i=0; i<len; i++)
{
    Scanf("%d", &arr[i]);
}
sorting(arr, len);
Printf("Enter number to search : ");
Scanf("%d", &num);
binary(arr, 0, len, num);
}
Void sorting(int arr[], int len)
{
int temp, i, j;
for(i=0; i<len; i++)
{
    for(j=1, j<len; j++)
    {
      if(arr[i] > arr[j])
      {
```

```
            temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
    }
}
Void binary (int arr[], int a, int b, int num)
{
  int mid;
  if(a>b)
  {
    Printf("number not found");
  }
mid = (a+b)/2;
  if(arr[mid] == num)
  {
    Printf("number found");
  }
else if(arr[mid]>num)
  {
    binary(arr, a, mid-1, num);
  }
```

```
else if (arr[mid] < num)
{
    binary (arr, mid+1, b, num);
}
}
```