Assignment - 6

K. Naineeka
AP19110010134
CSE-G.

1. Take the elements from the user and sort them in descending order and do the following.

(a) using binary search find the elements and the location in the array where the element is asked from user

(b) Ask the user to enter any two location print the sum and product of values at those locations in the sorted array

Codes

(a)
```
# include<stdio.h>
void binary_search();
int a[50], n, item, loc, beg, mid, end, i;
void main()
{
    printf("Enter size of an array:");
    scanf("%d", &n);
    printf("\n Enter elements of an array in sorted
            form:\n");
    for (i=0; i<n; i++)
        scanf("%d", &a[i]);
    printf("\n Enter ITEM to be searched:");
```

```
    binary_search();
    getch();
}
void binary_search()
{
    beg=0;
    end=n-1;
    mid=(beg+end)/2
    while((beg<=end)&&(a[mid]!=item))
    {
        if(item<a[mid])
            end=mid-1;
        else
            beg=mid+1;
        mid=(beg+end)/2
    }
    if(a[mid]==item)
        printf("\n\n ITEM found at location %d",mid+
    else
        printf("\n\n ITEM doesn't exist");
}
```

Output :-

Enter size of an array : 5

Enter elements of an array in sorted form:

12

16

18

20
25

Enter ITEM to be Sorted Searched : 18

ITEM found at location 3

(6)

```
#include<stdio.h>
int main()
{
    int arr[5];
    int sum, product, i;
    printf("\n Enter elements:\n");
    for(i=0; i<5; i++)
    {
        printf("Enter arr[%d]:", i);
        scanf("%d", &arr[i]);
    }
    sum=0;
    product=1;
    for(i=0; i<5; i++)
    {
        sum=sum+arr[i];
        product=product*arr[i];
    }
    printf("\n sum of array is: %d", sum);
    printf("\n product of array is: %d\n", product);
    return 0;
}
```

```
}
void mergesort(int arr[], int l, int r)
{
  if(l<r)
  {
   int m: l+(r-1)/2
   mergesort(arr, l, m);
   mergesort(arr, m+1, r);
   mergesort(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 arr[] = {12, 11, 13, 5, 6, 7};
  int arr_size = sizeof(arr)/sizeof(arr[0]);
  Printf("Given array is \n");
```

```
printArray(arr, arr_size);
mergeSort(arr, 0, arr_size-1);
printf("\n Sorted array is \n");
printArray(arr, arr_size);
return 0;
}
```

Output:-

Given array is

12 11 13 5 6 7

Sorted array is

5 6 7 11 12 13

3. Discuss insertion sort and Selection sort with examples.

Insertion Sort:-

Insertion sort is the sorting mechanism where the sorted array is built having one item at a time. the array elements are compared with each other sequentially and then arranged simultaneously in some particular order. the analogy can be understood from the style we arrange a deck of cards. this sort works on the principle of inserting an element at a particular position, hence the name insertion sort.

Code:-

```
#include<math.h>
#include<stdio.h>
void insertionsort(int arr[], int n)
{
  int i, key, j;
  for(i=1; i<n; i++){
    key=arr[i];
    j=i-1;
    while(j>=0 && arr[j]>key){
      arr[j+1]=arr[j];
      j=j-1;
    }
    arr[j+1]=key;
  }
}
void printArray(int arr[], int n)
{
  int i;
  for(i=0; i<n; i++)
    printf("%d", arr[i]);
  printf("\n");
}
int main()
{
```

```
int arr[] = {12, 11, 13, 5, 6};
int n = sizeof(arr) / sizeof(arr[0]);
insertionSort(arr, n);
printArray(arr, n);
return 0;
}
```

output:-

5 6 11 12 13

Selection Sort:-

Selection sort is another algorithm that is used for sorting. The sorting algorithm iterates through the array and finds the smallest number in the array and swaps it with the first element if it is smaller than the first element. Next, it goes to the second element and so on until all elements are sorted.

Code:-

```
#include <stdio.h>
void swap(int *a, int *b){
  *a = *b;
  *b = temp;
}
```

```
void selectionSort(int array[], int size){
  for(int step=0; step< size-1; step++){
    int min_idx=step;
    for(int i=step+1; i< size; i++){
      if(array[i] < array[min_idx])
        min_idx=i;
    }
    swap(&array[min_idx], &array[step]);
  }
}

void printArray(int array[], int size){
  for(int i=0; i<size; i++){
    printf("%d", arr[i]);
  }
  printf("\n");
}

int main(){
  int data[] ={20, 12, 10, 15, 2};
  int size = sizeof(data)/ sizeof(data[0]);
  SelectionSort(data, size);
  printf("sorted array in Ascending order: \n");
  printArray(data, size);
}
```

output:-

Sorted array in Ascending order:

2 10 12 15 20

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

(i) in alternate order.

(ii) sum of elements in odd position and product of elements in even positions.

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

code:-

(i)
```
#include<stdio.h>
#include<math.h>
int main()
{
  int a[]={16,19,11,15,10,12,14};
  int i,j;
  for(j=0;j<7;j++)
  {
    int swapped=0;
    i=0;
    while(i<7-1)
    {
      if(a[i]>a[i+1])
      {
```

```
            int temp = a[i];
            a[i] = a[i+1];
            a[i+1] = temp;
            swapped = 1;
        }
        i++;
    }
    if(!swapped)
        break;
}
for(i=0; i<7; i++)
    printf("%d\n", a[i]);
return 0;
}
```

Output:-

10
11
12
14
15
16
19.

Code:-

(ii)

```
#include<stdio.h>
#include<conio.h>
int main()
{
    int num, evensum=0, oddprod=1, rem, temp;
    printf("Enter any number:");
    scanf("%d", &num);
    while(num>0)
    {
        rem=num%10;
        if(rem%2==0)
            evensum=evensum+rem;
        else
            oddprod=oddprod*rem;
        num=num/10;
    }
    printf("\n Sum of even digit = %d", evensum);
    printf("\n product of odd digit= %d", oddprod);
    getch();
    return 0;
}
```

output:-

Enter any number: 2 4 3 7 6 9

Sum of Even digit = 2

product of odd digit = 1

(iii) code:-

```
#include <stdio.h>
void swap(int *xp, int *yp)
{
  int temp = *xp;
  *xp = *yp;
  *yp = temp;
}
void bubbleSort(int arr[], int n)
{
  int i, j;
  for (i=0; i<n-1; i++)
    for(j=0; j<n-i-1; j++)
      if (arr[j] > arr[j+1])
        swap(&arr[j], &arr[j+1]);
}
void printArray(int arr[], int size)
{
  int i;
  for (i=0; i<size; i++)
    printf("%d", arr[i]);
  printf("\n");
}
int main()
{
  int arr[] = {64, 34, 25, 12, 22, 11, 90};
```

```
int n=sizeof(arr)/sizeof(arr[0]);
bubbleSort(arr,n);
printf("sorted array:\n");
printArray(arr,n);
return 0;
}
```

output:-

Sorted array:

11 12 22 25 34 64 90.

5. write a recursive program to implement binary search?

code:-

```
#include<stdio.h>
void binary_search(int[],int,int,int);
void bubble_sort(int[],int);
int main()
{
int key, size, i;
int list[25];
printf("Enter size of a list:");
scanf("%d",&size);
printf("Enter elements\n");
for(i=0;i<size;i++)
{
scanf("%d",&list[i]);
}
```

```
    bubble_sort(list, size);
    printf("\n");
    printf("Enter key to search\n");
    scanf("%d", &key);
    binary_search(list, 0, size, key);
}
void bubble_sort(int list[], int size)
{
  int temp, i, j;
  for(i=0; i<size; i++)
  {
    for(j=i; j<size; j++)
    {
      if(list[i] > list[j])
      {
        temp = list[i];
        list[i] = list[j];
        list[j] = temp;
      }
    }
  }
}
void binary_search(int list[], int lo, int hi,
                   int key)
{
  int mid;
```

```
if (lo > hi)
{
  printf("key not found\n");
  return;
}
mid = (lo+hi)/2
if(list[mid] == key)
{
printf("key found\n");
}
else if (list[mid] > key)
{
binary_search(list, lo, mid-1, key);
}
elseif(list[mid] < key)
{
binary_search(list, mid+1, hi, key);
}
}
```

output :-

```
Enter size of a list: 4
Enter elements 1 3 5 7
Enter key to search
9
key not found.
```