Assignment-6

B. Jayanth
AP19110010394
CSE-F

1. Take the elements from the user and sort them in descending order and do the following
(a) Using Binary search find the element and the location in the array where the element is asked from User
(b) Ask the user to enter any two locations print the sum and product of values at those Locations in the sorted array

```
#include<stdio.h>
int main()
{
int a[10],n,i,j,temp,se,found=0,top,mid,bot,one,two,sum,prod;
printf("enter the number of variables to be used: ");
scanf("%d",&n);
for(i=0;i<n;i++)
{
    printf("enter the value of a[%d]",i);
    scanf("%d",&a[i]);
}
for(i=0;i<n-1;i++)
{
    for(j=0;j<n-i-1;j++)
    {
        if(a[j]<a[j+1])
        {
            temp=a[j];
            a[j]=a[j+1];
            a[j+1]=temp;
        }
    }
}
```

```
printf("After sorting\n");
for(i=0;i<n;i++)
{
    printf("a[%d] = %d\n", i, a[i]);
}
printf("enter the searching element\n");
scanf("%d", &se);
top=0;
bot=n-1;
while(top<=bot)
{
    mid=(top+bot)/2;
        if(a[mid]==se)
        {
            found=1;
            break;
        }
        else if(a[mid]>se)
        {
            bot=mid-1;
        }
        else if(a[mid]<se)
        {
            top=mid+1;
        }
}
if(found==1)
{
    printf("element found at %d position\n", mid);
}
else
{   printf("element not found\n");
}
```

```
printf("enter two locations to find sum and product");
scanf("%d", &one);
scanf("%d", &two);
sum = (a[one] + a[two]);
prod = (a[one] * a[two]);
printf("The sum of elements is %d", sum);
printf("The product of elements is %d", prod);
return 0;
}
```

Output :-

enter the number of variables to be used: 5
enter the value of a[0] 100
enter the value of a[1] 700
enter the value of a[2] 500
enter the value of a[3] 400
enter the value of a[4] 600
After sorting
a[0] = 700
a[1] = 600
a[2] = 500
a[3] = 400
a[4] = 100
enter the searching element 400
element found at 3 position
enter two locations to find sum and product 0
3
The sum of elements is 1100
The product of elements is 280000

2. Sort the array using Merge sort where elements are taken from the user and find the product of kth elements from first and last where k is taken from user

```
#include<stdio.h>
#include<stdlib.h>
void merge(int a[], int l, int m, int r)
{
    int i,j,k
    int n1 = m-l+1;
    int n2 = r-m;
    int L[n1], R[n2];
    for(i=0;i<n1;i++)
        L[i] = a[l+i];
    for(j=0;j<n2;j++)
        R[j] = a[m+1+j];
    i=0;
    j=0;
    k=l;
    while(i<n1 && j<n2)
    {
        if(L[i]<=R[j])
        {
            a[k] = L[i];
            i++;
        }
        else
        {
            a[k] = R[j];
            j++;
        }
        k++;
    }
```

```
    while(i<n1)
    {
        a[k] = L[i];
        i++;
        k++;
    }
    while(j<n2)
    {
        a[k] = R[j];
        j++;
        k++;
    }
}
void mergeSort(int a[], int l, int r)
{
    if(l<r)
    {
        int m = l+(r-l)/2;
        mergeSort(a, l, m)
        mergeSort(a, m+1, r)
        merge(a, 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 siz, v;
```

```
printf("Enter array size ");
scanf("%d", &siz);
int val[siz];
for(v=0; v<siz; v++)
{
    printf("Enter value: ");
    scanf("%d", &val[v]);
}
printf("Given array is\n");
printArray(val, siz);
mergeSort(val, 0, siz-1);
printf("\nSorted array is\n");
printArray(val, siz);
int k, f, l, p1, p2, temp;
printf("Enter the value of k to find the product of
        elements from first and last: ");
scanf("%d", &k);
p1 = p2 = 1;
for(f=0; f<=k; f++)
{
    temp = val[f];
    p1 *= temp;
}
for(l=siz-1; l>=k; l--)
{
    temp = val[l];
    p2 *= temp;
}
printf("Product of Kth element from first and
        last are: %d %d", p1, p2);
}
```

output:-

```
Enter array size: 3
Enter Value: 1
Enter value: 2
Enter Value: 3
Given array is
1 2 3
Sorted array is
1 2 3
Enter the value of K to find the product of elements fro
first and last : 2
Product of Kth element from first and last are: 6 3
```

3. Discuss Insertion and selection sort with examples

Insertion sort is a simple sorting algorithm that builds the final sorted array one item at a time. It is much less efficient on large lists than more advanced algorithms such as quicksort, heapsort, or merge sort

Worst Complexity: n^2

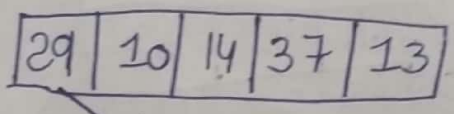
Avg Complexity: n^2

Best Complexity: n

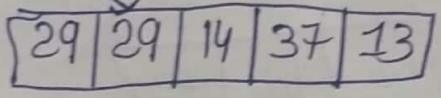
Space Complexity: 1.

Example

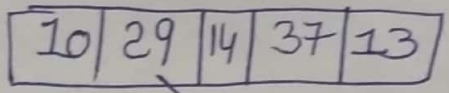
Initial array:-



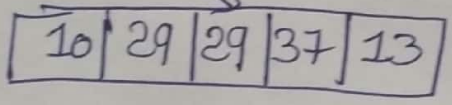
Copy 10



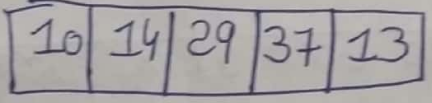
shift 29



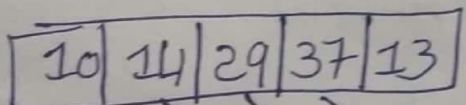
Insert 10, Copy 14



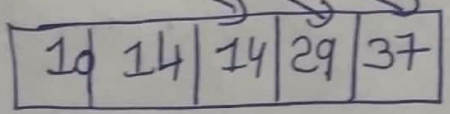
shift 29



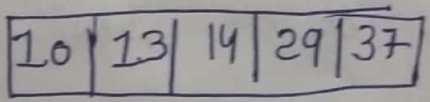
Insert 14, Copy 37, insert 37 on top of itself



Copy 13



Shift 37, 29, 14

Sorted array 

Insert 13

Selection sort:- This algorithm will first find the smallest element in the array and swap it with the element in the 1st position. Then it will find the second smallest element and swap it with the element in the 2nd position

and it will keep on doing this entire array is sorted. It is called selection sort because it repeatedly selects the next-smallest element and swaps it onto the right place.

Example:-

Index	0	1	2	3	4	5	6	7
1st Pass	27	63	1	72	64	58	14	9
2nd Pass	1	63	27	72	64	58	14	9
3rd Pass	1	9	27	72	64	58	14	63
4th pass	1	9	14	72	64	58	27	63
5th pass	1	9	14	27	64	58	72	63
6th pass	1	9	14	27	58	64	72	63
7th pass	1	9	14	27	58	63	72	64
8th pass	1	9	14	27	58	63	64	72

Worst Complexity: n^2
Avg Complexity: n^2
Best Complexity: n^2
Space Complexity: 1

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

(i) in alternate order

(ii) Sum of elements in odd positions and product of elements in even positions

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

```
#include<stdio.h>
void bubbleSort(int a[], int n)
{
    int i, j, temp;
    for(i=0; i<n-1; i++)
```

```
for(j=0;j<n-i-1;j++)
    if(a[j]>a[j+1])
    {
        temp=a[i]
        a[j]=a[j+1]
        a[j+1]=temp;
    }
}
int main()
{
    int x,i;
    printf("enter the no of variables to be used:");
    scanf("%d",&x);
    for(i=0;i<x;i++)
    {
        printf("enter the value of a[%d]",i);
        scanf("%d",&a[i]);
    }
    printf("Enter your choice");  printf("***MENU***\n");
    printf("1. Alternate order\n");
    printf("2. Sum of elements in odd position and Product of
                                    elements in even position\n");
    printf("3. Divisible by m\n");
    int ch,sum=0,product=1,m;
    printf("Enter one of option:");
    scanf("%d",&ch);
    switch(ch);
    {
        case1:
        for(i=0;i<x;i+=2)
```

```
{
    printf("%d\t", a[i]);
}
case 2:
for(i=0; i<x; i+=2)
{
    sum = sum + a[i];
}
for(i=0; i<x; i+=2)
{
    product = product * a[i];
}
printf("The sum is %d\n", sum);
printf("The product is %d\n", product);
case 3:
printf("Enter the value of m: ");
scanf("%d", &m);
printf("Numbers divisible by %d are:\n", m);
for(i=0; i<x; i++)
{
    if(a[i] % m == 0)
    {
        printf("%d\t", a[i]);
    }
}
}
```

Output:-

```
enter the no. of variables to be used: 5
enter the value of a[0] 5
enter the value of a[1] 3
enter the value of a[2] 4
enter the value of a[3] 7
enter the value of a[4] 1
***MENU***
1 Alternate orders
2. Sum of elements in odd position and Product of elements in even position
3 Divisible by m
Enter one of option : 1
5 4 1
***MENU***
1. Alternate orders
2. Sum of elements in odd position and Product of elements in even position.
3. Divisible by m
Enter one of option: 2
sum: The sum is 10
The product is 21
***MENU***
1. Alternate orders
2. Sum of elements in odd position and Product of elements in even position.
3. Divisible by m
Enter one of option 3
Enter the value of m: 4
Numbers divisible by 4 are:
4
```

9. Write a recursive program to implement binary search?

```
#include<stdio.h>
int binarysearch(int a[], int l, int h, int k)
{
    int m;
    if(l>h)
        return -1;
    m=(l+h)/2;
    if(k==a[m])
        return m;
    else if(k<a[m])
        binarysearch(a,l,m-1,k);
    else
        binarysearch(a,m+1,h,k);
}
void main()
{
    int a[50], n,i, k;
    printf("Enter the number of variables to be used: ");
    scanf("%d", &n);
    printf("Enter the %d elements\n", n);
    for(i=0; i<n; i++)
    scanf("%d", &a[i]);
    printf("enter the searching element: ");
    scanf("%d", &k);
    i = binarysearch(a, 0, n-1, k);
    if(i!=-1)
        printf("\n element found at index %d", i);
    else
        printf("\n element not found");
}
```

output:-

```
Enter the number of variable to be used : 4
Enter the 4 elements
100
200
300
400
enter the searching element : 300
element found at index 2
```