P. Hema Sri Harshitha
AP19110010130
CSE-G

ASSINGMENT - 6

1.
```
# include < Stdio.h>
Void main ()
{
int array [10], Sumloc, Proloc;
int i, j, z, k, num, temp, Keynum;
int low, mid, high;
Printf ("Enter the value of Sort \n");
Scanf ("%d", &num);
Printf (" Enter the elements \n");
for (i=0, i<num; i++)
{
Scanf ("%d", &array [i]);
}
Printf ("Input array elements \n");
for (i=0, i<num; i++)
{
printf ("%d \n", array [i]);
}
for (i=0; i<num, i++)
{
for (j=0; j<(num-i-1); j++)
{
if (array [j] < array [j+1])
{
temp = array [j];
array [j] = array [j+1];
array [j+1] = temp;
}
}
}
Printf ("The Sorted array \n");
for (i=0; i<num; i++)
{
Printf ("%d \n", array [i]);
}
Printf ("Enter the element that need to be Searched \n");
Scanf ("%d", & Keynum);
low = 1;
high = num;
do
{
mid = (low + high)\2;
if (Keynum > array [mid])
high = mid - 1;
else if (Keynum > array [mid])
low = mid + 1;
}
```

```
While (Keynum! = array [mid] && low <=high);
if (Keynum == array [mid])
{
Printf ("Search Successful and %d found at location %d
                                        \n", keynum, mid+1);
}
else
{
printf ("Search failed \n");
}
Printf ("Enter the location in the Sorted array \n");
Scanf ("%d ", &z, &k);
 z--;
 K--;
for (i=0; i<num; i++)
{
Sumloc = array [z] + array [k];
Proloc = array [z] * array [K];
}
Printf ("\n Sum of the locations is %d," Sumloc);
Printf ("\n Product of the locations is %d," Proloc);
}
```

Output

```
Enter the Value of Sort
4
Enter the elements
1
2
3
4
The Sorted Array
4
3
2
1
Enter the element that needed to be Searched
3
Search Successful 3 is found at location 2
Enter the location in the Sorted array
2
3
Sum of the Location is 5
Product of the location is 6
```

2.
```
# include < stdio.h>
void mergesort (int array [], inti, intj);
void merge (int array [], inti1, intj1, inti2, intj2);
void main ()
{
int array [30], n, i, k;
printf ("Enter the value of sort");
scanf ("%d", &n);
printf ("Enter the values in array");
for (i=0; i<n; i++)
scanf ("%d", & array [i]);
mergesort (array, 0, n-1);
printf ("\n sorted array is");
for (i=0; i<n; i++)
printf ("%d", array [i]);
int prod b=1, prodl=1;
printf ("\n 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--)
{
Prodl = prodl * array [i];
}
Printf ("\n the product from Start is Equal to %d,"
                                              Prodb);
printf ("\n the product from last is equal to %d",
                                              prodl);
}
void mergesort (int array [], int i, int j)
{
int mid;
if (i<j)
{
mid = (i+j) / 2;
merge sort (array, i, mid);
merge sort (array, mid+1, j);
merge (array, i, mid, mid+1, j);
}
}
void merge (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<=j1 && j<=j2)
{
if (array [i] < array [j])
temp [k++] = array [i++];
else
temp [k++] = array [j++];
}
while (i<=j1)
temp [k++] = array [i++];
while (j<=j2)
temp [k++] = array [j++];
for (i=i1, j=0; i<=j2; i++, j++)
array [i] = temp [j];
}
```

Output:

```
Enter the Value of Sort 4
Enter the value in the array 3
4
```

```
2
1
Sorted Array is  1 2  3 4
Enter the value of K 2
The product from start is equal to 2
The product from last is equal to 24
```

3. Insertion Sort

Insertion Sort in C is a Simple and 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. Average & wrost - Case Complexity of this algorithm is $o(n_2)$. Insertion Sort is not good for large data sets.

Eg: Initial Array

↳

120	82	110	130	100

120	120	110	130	100

82	120	110	130	100

82	120	120	130	100

82	110	120	130	100

82	110	120	130	100

82	110	110	120	130

Sorted Array →

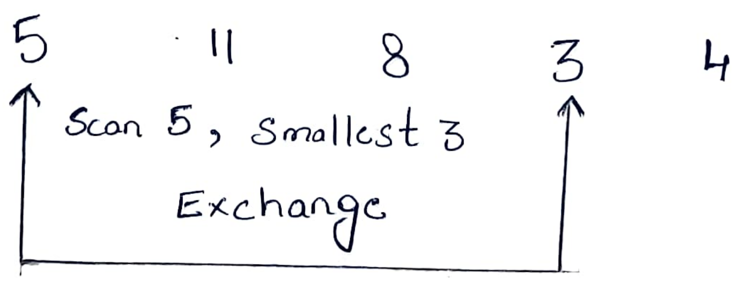
82	100	110	120	130

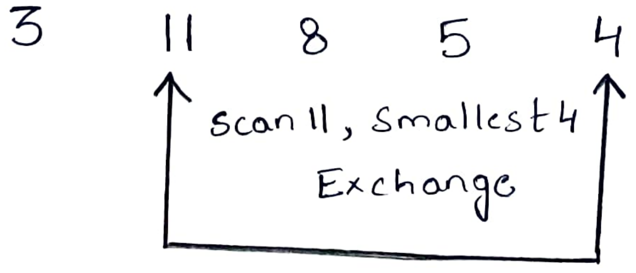
Selection Sort

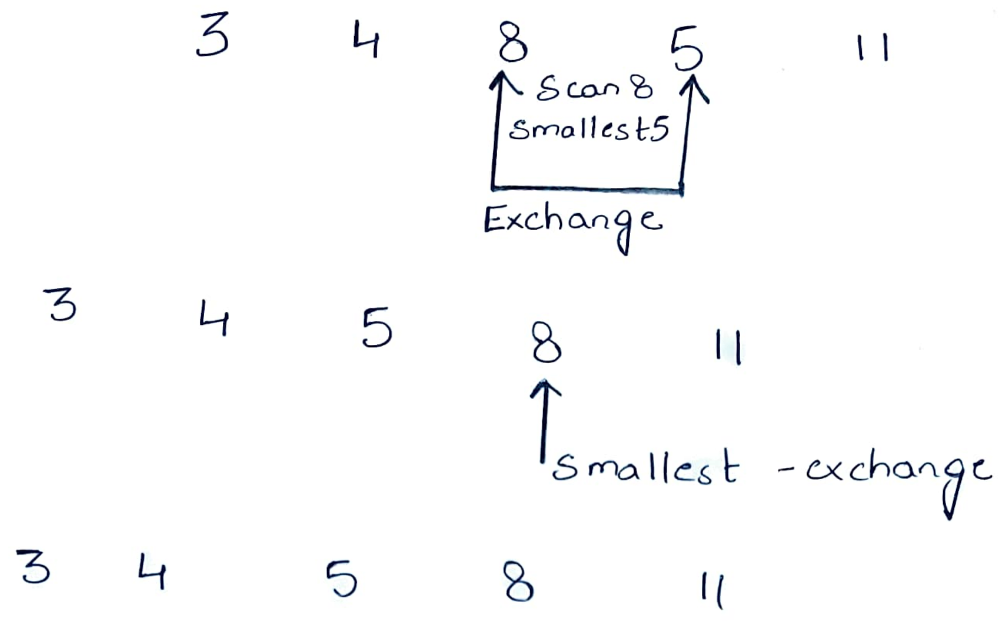
In Selection Sort, the smallest element is exchanged with the first element of the unsorted list of elements. Then the second smallest element is exchanged with the second element of the unsorted list of elements and so on until all the elements are sorted.

Average & worst-case Complexity of this algorithm is $o(n^2)$

Eg:







4.
```
# include <stdio.h>
int main ()
{
int array [100], n, c, d, i, m, swap, sumo=0, proo=1;
Printf ("Enter the elements \n");
Scanf ("%d", &n);
Printf ("Enter %d Integers \n", n);
for (c=0; c<n; c++)
Scanf ("%d", & array [c]);
for (c=0; c<n-1; c++)
{
for (d=0; d<n-c-1; d++)
{
```

```
if (array [d] < array [d+1])
{
Swap = array (d];
array [d] = array [d+1];
array (d+1) = Swap;
}
}
}
Printf ("Sorted Array in Ascending Order \n");
for (c=0 ; c<n; c++)
printf ("%d \n", array [c]);
printf ("The 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)
{
```

```
Sumo = Sumo + array [i];
}
else
{
Pro = proo * array [i];
}
}
Printf ("\n sum in odd positions is %d," sumo);
printf ("\n Product in even position %d, "proo);
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 elements 4

Enter 4 Integers

4
3
2
1
Sorted Array in Ascending Order

1
2
3
4
The Alternative Series is 1 3

Sum in ODD postions is 4

Product in Even position 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 [] = { 100, 130, 150, 170, 110 };
int num = 130;
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
```