ASSIGNMENT-6

M.SURYA PRAKASH
AP19110010082

1.

```
#include <stdio.h>
Void main( )
{
int array[10], Sumpla, Propla;
int h ,u, z, k, num, temp, keynum;
int little,Cen, rise;
Printf(" Enter The Value of Sort \n");
Scanf (" %d", &num);
Printf (" Enter The Elements \n");
for (h=0;h<num;h++)
{
Scanf("%d", &array[h]);
}
printf ("Input Array Elements \n");
for (h=0; h<num;h++)
{
printf ("%d\n", array[h]);
}
for (h=0;h<num;h++)
{
for (u=0;u<(num-h-1);u++)
{
```

```
if(array[u] < array[u+1])
{
 temp = array[u];
 array[u] = array[u+1];
  array[u+1] = temp;
}
}
}
printf("THE SORTED ARRAY \n");
for (h = 0; h<num;h++)
{
printf(" %d\n", array[h]);
}
printf(" ENTER THE ELEMENT THAT NEED TO BE SEARCHED\n");
Scanf ("%d", &keynum);

little= 1;
rise = num;

do
{
Cen = (little+rise)/2;
if (keynum > array[cen])
rise = Cen -1;
else if (keynum > array [cen])
little = cen +1;
}
```

```
while (keynum != array[cen] && little<=rise );
if (keynum== array[cen])
{
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 %d", &z,&k);
z--;
k--;
for (h=0;h<num;h++)
{
Sumloc = array[z]+ array[k];
proloc = array[z]* array[k];
}
printf ("\n SUM OF THE LOCATIONS IS %d", Sumpla);
printf ("\n PRODUCT OF THE LOCATIONS IS %d," Propla);
}
```

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 Need To Be Searched
3
Search Successful 3 is Found at Location 2
Enter The Location in The Sorted Array
2
3
Sum of The Locations is 5
Product of The Locations is 6
```

4.

```
#include <stdio.h>
int main ( )
{
int array[70], j, c, x, i, m, Swap, ofsum=0, ofpro=1;
printf(" ENTER THE ELEMENTS \n");
Scanf(" %d", &j);
printf(" ENTER %d INTEGERS \n", j);
for(c=0; c<j; c++)
Scanf("%d", &array[c]);
for(c=0; c<j-1; c++)
{
for(x=0; x<j-c-1; x++)
{
 if(array[x] > array[x+1])
{
Swap = array[x];
array[x]= array[x+1];
array[x+1]= Swap;
}
}
}
Printf(" SORTED ARRAY IN ASSENDING ORDER \n");
```

```
for(c=0;c<j;c++)
printf("%d\n",array[c]);
printf("THE ALTERNATIVE SERIES IS ");

for(i=0;i<j;i++)
{
if(i%2!=0)
{
printf(" %d ",array[i]);
}
}
for(i=0;i<j;i++)
{
if(i%2!=0)
{
ofsum=ofsum+array[i];
}
else
{
ofpro=ofpro*array[i];
}
}
printf("\n SUM IN ODD POSITIONS IS %d",ofsum);
printf("\n PRODUCT IN EVEN POSITION %d",ofpro);
printf("\n ENTER THE VALUE");
Scanf("%d",&m);
```

```
for (i=0;i<j;i++)
{
if (array[i]%m==0)
{
printf ("%d ", array[i]);
}
}
}
```

OUTPUT

ENTER THE ELEMENTS 5

ENTER 5 INTEGERS

130
100
70
90
80

SORTED ARRAY IN ASSENDING ORDER

70
80
90
100
130

THE ALTERNATIVE SERIES IS 80 100

SUM IN ODD POSITIONS IS 290

PRODUCT IN EVEN POSITION 8000

ENTER THE VALUE

10

70 80 90 100 130

5.

```
#include <stdio.h>
#include <stlib.h>
int BinarySearch (int arr[ ], int num, int first, int last)
{
if (first >last)
printf("NUMBER ENTERED IS NOT FOUND");
}
else
{
int mid;
mid= (first + last)/2;
if(arr[mid]==num)
{
printf("ELEMENT YOU DESIRED IS FOUND AT INDEX %d",mid);
exit(0);
}
else if(arr[mid]>num)
{
BinarySearch(arr, num, first, mid-1);
}
else
{
BinarySearch (arr, num, mid+1, last);
}
}
}
```

```
int main ( )
{
int arr [ ] = { 10, 82, 70, 130, 156 };
int num = 130;
int first = 0, last = (sizeof(arr) / sizeof(arr[0])) - 1;
Binary Search (arr, num, first, last);
}
```

OUTPUT

ELEMENT YOU DESIRED IS FOUND AT INDEX 3

3.

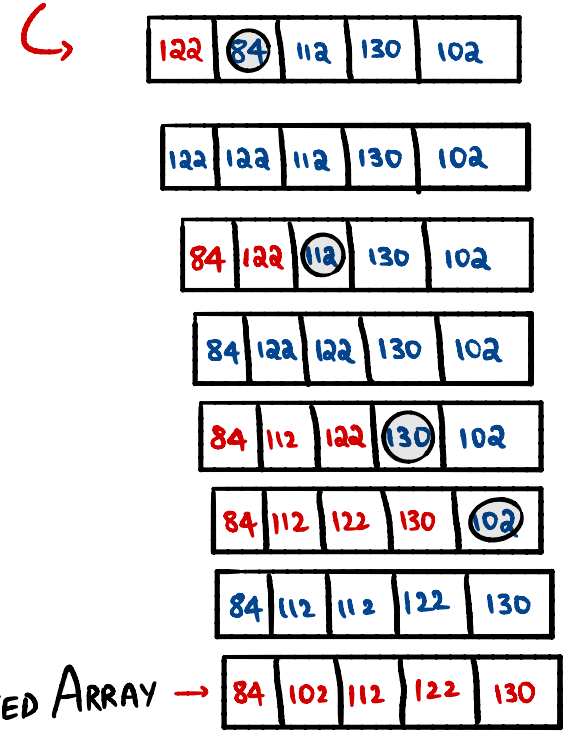
Insertion Sort

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 & worst-Case Complexity of this algorithm is O(n2). Insertion Sort is not good for large data Sets.

Eg: Initial Array

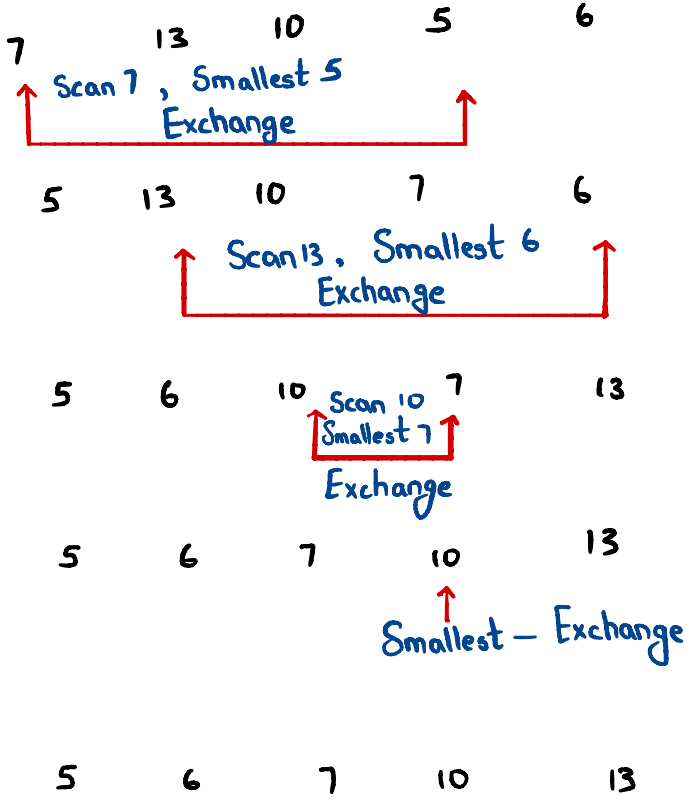


Selection Sort

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:



20

```
#include<stdio.h>
void mergsort(int array[], int i, int j);
void merg(int array[], int i1, int j1, int i2, int j2);
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 bofpso =1, lofpso =1;
printf("\n Enter The Value of k ");
Scanf("%d", &k);
k = k-1;
for(i=0;i<=k;i++)
{
```

```
bofpro = bofpro * array[i];
}
for (i=k; i<n; i++)
{
lofpro = lofpro * array[i];
}
printf("\n THE PRODUCT FROM START IS EQUAL TO %d", bofpro);
printf("\n THE PRODUCT FROM LAST IS EQUAL TO %d", lofpro);
}
Void mergesort (int array[], int i, int j)
{
int mid;
if (i<j)
{
mid= (i+j)/2;
mergesort (array, i, mid);
mergesort (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 = i2;
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 VALUES IN THE ARRAY 30
45
20
10
SORTED ARRAY IS 10 20 30 45
ENTER THE VALUE OF K 20
THE PRODUCT FROM START IS EQUAL TO 200
THE PRODUCT FROM LAST IS EQUAL TO 27000
```