ASSIGNMENT-6

M. Tanusha
AP19110010416
CSE-F

3)

```
#include <stdio.h>
void main()
{
int array[10], Sumpla, Propla;
int a, b, c, d, num, temp, keynum;
int small, cen, rise;
printf("Enter value of sort \n");
scanf("%d", &num);
printf("enter the elements \n");
for (a=0; a<num, a++)
 {
 scanf("%d", &array[a]);
 }
 printf("Array elements \n");
for (a=0; a<num; b++)
 {
 for (b=0; b<(num-a-1); b++)
  {
  if(array[b] < array[b+1])
  {
   temp = array[b];
   array[b] = array[b+1];
    array[b+1] = temp;
 }
 }
 }
 printf("The sorted array\n");
for(a=0; a<num, a++)
 {
  printf("%d \n", array[a]);
}
```

```
printf("enter the element that need to search in");
scanf("%d", &keynum);
small = 1;
rise = num;
do
 {
  cen = (small + rise)/2;
 if (keynum > array[cen])
  rise = cen - 1;
  else if (keynum > array[cen])
  small = cen + 1;
 }
while (keynum != array[cen] && small <= rise);
if (keynum == array[cen])
{
 printf("search success and %d found location %d\n", keynum, mid+1);
}
else
 {
  printf("search failed \n");
 }
printf("enter the location in sorted array\n");
scanf("%d %d", &c, &d);
 c--;
 d--;
for(a=0, a<num, a++)
{
 sumloc = array[z] + array[t];
 proloc = array[z] * array[t];
}
printf("\n sum of the locations is %d", sumpla);
printf("\n product of the location %d", propla);
}
```

Output :-

```
Enter the value of sort
3
Enter the elements
2
4
5
The sorted array
5
4
2
Enter the element that need to be search
4
search success 4 is found at location 2
Enter the location in sorted array.
4
2
Sum of the locations 6
product of the location 8
```

3) Insertion Sort :-

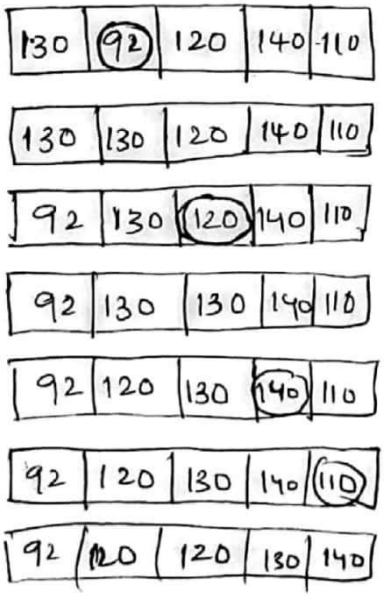
Insertion Sort in C is a simple and efficient algorithm, that creates the final sorted array one element at a time.

Insertion Sort works in a similar manner as we arrange a deck of cards.

Avg & worst-case complexity of this algorithm is $O(n_2)$.

Insertion Sort is not good for large data sets.

Eg :- Initial Array



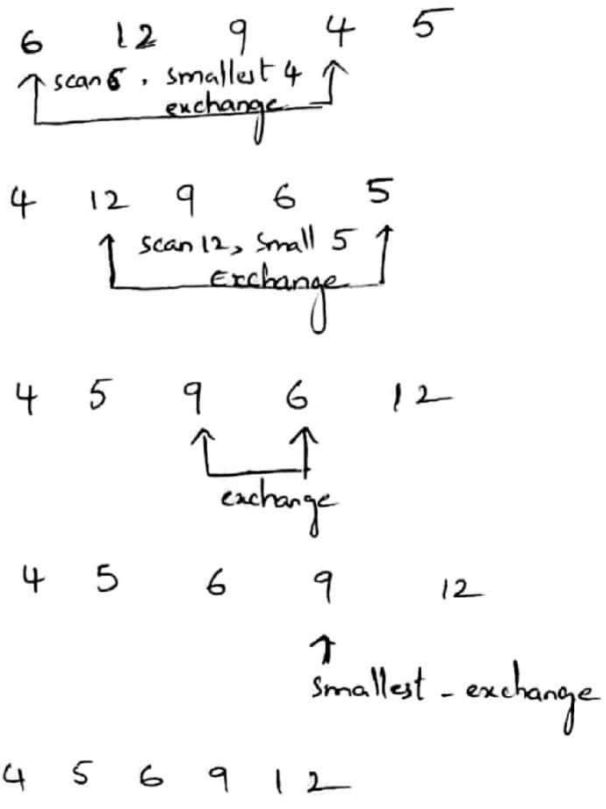
Sorted Array → 92 | 110 | 120 | 130 | 140

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.

Avg & worst-case complexity of the algorithm is $O(n^2)$

Eg :-



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[ ] = {110, 140, 160, 180, 120};
 int num =140;
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.

4)

```
#include <stdio.h>
int main()
{
int array[100], n, a, b, i, m, swap, sum = 0, proo = 1;
printf("enter the elements \n");
scanf("%d", &n);
printf("enter %d integers \n", n);
for(a=0; a<n; a++)
scanf("%d", &array[a]);
for(a=0; a<n-1; a++)
{
for(b=0; b<n-a-1; b++).
{
if(array[b] > array[b+1])
{
 swap = array[b];
 array[b]= array[b+1];
 array[b+1] = swap;
}
}
}
Printf("sorted array in Ac \n");
for(a=0; a<n, a++)
Printf("%d \n", array[a]);
Printf(" 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)
{
sum = sum + array[i];
}
else
{
proo = proo * array[i];
}
}
printf(" \n Sum in odd position is %.d", sumo);
printf(" \n product in even %.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 5
enter 5 integers
5
4
3
2
1
Sorted array in AO
1
2
3
4
5
The Alternative Series 1 3 5
sum in odd is 9
product in even is 8
Enter the value.
2
2 4
```

2)

```
# 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[40], n, i, k;
printf("enter the value of fost");
scanf("%d", &n);
printf("enter the value in array");
for(i=0, i<n, i++)
scanf("%d", &array[i]);
mergesort (array, 0, n-1);
```

```
printf("In Sorted array is");
for (i=0; i<n; i++)
printf("%d", array[i]);
int prodb=1, prodl=1;
Printf("In 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("In the product from start is equal %d", prodb);
printf("In the product from last is equal %d", prodl);
}
void mergesort(int array[], int i, int j)
{
 int mid;
 if(i<j)
 {
  mid=(i+j)/2;
 merge sort(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 = j2 ;
k = 0 ;
while ( i<=j, && 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[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 5
enter the value of array 3
4
3
2
sorted array is 1 2 3 5
enter the value of k 2
product form start is equal to 2
product from last is equal to 24.
```