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

```
#include<stdio.h>
void sort(int a[], int n)
{
 int i,j,temp;
 for(i=0;i<n;i++)
  {
   For(j=i+1;j<n;j++)
   {
   if(a[i]<a[j])
    {
    temp = a[i];
    a[i] = a[j];
    a[j] = temp;
   }
  }
 }
}
int binary(int a[], int e, int n)
{
  int i=0,j=n-1,mid;
```

```
while(i<=j)
{
  mid=(i+j)/2;
  if(a[mid]==e)
     return mid+1;
  else
   {
    if(e<a[mid])
       j=mid-1;
    else
       i=mid+1;
   }
}
if(i>j)
 {
   return 0;
 }
}
int main()
{
  int n,i,a[20],f,e,m1,m2;
  printf("Enter no. of elements in array");
  scanf("%d",&n);
  printf("Enter elements in array");
 for(i=0;i<n;i++)
```

```
        scanf("%d",&a[i]);
sort(a,n);
for(i=0;i<n;i++)
 printf("\n%d",a[i]);
printf("Enter element to find in array");
scanf("%d", &e);
 f=binary(a,e,n);
if(f!=0)
{
   printf("element %d position", f);
 }
else
{
  printf("elements not found");
 }
printf("Enter the position of array to find sum&
        product");
scanf("%d%d",&m1,&m2);
m1--;
m2--;
```

```
Printf("sum is %d", a[m1] + a[m2]);
printf("product is %d", a[m1] * a[m2]);
}
```

2) C program for merge sort

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

```
        i++;
    }
    else
    {
        arr[k] = R[j];
        j++;
    }
    k++;
}
while(i<n1)
{
arr[k] = L[i];
    i++;
    k++;
}
while (j<n2)
{
arrk[k] = R[j];
    j++;
    k++;
    }
}
void mergesort(int arr[], int l, int r)
{
if(l<r)
{
```

```
    int m = l + (r-l)/2;
    mergeSort(arr, l, m);
    mergeSort(arr, m+1, r);
    merge(arr, l, m, r);
  }
}

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

int main()
{
  int arr[5];
  int i;
  int arr_size = sizeof(arr) / sizeof(arr[0]);
  for(i=0; i<arr_size; i++)
  {
    printf("Enter the elements");
    scanf("%d", &arr[i]);
  }
```

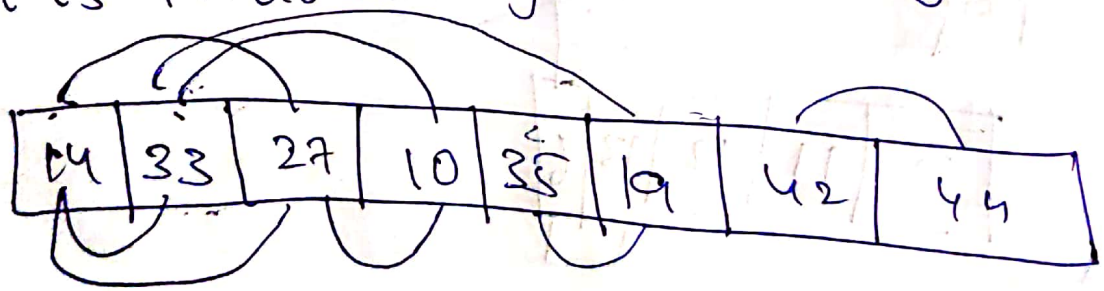
```
Printf("array is");
PrintArray(arr, arr - size);
mergeSort(arr, 0, arr - size - 1);
Printf("Enter Sorted list");
scanf("%d", &K);
int fromfirst = arr[K-1];
int fromlast = arr[5-(K)];
printf("%d", fromlast * fromfirst);
return 0;
}
```

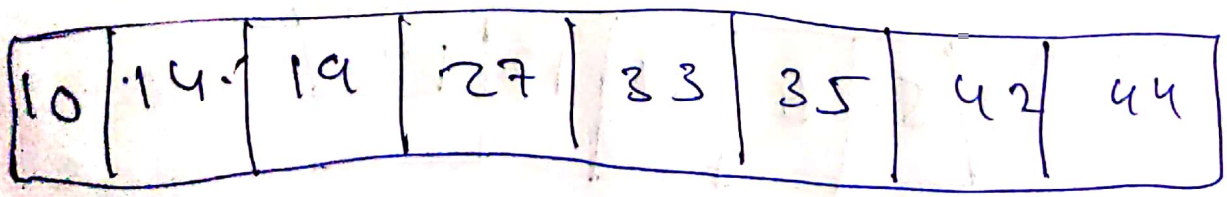
3) Insertion Sort:-

* The data is sorted by inserted by inserting the data into an exisiting sorted file.
* Elements are known beforehand which location to be placed them in searched.
* Insertion sort is live sorting technique which can ~~tead~~ deal with immediate data

eg:- Best Case Complerity O(n)

1) we take an unsorted array
2) Insertion sort compares the first two elements.
2) Comparing first element with next one whether it is in ascending or descending order

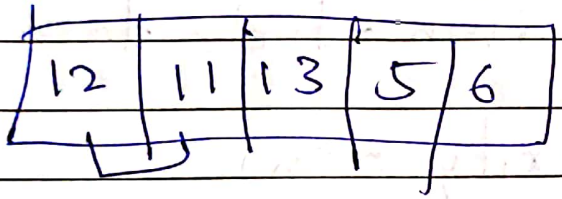


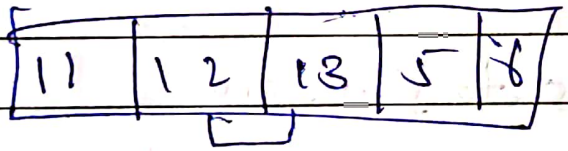


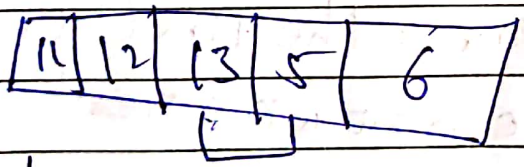
2) Selection Sort:-

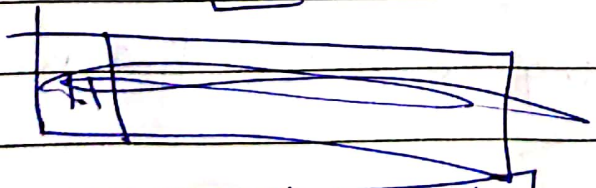
* The data is sorted by selecting and placing the consecutive elements in sorted location
* location is previously known while elements are searched.
* It can not deal with immediate data, it needs to be present at the beggining

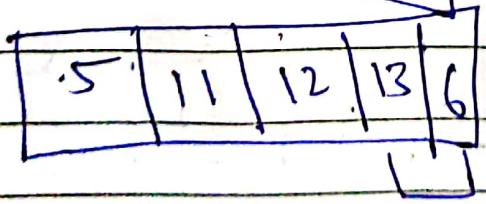
eg:- Best Complexity $O(n^2)$

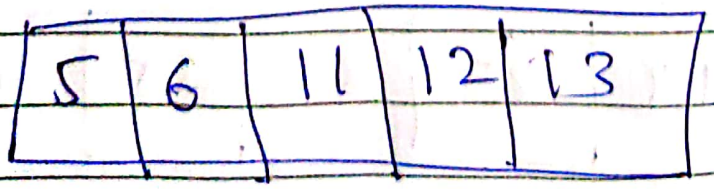












4) Sort the array using bubble sort where elements are taken from the user and display the element.

```
#include<stdio.h>
void main( )
 {
  int a[100], b, i, j, temp, sum=0, pro=1, c;
Printf("No. of elements");
Scanf("%d", &b);
 Printf("Enter %d integers", b);
For(i=0; i<b; i++)
 {
  Scanf("%d", a[i]);
 }
 For(i=0; i<b-i-1; i++)
 {
  For(j=0; j<b-i-1; j++)
 {
 if (a[j] > a[j+1])
  {
  temp=a[j];
  a[j] = a[j+1];
  a[j+1] =temp;
 }
 }
}
```

```
printf("sorted list");
for(i=0;i<b;i++)
{
printf("%d\n",a[i]);
}
printf("alternate");
for(i=0;i<b;i++)
{
if(i%2==0)
{
printf("%d",a[i]);
}
}
for(i=0;i<b;i++)
{
if(i%2!=0)
{
sum=sum+a[i];
}
}
printf("sum %d",sum);
for(i=0;i<b;i++)
{
```

Date :

```
printf("sorted list");
for(i=0;i<b;i++)
{
printf("%d\n",a[i]);
}
printf("alternate");
for(i=0;i<b;i++)
{
if(i%2==0)
{
printf("%d",a[i]);
}
}
for(i=0;i<b;i++)
{
if(i%2!=0)
{
sum=sum+a[i];
}
}
printf("sum %d",sum);
for(i=0;i<b;i++)
{
if(i%2==0)
```

```
{
pro = pro * a[i];
}
}
Printf("Product is %d", Pro);
Printf("Enter the value");
Scanf("%d", &c);
for(i=0; i<b; i++)
{
if(a[i] % c == 0)
{
printf("%d", a[i]);
}
}
}
```

5) Program to Implement recursive method For binary Search.

```
#include<stdio.h>
int binarysearch(int n[], int a, int b, int c)
  {
    int x = (a+b)/2;
  if (a>b) return -1;
  if (n[x]==c)
        return x;
  if (n[x]<c)
     return binarysearch(n, x+1, b, c);
    else
     return binarysearch (n, a, x-1, c);
  }
int main(void){
    int n[50];
  int len, num, search;
  Printf("Enter length");
  Scanf("%d", &len);
  Printf("Enter the elements");
  for(int i=0; i<len; i++)
     Scanf("%d", &n[i]);
  Printf("Enter element to search");
  Scanf("%d", &search);
  Pos = binarysearch(n, 0, len-1, search);
```

```
if(pos<0)
  printf("Cannot find %d", search);

else
printf("position %d is %d", search, pos+1);

return 0;
}
```