Iterative - GCD

```
#include < stdio.h>.
int euclid(int a, int b).
{ while ( a != b).
  { if ( a > b) {
        a = a - b; }

     else {
       b = b - a; }
  }
return a;
}

int main ()
{ int a, b;
printf (" Enter a & b resp");
 Scanf (" %d %d", &a , &b);
 printf (" "euclid (%d , %d)= %d
            a, b, euclide(a, b)};
 return 0; }
```

GCD - recursive

```
# include<stdio.h>.
# include <stdlib.h>.
int hcf ( int n1, int n2);
int main () {
int n1, n2 ;
printf ("Enter 2 positive nos:");
scanf ("%d %d", &n1 , &n2);
printf ("GCD is : %d , hcf(n1, n2));
return 0 ; }

int hcf (int n1, int n2) {
if (n2 != 0)
  return hcf (n2 , n1 % n2);
  else
  return n1; }
```

```
# include<stdio.h>
 int recsearch (int arr[], int l, int r, int x)
 { if(r<l).
      return -1;
      if (arr[l] == x).
        return l;
      if ( arr [r] == x)
        return r;

   else  return recsearch(arr, l+1, r-1, x);
 }

int main ()
{ int arr[] = {12, 34, 54, 2, 3}, i;
   int n = sizeof(arr)/ sizeof(arr[0])
   int x;

  printf ("Enter item to be searched");
  scanf ("%d", &x);
   int   index = recSearch (arr, 0, n-1, x)

   if ( index != -1)
    printf ("item is present at index%d, index);
else    printf ("NO");  return 0; }
```

Binary - Search

```
#include<stdio.h> .

int binarySearch( int arr[], int l, int r,
                  int x)
{ if ( r>= 1) {
    int mid = l+(r-l) /2;
  if (arr[mid] = = x).
     return mid;
  if ( arr[mid] -> x
   return binarySearch( arr, l, mid-1, x)
  else
  return binarySearch (arr, mid+1, r, x);
 }
return -1;
}
int main (void).
{ int x;
  int arr[] = {2, 3, 4, 10, 40};
  int n = sizeof (arr) / sizeof (arr[0]);
```

~~int o~~

```
printf ("enter ITEM TO BE SEARCHED");
scanf ("%d", &x);

int result = binarySearch( arr, 0, n-1, x);

if ( result == -1)
Printf (" NOT FOUND");

else
printf (" Element present in index %d",
                result);

return 0;

}
```