15/4/21

Page No: 1
Date: / /

① Recursive program for GCD.

```
#include <stdio.h>
int gcd (int n1, int n2) {
    if (n2 != 0) {
        return gcd (n2, n1 % n2); }
    else
        return n1;
}
int main () {
    int n1, n2;
    printf ("Enter two for integers : \n");
    scanf ("%d %d", &n1, &n2);
    printf ("GCD of %d & %d is %d",
            n1, n2, gcd (n1, n2));
    return 0;
}
```

② Iterative program for GCD

```
#include <stdio.h>
int main () {
    int n2, n1, K;
    printf ("Enter two values :- \n");
    scanf ("%d %d", &n1, &n2);
    while (n2 != 0) {
        K = n2;
```

```
        m2 = m1 % m2;
        m1 = k;
    }
    printf("GCD = %d", m1);
    return 0;
}
```

(3) Linear & binary search using Recursion

```
#include <stdio.h>
#include <stdlib.h>
int linear(int a[], int l, int r, int Key){
    if (r < l) {
        return -1;
    if (a[l] == Key)
        return l;
    if (a[r] == Key)
        return r;
    return linear(a, l+1, r-1, Key);
}
int binary(int a[], int first, int
                last, int key) {
    if (last >= first) {
        int m = first + (last - first)/2;
        if (a[m] == key) {
            return m;}
```

```
    if (a[m] > key) {
        return binary(a, first, m-1, key);
    }
    return binary(a, m+1, last, key);
}
return -1;
}
int main() {
    int a[200], i, choice, key, m, res;
    printf("Enter size of array : ");
    scanf("%d", &m);
    printf("Enter values : ");
    for(i=0; i<m; i++){
        scanf("%d", &a[i]);
    }
    for(;;){
        printf("Enter value to find \n");
        scanf("%d", &key);
        printf("1. Linear 2 Binary ");
        scanf("%d", &choice);
        switch(choice){
            case 1: res = linear(a, 0, m-1, key);
                if(res != -1) { printf("%d is
                    at %d", key, res+1);
                else { printf("%d Not found", key);
                break;
```

```
case 2 : res = binary(a, 0, n-1, key);
    if (res == -1) {
        printf("Element Not found"); }
    else {
        printf("Element found at %d",
                (res+1));
    }
    break;
default : exit(0);
}
}
return 0;
}
```