Recursion: Factorial of a no.

```
#include <stdio.h>
int factorial(int n);
int main()
{   int n;
    printf("Enter a no.");
    scanf("%d", &n);
    printf("factorial of %d is %d", n, factori
    -al(n));
      return 0;
}
    int factorial(int n) {
    { if(n>=1)
        return n* factorial(n-1);
      else
      return 1;
    }
```

GCD of 2 nos. using recursion.

```
#include <stdio.h>
int hcf(int n1, int n2)
int main()
{ int n1, n2;
  printf("Enter 2 integers");
  scanf("%d %d", &n1, &n2);
  printf("G.C.D of %d and %d is %d", n1, n2,
    hcf(n1, n2));
  return 0;
}
```

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

   else
    return 1;
}
```

Tower of hanoi using recursion.

```
#include<stdio.h>
void towerOfHanoi(int n, char from_rod,
char to_rod, char aux_rod);
 { if(n==1)
   { printf("\n Move disc 1 from rod %c to
   to rod %c", from_rod, to_rod);
    return;
   }
  towerOfHanoi(n-1, from_rod, aux_rod, to_rod);
  printf(" move disc %d from rod %c to
    rod %c ", n, from_rod, to_rod);
  towerOfHanoi(n-1, aux_rod, to rod, from_rod);
}
  int main()
  { int n=4;
    towerOfHanoi(n, 'A', 'C', 'B');
    return 0;
  }
```

Ques implimentation

Binary Search-

```
#include <stdio.h>
void binary_search (int [], int, int);
void bubble_sort( int [], int );
int main()
{ int key, size, i;
  int list[25];
  printf("Enter size of list");
  scanf(" %d ", &size);
  printf(" Enter Elements. \n");
    for(i=0; i< size ; i++)
        { scanf(" %d", &list [i]); }
  bubble_sort(list, size) ;
  printf(" Enter key to search ");
  scanf(" %d ", &key);
  binary_search (list, 0, size, key);
}
void bubble_sort (int list[], int size)
{ int temp, i, j;
  for(i=0; i<size ; i++)
  { for (j=i; j<size ; j++)
      { if (list [i] > list[j])
          { temp = list [i];
            list [i]= list [j];
            list [j]= temp;
          }}}}
void binary_search (int list [], int low, int high, int key)
{ int mid;
  if (low > high)
{ printf("key not found");
```

```
    return ; }
mid = (low + high) / 2;
if (list[mid] == key)
{ printf(" Key found "); }
  else if (list[mid] < key)
  { binary_search(list, mid + 1, high, key);
  } }
```