1. **Sum of Digits**

import java.io.\*;

class sum\_of\_digits

{

static int sum\_of\_digit(int n)

{

return n==0?0:(n % 10 + sum\_of\_digit(n / 10));

}

public static void main(String args[])

{

Scanner sc=new Scanner(System.in)

int num=sc.nextInt();

int result = sum\_of\_digit(num);

System.out.println("Sum of digits in " +num + " is " + result);

}

}

1. **Is magic?**

import java.io.\*;

public class GFG

{

public static boolean isMagic(int n)

{

int sum = 0;

while (n > 0 || sum > 9)

{

if (n == 0)

{

n = sum;

sum = 0;

}

sum += n % 10;

n /= 10;

}

return (sum == 1);

}

public static void main(String args[])

{

int n = 1234;

n==0? System.out.println("0") :System.out.println("1");;

}

}

1. **Implement Power Function**

#define ll long long int

int Solution::pow(int base, int expo, int m) {

if(m==1) return 0;

ll ans=1,y=base%m;

while(expo>0){

if(expo&1)

ans=((ans)\*y)%m;

y=(y\*y)%m;

if(y<0) y+=m;

expo=expo>>1;

}

if(ans<0){

ans=(m-abs(ans)%m);

return ans;

}

return ans%m;

}

1. **Tower of Hanoi**

import java.util.\*;

public class Main

{

static void tower\_of\_hanoi(int N,int source,int aux,int Dest){

if(N==0) return;

tower\_of\_hanoi( N-1,source,Dest,aux);

System.out.println("[ " + N + " "

+ source + " "

+ Dest + " ]");

tower\_of\_hanoi( N-1, Dest,aux, source);

}

public static void main(String[] args) {

Scanner sc=new Scanner(System.in);

int N=sc.nextInt();

tower\_of\_hanoi(N,1,3,2);

}

}

1. **Gray Code**

public class Solution {

public ArrayList<Integer> grayCode(int n) {

ArrayList<Integer> res = new ArrayList<Integer>();

if(n <= 1){

res.add(0);

if(n == 1) res.add(1);

return res;

}

ArrayList<Integer> prev = grayCode(n - 1);

int highest\_bit = 1 << (n - 1);

for(int i = prev.size() - 1; i >= 0; i--)

res.add(prev.get(i) + highest\_bit);

prev.addAll(res);

return prev;

}

}

Homework  
1**. Find Fibonacci – II**

#include <stdio.h>

int fib(int n)

{

if (n <= 1)

return n;

return fib(n-1)+fib(n-2);

}

int main()

{

int n;

scanf("%d",&n);

printf("%d", fib(n-1)+fib(n-2));

getchar();

return 0;

}

**Find Factorial!**

#include <stdio.h>

int factorial(unsigned int n)

{

if (n == 0)

return 1;

return n \* factorial(n - 1);

}

int main()

{

int num = 5;

printf("Factorial of %d is %d",

num, factorial(num));

return 0;

}

**Print reverse string**

# include <stdio.h>

void reverse(char \*str)

{

if (\*str)

{

reverse(str+1);

printf("%c", \*str);

}

}

int main()

{

char a[] = "Geeks for Geeks";

reverse(a);

return 0;

}

**Another sequence problem**

#include <stdio.h>

int fib(int n)

{

if (n <= 1)

return n;

return fib(n-1)+fib(n-2);

}

int main()

{

int n;

scanf("%d",&n);

printf("%d", fib(n-1)+fib(n-2)+fib(n));

getchar();

return 0;

}

**Kth Symbol**

class Solution {

public int kth(int n, int k) {

if (n == 1)

return 0;

if (k % 2 == 1)

return kth(n - 1, (k + 1) / 2) == 0 ? 0 : 1; // Left node

return kth(n - 1, k / 2) == 0 ? 1 : 0; // Right node

}

Public static void main(String[] args)

{

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int k=sc.nextInt();

System.out.println(kth(n,k));

}

}