#include <iostream>

using namespace std;

//Recursive approach

int fib(int n,int dp[]){

if(n==0 || n==1){

return n;

}

if(dp[n]!=0){

return dp[n];

}

int ans=fib(n-1,dp)+fib(n-2,dp);

return dp[n]=ans;

}

//Bottom up approach

int BottomUp(int n){

int dp[100]={0};

dp[1]=1;

for(int i=2;i<=n;i++){

dp[i]=dp[i-1]+dp[i-2];

}

return dp[n];

}

//Bottom up approach with space optimisation

int fibSpaceOpt(int n){

if(n==0 || n==1){

return n;

}

int a=0;

int b=1,c;

for(int i=2;i<=n;i++){

c=a+b;

a=b;

b=c;

}

return c;

}

int main() {

int n;

cin>>n;

int dp[100]={0};

cout<<fib(n,dp)<<endl;

cout<<BottomUp(n)<<endl;

cout<<fibSpaceOpt(n)<<endl;

return 0;

}

Input-6

Output-

8

8

8