Your company has an annual holiday party with a secret Santa event. In recent years, people have been getting assigned as their own secret Santa and buying themselves foot baths. As the analyst of the party planning committee, you are starting to get almost deranged because of how frequently this activity is occurring. Help keep yourself sane by determining how many ways the people in a group of size n can get assigned so that no one gets assigned to be their own secret Santa.

**Example**  
For n = 3, the output should be  
deranged(n) = 2.  
There are 3 people in the group. Let's call them Bob, Joe, and Jane. There are 6 ways to assign these people to each other:

Bob -> Bob, Joe -> Joe, Jane -> Jane  
Bob -> Bob, Joe -> Jane, Jane -> Joe  
Bob -> Joe, Joe -> Bob, Jane -> Jane  
Bob -> Jane, Joe -> Joe, Jane -> Bob

Bob -> Joe, Joe -> Jane, Jane -> Bob  
Bob -> Jane, Joe -> Bob, Jane -> Joe

Only in the last two combinations is no one assigned to themselves. So the answer is 2.

* **[time limit] 3000ms (cs)**
* **[input] integer n**

The number of people participating in the secret Santa activity.

*Guaranteed constraints:*  
0 ≤ n ≤ 18.

* **[output] integer64**

The number of ways that people in a group of size n can be assigned so that no one is assigned as their own secret Santa.

<https://codefights.com/challenge/RmHkZf2atLqkBNXud/solutions>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static double factorial(long n)

{

double prod = 1;

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

{

prod \*= i;

}

return prod;

}

static long deranged(int n)

{

double nf = (double) factorial(n);

double sum = 0;

for (int k = 0; k <= n; k++)

{

sum += (long)Math.Pow(-1, k)/factorial(k) ;

}

return (long)(nf \* sum);

}

static void Main(string[] args)

{

int n = 7;

Console.WriteLine(deranged(n));

Console.ReadLine();

}

}

}