**Recursion: Davis' Staircase**

Davis has a number of staircases in his house and he likes to climb each staircase 1,2, or 3 steps at a time. Being a very precocious child, he wonders how many ways there are to reach the top of the staircase.

Given the respective heights for each of the s staircases in his house, find and print the number of ways he can climb each staircase, module 10^10+7 on a new line.

For example, there is s=1 staircase in the house that is  steps high. Davis can step on the following sequences of steps:

1 1 1 1 1

1 1 1 2

1 1 2 1

1 2 1 1

2 1 1 1

1 2 2

2 2 1

2 1 2

1 1 3

1 3 1

3 1 1

2 3

3 2

There are 13 possible ways he can take these 5 steps 13%10000000007=13.

**Function Description**

Complete the *stepPerms* function in the editor below. It should recursively calculate and return the integer number of ways Davis can climb the staircase, modulo 10000000007.

stepPerms has the following parameter(s):

* *n*: an integer, the number of stairs in the staircase

**Input Format**

The first line contains a single integer, s, the number of staircases in his house.   
Each of the following s lines contains a single integer, n, the height of staircase i.

**Constraints**

* 1<=s<=5
* 1<=n<=36

**Subtasks**

* 1<=n<20 for 50% of the maximum score.

**Output Format**

For each staircase, return the number of ways Davis can climb it as an integer.

**Sample Input**

3

1

3

7

**Sample Output**

1

4

44

**Explanation**

Let's calculate the number of ways of climbing the first two of the Davis' s=3 staircases:

1. The first staircase only has n=1 step, so there is only one way for him to climb it (i.e., by jumping 1 step). Thus, we print 1 on a new line.
2. The second staircase has n=3 steps and he can climb it in any of the four following ways:   
   1. 1 -> 1 -> 1
   2. 1 -> 2
   3. 2 ->1
   4. 3

Thus, we print 4 on a new line.

using System.CodeDom.Compiler;

using System.Collections.Generic;

using System.Collections;

using System.ComponentModel;

using System.Diagnostics.CodeAnalysis;

using System.Globalization;

using System.IO;

using System.Linq;

using System.Reflection;

using System.Runtime.Serialization;

using System.Text.RegularExpressions;

using System.Text;

using System;

class Solution {

// Complete the stepPerms function below.

static int stepPerms(int n) {

if(n==1) return 1;

if(n==2) return 2;

if(n==3) return 4;

int[] arr=new int[n+1];

arr[0]=0;

arr[1]=1;

arr[2]=2;

arr[3]=4;

for(int i=4;i<=n;i++){

arr[i]=arr[i-1]+arr[i-2]+arr[i-3];

}

return arr[n];

}

static void Main(string[] args) {

TextWriter textWriter = new StreamWriter(@System.Environment.GetEnvironmentVariable("OUTPUT\_PATH"), true);

int s = Convert.ToInt32(Console.ReadLine());

for (int sItr = 0; sItr < s; sItr++) {

int n = Convert.ToInt32(Console.ReadLine());

int res = stepPerms(n);

textWriter.WriteLine(res);

}

textWriter.Flush();

textWriter.Close();

}

}

**Congratulations**

You solved this challenge. Would you like to challenge your friends?

[Next Challenge](https://www.hackerrank.com/challenges/crossword-puzzle?h_l=interview&playlist_slugs%5B%5D=interview-preparation-kit&playlist_slugs%5B%5D=recursion-backtracking&h_r=next-challenge&h_v=zen)

* **Test case 0**
* **Test case 1**
* **Test case 2**
* **Test case 3**
* **Test case 4**
* **Test case 5**
* **Test case 6**
* **Test case 7**
* **Test case 8**
* **Test case 9**
* **Test case 10**

Compiler Message

**Success**

Input (stdin)

Download

* **3**
* **1**
* **3**
* **7**

Expected Output

Download

* **1**
* **4**
* **44**