Full Name:	GUILHERME FERNANDES
Email:	contato@guifr.com.br
Test Name:	**Mock Test**
Taken On:	6 Jun 2022 17:15:24 IST
Time Taken:	19 min 26 sec/ 20 min
Invited by:	Ankush
Invited on:	6 Jun 2022 17:15:01 IST
Skills Score:	
Tags Score:	Algorithms 120/120
	Core CS 120/120
	Dynamic Programming 120/120
	Medium 120/120
	problem-solving 120/120

100%

120/120

scored in **Mock Test** in 19 min 26 sec on 6 Jun 2022 17:15:24 IST

Recruiter/Team Comments:

No Comments.

	Question Description	Time Taken	Score	Status
Q1	Fibonacci Modified > **Coding**	19 min 7 sec	120/ 120	✓

QUESTION 1

✓

Correct Answer

Score 120

Fibonacci Modified > Coding

Dynamic Programming | Algorithms | Medium | problem-solving | Core CS

QUESTION DESCRIPTION

Implement a *modified* Fibonacci sequence using the following definition:

Given terms $t[i]$ *and* $t[i+1]$ *where* $i \in (1, \infty)$, *term* $t[i+2]$ *is computed as:*

$$t_{i+2} = t_i + (t_{i+1})^2$$

Given three integers, $t1$, $t2$, and n, compute and print the n^{th} term of a *modified Fibonacci sequence*.

Example

$t1 = 0$
$t2 = 1$
$n = 6$

- $t3 = 0 + 1^2 = 1$
- $t4 = 1 + 1^2 = 2$

- $t5 = 1 + 2^2 = 5$
- $t6 = 2 + 5^2 = 27$

Return 27.

Function Description

Complete the *fibonacciModified* function in the editor below. It must return the n^{th} number in the sequence.

fibonacciModified has the following parameter(s):

- *int t1*: an integer
- *int t2*: an integer
- *int n*: the iteration to report

Returns

- *int:* the n^{th} number in the sequence

Note: The value of $t[n]$ may far exceed the range of a 64-bit integer. Many submission languages have libraries that can handle such large results but, for those that don't (e.g., C++), you will need to compensate for the size of the result.

Input Format

A single line of three space-separated integers, the values of $t1$, $t2$, and n.

Constraints

- $0 \leq t1, t2 \leq 2$
- $3 \leq n \leq 20$
- t_n may far exceed the range of a 64-bit integer.

Sample Input

```
0 1 5
```

Sample Output

```
5
```

Explanation

The first two terms of the sequence are $t1 = 0$ and $t2 = 1$, which gives us a modified Fibonacci sequence of $\{0, 1, 1, 2, 5, 27, \ldots\}$. The 5^{th} term is 5.

CANDIDATE ANSWER

Language used: **Java 8**

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.function.*;
import java.util.regex.*;
import java.util.stream.*;
import static java.util.stream.Collectors.joining;
import static java.util.stream.Collectors.toList;



class Result {
    /*
     * Complete the 'fibonacciModified' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts following parameters:
     *  1. INTEGER t1
     *  2. INTEGER t2
     *  3. INTEGER n
     */

    public static String fibonacciModified(BigDecimal t1, BigDecimal t2, int
n) {
        if(n == 1) return t1.toString();
        if(n == 2) return t2.toString();

        BigDecimal tNext = t1.add(t2.pow(2));

        return fibonacciModified(t2, tNext, --n);
    }

}

public class Solution {
    public static void main(String[] args) throws IOException {
        BufferedReader bufferedReader = new BufferedReader(new
InputStreamReader(System.in));
        BufferedWriter bufferedWriter = new BufferedWriter(new
FileWriter(System.getenv("OUTPUT_PATH")));

        String[] firstMultipleInput =
bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");

        BigDecimal t1 = new BigDecimal(firstMultipleInput[0]);

        BigDecimal t2 = new BigDecimal(firstMultipleInput[1]);

        int n = Integer.parseInt(firstMultipleInput[2]);

        String result = Result.fibonacciModified(t1, t2, n);

        bufferedWriter.write(String.valueOf(result));
        bufferedWriter.newLine();

        bufferedReader.close();
        bufferedWriter.close();
    }
}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 1	Easy	Sample case	✓ Success	0	0.1115 sec	23.9 KB
Testcase 2	Easy	Sample case	✓ Success	0	0.0943 sec	24 KB
Testcase 3	Easy	Hidden case	✓ Success	15	0.1766 sec	38.4 KB
Testcase 4	Easy	Hidden case	✓ Success	15	0.4377 sec	52.4 KB
Testcase 5	Easy	Hidden case	✓ Success	15	0.1174 sec	24.7 KB
Testcase 6	Easy	Hidden case	✓ Success	15	0.0935 sec	24 KB
Testcase 7	Easy	Hidden case	✓ Success	15	0.5395 sec	58.2 KB
Testcase 8	Easy	Hidden case	✓ Success	15	0.1218 sec	24.1 KB

Testcase 9	Easy	Hidden case	✓ Success	15	0.0931 sec	24.2 KB
Testcase 10	Easy	Hidden case	✓ Success	15	0.089 sec	23.6 KB

No Comments

PDF generated at: 6 Jun 2022 12:06:49 UTC