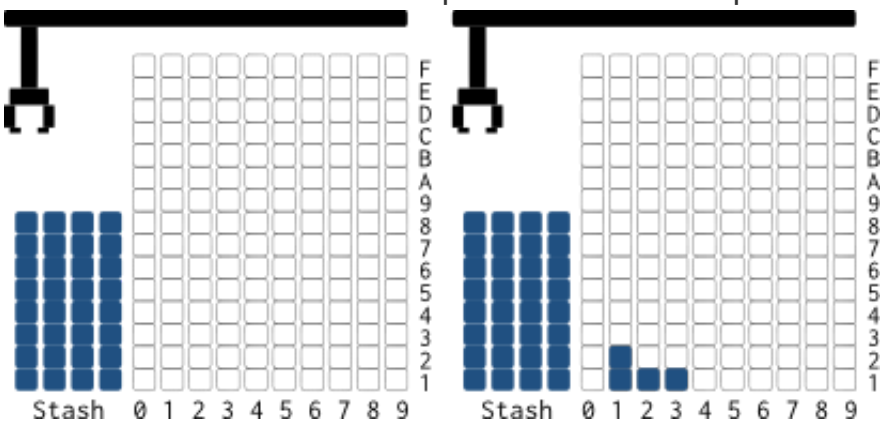
Robot - 65 points (Coding)

We have a robot that can pick up blocks from a stash, move them horizontally, and lower them in place. There are 10 positions available to lower blocks, numbered from 0 to 9. Each position can hold up to 15 blocks.



The robot understands the commands 'P', 'M' and 'L':

P: **Pickup** from the stash and move to position 0

M: **Move** to the next position

L: **Lower** the block

The robot is safe to operate and very forgiving:

- There are always blocks in the stash (**Pickup** will always get a block).
- If the robot already holds a block, **Pickup** will reset the robot to position 0.
- The robot will not go beyond position 9. Trying to **Move** it further does nothing.
- **Lower**ing a block on a pile of 15 blocks does nothing (and the robot will keep any block it holds).
- **Lower**ing without a block does nothing.
- The robot ignores any command that is not 'P', 'M' or 'L'.

Implement a function that takes a `String` of commands for the robot. The function should output a `String` representing the number of blocks (in hexadecimal) at each position after running all the commands.

For example, given the String `"PMLPMMMLPMLPMML"`, the function should return `"0211000000"` (see the image above). And `"PLPLPLPLPLPLPLPLPLPL"` should return `"A000000000"`.

YOUR ANSWER

Java

```java
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

/*
 * Complete the function below.
 */

    static String compute(String instructions) {


    }
```

```java
    public static void main(String[] args) throws IOException{
        Scanner in = new Scanner(System.in);
        final String fileName = System.getenv("OUTPUT_PATH");
        BufferedWriter bw = new BufferedWriter(new FileWriter(fileName));
        String res;
        String _instructions;
        try {
            _instructions = in.nextLine();
        } catch (Exception e) {
            _instructions = null;
        }

        res = compute(_instructions);
        bw.write(res);
        bw.newLine();

        bw.close();
    }
}
```

Line: 12 Col: 1

Test against custom input

Run Code

Submit code & Continue

Download sample testcases *The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*