Ioiwari Game

PROBLEM

The Mancala family of games with beads and pits is among the oldest forms of human entertainment. This task introduces a version of the game especially developed for the IOI. The game is played by two players on a round board with seven pits around the edge. In addition, there is a bank for each player. The game begins by randomly distributing 20 beads into the pits so that each pit contains at least 2 and at most 4 beads. The two players move alternately. To move, the player chooses a non-empty pit and takes all beads out of the pit, and holds them in her hand. As long as there are beads in the player's hand, she considers the pits in clockwise order, starting one after the emptied one, and performs the following operations:

- More than one bead in your hand: If the current pit already contains 5 beads, then take one bead out of the current pit and place it into your bank, otherwise place one bead from your hand into the current pit.
- One bead in your hand: If the current pit contains at least one and at most four beads then move all beads from the pit and the one from your hand into your bank, otherwise (the pit contains 0 or 5 beads) place the bead in your hand into the opponent's bank.

The game is over when after a move all pits are empty and the winner is the player with most beads in her bank.

The starting player always has a winning strategy. You are to write a program, which plays Ioiwari as the starting player and wins. The evaluation opponent plays optimally, that is, once given a chance, it will win and your program will lose.

INPUT AND OUTPUT

Your program reads input from standard input and writes output to standard output. Your program is player 1, and the opponent is player 2. When your program is started, it must first read a line with 7 integers p_1,..p_7: the initial number of beads in pits 1,..7, respectively. The pits are labeled with integers from 1 to 7 in clockwise direction on the board. After this, the game starts with empty banks. Your program should play as follows:

- If it is your program's turn to move, then your program should write the label of the pit describing the move to standard output
- If it is your program's opponent's turn to move, then your program should read the label of the pit defining the move (the pit from which the beads are removed) from standard input.

TOOLS

You are given a program (`ioiwari2` on Linux, `ioiwari2.exe` on Windows), which plays from one initial game position optimally as Player 2. It will first write to

standard output the first line your program is supposed to read, describing the initial values of beads in that game: `4 3 2 4 2 3 2`

After this, the program will play the game, trying to read Player 1's moves from standard input and writing its own moves to standard output. You can run your program and ioiwari2 in separate windows and transfer the conversation manually to both programs. Ioiwari2 records the dialogue in the file `ioiwari.out`.

PROGRAMMING INSTRUCTIONS

In the examples below, you are reading the last integer of the input into variable `last` and the variable `mymove` contains your move.

If you program in C++ and use iostreams, you should use the following implementation for reading standard input and writing to standard output:

```
cout<<mymove<<endl<<flush;
cin>>last;
```

If you program in C or C++ and use scanf and printf, you should use the following implementation for reading standard input and writing to standard output:

```
printf("%d\n",mymove); fflush (stdout);
scanf ("%d", &last);
```

If you program in Pascal, you should use the following implementation of reading standard input and writing to standard output:

```
Writeln(mymove);
Readln(last);
```

EXAMPLE

Here is a correct sequence of 6 moves

	Pit and bank contents after the operation								
Operation/Pit label	1.	2.	3.	4.	5.	6.	7.	Bank1	Bank2
Initial situation	4	3	2	4	2	3	2	0	0
Player 1's move: 2	4	0	3	5	0	3	2	3	0
Player 2's move: 3	4	0	0	4	1	4	0	3	4
Player 1's move: 5	4	0	0	4	0	0	0	8	4
Player 2's move: 4	0	0	0	0	1	1	1	8	9
Player 1's move: 5	0	0	0	0	0	0	1	10	9
Player 2's move: 7	0	0	0	0	0	0	0	11	9

SCORING

If your program wins a test run, then you get 4 points for that test, a tie in a test gives you 2 points for that test, and otherwise you get 0 points for a test.