Programming Marvin

Marvin, the paranoid android is in another fit of existential depression. To relieve him of his manic tendencies, you have to issue commands to him to complete certain tasks. Unfortunately, he does not understand C++ or Python. You will have to write the code in Marvin's own programming language as described below.

Operation Area

Marvin stands on a path of 100 squares. He starts out at square 1, the leftmost square, and can go up till square 100 by moving towards the right. Each square may have blue and/or red stones placed on it. Marvin can pick up as many as you want, and he has an infinite supply of stones to drop.

Controlling Marvin's Actions

A list of instructions has to be supplied to Marvin, consisting of the following commands-

Syntax	Action
left	Moves one square to the left, unless on square 1 (leftmost square)
right	Moves one square to the right, unless on square 100 (rightmost square)
getb	Picks up one blue stone, if there is one on the current square, otherwise does nothing
getr	Picks up one red stone, if there is one on the current square, otherwise does nothing
putb	Adds one blue stone to the current square
putr	Adds one red stone to the current square
halt	Terminates execution

These commands are executed in the order that you write them

Example 1

```
right
putb
left
left
```

Marvin will move right to square 2, add a blue stone to that square and come back to square 1.

Labels and Jumps

Just like you can define labels for 'goto' in C++, you may define labels in this program. A label 'L' must consist only of alphanumeric characters.

Syntax (for any label L)	Action
L:	Declares a point in the program to which you can jump to later. Label definitions must be unique.
jump L	Continue executing after jumping to the line with label L
blue L	If there is a blue stone on the current square, continue executing after jumping to the line with label L
red L	If there is a red stone on the current square, continue executing after jumping to the line with label L

Example 2

```
red alphabit
halt
alphabit:
getr
right
putb
```

The label ‘alphabit’ is used in this program. If there is a red stone on square 1, Marvin will pick it up, move right and drop a blue stone. Otherwise, the program will halt.

THE TASKS

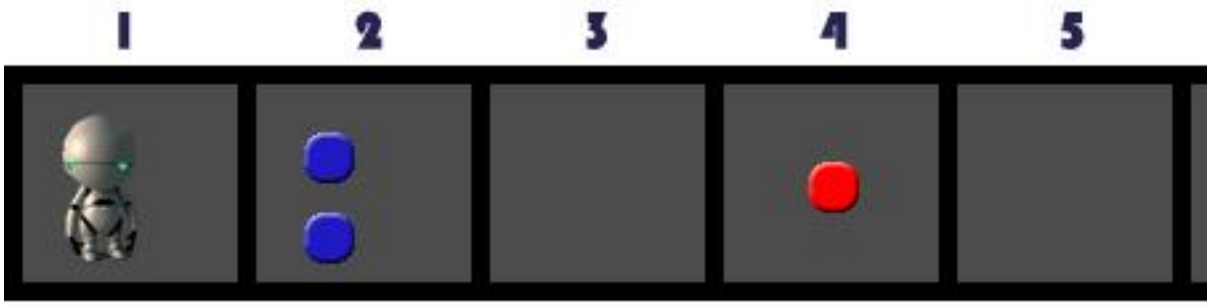
You will create and edit a file in notepad for each subtask you attempt. The files will be named “s1.txt”, “s2.txt”, “s3.txt”, “s4.txt” and “s5.txt” in the folder ‘AlphaBit’ on the desktop.

In case of ties, the entry with the shorter total execution length will be adjudged the winner.

Subtask 1 (20)

There is exactly one red stone on the path. Marvin must move right until he encounters the red stone.

Example case- Consider the path laid out as follows:



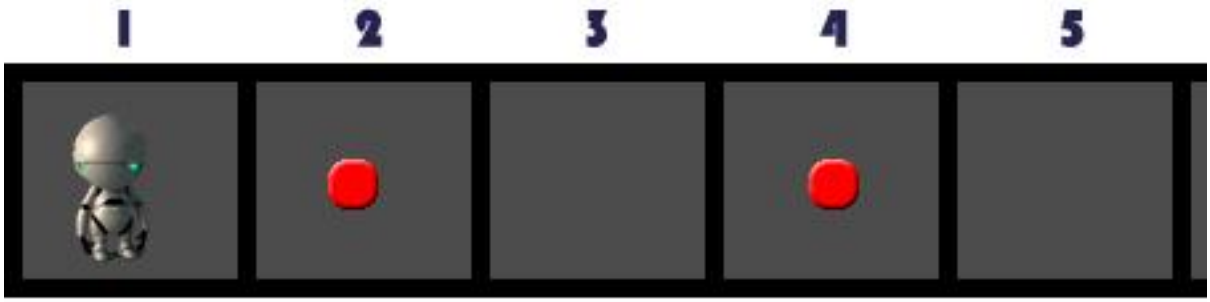
After execution, Marvin must stop on square 4 because it contains the red stone.

Only the final position of Marvin will determine your score.

Subtask 2 (20)

There are exactly two red stones on the path. Marvin must make both of them blue.

Example case- Consider the path laid out as follows:



After execution, it should look like this:

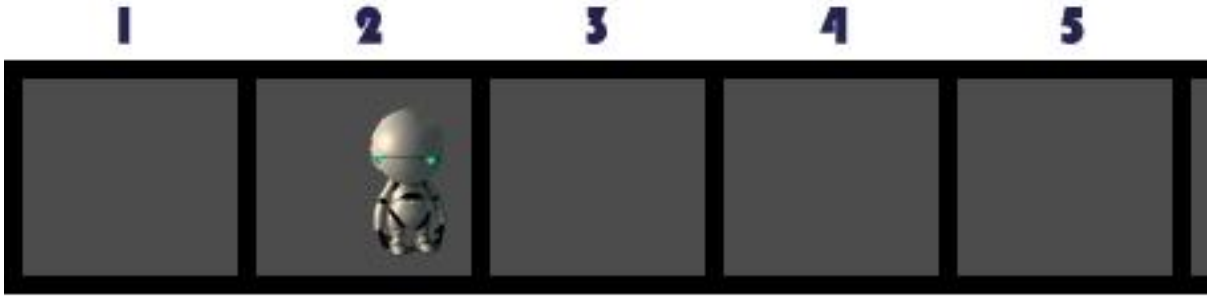
The final position of Marvin will not affect your score.

Subtask 3 (20)

There are a few red stones on square 1 and square 2. Marvin must stop at the square with the minimum number of red stones.

Example case- Consider the path laid out as follows:

After execution, Marvin must stop on square 2 in this case, because it contained the lesser number of stones.



Only the final position of Marvin will determine your score. (Hint; It is not necessary that all the stones remain on the path. They can be removed if required)

Subtask 4 (20)

Squares 1 and 2 contain stones of both colours. Marvin must sort them so that all red stones are on square 1 and all blue stones are on square 2.

Example case- Consider the path laid out as follows:

After sorting, the path should look like this:

The final position of Marvin will not affect your score.

Subtask 5 (20)

Each square will contain up to 2 blue stones. Marvin's task is to bring all blue stones to square 1.

You will be scored depending on how many execution steps your program takes. Thus, even if you solve this problem, try and make your algorithm as optimal as possible. No test case will be provided beforehand.

TESTING YOUR PROGRAM

A simulator has been provided which will test the program for you. For each subtask, the example given has been provided in files "t1.grid", "t2.grid", "t3.grid" and "t4.grid" respectively. To use the simulator, open the command prompt (which will open by default in the AlphaBit directory). Type

```
marvin.py <file name> -g <test file name>
```

For example, to test your code for Subtask 1, type

```
marvin.py s1.txt -g t1.grid
```