The PicoBot Challenge!

Adapted from the "Nifty Assignment" presented at SIGCSE 2010 by Zach Dodds and Wynn Vonnegut, Harvey Mudd College.

Computation - "A process that transitions from state to state under control of a program."

The key elements in our definition of a computation are:

- Process
- State
- Program

Process implies an algorithm, and a program defines the algorithm in terms of actions and state transitions. State defines the internal context of the computation. It is the state of the process and should not be confused with the state of the external view of the process.

In this lab, we are going to define computations for a robot called a PicoBot. "YAK! – Yet Another Karel!" you say. Actually, it is quite different.

A PicoBot is aware of its immediate surroundings in all four cardinal directions. That is, it knows whether or not it can move to the north, east, west or south. A PicoBot maintains a single integer variable which denotes its current state. A PicoBot program consists of a series of moves depending on the surroundings and the state. The resulting process can solve a number of interesting problems.

At each step, the PicoBot can optionally make a move according to its surroundings and it will always specify its next state. A state in a PicoBot program must specify exactly what the PicoBot is to do for each of the possible surroundings.

For example, a program that causes a PicoBot to move north until it can no longer move might be specified as follows:

```
State 0:
     If North is not blocked then move North and set state to 0
     else set state to 1
State 1:
     No matter what the surroundings are, set state to 1
```

Notice that each statement specifies what to do for all possible values of the surroundings.

PicoBot Programming

NOTE: This is an individual assignment. You may not work with anyone else. You must work by yourself, not with a partner, alone, not in a group, without the help of anyone other than your instructor.

A PicoBot program consists of a series of statements of the form:

```
        state    surroundings   ->       move   nextState
```

For each state, all possible combinations of the surroundings must be specified.

In your notebook, write down the number of possible surroundings for each state.

A state consists of an integer. States are numbered starting from 0.

Surroundings are specified using the format NEWS. If a direction is blocked, specify that with the capital letter corresponding to the direction. If a direction is not blocked, specify that with the letter x. If it does not matter whether or not the direction is blocked, specify that with the character *. For example, if the PicoBot is blocked to the north, not blocked to the south and we are not interested in the east and west directions, the surrounding are specified as "N**x".

A move is similarly specified using N,E,W,S or X where X means no move. Note that you may move in one, and only one, direction and that you do not have to move at all.

For each state, the PicoBot executes the first statement that matches its surroundings. You are not allowed to specify duplicate surroundings nor are you allowed to create two conflicting surrounding statements.

Here is an example of a PicoBot program that moves the robot as far north as possible:

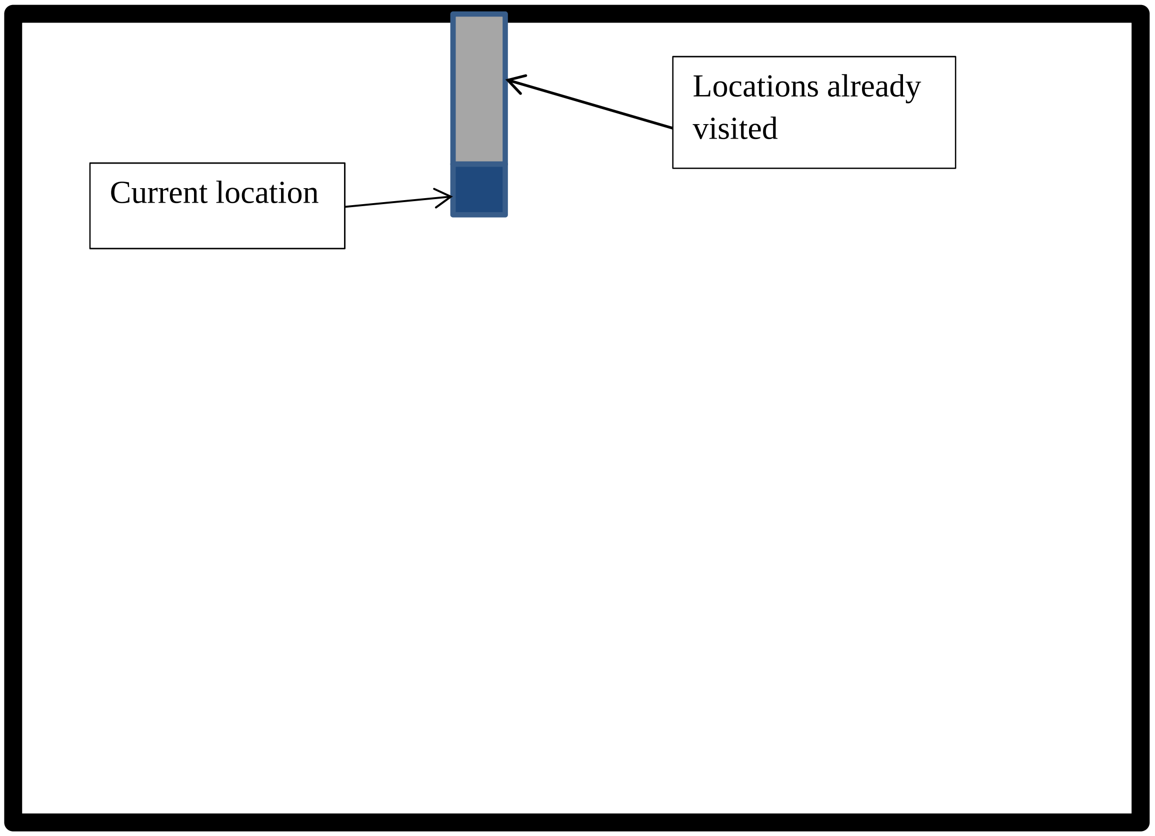
```
0       x***   ->     N          0
        N***   ->     x          1

1       ****   ->     x          1
```

It is possible to shorten this program to only one state. In your notebook, write the one state equivalent program.

Challenge Number 1 – The Empty Room

In this challenge, the PicoBot starts somewhere inside of an enclosed room that is otherwise empty. Your challenge is to specify a PicoBot program using the PicoBot language that will cause the PicoBot to visit every location within the room. This is equivalent to the "Roomba" vacuum robot problem. You must test and verify your program using the PicoBot simulator. To get checked off, submit your PicoBot program to Athena and demonstrate your program to your teacher.

Is it possible to devise a room where your PicoBot is incapable of visiting every possible location? In other words, are there rooms that are not computable using your PicoBot?

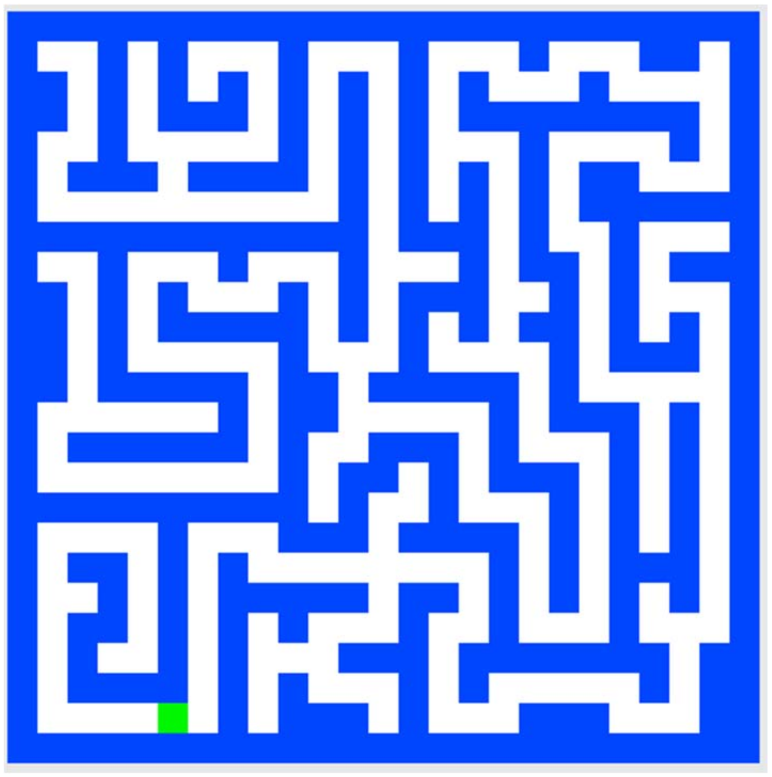


Example room showing PicoBot and area already visited

Challenge Number 2 – The Connected Maze

In this challenge, the PicoBot starts somewhere within the connected maze and your job is to write a PicoBot program that causes the PicoBot to visit every location within the maze. An example is shown below. You must test and verify your program using the PicoBot simulator. To get checked off, submit your PicoBot program to Athena and demonstrate your program to your teacher.

Is it possible to write a PicoBot program that will cause a PicoBot to solve any maze? Write your answer along with your reasoning in your notebook.



Instructions for completing the challenge

1. You **must** work alone, not in a group, not with another person.
2. Using the PicoBot programming language as shown in the introduction, you must define a sequence of commands that cause the PicoBot to perform its task.
3. The deliverable for this lab is a fully documented PicoBot program for each challenge submitted to Athena, a demonstration of your program using the PicoBot simulator and your notebook entries illustrating your program design as well as answering any questions posed. You may begin lines with the character # to denote a comment that is not part of the PicoBot program. Commenting is expected.
4. Here is the URL for the PicoBot simulator:

   For Firefox, Chrome, and other non-IE browsers:

   http://nifty.stanford.edu/2010/dodds-picobot/picobotPlain.html

   For IE browsers

   http://nifty.stanford.edu/2010/dodds-picobot/iePicobot/picobot.html

5. Once you have read the default rules, select the rules set, and delete, then enter your own rules. Below the Pico Room, select Go to execute your rules.