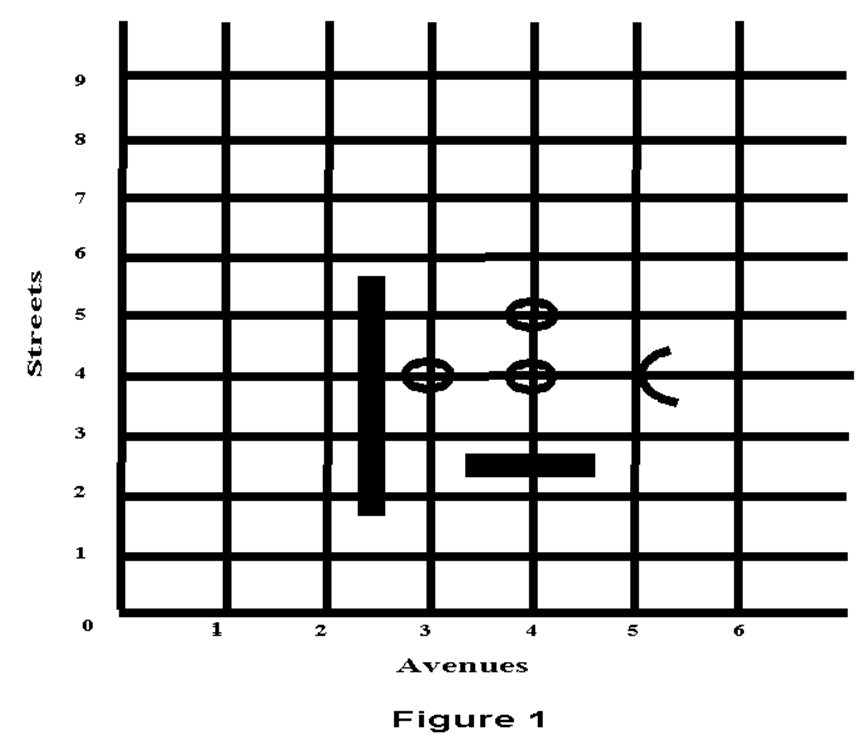
BILL’s WORLD ROBOT PROGRAMMING EXERCISE



Bill the robot exists in two dimensional space. His world is bounded on the west and south by two walls that extend infinitely to the north and east respectively. These walls prevent Bill from falling over the edge of his world. Crisscrossing Bill's world are horizontal streets (running east-west) and vertical avenues (running north-south) at regular one block intervals. A (street) corner is located at the intersection of a street and an avenue. Bill can be positioned on any corner, facing one of the four directions - east, west, north or south. Each corner can be identified uniquely by its street and avenue numbers.

In addition to Bill, there are two types of objects in this world. The first type of object is a wall section. Wall sections exist in various lengths and are positioned sideways between adjacent street corners, effectively blocking Bill's path from one corner to the next. Bill must navigate around these wall sections. The second type of object is a beeper. These beepers are placed on street corners and they emit a soft beeping sound that can only be heard by Bill if he is situated on the same corner as the beeper. Some of Bill's tasks involve picking up, transporting and laying down these beepers.

### Bill's Capabilities

Bill is a mobile robot. He can move forward in the direction he is facing, and he can turn in place. He has an internal compass that he can consult to determine the direction he is facing. Bill can see using his three TV cameras which point straight ahead, to his left, and to his right. These three cameras are focused to detect walls exactly one half of a block away from Bill. As stated earlier, Bill can hear beepers - only if he and the beeper are situated on the same street corner. Finally, Bill has a mechanical arm which he can use to pick up and lay down beepers. He carries these beepers in a soundproof bag. He can determine if there are any beepers in the bag by probing it with his arm.

### Bill's Language

Your task is to make Bill perform a variety of tasks. In order to do this you must supply him with a detailed set of instructions. Each instruction must be terminated with a semicolon. Bill is capable of receiving and memorizing these instructions - whch constitute a program. Bill cannot understand standard English. He has a very limited vocabulary and you must program him using a special programming language designed for programming robots. The following is the set of words that comprises Bill's vocabulary. Instructions for changing position, and handling beepers form Bill's basic or primitive command set.

### Bill's Command Set

* **Commands for Changing Position**

|  |  |
| --- | --- |
| move | This moves Bill forward one block in the direction he is facing. If there is a wall section obstructing his path, he will perform an error shutoff. |
| turnleft | This will make Bill pivot 90o to the left. |

* **Commands for Handling Beepers**

|  |  |
| --- | --- |
| pickbeeper | This instructs Bill to pick up the beeper on the current street corner and put it in his beeper bag. If there is no beeper, Bill will perform an error shutoff. If there is more than one beeper, Bill will randomly pick up one. |
| putbeeper | This instructs Bill to retrive a beeper from his beeper bag and put it on the current street corner. If there are no beepers in his bag, Bill will perform an error shutoff. |

* **[Defining New Commands](http://www.umsl.edu/~subramaniana/robot_commands.html)** (EXTENDING BILL’S Vocabulary below)
* **Conditional Execution Commands**

|  |  |
| --- | --- |
| IF [<test>](http://www.umsl.edu/~subramaniana/robot_test.html) THEN <instruction> | This tells Bill that IF test is true THEN the instructions should be executed. |
| IF [<test>](http://www.umsl.edu/~subramaniana/robot_test.html) THEN <instruction> ELSE <instruction> | Same as above, additionally it provides Bill with an alternate set of instructions to be executed if the test is not true. [Examples ...](http://www.umsl.edu/~subramaniana/robot_if_then.html) |

* **Repetitive Commands**

|  |  |
| --- | --- |
| ITERATE <positive-number> TIMES <instruction> | This instructs Bill to repeat the instruction positive-number of times. [Examples...](http://www.umsl.edu/~subramaniana/robot_iterate.html) |
| WHILE [<test>](http://www.umsl.edu/~subramaniana/robot_test.html) DO <instruction> | Instructs Bill to execute the instruction as long as the test is true. [Examples...](http://www.umsl.edu/~subramaniana/robot_while.html) |

* **Command for Finishing a Task**

|  |  |
| --- | --- |
| turnoff | This will switch Bill off. He will need to be restarted in order to perform another task. This should be the last instruction in a program. |

### Extending Bill's Vocabulary

Bill's vocabulary can be extended to include new words. However, these words are constructed from the set of primitive words. For example, a turnright instruction can be introduced as a set of three turnleft instructions. Because the turnright instruction comprises multiple primitive turnleft instructions, it must be defined as a block of turnleft instructions. The following is a sample structure of a program:

BEGINNING-OF-PROGRAM

DEFINE-NEW-INSTRUCTION turnright AS

BEGIN

turnleft;

turnleft;

turnleft;

END

BEGINNING-OF-EXECUTION

move;

turnleft;

move;

turnright;

pickbeeper;

turnoff;

END-OF-EXECUTION

END-OF-PROGRAM

### The Conditions Bill Can Test

|  |
| --- |
| front-is-clear, front-is-blocked |
| left-is-clear, left-is-blocked |
| right-is-clear, right-is-blocked |

* next-to-beeper, not-next-to-beeper

|  |
| --- |
| facing-north, not-facing-north |
| facing-south, not-facing-south |
| facing-east, not-facing-east |
| facing-west, not-facing-west |

* beepers-in-bag, no-beepers-in-bag

### Example of IF-THEN-ELSE

DEFINE-NEW-INSTRUCTION replant-exactly-one-beeper AS

BEGIN

IF not-next-to-beeper

THEN putbeeper;

ELSE next-to-one-replant-one;

END

DEFINE-NEW-INSTRUCTION next-to-one-replant-one AS

BEGIN

pickbeeper;

IF not-next-to-beeper

THEN putbeeper;

END

### Example of the ITERATE Command

DEFINE-NEW-INSTRUCTION harvest-a-row AS

BEGIN

pickbeeper;

ITERATE 4 TIMES

BEGIN

move;

pickbeeper;

END

END

### Example of the WHILE Command

DEFINE-NEW-INSTRUCTION go-to-beeper AS

BEGIN

WHILE not-next-to-beeper DO

move;

END

DEFINE-NEW-INSTRUCTION clear-corner-of-beepers AS

BEGIN

WHILE next-to-beeper DO

pickbeeper;

END

### Example of a Complete Robot Program

Write a program to enable Bill to do the following:  
Bill is currently positioned on the corner of 2nd street and 3rd avenue, facing east. Beepers are placed randomly on corners encompassed by 2nd and 7th streets, and 3rd and 7th avenues respectively. The area is framed by wall sections between the 7th and 8th avenues, and the 2nd and 3rd avenues. Bill is required to pick up all the beepers (without performing an error shutoff!!). NOTE: There may be more than one beeper on each street corner.

[Figure...](http://www.umsl.edu/~subramaniana/figure2.gif)

BEGINNING-OF-PROGRAM

DEFINE\_NEW-INSTRUCTION turnright AS

BEGIN

ITERATE 3 TIMES

turnleft;

END

DEFINE-NEW-INSTRUCTION position-for-next-two AS

BEGIN

turnright;

move;

turnright;

END

DEFINE-NEW-INSTRUCTION go-to-next-row AS

BEGIN

turnleft;

move;

turnleft;

END

DEFINE-NEW-INSTRUCTION clear-corner-of-beepers AS

BEGIN

WHILE next-to-beeper DO

pickbeeper;

END

DEFINE-NEW-INSTRUCTION pick-beeper-if-present AS

BEGIN

IF next-to-beeper

THEN clear-corner-of-beepers;

END

DEFINE-NEW-INSTRUCTION harvest-a-row AS

BEGIN

WHILE front-is-clear DO

BEGIN

pick-beeper-if-present;

move;

END

pickbeeper-if-present;

END

DEFINE-NEW-INSTRUCTION harvest-2-rows AS

BEGIN

harvest-a-row;

go-to-next-row;

harvest-a-row;

END

BEGINNING-OF-EXECUTION

ITERATE 3 TIMES

BEGIN

harvest-2-rows;

position-for-next-two;

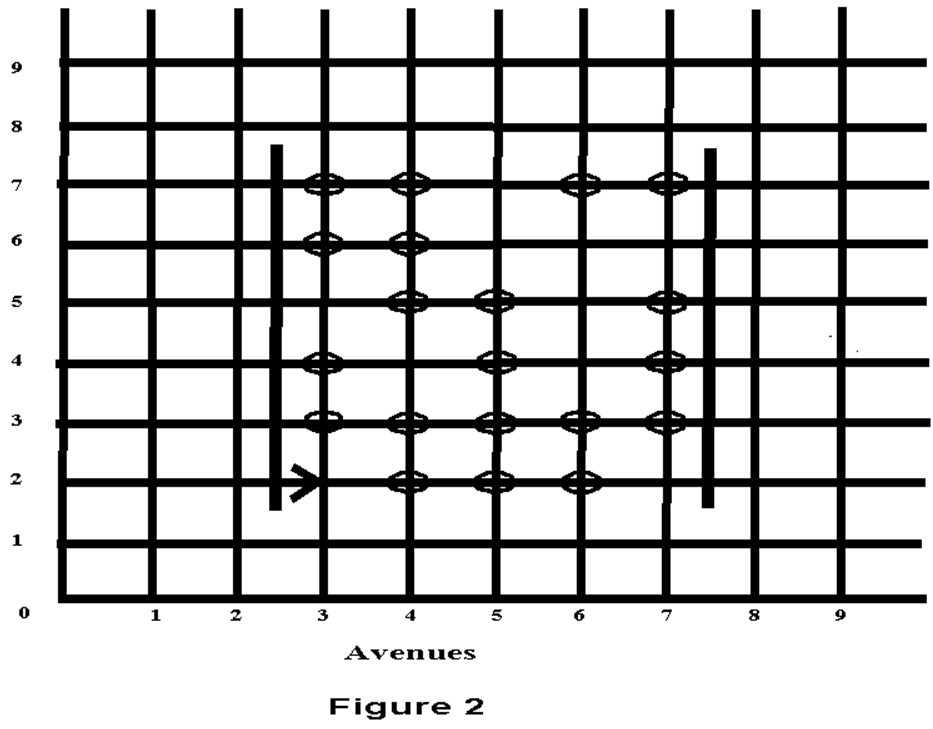
END

move;

turnoff;

END-OF-EXECUTION

END-OF-PROGRAM



### Exercises Based on Robot Bill

|  |  |
| --- | --- |
| 1. | Modify the program in the Example to program Bill to pick up beepers if the wall sections are removed. |
| 2. | Write a new instruction for Bill named empty-beeper-bag. After Bill executes this, his beeper bag should be empty. |
| 3. | Write a new instruction for Bill named go-to-origin. This should position Bill on 1st street and 1 st avenue regardless of his initial position and the direction he is facing. |
| 4. | Write an instruction that faces Bill east if he is on a corner with an even number of beepers, and faces him west if he is on a corner with an odd number of beepers (zero is considered an even number). |
| 5. | Program Bill to run a [Steeplechase](http://www.umsl.edu/~subramaniana/figure3.gif). In this race the hurdles are arbitrarily high and arbitrarily wide. The finish is marked by a beeper, which Bill must pick up before turning himself off. |

