The Bug class models a bug on a tightrope wire which is divided into unit positions where the bug can be placed.

The Bug has two behaviors.

The Bug can move ONE unit at a time on the wire in the direction which it is facing (right or left).

The Bug can turn around to face the other direction.

The Bug has two attributes: its position on the wire and the direction that it is facing: right or left.

The Bug has these behaviors

        void moveOne()

        void turnAround()

        int  getPosition()

        int getDirection()     // you may use another data type for the direction, Using 1 and -1 as values is shown here.

       void setPosition(int initialPosition)

Think carefully about the best way to model the bug

The wire is marked on in units and contains positions from -100 to 100

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ... | -5 | -4 | -3 | -2 | -1 | 0 | 1 | 2 | 3 | 4 | 5 | ... |
| left |  |  |  |  |  | D\*> |  |  |  |  |  | right |

This diagram is attempting to show the bug facing right at position 0.

The Bug class includes two constructors:

A.   Constructor which is passed the initial position on the wire and the direction it is currently facing.

B. Constructor which defaults to initial position of 0 and facing in the right direction.

Testing - create a BugTester application which uses all of the possible combinations of actions of behaviours.

Make a list of things to test and check them off.  
Instantiate two Bug objects (you can name it ladybug or beetle or whatever you like)

Test that an object can be constructed properly with both of the constructors.

Test the methods to make sure that the moveOne, turnAround, and getPosition methods are all working properly. Hint, you need to turn around a few times to be sure it works properly.

Finally  allow the user to input values for a new bug object and verify  that it works with all methods.

**Rubric**

Bug Class

| Bug Class | | |
| --- | --- | --- |
| **Criteria** | **Ratings** | **Pts** |
| This criterion is linked to a Learning Outcome A comment with your name as author and your honor code statement  At the beginning of each file, write a comment with your name as author and the honor code indicating that you did the work yourself. | |  |  | | --- | --- | | 5.0 pts  Full Marks | 0.0 pts  No Marks | | 5.0 pts |
| This criterion is linked to a Learning Outcome Create a Bug class with instance variables  Declare a Bug class as a blueprint. Use the attributes of the class to declare the instance variables Theses variables must be private. | |  |  | | --- | --- | | 10.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 10.0 pts |
| This criterion is linked to a Learning Outcome Supply two constructors for the Bug  Supply two constructors for the Bug: Write a constructor that contains formal parameters used to set all of the instance variables, AND Write a constructor that contains no formal parameters and sets the instance variables to be the default values. .   Range | |  |  | | --- | --- | | 10.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 10.0 pts |
| This criterion is linked to a Learning Outcome Implement the behaviors of the Bug  The Bug has these behaviors  void moveOne() ONLY one move one unit  void turnAround() change the direction  int getPosition() return position on wire int setPosition() change position on wire int getDirection() return facing which direction | |  |  | | --- | --- | | 15.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 15.0 pts |
| This criterion is linked to a Learning Outcome Declare a BugTester class and list the items to be tested  The list of items to be tested should include creating at least two objects of Bug class, one using the constructor to pass parameter values and one that is the default constructor. List the methods to be tested. | |  |  | | --- | --- | | 10.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 10.0 pts |
| This criterion is linked to a Learning Outcome Instantiate at least two Bug objects in the tester class  Instantiate two Bug objects by using the two different forms of the Bug constructor. | |  |  | | --- | --- | | 15.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 15.0 pts |
| This criterion is linked to a Learning Outcome Use the methods to manipulate the Bug objects  test all of the methods of the Bug class | |  |  | | --- | --- | | 20.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 20.0 pts |
| This criterion is linked to a Learning Outcome Have user input the values to create a new Bug object  Prompt the user to enter values for the instantiation of a Bug object using the constructor that allows values to be passed for instantiation. Verify that this Bug object works correctly. | |  |  | | --- | --- | | 15.0 to >0.0 pts  Full Marks | 0.0 pts  No Marks | | 15.0 pts |
|  | | | |