**Concordia University**
**COMP 248 – Winter 2017**
**Assignment 4**

| | |
|---|---|
| **Due Date:** | By 11:59pm Wednesday April 12, 2017 |
| **Evaluation:** | 5% of final mark (see marking rubric at the end of handout) |
| **Late Submission:** | none accepted |
| **Purpose:** | The purpose of this assignment is incorporate arrays of objects and the different types of statements we have seen since the beginning of the course into one more elaborate scenario. |
| **CEAB/CIPS Attributes:** | Design/Problem analysis/Communication Skills |

## General Guidelines When Writing Programs:

Please refer to the handout of Assignment #1.

## Question 1- Class Robot & Driver to test it

Part a) Class *Robot*

A robot has a name and moves in a 2-dimensional plane (square grid) in a specified direction. It starts at location (0,0) facing East and moves a randomly generated number of steps each turn until it reaches the top right corner of the grid, whose coordinates will be N x N, where N is the size of the square grid.

The (x,y) coordinates are to be modified as follows (see figure 1):

- If robot is facing E, it will move right, so the x coordinate is increased.
- If the robot is facing W, it will move left, so the x coordinate is decreased.
- If the robot is facing N, it will move up, so the y coordinate is increased.
- If the robot is facing S, it will move down, so the y coordinate is decreased.
-

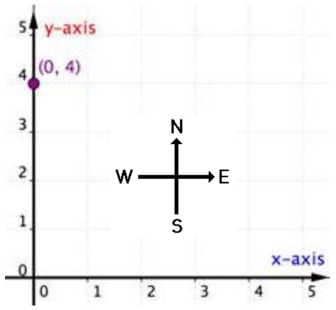


*Figure 1- Grid robot must move in if a 5x5 grid*

The robot cannot walk off the grid. So when it reaches the edge of the grid during any of its turns it needs to change directions and complete the remaining number of steps in the new direction.

Implement the class `Robot` based on the following specification. The class has

- four instance variables:
  - *name* that holds the name of the robot (which can be more than one word long)
  - *x* and *y* which are the integer coordinates of the robot's location (between 0 and N, where n is the size of the grid)

- *direction* a character indicating the direction the rboot is facing (possible values E, W, N and S)

- 3 constructors:
  - Default constructor: set name to "noName", direction to 'E' and both position corrdinates to 0.
  - One parameter constructor: sets the name to the passed name, direction to 'both position corrdinates to 0.
  - Copy constructor

- Accessor methods for each of the attributes
  - *getName()*
  - *getDirection()*
  - *getX()*
  - *getY()*

- Mutator method
  - *setName()* a void method which sets the name of the object to the passed String
  - no mutator methods for the *x* and *y* attributes or *direction* attribute. The only way these attributes can be changes is with the methods *move()* and *changeDirection()*, which are described below.

- *toString()* to return a robot's information in the following format:
  `name is facing direction and at position (x,y)`
  where the words in *italics* and orange are the values stored in the corresponding attributes of the object.

- *equals()* to return *true* if the position and direction of two objects are the same, `false` otherwise.

- *changeDirection()* a void method which changes the direction of the robot as follows (assigns a new value to the attribute *direction*):
  - if it is facing E, it changes to N
  - if it is facing W, it changes to S
  - if it is facing N, it changes to W
  - if it is facing S, it changes to E

- *move()* which has 2 parameters: the first specifies the numbers of steps the robot must take, and the second specifies the size of the grid. This method moves the robot the specified number of steps. It must make sure that the robot does not move off the grid.
  Here are a few examples for a robot moving on a 5 x 5 grid to illustrate the general behaviour:
  - if the robot is at position 0,0 and facing N and needs to move 4, after the call to the method move(), it will be at position 0,4 still facing N (figure 2).

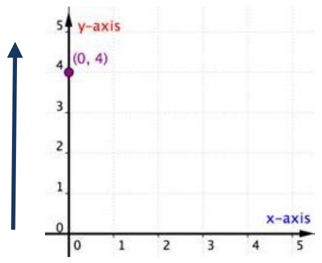


*Figure 2- Move from (0,0) to (0,4)*

  - if the robot is at position (5,4) facing N (figure 3) and needs to move 3 steps, it can move 1 step North without falling of the grid, meaning that it is at location (5,5) (figure 4). It now needs to change directions to complete the remaining 2 steps. When facing N, it changes directions to W, meaning it will move left (decrease x position). So the final position is (3,5) and the robot is facing W at the end of this turn (figure 5).

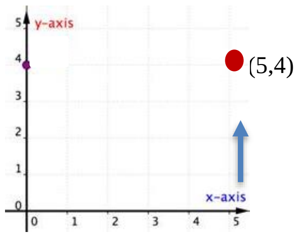


*Figure 3- Starting position, facing North*

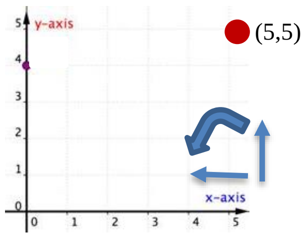


*Figure 4- Moves 1 of 3 steps North, still has 2 steps to make. Changes directions.*

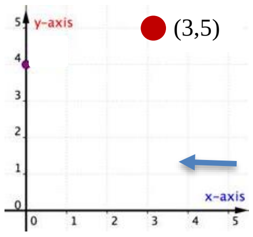


*Figure 5- Final position and facing West*

- *won()* is a boolean method which determines if a robot has reached the top left corner of the grid once it has completed a turn. This method receives an an integer parameter which represents the size of the square grid. The method returns *true* if both the *x* and *y* coordinates of the robot are equal to the size of the grid, *false* otherwise.

These methods must be in implemented in your class, and work as described above. You can add other public or private methods if you want.

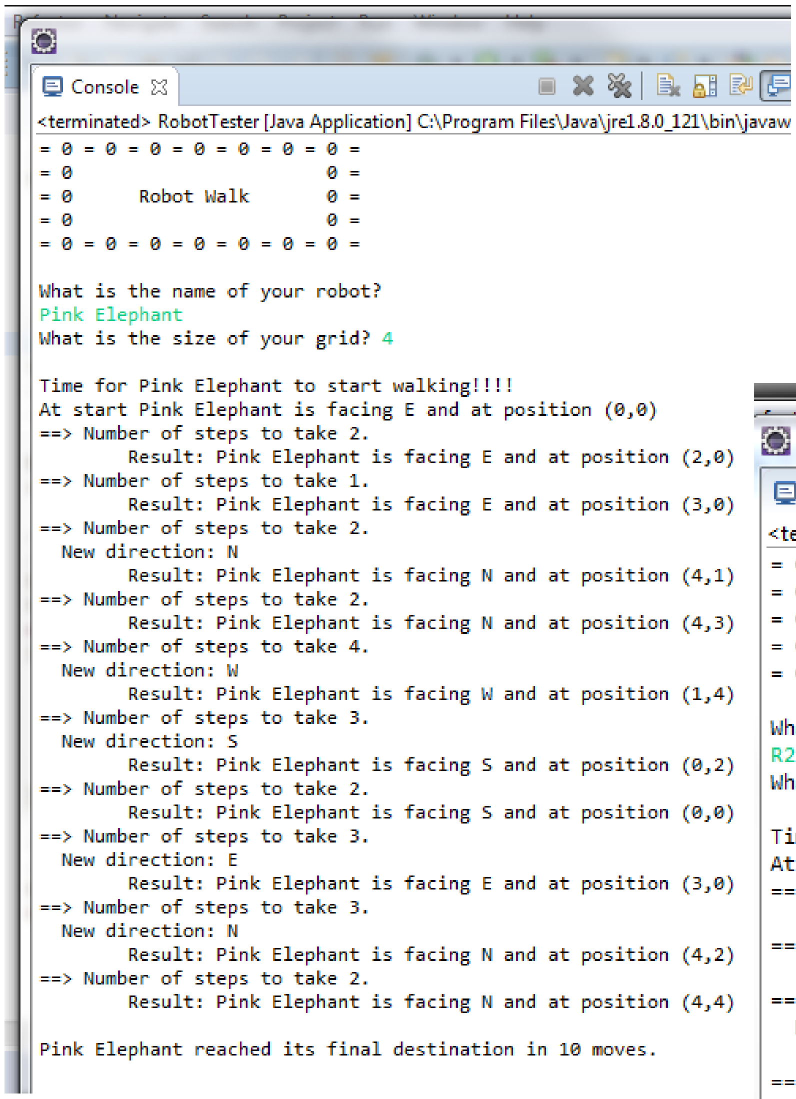
**Note**: It is possible that you don't use all of the methods you have implemented in *Robot* in this assignment.

Part b) Driver to test Class *Robot*
Write a java program using the *Robot* class from part a) of this question, to simulate the walking of a robot from the bottom left hand size of a NxN grid to the top right hand side of this grid.
Your program must prompt the user for the name of the robot (which can be longer than 1 word) and the size of the grid. It then randomly generates the number of steps (between 1 and N) and moves the robot based on the behaviour described in the class until it reaches its final destination of (N, N).
Trace the behaviour of the robot for each turn by displaying the number of steps it needs to walk, the direction changes and the coordinates of its position.
Once the robot has reached its final destination, write a closing message specifying the number of turns it took the robot to reach its final destination. Be sure to include its name in the message. Eventhough they are robots, they are sensitive and gets offended if we don't adress them with their name

Your driver must have 3 static methods:

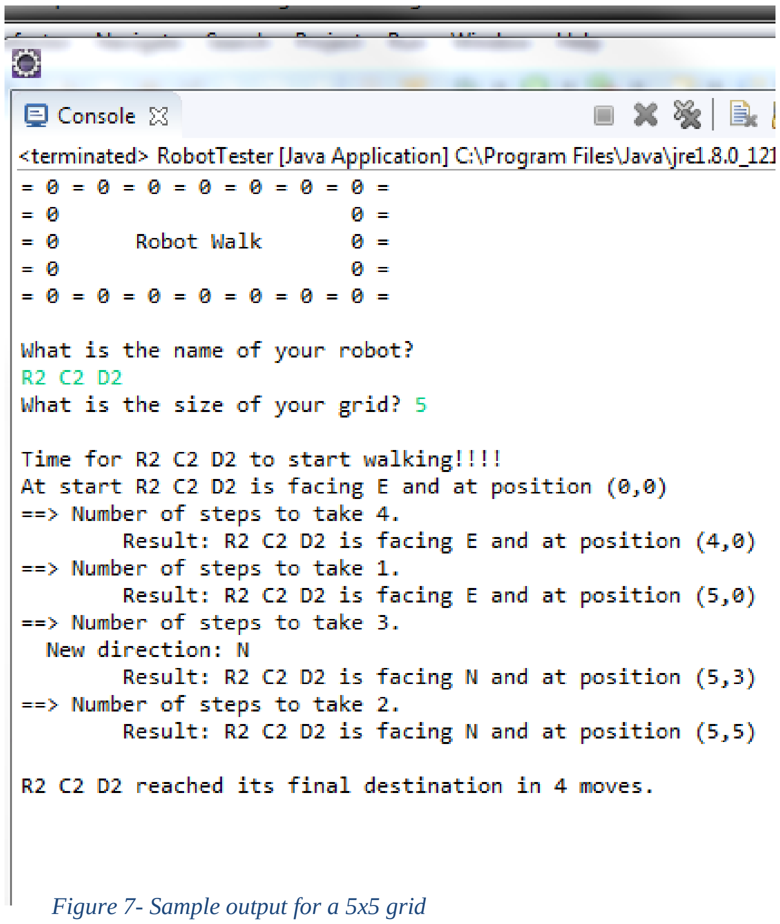
- a void method to display the header.
- a method which prompts the user for the size of the grid, reads it and returns the inputted value. This method requires the Scanner object as a parameter.
- A void method which displays the closing message. This method requires 2 parameters: the object and the number of turns.

Here are a couple of sample outputs to illustrate the expected behaviour of your code (figures 6 and 7).



```
<terminated> RobotTester [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw
= 0 = 0 = 0 = 0 = 0 = 0 = 0 =
= 0                     0 =
= 0       Robot Walk    0 =
= 0                     0 =
= 0 = 0 = 0 = 0 = 0 = 0 = 0 =

What is the name of your robot?
Pink Elephant
What is the size of your grid? 4

Time for Pink Elephant to start walking!!!!
At start Pink Elephant is facing E and at position (0,0)
==> Number of steps to take 2.
        Result: Pink Elephant is facing E and at position (2,0)
==> Number of steps to take 1.
        Result: Pink Elephant is facing E and at position (3,0)
==> Number of steps to take 2.
  New direction: N
        Result: Pink Elephant is facing N and at position (4,1)
==> Number of steps to take 2.
        Result: Pink Elephant is facing N and at position (4,3)
==> Number of steps to take 4.
  New direction: W
        Result: Pink Elephant is facing W and at position (1,4)
==> Number of steps to take 3.
  New direction: S
        Result: Pink Elephant is facing S and at position (0,2)
==> Number of steps to take 2.
        Result: Pink Elephant is facing S and at position (0,0)
==> Number of steps to take 3.
  New direction: E
        Result: Pink Elephant is facing E and at position (3,0)
==> Number of steps to take 3.
  New direction: N
        Result: Pink Elephant is facing N and at position (4,2)
==> Number of steps to take 2.
        Result: Pink Elephant is facing N and at position (4,4)

Pink Elephant reached its final destination in 10 moves.
```

*Figure 6- Sample output for a 4x4 grid*



```
<terminated> RobotTester [Java Application] C:\Program Files\Java\jre1.8.0_121
= 0 = 0 = 0 = 0 = 0 = 0 = 0 =
= 0                     0 =
= 0       Robot Walk    0 =
= 0                     0 =
= 0 = 0 = 0 = 0 = 0 = 0 = 0 =

What is the name of your robot?
R2 C2 D2
What is the size of your grid? 5

Time for R2 C2 D2 to start walking!!!!
At start R2 C2 D2 is facing E and at position (0,0)
==> Number of steps to take 4.
        Result: R2 C2 D2 is facing E and at position (4,0)
==> Number of steps to take 1.
        Result: R2 C2 D2 is facing E and at position (5,0)
==> Number of steps to take 3.
  New direction: N
        Result: R2 C2 D2 is facing N and at position (5,3)
==> Number of steps to take 2.
        Result: R2 C2 D2 is facing N and at position (5,5)

R2 C2 D2 reached its final destination in 4 moves.
```

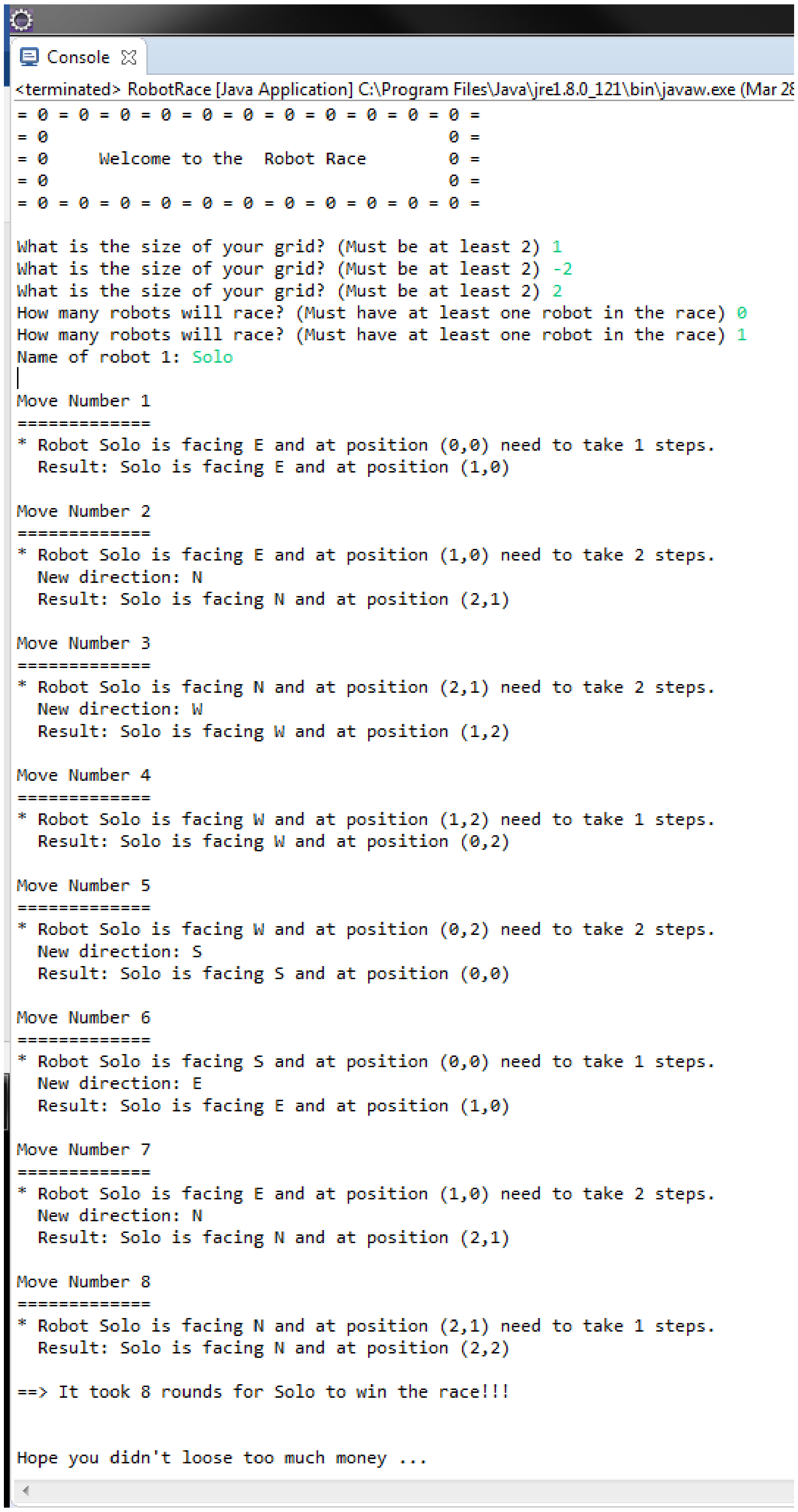
*Figure 7- Sample output for a 5x5 grid*

## **Question 2- Robot race**

Using the class Robot from Question 1 of this assignment, write a java program to simulate a robot race. Your program should ask the user the size of the grid, and the number of robots that will race, as well as their names. The robot objects must be stored in an array.
The winner is the 1st robot to reach the top right corner of the grid.

Your program must work for any size grid and any number of robots. Make sure the number of robots is at least 1 and that the grid size is at least 2x2. Make use of the static methods you wrote for the driver in Question 1. (Write a static method to read and validate the number of robots.)

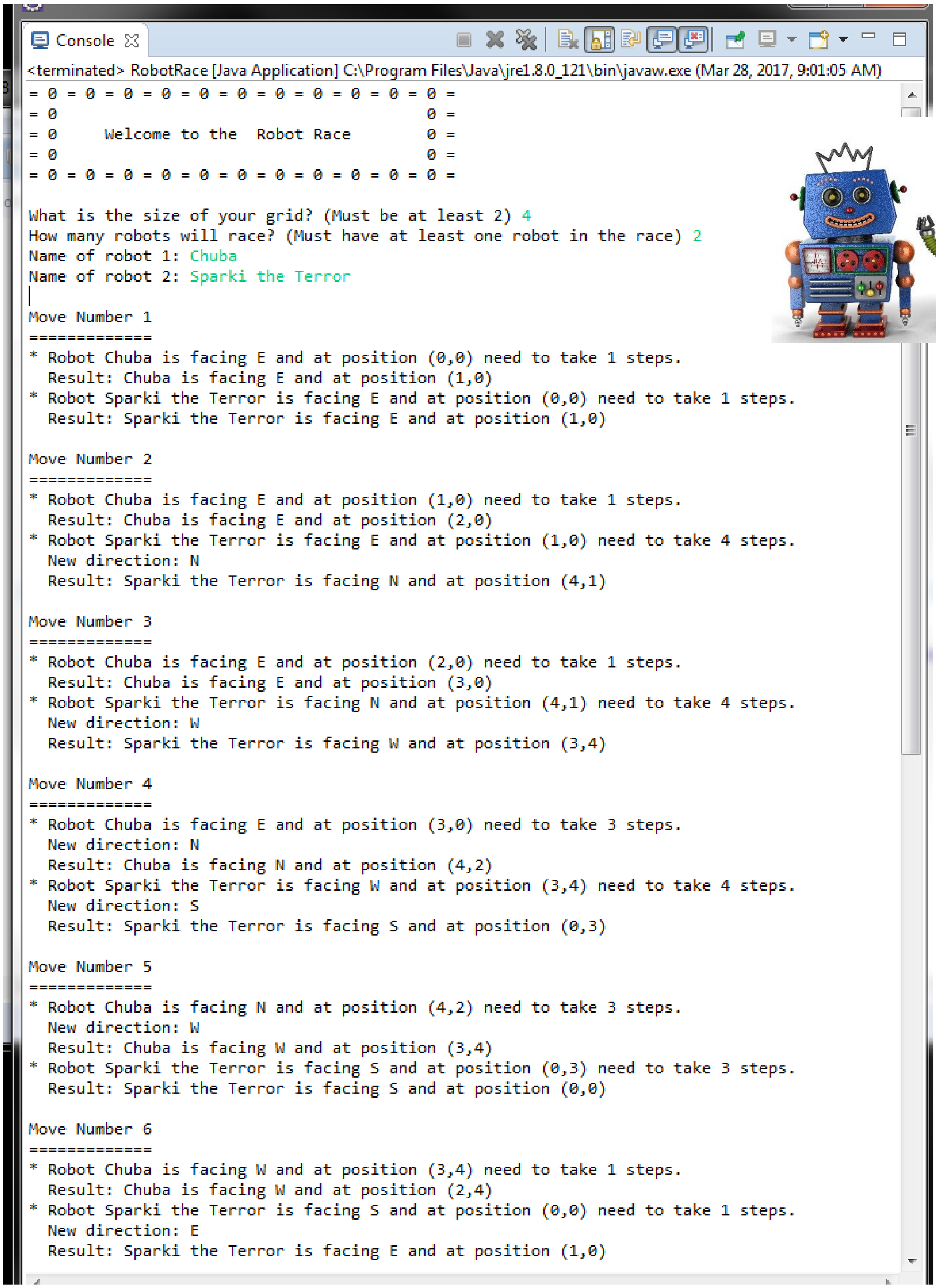
Here are a couple of screen shots to illustrate the expected behaviour of the race (figures 8 & 9).



Console

```
<terminated> RobotRace [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (Mar 28
= 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 =
= 0                                       0 =
= 0     Welcome to the  Robot Race        0 =
= 0                                       0 =
= 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 =

What is the size of your grid? (Must be at least 2) 1
What is the size of your grid? (Must be at least 2) -2
What is the size of your grid? (Must be at least 2) 2
How many robots will race? (Must have at least one robot in the race) 0
How many robots will race? (Must have at least one robot in the race) 1
Name of robot 1: Solo

Move Number 1
=============
* Robot Solo is facing E and at position (0,0) need to take 1 steps.
  Result: Solo is facing E and at position (1,0)

Move Number 2
=============
* Robot Solo is facing E and at position (1,0) need to take 2 steps.
  New direction: N
  Result: Solo is facing N and at position (2,1)

Move Number 3
=============
* Robot Solo is facing N and at position (2,1) need to take 2 steps.
  New direction: W
  Result: Solo is facing W and at position (1,2)

Move Number 4
=============
* Robot Solo is facing W and at position (1,2) need to take 1 steps.
  Result: Solo is facing W and at position (0,2)

Move Number 5
=============
* Robot Solo is facing W and at position (0,2) need to take 2 steps.
  New direction: S
  Result: Solo is facing S and at position (0,0)

Move Number 6
=============
* Robot Solo is facing S and at position (0,0) need to take 1 steps.
  New direction: E
  Result: Solo is facing E and at position (1,0)

Move Number 7
=============
* Robot Solo is facing E and at position (1,0) need to take 2 steps.
  New direction: N
  Result: Solo is facing N and at position (2,1)

Move Number 8
=============
* Robot Solo is facing N and at position (2,1) need to take 1 steps.
  Result: Solo is facing N and at position (2,2)

==> It took 8 rounds for Solo to win the race!!!


Hope you didn't loose too much money ...
```

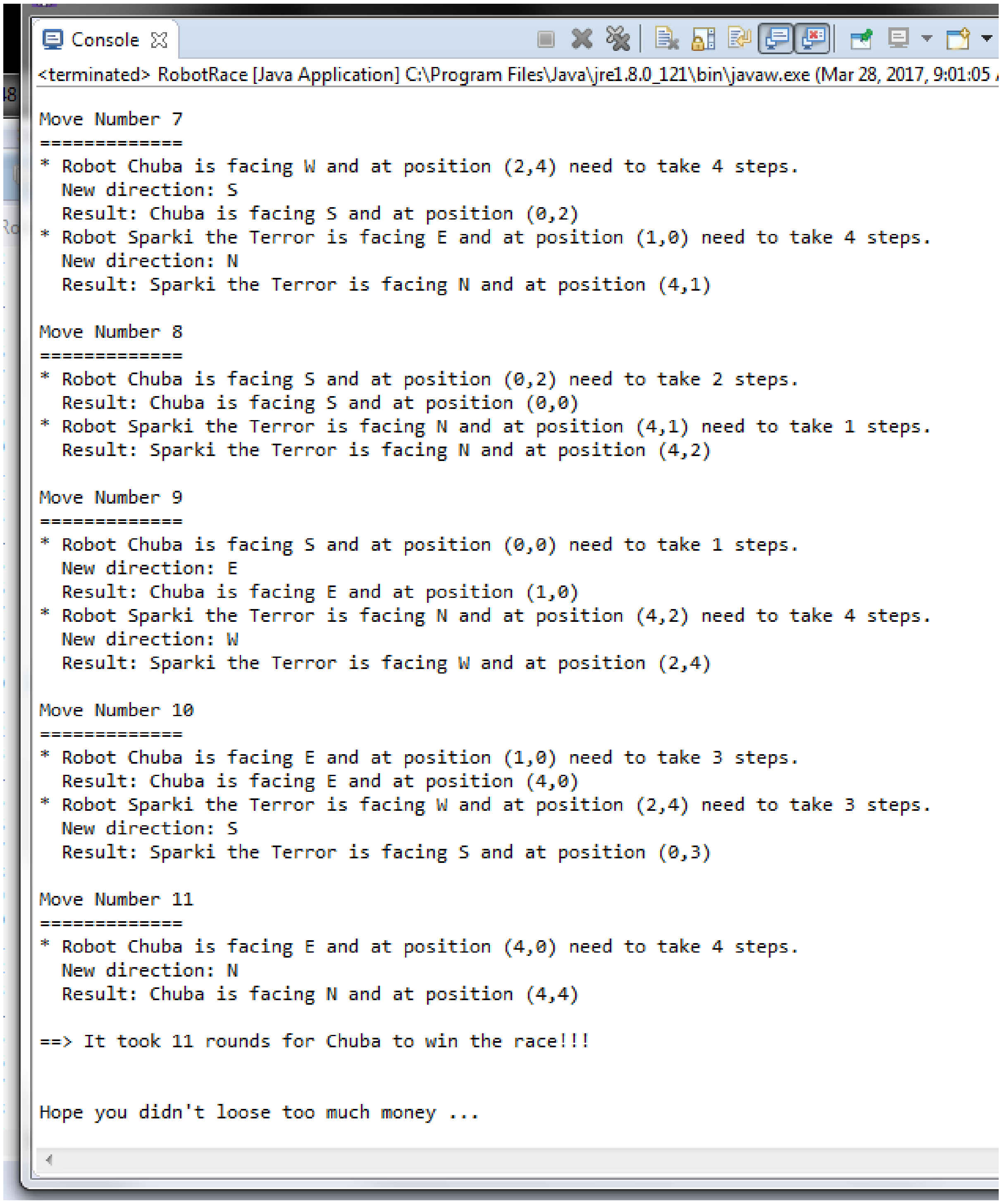
*Figure 8 - Screen shot to show behaviour if incorrect input*


Console

<terminated> RobotRace [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (Mar 28, 2017, 9:01:05 AM)

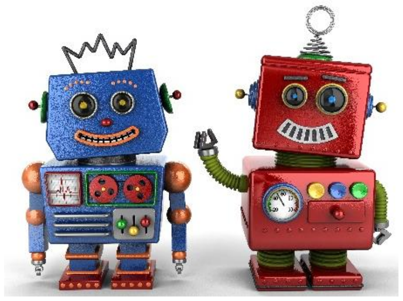
```
= 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 =
= 0                                       0 =
= 0     Welcome to the  Robot Race        0 =
= 0                                       0 =
= 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 = 0 =

What is the size of your grid? (Must be at least 2) 4
How many robots will race? (Must have at least one robot in the race) 2
Name of robot 1: Chuba
Name of robot 2: Sparki the Terror

Move Number 1
=============
* Robot Chuba is facing E and at position (0,0) need to take 1 steps.
  Result: Chuba is facing E and at position (1,0)
* Robot Sparki the Terror is facing E and at position (0,0) need to take 1 steps.
  Result: Sparki the Terror is facing E and at position (1,0)

Move Number 2
=============
* Robot Chuba is facing E and at position (1,0) need to take 1 steps.
  Result: Chuba is facing E and at position (2,0)
* Robot Sparki the Terror is facing E and at position (1,0) need to take 4 steps.
  New direction: N
  Result: Sparki the Terror is facing N and at position (4,1)

Move Number 3
=============
* Robot Chuba is facing E and at position (2,0) need to take 1 steps.
  Result: Chuba is facing E and at position (3,0)
* Robot Sparki the Terror is facing N and at position (4,1) need to take 4 steps.
  New direction: W
  Result: Sparki the Terror is facing W and at position (3,4)

Move Number 4
=============
* Robot Chuba is facing E and at position (3,0) need to take 3 steps.
  New direction: N
  Result: Chuba is facing N and at position (4,2)
* Robot Sparki the Terror is facing W and at position (3,4) need to take 4 steps.
  New direction: S
  Result: Sparki the Terror is facing S and at position (0,3)

Move Number 5
=============
* Robot Chuba is facing N and at position (4,2) need to take 3 steps.
  New direction: W
  Result: Chuba is facing W and at position (3,4)
* Robot Sparki the Terror is facing S and at position (0,3) need to take 3 steps.
  Result: Sparki the Terror is facing S and at position (0,0)

Move Number 6
=============
* Robot Chuba is facing W and at position (3,4) need to take 1 steps.
  Result: Chuba is facing W and at position (2,4)
* Robot Sparki the Terror is facing S and at position (0,0) need to take 1 steps.
  New direction: E
  Result: Sparki the Terror is facing E and at position (1,0)
```

*Figure 9 - Sample of race with 2 robots on a 4x4 grid (continued on next page)*



```
<terminated> RobotRace [Java Application] C:\Program Files\Java\jre1.8.0_121\bin\javaw.exe (Mar 28, 2017, 9:01:05

Move Number 7
=============
* Robot Chuba is facing W and at position (2,4) need to take 4 steps.
  New direction: S
  Result: Chuba is facing S and at position (0,2)
* Robot Sparki the Terror is facing E and at position (1,0) need to take 4 steps.
  New direction: N
  Result: Sparki the Terror is facing N and at position (4,1)

Move Number 8
=============
* Robot Chuba is facing S and at position (0,2) need to take 2 steps.
  Result: Chuba is facing S and at position (0,0)
* Robot Sparki the Terror is facing N and at position (4,1) need to take 1 steps.
  Result: Sparki the Terror is facing N and at position (4,2)

Move Number 9
=============
* Robot Chuba is facing S and at position (0,0) need to take 1 steps.
  New direction: E
  Result: Chuba is facing E and at position (1,0)
* Robot Sparki the Terror is facing N and at position (4,2) need to take 4 steps.
  New direction: W
  Result: Sparki the Terror is facing W and at position (2,4)

Move Number 10
=============
* Robot Chuba is facing E and at position (1,0) need to take 3 steps.
  Result: Chuba is facing E and at position (4,0)
* Robot Sparki the Terror is facing W and at position (2,4) need to take 3 steps.
  New direction: S
  Result: Sparki the Terror is facing S and at position (0,3)

Move Number 11
=============
* Robot Chuba is facing E and at position (4,0) need to take 4 steps.
  New direction: N
  Result: Chuba is facing N and at position (4,4)

==> It took 11 rounds for Chuba to win the race!!!


Hope you didn't loose too much money ...
```

*Figure 9 - Sample of race with 2 robots on a 4x4 grid (continued from previous page)*



## Submitting Assignment 4

- Make one zip file that includes all of the .java files.
- Naming convention for zip file: Refer to the handout for Assignment #3.
- For submission instructions please refer to the course web page.

- **<u>Assignments not submitted to the correct location or not in the requested format will not be graded</u>.**

## Evaluation Criteria for Assignment 4 (25 points)

| **Programming Style for the assignment** | **5 pts** |
|---|---|
| Comments/description of variables/description of the program (authors, date, purpose) | 1 |
| Clear prompts to user & clear message with output | 1 |
| Use of significant names for identifiers | 1 |
| Efficiency (no repeated code unnecessarily) | 1 |
| Indentation and readability | 1 |
| **Question 1 - Part a** (Class Robot) | **10 pts** |
| Constructors | 2.5 pts |
| Accessor/mutator methods | 2 pts |
| changeDirection() | 1 pt |
| move() | 2 pts |
| won() | 0.5 pts |
| equals() | 1 pt |
| toString() | 1 pt |
| **Question 1 – Part b** (Testing Driver) | **5 pts** |
| Static methods(3) | 2.5 pts |
| driver | 2.5 pts |
| **Question 2** (Driver for Race) | **5 pts** |
| Static method for number of robots | 0.5 pt |
| Set-up and use of array of Robots | 1 pt |
| Running race & declaring winner | 3.5 pts |
| **Total** | **25 pts** |