Title: If/Else statements

Author #1 Krishna Balasubramanian

Author #2 Kristian Horman

Assignment Definition And General Feedback By Michael Panitz at Cascadia Community College (http://www.cascadia.edu)

Table of contents:

- Quick Summary
- When To Use (or Avoid) This
- Example Of Usage
- Example Trace
- Syntax Explanation
- Help With The Logic
- Important Details
- Common Mistakes (And How To Fix Them)

Quick Summary:

In Java, you can use if/else statements to choose between two different commands or actions. The initial “if” part of the statement creates a test. If the parameters of the test are met, then the command or action associated with the “if” part will be performed. If the parameters are not met, then the “else” command or actions will be performed.

When To Use This / Avoid This:

There are several situations where you will want to use this technique. When there is a certain parameter that must be met before using a command or action, the if/else statement comes in very handy. It is best used to run a test that will determine whether the robot should use one action (given in the “if” part of the statement) or another action (given in the “else” part of the statement).

There are several situations where you will want to avoid this technique. The if/else statement doesn’t loop, which means that it will go through the if/else statement once and then move on to the next lines of code. The if/else statements are best to be avoided if the user wishes to use a command or action multiple times (depending on the parameters of a test), in those cases, a while statement would be more suitable.

Example Of Usage:

Let's say you want to have your robot pick up something in the next intersection, but you're not sure if there is a wall in front of it or if there is even a thing in the next intersection. The following program will demonstrate one way in which you can use **if/else statements** to have your robot maneuver around a wall (if there is a wall blocking its path) and pick up a thing (if it is even there!).

There are three simple steps:
First, you will need to have a regular old robot set up in a city, with the robot starting at (2,2), a wall at (2,2,E) and a thing at (2,3). This is done in lines 3-9

Second, you will need to write the code for the robot that will determine whether there is in fact a wall in front of it

Third

Line	**Code**
1.	**import** becker.robots.*;
2.	public class If_Else_Statements extends Object
3.	{
4.	public static void main(String[] args)
5.	{
6.	City woodinville = new City();
7.	Robot robocop = new Robot(woodinville, 2, 2, Direction.EAST, 0);
8.	
9.	new Thing(woodinville, 2, 3);
10.	new Wall(woodinville, 2, 2, Direction.EAST);
11.	
12.	if (robocop.frontIsClear())
13.	{
14.	robocop.move();
15.	}
16.	else
17.	{
18.	robocop.turnLeft();
19.	robocop.move();
20.	robocop.turnLeft();
21.	robocop.turnLeft();
22.	robocop.turnLeft();
23.	robocop.move();

```
24.             robocop.turnLeft();
25.             robocop.turnLeft();
26.             robocop.turnLeft();
27.             robocop.move();
28.             robocop.turnLeft();
29.         }
30.
31.         if (robocop.canPickThing())
32.         {
33.             robocop.pickThing();
34.         }
35.         else
36.         {
37.             System.out.println("THERE IS NOTHING TO
    PICK!");
38.         }
39.
40.     }
41. }
42. < Put the cursor here, then hit the 'Tab' key to get more rows >
```

Example Trace:

In a nutshell, this technique allows you to have a robot do one of two things based on a simple set of parameters that return either a true (do the first command listed) or a false (skip first command and do the second) answer.

In order to go over these details more thoroughly, here is a (partial) trace of the above program, with some additional explanation afterwards

Line #	Program Statement	Robocop St #	Robocop Ave #	Robocop Dir	Robocop Backpack	Thing	Wall	TRUE / FALSE
6	`City woodinville = new City();`	-	-	-	-	-	-	-
7	`Robot robocop = new Robot(woodinville, 2, 2, Direction.EAST, 0);`	2	2	E	0	-	-	-
9	`new`	2	2	E	0	(2,3)	-	-

	`Thing(woodinville, 2, 3);`							
10	`new Wall(woodinville, 2, 2, Direction.EAST);`	2	2	E	0	(2,3)	(2,2,E)	-
12	`if (robocop.frontIsClear())`	2	2	E	0	(2,3)	(2,2,E)	FALSE
16	`Else`	2	2	E	0	(2,3)	(2,2,E)	
18	`robocop.turnLeft();`	2	2	N	0	(2,3)	(2,2,E)	
19	`robocop.move();`	1	2	N	0	(2,3)	(2,2,E)	
20	`robocop.turnLeft();`	1	2	W	0	(2,3)	(2,2,E)	
21	`robocop.turnLeft();`	1	2	S	0	(2,3)	(2,2,E)	
22	`robocop.turnLeft();`	1	2	E	0	(2,3)	(2,2,E)	
23	`robocop.move();`	1	3	E	0	(2,3)	(2,2,E)	
24	`robocop.turnLeft();`	1	3	N	0	(2,3)	(2,2,E)	
25	`robocop.turnLeft();`	1	3	W	0	(2,3)	(2,2,E)	
26	`robocop.turnLeft();`	1	3	S	0	(2,3)	(2,2,E)	
27	`robocop.move();`	2	3	S	0	(2,3)	(2,2,E)	
28	`robocop.turnLeft();`	2	3	E	0	(2,3)	(2,2,E)	
31	`if (robocop.canPickThing())`	2	3	E	0	(2,3)	(2,2,E)	TRUE
33	`robocop.pickThing();`	2	3	E	1		(2,2,E)	

You'll notice that the trace starts at beginning of the **main** function, on line 6, like normal. It proceeds normally until line 12, were it goes into the first of 2 if/else statements. The first if / else statement is false and proceeds to the else statement in the code. Then on line 31 it proceeds into the second if / else statement which is true and once it completes that if statement you skip the else and the program ends.

Syntax Explanation:

Note on syntax of command: if/else statements are used to add intelligence to our robots so that they have more flexibility by testing a situation prior to taking action.

Line #	Program Source Code
1.	`import becker.robots.*;`
2.	`public class If_Else_Statements extends Object`
3.	`{`

```
4.      public static void main(String[] args)
5.      {
6.          City woodinville = new City();
7.          Robot robocop = new Robot(woodinville, 2, 2,
    Direction.EAST, 0);
8.
9.          new Thing(woodinville, 2, 3);
10.         new Wall(woodinville, 2, 2, Direction.EAST);
11.
12.         if (robocop.frontIsClear())
13.         {
14.                 robocop.move();
15.         }
16.         else
17.         {
18.                 robocop.turnLeft();
19.                 robocop.move();
20.                 robocop.turnLeft();
21.                 robocop.turnLeft();
22.                 robocop.turnLeft();
23.                 robocop.move();
24.                 robocop.turnLeft();
25.                 robocop.turnLeft();
26.                 robocop.turnLeft();
27.                 robocop.move();
28.                 robocop.turnLeft();
29.         }
30.         }
```

It is important to understand the importance of parts of an If/Else statement. Look at the highlighted portion of the code above. Note that on line 12 there is no semicolon at the end of the line because it is not a command, it is only outlining a test that will return either a true or false response. In the case of this example the test outlined is to check if the area in front of the robot is clear, we are not telling the robot to do anything but check to see if this statement is true or false. Next on line 13 we open bracket, which tells the program that anything after it until the close bracket are commands to execute if the test is true. For our example we simply want the robot to move forward once if the front is clear and you will notice on line 15 that the close bracket is used to signify the end of commands. The else portion of the statement is laid out the same way as the if portion it simply outlines what to do if the test response is false. Note that again the actual the brackets outline the commands that we want executed. You will also note that there is no test outlined by the else portion of the statement, this is due to it already being outlined above and the else portion is simply the response to a false answer returned. Remember an if/else statement only runs once and then the program will continue on, if you need a robot to do something multiple times look into how to create while statements.

Help With The Logic:

This is good to use whenever you want the robot to test a parameter before doing an action and based on the outcome of that test it will follow one command or another. This can be good if you only want the robot to do something once and then continue on with a program. Remember using if/else statements is a singular event and only happens once when used under main in a program. If/else statements can be very useful when used as part of a new command to get a robot to make decisions based on the parameters that you set.

How do we tell Java where the if/else statement begins? It starts on line 12 with `if (robocop.frontIsClear())`

Remember that in Java words that are in purple have to be typed letter for letter. Note that on line 13 the (**{**) open bracket denotes where the commands start if a TRUE response is received from the test and those commands end at the (**}**) close bracket. The else portion of the statement starts the line after the close bracket with the word else in purple and followed on the next line (17) by another (**{**) open bracket which denotes where the commands start if a FALSE response is received from the test. Just as before the command is concluded when we have a line that is simply a (**}**) close bracket (Line 29).

Important Details:

- The if and else lines are not commands themselves the simply denote the two different options that could happen if the test is TRUE or FALSE. That is why you do not use a (**;**) semicolon after either of these lines.

- Correct use of (**{**) brackets: After your ***if*** statement, on the nest line you need to open bracket then skip to the next line and type in the commands you want if the statement ends up true and after your last command skip to the next line and close brackets **before** moving on to your ***else*** statement otherwise it will all read as part of the if statement commands. Also remember to close bracket after you complete the commands for the else statement.

Mistakes People Commonly Make (And How To Fix Them):

- **Quick Name Of Mistake:** Forgetting the parentheses
  Detailed Example Of Error: `if (robocop.canPickThing  )`

Detailed Example Of Fix: (unless it's obvious): Make sure the command for the robot has parameters inside of the "if" statement.

- **Quick Name Of Mistake:** Accidentally putting a semi-colon after "if" statement
  Detailed Example Of Error: `if (robocop.canPickThing());`
  Detailed Example Of Fix: (unless it's obvious): Don't put a semi-colon there; the robot doesn't use the "if" statement as an actual command.

- **Quick Name Of Mistake:** Forgetting/Misplacing the brackets
  Detailed Example Of Error:

```
if (robocop.canPickThing())
{
     robocop.pickThing();

else

System.out.println("THERE IS NOTHING TO PICK!");
}
```

Detailed Example Of Fix: (unless it's obvious):

```
if (robocop.canPickThing())
{
     robocop.pickThing();
}
else
{
System.out.println("THERE IS NOTHING TO PICK!");
}
```

Licensing