Creating and Using If/Else Statements

Author #1: Kristi Diep
Author #2: Lucas Kostenko

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

Table of Contents:

Quick Summary:

In Java, you can use the if-else statements to perform an action if-else statements allow you to choose between two actions, you MUST do one of the two. However, which one is performed depends on whether it meets the given conditions stated by the parenthesis after the "if" statement or not. This sequence chooses between the choice "if" which is performed when the condition is met or the "else" choice if it is not met, the "else" statement chooses the latter of the two actions. The "if-else" will always result in an action whereas the "if" statement alone will sometimes not.

When To Use This / Avoid This:

Basically, you want to use the if-else statements when you need to make making either-or decisions. There are several situations where you will want to use this technique. Such situations will appear when the robot needs to decide whether an action must be taken based on certain variables. For example, if a city is created where a thing can be placed in the location (1, 2) or

left out, then the robot must decide whether to continue on, or pick the thing up. This should be used if the circumstance is unique to one situation.

There are several situations where you will want to avoid this technique; situations where the circumstances are repeated and where the situation would not be random. An example of this would be if there are several things placed in a row in front of the robot, it would be better to use a "while" loop rather than a long list of "if-then" statements.

Example Of Usage:

Let's say, in the city of Bothell a wall may or may not be generated at (3, 6) facing west, there is a thing placed at (3, 6), and a robot named jo is placed at (3, 5). The robot m

There are three steps:

First, you would ask the robot to check if there is a wall in front of it, at (3,6).

Second, if there isn't a wall there, you would ask the robot to directly forward to (3,6). Otherwise (this is the "else" part), the robot will go around the wall.

Third, the robot will pick the thing up.

Line	Code
1	`import becker.robots.*;`
2	`import java.util.Random;`
3	`class GoodRobot extends Robot`
4	`{`
5	`GoodRobot(City c, int st, int ave, Direction dir, int num)`
6	`{`
7	`super(c, st, ave, dir, num);`
8	`}`
9	`public void turnRight()`
10	`{`
11	`this.turnLeft();`
12	`this.turnLeft();`
13	`this.turnLeft();`
14	`}`
15	`public void goAround()`
16	`{`
18	`this.turnLeft();`
19	`this.move();`
20	`this.turnRight();`
21	`this.move();`
22	`this.turnRight();`
23	`this.move();`
24	`}`
25	`}`
26	`public class Starting_Template extends Object`
27	`{`
28	`public static void main(String[] args)`

```
29      {
30          City toronto = new City();
31          GoodRobot jo = new GoodRobot(toronto, 3, 0, Direction.EAST, 1);
32        new Thing(toronto, 3, 6);
36          Random generator = new Random();
37          int r;
38          r = generator.nextInt(2);
39          if(r == 0)
40          {
41          new Wall(toronto, 3, 6, Direction.WEST);
42          }
44            jo.move();
45            jo.move();
46            jo.move();
47            jo.move();
48            jo.move();
54          if(jo.frontIsClear())
55          {
56           jo.move();
57          }
58          else
59          {
60             jo.goAround();
62          }
63          jo.pickThing();
65      }
66  }
```

Example Trace:(True)

In a nutshell, this technique allows you to tell the robot to check if a condition is met and then to choose between two actions depending on whether the condition is met or not.

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

Line #	Program Statement	If wall	Robot St #	Robot Ave #	Robot Direction	Robot Back pack	Wall Pos.	Thing Pos
1	import becker.robots.*;	-	-	-	-	-	-	-
2	import java.util.Random;	-	-	-	-	-	-	-
28	public static void main(String[] args)	-	-		-	-	-	-
30	City toronto = new City();	-	-	-	-	-	-	-
31	GoodRobot jo = new GoodRobot(toronto, 3, 0, Direction.EAST, 1);	-	3	0	east	0	(3,6,W)	-
32	new Thing(toronto, 3, 6);	-	3	0	east	0	(3,6,W)	(3,6)

36	Random generator = new Random();	-	3	0	east	0	(3,6,W)	(3,6)
37	int r;	-	3	0	east	0	(3,6,W)	(3,6)
38	r = generator.nextInt(2);	-	3	0	east	0	(3,6,W)	(3,6)
39	if(r == 0)	-	3	0	east	0	(3,6,W)	(3,6)
41	new Wall(toronto, 3, 6, Direction.WEST);	-	3	0	east	0	(3,6,W)	(3,6)
44	jo.move();	-	3	1	east	0	(3,6,W)	(3,6)
45	jo.move();	-	3	2	east	0	(3,6,W)	(3,6)
46	jo.move();	-	3	3	east	0	(3,6,W)	(3,6)
47	jo.move();	-	3	4	east	0	(3,6,W)	(3,6)
48	jo.move();	-	3	5	east	0	(3,6,W)	(3,6)
54	if(jo.frontIsClear())	F	3	5	east	0	(3,6,W)	(3,6)
55	jo.move();	-	3	5	east	0	(3,6,W)	(3,6)
58	else	-	3	5	east	0	(3,6,W)	(3,6)
60	jo.goAround();	T	3	5	east	0	(3,6,W)	(3,6)
19	this.turnLeft();	-	3	5	north	0	(3,6,W)	(3,6)
20	this.move();	-	2	5	north	0	(3,6,W)	(3,6)
21	this.turnRight	-	2	5	north	0	(3,6,W)	(3,6)
12	this.turnLeft();	-	2	5	west	0	(3,6,W)	(3,6)
13	this.turnLeft();	-	2	5	south	0	(3,6,W)	(3,6)
14	this.turnLeft();	-	2	5	east	0	(3,6,W)	(3,6)
22	this.move();	-	2	6	east	0	(3,6,W)	(3,6)
23	this.turnRight();	-	2	6	east	0	(3,6,W)	(3,6)
12	this.turnLeft();	-	2	6	north	0	(3,6,W)	(3,6)
13	this.turnLeft();	-	2	6	west	0	(3,6,W)	(3,6)
14	this.turnLeft();	-	2	6	south	0	(3,6,W)	(3,6)
24	this.move();	-	3	6	south	0	(3,6,W)	(3,6)
25	jo.pickThing();	-	3	6	south	1	(3,6,W)	-

Example Trace:(False)

Line#	Program Statement	If wall	Robot St #	Robot Ave #	Robot Direction	Robot Backpack	Wall Pos.	Thing Pos
1	import becker.robots.*;	-	-	-	-	-	-	-
2	import java.util.Random;	-	-	-	-	-	-	-
28	public static void main(String[] args)	-	-		-	-	-	-
30	City toronto = new City();	-	-	-	-	-	-	-
31	GoodRobot jo = new GoodRobot(toronto, 3, 0, Direction.EAST, 1);	-	3	0	east	0	(3,6,W)	-
32	new Thing(toronto, 3, 6);	-	3	0	east	0	(3,6,W)	(3,6)
36	Random generator = new Random();	-	3	0	east	0	(3,6,W)	(3,6)
37	int r;	-	3	0	east	0	(3,6,W)	(3,6)
38	r = generator.nextInt(2);	-	3	0	east	0	(3,6,W)	(3,6)
39	if(r == 0)	-	3	0	east	0	(3,6,W)	(3,6)
41	new Wall(toronto, 3, 6, Direction.WEST);	-	3	0	east	0	(3,6,W)	(3,6)
44	jo.move();	-	3	1	east	0	(3,6,W)	(3,6)
45	jo.move();	-	3	2	east	0	(3,6,W)	(3,6)

46	jo.move();	-	3	3	east	0	(3,6,W)	(3,6)
47	jo.move();	-	3	4	east	0	(3,6,W)	(3,6)
48	jo.move();	-	3	5	east	0	(3,6,W)	(3,6)
54	if(jo.frontIsClear())	T	3	5	east	0	(3,6,W)	(3,6)
55	jo.move();	-	3	5	east	0	(3,6,W)	(3,6)
25	jo.pickThing();	-	3	6	south	1	(3,6,W)	-

You'll notice that the trace starts at beginning of the **main** function, on line 23, like normal. It proceeds normally until line 33, although it's worth noting that lines 26 and 27 are creating RobotThatCanTurnRight robots, instead of the normal "plain vanilla" Robot robots.

Syntax Explanation:

Note on syntax of command:

Let's start with the program as it's written here.

Line #	Program Source Code	Expanation
1.	class GoodRobot extends Robot	
2.	{	
3.	GoodRobot(City c, int st, int ave, Direction dir, int num)	
4.	{	
5.	super(c, st, ave, dir, num);	
6.	}	
7.	public void turnRight()	----------
8.	{	|
9.	this.turnLeft();	|
10.	this.turnLeft();	this is used for turning right
11.	this.turnLeft();	|
12.	}	----------
13.	public void goAround()	----------
14.	{	|
15.	this.turnLeft();	|
16.	this.move();	|
17.	this.turnRight();	if the robot needs to go around do this
18.	this.move();	|
19.	this.turnRight();	|
20.	this.move();	----------
21.	}	
22.	}	
23.	public class Starting_Template extends Object	

24.	{	
25.	public static void main(String[] args)	
26.	{	
27.	City toronto = new City();	
28.	GoodRobot jo = new GoodRobot(toronto, 3, 0, Direction.EAST, 1);	
29.	new Thing(toronto, 3, 6);	
30.	Random generator = new Random();	----------
31.	int r;	|
32.	r = generator.nextInt(2);	|
33.	if(r == 0)	|
34.	{	|randomly creates a wall
35.	new Wall(toronto, 3, 6, Direction.WEST);	|
36.	}	----------
37.	jo.move();	
38.	jo.move();	Moves up to the wall
39.	jo.move();	
40.	jo.move();	
41.	jo.move();	moves up to the possible wall
42.	if(jo.frontIsClear())	***If*** the front of the robot is clear
43.	{	
44.	jo.move();	move
45.	}	
46.	else	if not?
47.	{	
48.	jo.goAround();	initiates the "go around" method
49.		
50.	}	
51.	jo.pickThing();	picks it up

Help With The Logic:

This is good to use whenever a variable is implemented where different outcomes are needed and when the robot has a chance of breaking if it does not take the appropriate action. It is good to use as a safety net if you know where the variable might change and when the action has an uncertain outcome. For instance, when a wall or thing has a chance of being there, but isn't guaranteed to be there. The robot will break if it tries to pick an object up that isn't there or walk through the wall. The "if-else" statements will prevent the break by taking one of two actions depending on certain the whether the wall or thing is there or not. However, this is not good to use when you keep having to repeat the "if" statement. In that case, you would want to use a "while" loop statement so that it is less work for you and the program is easier to understand.

Important Details:

- Placement of the {}

The if stating the condition does not need the curly brackets, but both the action it will take if the condition is met or not met needs to have brackets.

```
Ex.    If( robot. frontIsClear())
```

Also, the command telling the robot to choose the second action is noted with "else" is not in brackets, but the action following the else is.

```
Ex.    Else
       {
       robot.goAround
       }
```

- When placed in **main** the name of the robot goes in front of the commands and check whereas when placed inside an extension the name in front of the commands and check is "this"

Ex. Inside Main :	Ex. Inside Extension:

if(robot.frontIsClear())
{
robot.move();
}
else
{
robot.goAround();
}
	if(this.frontIsClear())
	{
	this.move();
	}
	else
	{
	this.goAround();
	}

- There are two sets of parenthesis for the checking portion of the command. One for the what the robot is checking for and one for the action of checking.

```
Ex.    If( robot. frontIsClear())
```

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

Quick Name Of Mistake: Inside Extension with the name of the robot
Detailed Example Of Error:

```
if(robot.frontIsClear())
{
robot.move();
```

```
}
else
{
robot.goAround();
}
```

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

```
if(this.frontIsClear())
{
this.move();
}
else
{
this.goAround();
}
```


Plagiarism

If you believe that some or all of this document infringes on your intellectual property (i.e., part or all of this document is copied from something you've written) please immediately contact Mike Panitz at Cascadia Community College (perhaps using the Faculty And Staff Directory at http://www.cascadia.edu/pages/searchtemplate.aspx)