

CS 1301 Homework – Robot Web Writer

Due: Friday February 28th, before 11:55pm

This is a pair programming assignment!

You are expected to work with the person you have been paired with in class, and you are **both** responsible for submitting the exact same code to T-Square. Follow good pair-programming practices by working together at a single computer and switching the driver/navigator role frequently.

Your pair may collaborate with other students in this class. Collaboration means talking through problems, assisting with debugging, explaining a concept, etc. You should not exchange code or write code for other pairs. Collaboration at a reasonable level will not result in substantially similar code.

For pair programming assignments, you and your partner should turn in identical assignments.

Files to submit: **hw5.py**

For help:

-TA Helpdesk – Schedule posted on class website

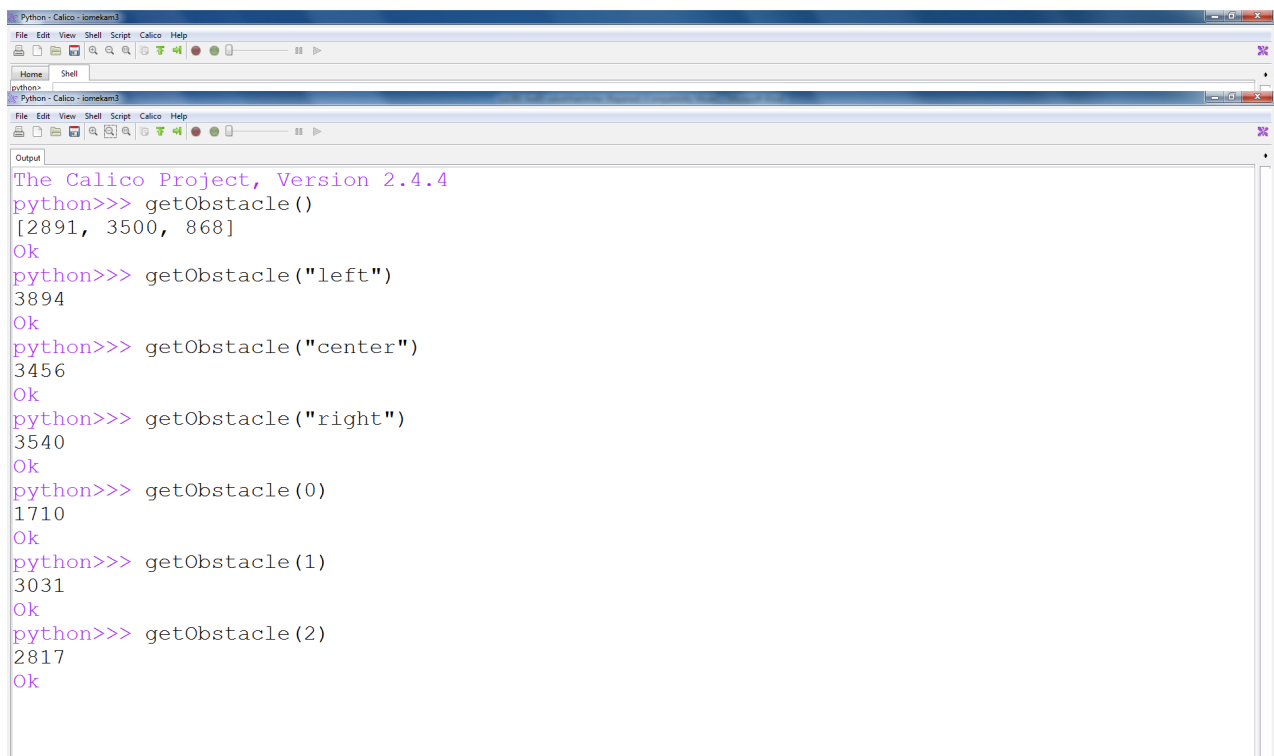
-Email TAs

Notes:

- **Don't forget to include the required comments and collaboration statement (as outlined on the course syllabus)**
- **Do not wait until the last minute** to do this assignment in case you run into problems.
- If you find a significant error in the homework assignment, please let a TA know immediately.

For this assignment, you will be learning two things: how the obstacle sensors on the Scribbler work, and how to *dynamically* create html webpages.

The obstacle sensors are three sensors that are present on the Fluke. To get the values from these sensors, we call the function **getObstacle()**. Depending on the parameter, this function will return a single value or a list of three values, ranging from zero to a very large number. If given no parameters, getObstacle() will return a **list** containing the values from the left, center, and right light sensor. If given a single parameter (such as 0, 1, or 2 or the strings “left”, “center”, or “right”), it will return a **single** value from the specified sensor. Note that Fluke2's only have one “center” obstacle sensor, so they will return the same value for all three options.



```
Python - Calico - somekam3
File Edit View Shell Script Calico Help
Python - Calico - somekam3
File Edit View Shell Script Calico Help
Output
The Calico Project, Version 2.4.4
python>>> getObstacle()
[2891, 3500, 868]
Ok
python>>> getObstacle("left")
3894
Ok
python>>> getObstacle("center")
3456
Ok
python>>> getObstacle("right")
3540
Ok
python>>> getObstacle(0)
1710
Ok
python>>> getObstacle(1)
3031
Ok
python>>> getObstacle(2)
2817
Ok
```

Your job will be to make a **function** called **makeWebPage** that **takes one parameter, numberOfPictures**. This parameter represents the number of pictures to take.

The function should take pictures using your Scribbler. Each picture should be taken in a different position. A way to do accomplish this is to have the Scribbler rotate or move after each picture. Each picture should then be saved in the format:

picx.jpg

where x represents the current count of the picture. **The count starts at zero!** For example, if 4 pictures were taken, they would be saved as: pic0.jpg, pic1.jpg, pic2.jpg, and pic3.jpg.

In addition, after each picture is taken, you should store the obstacle sensor value from the Scribbler's center sensor in a list. *Hint: if you add them to the list in order, then the value at `aList[2]` corresponds with `pic2`.*

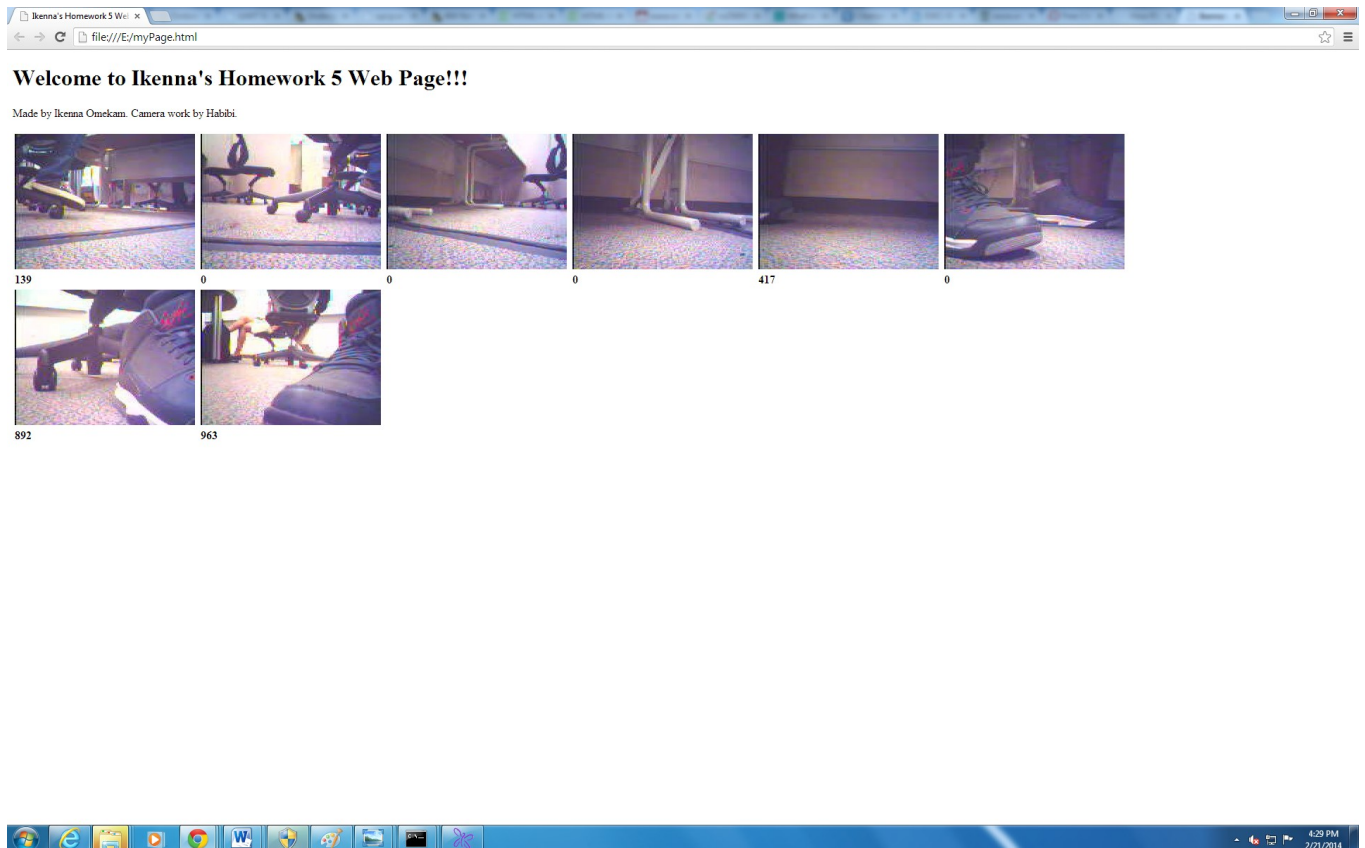
Once you have taken all your pictures and found all of the obstacle values, the function should then create an HTML file named **myPage.html**. On the page, have a descriptive title, a header welcoming the user to the page, your name(s) in the body of the document, a table showing the pictures you took with their corresponding obstacle sensor values below them, and the name of the robot that took the picture. *Your code should correctly generate the name of whatever robot is connected to the computer, and not just print your own robot's name!*

Each row of your table must be no wider than six pictures, so if the user asks you to take more than six pictures, you MUST wrap your pictures into one or more new row(s). (See examples below.)

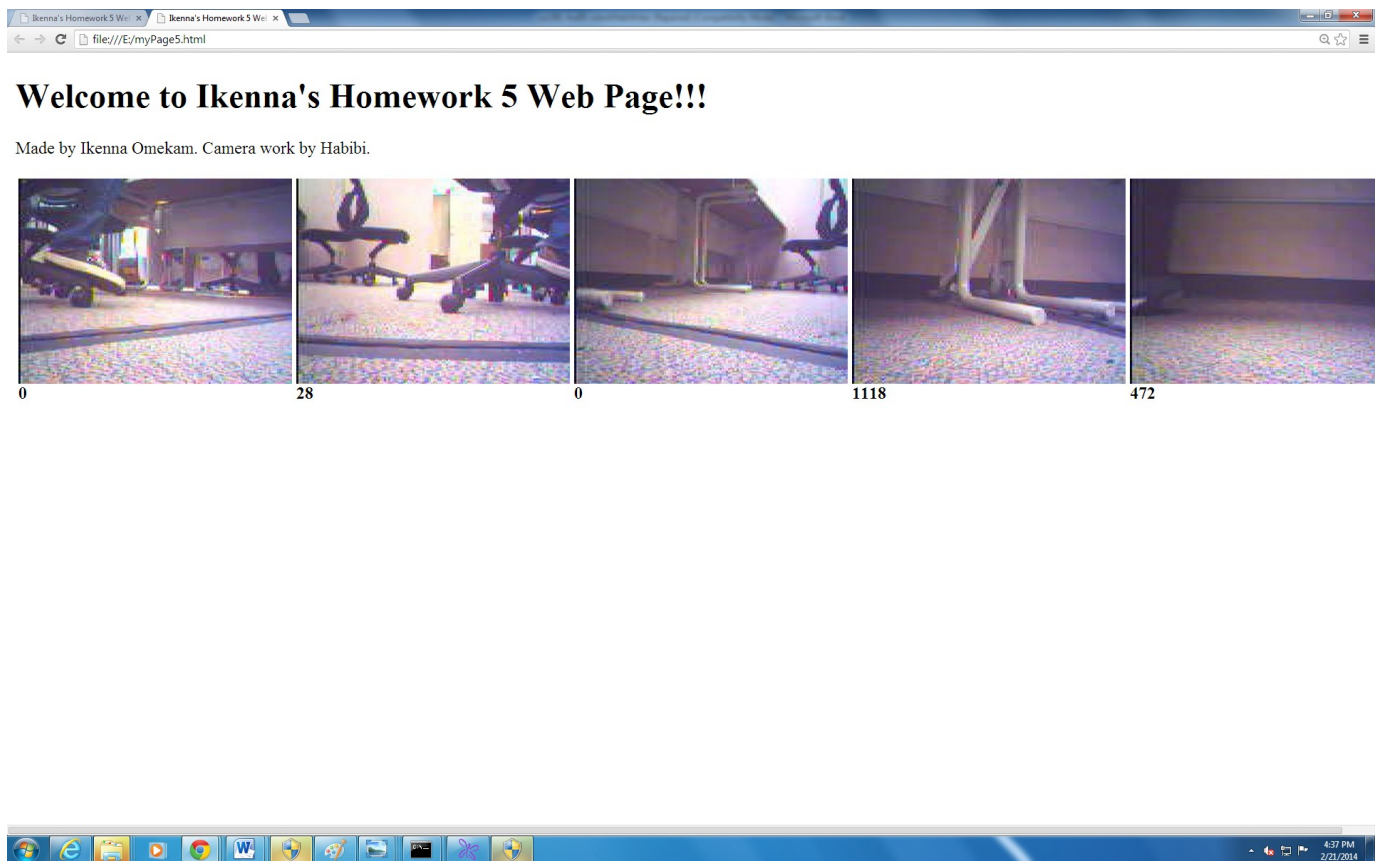
Your html file **MUST VALIDATE WITH NO ERRORS**, so be sure to use proper html tags and syntax in your file. To check if your html file validates, visit

http://validator.w3.org/#validate_by_upload

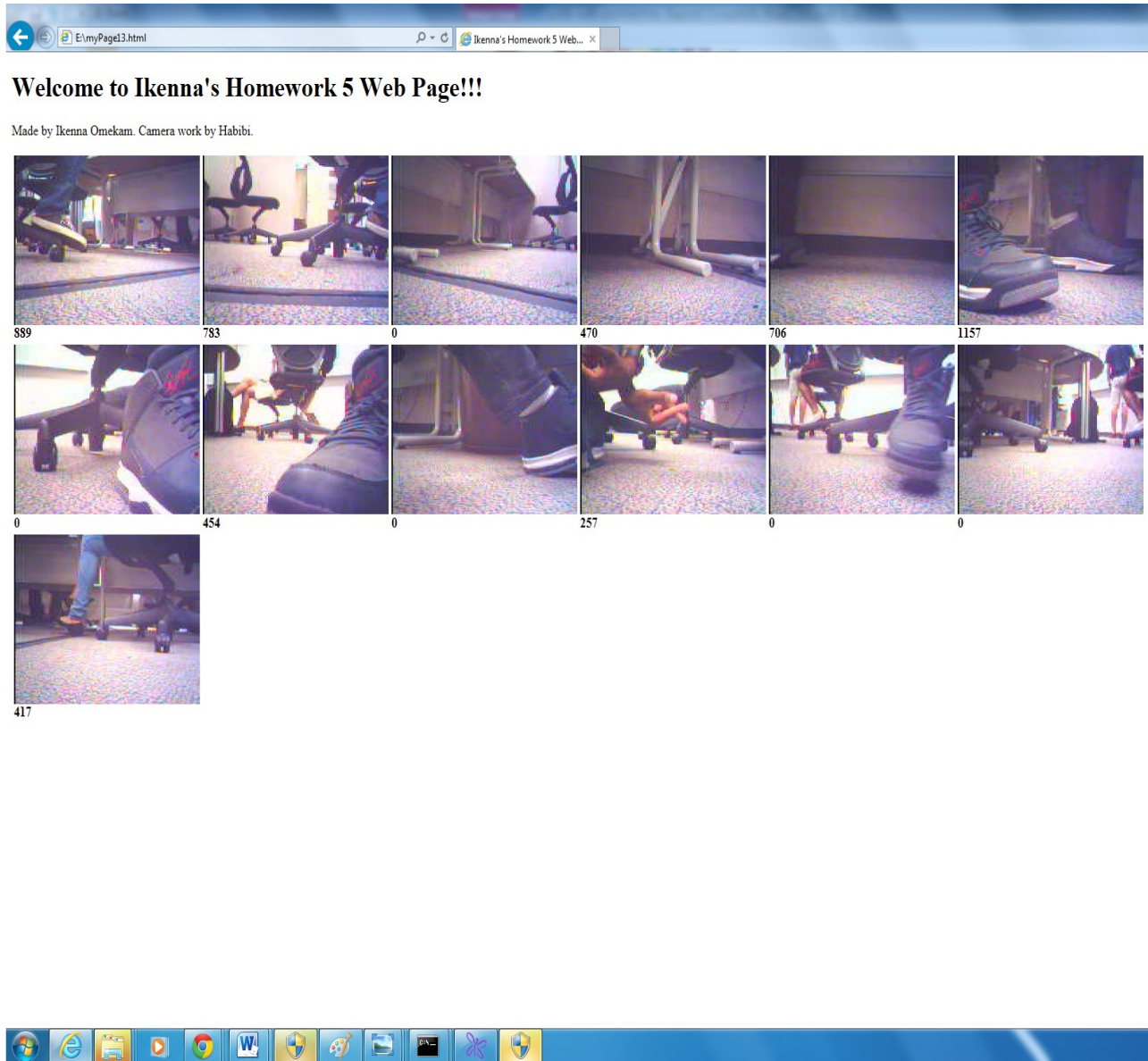
calling `makeWebPage(8)` should produce:



calling makeWebPage(5) should produce:



calling makeWebPage (13) should produce:



Grading:

Robot moves between taking each picture	5
Robot collects obstacle values with each picture	5
Program saves any number (X) pictures correctly	10
Program collects the obstacle sensor values correctly	10
Program opens correct file for writing	5
Program correctly closes the file.	5
Program writes required elements of the HTML page: (60 total)	
– Descriptive Title	5
– Welcome header text	5
– Name of student(s) in body	5
– Table showing the x photos and sensor values	10
– Table correct wraps every six pictures	10
– Photo credit giving name of the robot. (using getName())	5
– HTML Validates with no errors	20