Approximating π using Multiple Techniques

In this project you will approximate the value of π by 1) simulating throwing darts at a dartboard, and 2) using Wallis’s formula.

Open a new file called est_pi.py, in a new directory called proj6/, in your myCS106/ working directory.

Part 1: Monte Carlo Simulation

Think about drawing a circle on a piece of square plywood, such that the edges of the circle touch the 4 edges of the plywood. If you think about it, you'll see that the ratio of the area of the circle to the area of the square piece of plywood gives you an estimate for π, or actually π / 4. I.e., the circle covers about 78.5398163397% (approximately :-) of the square, which is about π / 4.

But, how do we get an estimate for the area in the circle? We can do this by simulating throwing lots and lots of darts at the plywood. We need to assume that the darts are thrown totally randomly, and, that all darts hit the plywood. Then, if we count the ratio of darts in the circle to the total number of darts, we get an estimate for π / 4. This is fairly easy, as all we have to do for each dart is figure out the distance from the dart to the center of the plywood square.

For the first part of this project, you are going to do this simulation. You need to:

1) Ask the user for the number of darts you want to throw at the plywood (using raw_input()).
2) Then, for each dart,
   a. Your code needs to get a random x value between [-1, 1] and a random y value between [-1, 1]. (You can assume the plywood square is 2 units high and 2 units wide.) This simulates where the dart lands on the plywood.
   b. Then, calculate the distance between the (x, y) for the dart and (0, 0) -- the center of the plywood.
   c. If the distance is < 1 (or is it <= 1?), increment a counter that keeps track of how many darts land in the circle.
3) After the loop, print out the ratio of how many darts landed in the circle to the total number of darts, times 4, as your estimate of π. Print out this value.

Part 2: Wallis Product
In this same file, under your code that computes pi using the Monte Carlo technique, write code to compute pi using the Wallis Product. Look at Wikipedia for a nice summary of Wallis Product. You just have to look at the top of the web page where it shows that $\pi/2 = 2/1 \times 2/3 \times 4/3 \times 4/5 \times 6/5 \times 6/7 \times 8/7 \ldots$

Your code should do this:
1) Print out a message saying you'll compute pi with the Wallis Formula. Then, ask the user for the number of terms to use.
2) Compute pi using the formula using that many terms.
3) Print out the results.

Grading Rubric:

<table>
<thead>
<tr>
<th>Category</th>
<th>Max points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program runs without crashing</td>
<td>8 (4 for each part)</td>
</tr>
<tr>
<td>Program produces correct output</td>
<td>4 (2 for each part)</td>
</tr>
<tr>
<td>Hospitable code!: Good variable names, comments, spacing, etc., following our coding conventions</td>
<td>3</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>15</strong></td>
</tr>
</tbody>
</table>

Submission mechanism: your file must be called `est_pi.py`. Create a folder ("directory") called `proj6` in your submission directory, located at:

```
/home/cs/106/current/<your-id>/
```

Thus, you'll have a directory called:

```
/home/cs/106/current/<your-id>/proj6/
```

Copy your proj6.py file into that directory. Thus, the end result is a file at:

```
/home/cs/106/current/<your-id>/proj6/est_pi.py
```