1 Administrative Topics

- Return the quiz.
- I will be emailing your project grades/comments to you today. Project grades will always be sent on the Mondays after they are due.
- Sign up for the csstu email list.

2 Math Operators

There are quite a few mathematical operators built in to Python. Table 1 contains a list of binary and unary operators.

Here are some interesting points regarding math in Python:

- The same precedence rules apply to the operations as in math. For operators with the same precedence, Python executes them from left to right.
- For floats, +,-,*,/ do what you expect.
- For integers +,-,* are also straightforward
Table 1: Unary and binary numeric operators in Python

<table>
<thead>
<tr>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>x + y</td>
<td>sum of $x$ and $y$</td>
</tr>
<tr>
<td>x - y</td>
<td>difference of $x$ and $y$</td>
</tr>
<tr>
<td>x * y</td>
<td>product of $x$ and $y$</td>
</tr>
<tr>
<td>x / y</td>
<td>quotient of $x$ and $y$</td>
</tr>
<tr>
<td>x % y</td>
<td>remainder of $x/y$</td>
</tr>
<tr>
<td>-x</td>
<td>$x$ negated</td>
</tr>
<tr>
<td>+x</td>
<td>$x$ unchanged</td>
</tr>
<tr>
<td>x ** y</td>
<td>$x$ to the power $y$</td>
</tr>
</tbody>
</table>

- What about `/` and `%` for integers? `/` results in the quotient and `%` results in the remainder, e.g. 5/3 is 1 and 5%3/ is 2.
- We can group together operations using parentheses the same way we can with mathematical expressions.

The rules of precedence for the binary operators are:

- `**` (highest)
- `*`, `/`,
- `+`, `-`

A word of advice, when making scale factor for your turtle drawings, never use 1/5. Because it is zero! There are several ways to make it become 0.2 (or, rather 0.200000000001 because the computer can’t represent 0.2 precisely). You can multiply by 1.0, or you can use the float function. Suppose you have two variables, A and B, and you want to scale your distances by A/B using float division. If A and B are floats, then you are all set. But there are no guarantees that A and B are floats. However, you can force them to perform float operations multiple ways, e.g.

```python
scale = 1.0 * A / B
scale = float(A) / B
scale = A / float(B)
```
3 Math Module

There are lots of mathematical functions available in the math module. For example, sin, cos, and other trigonometric functions. Also, there are a few “rounding” functions – floor, ceil, trunc. Finally, there are some interesting one to test whether or not a variable contains an odd value, like infinity or NaN (special symbol for “Not a Number”).

Also,

```python
import math
```

adds an entry in the symbol table with the name “math” and the value is of type `<module>`. The module is a collection of math functions. So we execute them like this:

```python
a = math.sin(0.0)
```

In other words, we provide Python the directions to the function – look up “math” in the main table, then “sin” within the module.

4 Random Module

The random module provides functions that generate pseudo-random numbers. I find this helpful for drawing starry skies and forests.

5 For loops

To draw a row of blocks, we could use a for loop.

```python
def rowOfBlocks( x, y, scale ):
    """draw a row of blocks with the bottom lefthand corner at (x,y)
    If scale is 1, then each block will be 200x200.""
    edge = 200
    for i in range(4):
        block( x+i*edge*scale, y, edge*scale )
```

We use a for loop to draw each of the four blocks. There are a few things to notice in the code
• We use scale to scale the edge length. The “natural” size for a block is 200x200.

• The “loop control variable” \( i \) has the value 0 the first time through the loop, 1 the second time through, 2 the third time through, and 3 the fourth time through. So we take advantage of that to determine the x-position of the \( i \)th block.

• We scale the offset in the x position, but not the x-position itself \((x + i \times \text{edge} \times \text{scale})\).

6 Debugging

Here are three bits of advice I can share with you about debugging your code:

1. Read the Python error message carefully. In particular, that message will give you the line on which the error was encountered. Set up your text editor to display line numbers to make it trivial to find the line with the problem.

2. When there is a syntax error reported on line \( X \), you might want to look at line \( X-1 \) to see if there is a problem like a missing colon or a missing close parenthesis.

3. Use \texttt{print} statements like crazy. If you are unsure about a calculation you are doing, then print it out before you use it.

7 Booleans

A Boolean value is either True or False. The name Boolean comes from George Boole, who invented Boolean Algebra and Boolean Logic.

they are \texttt{True} and \texttt{False}

And Python calls this type “\texttt{<bool>}”.

There are many built-in operators that return Boolean values.
8 Comparison (Relational) Operators

The comparison operators are listed below. They work for all types in Python, though we are going to focus on them for numerics.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;</td>
<td>strictly less than</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>strictly greater than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal</td>
</tr>
<tr>
<td>==</td>
<td>equal</td>
</tr>
<tr>
<td>!=</td>
<td>not equal</td>
</tr>
</tbody>
</table>

The precedence of these operators is lower than that of the mathematical operators. All of these operators have equal precedence to each other and are evaluated from left to right.