Creating list of lists

Can create list of lists in Python

\[
\text{table} = [[1, 2], [3, 6], [7, -3], [5, 6]]
\]

- Access an element with 2 subscripts.
- By convention first subscript is row and the second is the column

\[
\begin{array}{cc}
0 & 1 \\
0 & 1 \\
3 & 6 \\
7 & -3 \\
5 & 6 \\
\end{array}
\]

index of row

index of column

access element with 2 subscripts:

\[
\text{table}[2][0] \to 7
\]

Can also use list comprehension

\[
\text{table2} = [[0] * 12] * 10
\]

A list of lists with 10 rows and 12 columns per row.

\[
\text{flips} = [['H' if random.random() <= 0.5 else 'T' for x in range(12)] for x in range(10)]
\]

A table with 10 rows and 12 columns per row. Each element is a random coin flip.

Example of using a list of lists

Conway's Game of Life

- a cellular automaton designed by John Conway, a mathematician
- not really a game
- a simulation
- takes place on a 2d grid
- each element of the grid is occupied or empty by a simple organism, but not any known organism
Simulation

http://www.cuug.ab.ca/dewara/life/life.html

- Select pattern from menu
- Select region in large area with mouse by pressing the control key and left click at the same time
- Select the paste button

Generation 0

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.</td>
<td>*</td>
<td>.</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
<tr>
<td>1</td>
<td>*</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
</tr>
<tr>
<td>2</td>
<td>.</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
</tbody>
</table>

* indicates occupied, . indicates empty

Or

Generation 1

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>.</td>
<td>*</td>
<td>.</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
<tr>
<td>1</td>
<td>.</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
<tr>
<td>2</td>
<td>.</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
<tr>
<td>3</td>
<td>.</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>.</td>
<td>*</td>
</tr>
</tbody>
</table>

* indicates occupied, . indicates empty
Or, Generation 1

Rules of the "Game"

- If a cell is occupied in this generation:
  - it survives if it has 2 or 3 neighbors in this generation
  - it dies if it has 0 or 1 neighbors in this generation
  - it dies if it has 4 or more neighbors in this generation

- If a cell is unoccupied in this generation:
  - there is a birth if it has exactly 3 neighboring cells that are occupied in this generation

Neighboring cells are up, down, left, right, and diagonal. In general a cell has 8 neighboring cells

Case study

Design and implement a complete Python program to automate Conway's Game of Life

- text based
- user input for size of world
- wrapped or bounded?
- border or not?
- high level design first, then implement solution
- test, test, test, test