Recursion Continued
Visiting a Maze
How do we represent a maze in Python?
Step 1: String Definition

mazeText = ""

############################
........................#
############################
#.##.##........#.#........#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
#.##.##.###.#####.#####.#...#
Step 2: Replace dots by spaces
Step 3: Split into List of Strings
maze = [
',
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'.
'].

Step 3: Split into List of Strings

maze = [———, ——-,… ]
Result: Each row can be accessed with an index
```python
maze = [
    '#####################',
    '                     #',
    '#####################
    ' # # #   # #   # #   #',
    '# # #       # #       # #
    '# # #   # # # # # # # # #
    '#           # # # # # # #
    '# # # # # # # # # # # #
    '# #         # #       #
    '# ############### # # #
    '#               # # #
    '# # # # # # # # # # # #
    '#        #       # # #
    '#########################
]

maze[3]

Result: Each row can be accessed with an index
Result: a 2-Dimensional Structure!
Result: a 2-Dimensional Array

maze

each cell is defined by a row index, and a column index: maze[row][col]
Algorithm

maze[i][j]
maze[i][j]
$$\text{maze}[i][j+1]$$

$$\text{maze}[i][j]$$
maze[i][j]
maze
impass

path

wall

passage
Exploring the Code
Observing the Recursive Nature of visitMaze()

```python
for letter in line:
    letters.append( letter )
letters[i] = char
maze[i] = ''.join( letters )

def visitMaze( maze, i, j, win ):
    """recursive visit of the maze. Returns True when it
    has found the exit, False otherwise""
    setBreakCrumb( i, j, win )
    setChar( maze, i, j, '.' )
    # wait for user to click before recursing
    win.getMouse()
    #printMaze( maze )
    if ( i != STARTi or j != STARTj ) \n        and ( (i==0 or i==len( maze[0]) -1 ) or (j==0 or j==len( maze[0] )-1 ) ):
        return True
    #--- try the four directions around where we are ---
    #--- to the right? ---
    if j+1< len( maze[0] ) and maze[i][j+1]==' ': 
        if visitMaze( maze, i, j+1, win ) == True:
            return True  # found an exit by going right!
    #--- down? ---
    if i+1< len( maze ) and maze[i+1][j]==' ':
        if visitMaze( maze, i+1, j, win ) == True:
            return True  # found an exit by going down!
```
Fractal Trees
Let's Play with the Fractal Tree

- Make the order larger (e.g. 12)
- Make it draw the right side first
- Change the angle $\theta$
- Make the drawing of a branch random
- Make the size of a branch very large
- Make the color proportional to the order of recursion
color_rgb( order*20, 255-order*20, 100 )
Examples

https://www.youtube.com/watch?v=yWRFCS1zej0

https://www.youtube.com/watch?v=v5qvePPHM6E
https://www.youtube.com/watch?v=s0Nu1OaBXa8
The result...

https://www.youtube.com/watch?v=tiqS6J_A6Jc
color_rgb( 25*order, 25*order, 255-25*order )
Tough problems, simple solutions

Recursive Functions

Finding the Largest in a List
Finding the Smallest in a List
Factorial
Traversing a Maze
Recursive Trees

Towers of Hanoi
We stopped here last time...
Solving Problems Recursively: Towers of Hanoi

http://en.wikipedia.org
The Legend of the Towers of Hanoi

The puzzle was invented by the **French mathematician Édouard Lucas** in 1883. There is a story about an Indian temple in Kashi Vishwanath which contains a large room with three time-worn posts in it surrounded by **64 golden disks**. Brahmin priests, acting out the command of an ancient prophecy, have been moving these disks, in accordance with the immutable rules of the Brahma, since that time. The puzzle is therefore also known as the Tower of Brahma puzzle. **According to the legend, when the last move of the puzzle will be completed, the world will end.** It is not clear whether Lucas invented this legend or was inspired by it.

Coding

The Towers of Hanoi
How long would it take a good PC to move 64 disks on 3 pegs?
### Timing Analysis

#### TOWERS OF HANOI

<table>
<thead>
<tr>
<th>Number of Disks</th>
<th>Number of Moves</th>
<th>Number of Sec required</th>
<th>Number of Min required</th>
<th>Number of Hours required</th>
<th>Number of Days required</th>
<th>Laptop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3.33E-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1.00E-09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>2.33E-09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>5.00E-09</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>31</td>
<td>1.03E-08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>63</td>
<td>2.10E-08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>127</td>
<td>4.23E-08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>255</td>
<td>8.50E-08</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>511</td>
<td>1.70E-07</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1023</td>
<td>3.41E-07</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>2047</td>
<td>6.82E-07</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>4095</td>
<td>1.37E-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>8191</td>
<td>2.73E-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>16383</td>
<td>5.46E-06</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>32767</td>
<td>1.09E-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>65535</td>
<td>2.18E-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>131071</td>
<td>4.37E-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>262143</td>
<td>8.74E-05</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>524287</td>
<td>1.75E-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>1048575</td>
<td>3.50E-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>2097151</td>
<td>6.99E-04</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>4194303</td>
<td>1.40E-03</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

# Towers of Hanoi

<table>
<thead>
<tr>
<th>Number of Disks</th>
<th>Number of Moves</th>
<th>Sec required</th>
<th>Min required</th>
<th>Hours required</th>
<th>Days required</th>
<th>Laptop 3GHz Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1.33E-10</td>
<td></td>
<td></td>
<td></td>
<td>3.00E+09 moves/second</td>
</tr>
<tr>
<td>34</td>
<td>31</td>
<td>2147483647</td>
<td>7.16E-01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>32</td>
<td>4294967295</td>
<td>1.43E+00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>33</td>
<td>8589934591</td>
<td>2.86E+00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>34</td>
<td>17179859183</td>
<td>5.73E+00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>35</td>
<td>34359738367</td>
<td>1.15E+01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>36</td>
<td>6871947535</td>
<td>2.29E+01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>37</td>
<td>1.37439E+11</td>
<td>4.58E+01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>38</td>
<td>2.74878E+11</td>
<td>9.16E+01</td>
<td>1.527099483</td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>39</td>
<td>5.49756E+11</td>
<td>1.83E+02</td>
<td>3.054198966</td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>40</td>
<td>1.09951E+12</td>
<td>3.67E+02</td>
<td>6.108397932</td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>41</td>
<td>2.19902E+12</td>
<td>7.33E+02</td>
<td>12.21679586</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>42</td>
<td>4.39805E+12</td>
<td>1.47E+03</td>
<td>24.43359173</td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>43</td>
<td>8.79609E+12</td>
<td>2.93E+03</td>
<td>48.86718346</td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>44</td>
<td>1.75922E+13</td>
<td>5.86E+03</td>
<td>97.73436691</td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>45</td>
<td>3.51844E+13</td>
<td>1.17E+04</td>
<td>195.468738</td>
<td>3.25781223</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>46</td>
<td>7.03687E+13</td>
<td>2.35E+04</td>
<td>390.937467</td>
<td>6.51562446</td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>47</td>
<td>1.40737E+14</td>
<td>4.69E+04</td>
<td>781.874935</td>
<td>13.03124892</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>48</td>
<td>2.81475E+14</td>
<td>9.38E+04</td>
<td>1563.749671</td>
<td>26.06249784</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>49</td>
<td>5.62955E+14</td>
<td>1.88E+05</td>
<td>3127.499741</td>
<td>52.12495569</td>
<td>2.17187482</td>
</tr>
<tr>
<td>53</td>
<td>50</td>
<td>1.12595E+15</td>
<td>3.75E+05</td>
<td>6254.999482</td>
<td>104.2499914</td>
<td>4.33479641</td>
</tr>
<tr>
<td>54</td>
<td>51</td>
<td>2.25185E+15</td>
<td>7.51E+05</td>
<td>12509.99986</td>
<td>208.4999827</td>
<td>8.68749928</td>
</tr>
<tr>
<td>55</td>
<td>52</td>
<td>4.50361E+15</td>
<td>1.50E+06</td>
<td>25019.999793</td>
<td>416.9999655</td>
<td>17.3749856</td>
</tr>
<tr>
<td>56</td>
<td>53</td>
<td>9.0072E+15</td>
<td>3.00E+06</td>
<td>50039.99986</td>
<td>833.999931</td>
<td>34.7499712</td>
</tr>
<tr>
<td>57</td>
<td>54</td>
<td>1.80144E+16</td>
<td>6.00E+06</td>
<td>100079.9917</td>
<td>1669.999862</td>
<td>69.4999425</td>
</tr>
<tr>
<td>58</td>
<td>55</td>
<td>3.60288E+16</td>
<td>1.20E+07</td>
<td>200159.9834</td>
<td>3335.999724</td>
<td>138.999985</td>
</tr>
<tr>
<td>59</td>
<td>56</td>
<td>7.20575E+16</td>
<td>2.40E+07</td>
<td>400319.9669</td>
<td>6667.999448</td>
<td>277.999977</td>
</tr>
<tr>
<td>60</td>
<td>57</td>
<td>1.44115E+17</td>
<td>4.80E+07</td>
<td>800639.9338</td>
<td>13343.9989</td>
<td>555.999954</td>
</tr>
</tbody>
</table>

### Towers of Hanoi

<table>
<thead>
<tr>
<th>Number of Disks</th>
<th>Number of Moves</th>
<th>Number of Sec required</th>
<th>Number of Min required</th>
<th>Number of Hours required</th>
<th>Number of Days required</th>
<th>Laptop Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>3.33E-10</td>
<td>3.0519866</td>
<td>1.83E+02</td>
<td>3.00E+09 moves/second</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>3.67E-09</td>
<td>6.1083932</td>
<td>1.21E+02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1.00E-08</td>
<td>2.44339173</td>
<td>4.88E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>3.47E-08</td>
<td>2.44339173</td>
<td>4.88E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1.17E-07</td>
<td>1.954687338</td>
<td>3.25E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>3.12E-07</td>
<td>1.954687338</td>
<td>3.25E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>1.04E-06</td>
<td>1.563749871</td>
<td>2.60E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>3.11E-06</td>
<td>1.563749871</td>
<td>2.60E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>1.04E-05</td>
<td>1.21679586</td>
<td>1.83E+01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>3.11E-05</td>
<td>1.21679586</td>
<td>1.83E+01</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

![Graph showing exponential growth](image)

D. Thiebaut, Computer Science, Smith College
From Here to There...
Most Important Tips
Most Important Tips

• Small, start small
• Don't work with the original data
• Take a small sample that is representative
• Take all the shortcuts possible
Most Important Tips

• Create small test data set with all possible cases for exceptions

• Make sure your program works with
  • empty lists
  • empty strings
  • 0, 1 or \( n \) results
Most Important Tips

• Reuse code we have created before (solution programs, programs in the book)

• Solve only one problem at a time

• Do not be afraid of exceptions!

• "It would be nice if I could…" —> search python.org