CSc 110, Autumn 2017

Lecture 29: Lists of Lists

THINGS TO DO TODAY:
1. Finish yesterday's list.
Mountain peak

Write a program that reads elevation data from a file, draws it on a DrawingPanel and finds the path from the highest elevation to the edge of the region.

Data:
34 76 87 9 34 8 22 33 33 33 45 65 43 22
5 7 88 0 56 76 76 77 4 45 55 55 4 5
...

This data is a different shape. How should we store it?
Lists of lists

• You can put a list in a list

```python
list = [[1, 2, 3], [4, 5, 6]]
```

How can you access 2?

```
list[0][1]
```

How can you find the length of the second inner list ([4, 5, 6])?

```python
len(list[1])
```
Lists of lists - traversals

• We normally use nested loops to go through a list of lists
  
  ```python
  data = [[1, 2, 3], [4, 5, 6]]
  for i in range(len(data)):
    for j in range(len(data[i])):
      # do something with data[i][j]
  ```

  Why does the inner loop go to `len(data[i])`?
def mystery(data, pos, n):
    result = []
    for i in range(0, n):
        for j in range(0, n):
            result.append(data[i + pos][j + pos])
    return result

Suppose that a variable called grid has been declared as follows:
grid = [[8, 2, 7, 8, 2, 1],
        [1, 5, 1, 7, 4, 7],
        [5, 9, 6, 7, 3, 2],
        [7, 8, 7, 7, 9, 9],
        [4, 2, 6, 9, 2, 3],
        [2, 2, 8, 1, 1, 3]]

which means it will store the following 6-by-6 grid of values:

<table>
<thead>
<tr>
<th>8</th>
<th>2</th>
<th>7</th>
<th>8</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>1</td>
<td>7</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
<td>6</td>
<td>7</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>7</td>
<td>7</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>6</td>
<td>9</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>3</td>
</tr>
</tbody>
</table>

For each call at right, indicate what value is returned. If the function call results in an error, write error instead.

<table>
<thead>
<tr>
<th>Function Call</th>
<th>Contents of List Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>mystery(grid, 2, 2)</td>
<td>_____________________________</td>
</tr>
<tr>
<td>mystery(grid, 0, 2)</td>
<td>_____________________________</td>
</tr>
<tr>
<td>mystery(grid, 3, 3)</td>
<td>_____________________________</td>
</tr>
</tbody>
</table>
Exercise

Write a function called `max_row` that takes a list of lists as a parameter and returns the index of the row that contains the maximum value.
Exercise

Write a function called `flip` that takes a list of lists and two columns and swaps their contents. For example if `swap(data, 2, 3)` were called on the following list

\[
data = \begin{bmatrix}
    [1, 2, 3], [4, 5, 6], [7, 8, 9]
\end{bmatrix}
\]

data would contain the following afterwards:

\[
data = \begin{bmatrix}
    [1, 3, 2], [4, 6, 5], [7, 9, 8]
\end{bmatrix}
\]
Exercise

Write a function called `create_matrix` that takes a width and a height as parameters and returns a list of lists that is width by height and contains the numbers 0 to width - 1 in each row. For example a call to `create_matrix(5, 3)` would return the following list of lists:

```
[[0, 1, 2, 3, 4], [0, 1, 2, 3, 4], [0, 1, 2, 3, 4]]
```
Creating Lists of lists

• list = [[0] * 4] * 5 will NOT create a list of lists
  • This will create a list with 5 spots that all contain the SAME list that is 4 long.

• Instead, write the following:

```python
list = []
for i in range(0, 5):
    list.append([0] * 4)
```
Mountain peak

Write a program that reads elevation data from a file, draws it on a DrawingPanel and finds the path from the highest elevation to the edge of the region.

Data:

```
34 76 87 9 34 8  22 33 33 33 45 65 43 22
5 7  88 0 56 76 76 77 4  45 55 55 4  5
...
```