Matrices and Nested Loops Revision

In Python matrices are represented as lists of lists.

The general form of a matrix:

```python
matrix = [[], [], [], ..., []]
```

1. How to access elements of a matrix?

Since we have that a matrix is a list of lists we require to use two indexes to access a particular element of matrix. The first index identifies the inner list we want to refer (which corresponds to a row in the matrix), while the second index identifies the element inside the inner list (which corresponds to a column position). Let's take as example the following matrix.

```
[1  2]
[3  4]
```

This matrix is created in Python as follows:

```python
matrix = [[1, 2], [3, 4]]
```

The element 2 in row 0 and position 1 is accessed as follows:

```python
print(matrix[0][1])
```

Output:

```
>>> ============== RESTART ==============
>>> 2
>>>
```

2. How to iterate over every element of a matrix and how to determine its size?

For this we only require to use two nested loops. The loop structure can be any of your choice. Here we present a combination of while and for loop structures. It is more common, although, to use two for loops.

def main():
    matrix = [[1, 2], [3, 4]]
    nRow = len(matrix)
    nCol = len(matrix[0])
    i = 0
    while i < nRow:
        for j in range(nCol):
            print(matrix[i][j])
        i = i + 1

main()

Output:

>>> ================================= RESTART
================================
>>> 
1 
2 
3 
4 

Notice here how we determine the size of matrix. The number of rows is determined using len(matrix), which gives the number of inner lists within the outer list. The number of inner lists corresponds to the number of rows. The number of columns is obtained through len(matrix[0]), which gives the number of elements in the inner list. Be aware that the result is obtained if we use len(matrix[1])

3. How to create matrices with loop structures?

Matrices are represented in Python as lists of lists. Then, as long as we know the size of a matrix we can create it with any initial values of our choice using loop structures. The following example allows us to create matrices of size m x n whose initial values are zeros.

def main():
    m = eval(input('Enter number of rows m: '))
    n = eval(input('Enter number of rows n: '))

    myMatrix = []
    for i in range(m):
        myRow = []
        for j in range(n):
            myRow.append(0)
        myMatrix.append(myRow)
4. How to print matrices

Matrices can be print as any other variable in Python. We can print the outer list (which correspond to entire matrix) or iterate over each inner list and print them. Let's see some examples:

def main():
    m = eval(input('Enter number of rows m: '))
    n = eval(input('Enter number of rows n: '))

    myMatrix = [n*[1 * i for i in range(m)]

    print(\n', myMatrix, '\n')

    for i in range(m):
        print(myMatrix[i])

main()

Output:

>>> ==================================================================== RESTART
================================================================================
>>> Enter number of rows m: 3
Enter number of rows n: 3
[[0, 0, 0], [0, 0, 0], [0, 0, 0]]

[[0, 0, 0, 0], [1, 1, 1, 1], [2, 2, 2, 2], [3, 3, 3, 3]]
[0, 0, 0, 0]
[1, 1, 1, 1]
[2, 2, 2, 2]
[3, 3, 3, 3]

>>>
Notice that the recitation related to matrices a code was presented to achieve an even better look of our printing. For simplicity, this lab does not require that format.

5. How to generate random integers within a specific range?

First thing you need to do is to import the library `random`. After that you only need to call the function `randint(x, y)`, where `x` and `y` specify the range of random numbers to be generated inclusively.

```python
from random import randint
def main():
    for i in range(5):
        r = randint(-10, 10)
        print(r)
main()
```

Output:

```plaintext
>>> ============================ RESTART
===========================
>>> 5
-5
-9
4
5

6. How to determine whether a number is even or odd?

Generally when we say 2 divides 10, or 10 is divisible by 2, this means that the result of the division operation 10/2 is a whole number (integer) with no remainder.

Any number divisible by 2 is even. Likewise, any number no divisible by 2 is odd.

In Python and other programming languages the mod (%) is used to determine if a number is either even or odd. Let's see an example:

```python
def main():
    n = eval(input('Enter a number n: '))
    if (n%2 == 0):
        print('The number is even')
    else:
        print('The number is odd')
```
main()

Output:

```python
>>> =================== RESTART ===================
>>> Enter a number n: 4
    The number is even
>>> =================== RESTART ===================
>>> Enter a number n: 5
    The number is odd
>>> 
```

From: http://courses.cs.purdue.edu/ - Computer Science Courses

Permanent link: http://courses.cs.purdue.edu/cs17700:spring16:prelabs:prelab09

Last update: 2016/03/06 17:47