

Matrices and Nested Loops

A loop is often used to iterate through a list of items, performing some operation.

```
i=0
while i<len(someList)
    # do something with somelist[i]
    i=i+1
```

Nested loops can be used to iterate through 2-dimensional entities or comparing two loops (e.g., in Mastermind)

For nested loops we define two loop variables commonly named *i* and *j*:

```
i=0
while i<3:
    j=0
    while j<5:
        print i,
        print j
        j=j+1
    i=i+1
```

What is the output of this program?

Matrices

Consider the following list, which can be considered a 2x3 matrix

```
matrix = [ [10,20,30],[40,50,60] ] # ***
```

We index a matrix with two indices, as in the samples below. What is the value of:

```
matrix[0][1]
matrix[2][2]
matrix[1]
```

A matrix is represented as a list whose elements are lists (a list of lists).

For

```
matrix[0][1]
```

the 0 index gives us the list [10,20,30], the 1 index gives us the value 20.

We say that the first index is the row # and the second index the column #. So 'row' and 'column' would be appropriate loop variables (as an alternative to i and j)

Adding up all the elements in the matrix could be done with:

```
total = matrix[0][0]+matrix[0][1]+matrix[0][2]+  
matrix[1][0]+matrix[1][1]+matrix[1][2]
```

But this code is not very general-- it will only work for a 2x3 matrix. Can a nested loop be used instead?

In class-Problem:

Write a program that creates a matrix as in the statement on the previous page(***), then computes the sum of all elements in the matrix. The program should work no matter what size or content is in the matrix.