Ragged Array

- It is acceptable, when declaring a multi-dimensional array, to leave the **second** dimension's (i.e. the #columns) size unspecified

- e.g.

```
int[][] RagArray = new int[3][];
```

- We will have 3 rows of unknown length.

- However, it is **not** legal to have the first dimension unspecified.

- Non-rectangular two-dimensional arrays are called ragged arrays

Creating a ragged array

Recall:

```
int[][] RagArray = new int[3][];
   // will create an array of three rows of unknown length.
   // each row has an index, However none of the columns
   have indices! That is, the columns associated with all
   three rows can not contain any information until
   initialized to do so!
```

Therefore:

```
RagArray[2][3] = 54;
   //Will produce an error!
   // Although there is a second row for RagArray, There is
   no third column for the second row of RagArray.
```

Creating a ragged array

```
int[][] RagArray = new int[3][];

RagArray[0] = new int [3];
   //will create 3 columns for row 0
RagArray[1] = new int [2];
   //will create 2 columns for row 1
RagArray[2] = new int [4];
   //will create 4 columns for row 2
```

Creating a ragged array

The preceding lines will create the following array in memory.

	c0	c1	c3	c4
Row0	0	0	0	
Row1	0	0		
Row2	0	0	0	0

Creating a ragged array at declaration

- We can also initialize an array at declaration.

```
int[][] RagArray = {{0,0,0}, {0,0},
                    {0,0,0,0}};
```

creates

```
0       0       0
0       0
0       0       0       0
```

Using a ragged array

- An example
- Consider the ragged array on the previous slide. We wish to enter the value of sum=5 into each cell of our ragged array. Will this work?

```
int sum=5;

for (int row = 0; row < RagArray.length; row++){
    for (int col = 0; col < RagArray[0].length; col++){
            RagArray[row][col] = sum;
    }
}
```

Why not?

- RagArray[0].length is an integer that represents the number of columns for row 0.
- By repeating that code using RagArray[0].length as the number of columns per row, we are assuming that each row has the same number of columns
- HOWEVER, the number of columns is different for each row
  - RagArray[0] = 3 but there are only 2 values in the second row and 4 values in the third row

Try this!

```
int sum = 5;

for (int row = 0; row < RagArray.length; row++){
    for (int col = 0; col < RagArray[row].length; col++){
        RagArray[row][col] = sum;
    }
}
```

Notice that the upper bound of the inner loop now depends on the length of each row of the array.

e.g. row 0 has 3 columns, therefore the inner loop will repeat 3 times to read the values

Multi-dimensional arrays

- We can create 3-dimensional, 4-dimensional, and arrays of even higher dimensions.

- e.g. if a 2-D array is a table of rows and columns, a 3-D array can be visualized as a collection of tables.