CS159

Nathan Sprague

April 24, 2015

See Also

Bernstein Slides ↗

Some content here is borrowed from those slides.

Review of Arrays

- Declaration:
  - `int[] numbers;`
  - `String[] words;`

Review of Arrays

- Declaration:
  - `int[] numbers;`
  - `String[] words;`
- Instantiation:
  - numbers = new int[4];
  - words = new String[4];

Review of Arrays

- Declaration:
  - `int[] numbers;`
  - `String[] words;`
- Instantiation:
  - numbers = new int[4];
  - words = new String[4];
- How many strings have been created? How many ints?
- Let's draw a picture of memory...

Multiple Arrays...

Does this seem OK?

```
1    int[] donationsWeek0 = {10, 75};
2    int[] donationsWeek1 = {15, 20, 20};
3    int[] donationsWeek2 = {100, 63, 10};
4    //...
```

Multiple Arrays...

Does this seem OK?

```
1    int[] donationsWeek0 = {10, 75};
2    int[] donationsWeek1 = {15, 20, 20};
3    int[] donationsWeek2 = {100, 63, 10};
4    //...
```

No reason we can't create an array of arrays.

Multi-Dimensional Arrays

- Declaration:
  - `int[][] donations;`
  - `String[][] words;`

Multi-Dimensional Arrays

- Declaration:
  - `int[][] donations;`
  - `String[][] words;`
- Instantiation:
  - `numbers = new int[4][3];`
  - `words = new String[4][3];`

Multi-Dimensional Arrays

- Declaration:
  - `int[][] donations;`
  - `String[][] words;`
- Instantiation:
  - `numbers = new int[4][3];`
  - `words = new String[4][3];`
- How many strings have been created? How many ints?
- Let's draw a picture of `numbers`...

Multi-Dimensional Arrays

- Declaration:
  - `int[][] donations;`
  - `String[][] words;`
- Instantiation:
  - `numbers = new int[4][3];`
  - `words = new String[4][3];`
- How many strings have been created? How many ints?
- Let's draw a picture of `numbers`...
- How would this change the picture:
  - `numbers[1][2] = 7;`

Multi-Dimensional Arrays

- Declaration:
  - `int[][] donations;`
  - `String[][] words;`
- Instantiation:
  - `numbers = new int[4][3];`
  - `words = new String[4][3];`
- How many strings have been created? How many ints?
- Let's draw a picture of `numbers`...
- How would this change the picture:
  - `numbers[1][2] = 7;`
- How would this change the picture:
  - `numbers[2] = new int[5];`

MD-Array Literals

```
1  public static int[][] sampleDonationsFromLiteral()
2    {
3        int[][] donations = { {10, 75},
4                              {15, 20, 20},
5                              {100, 63, 10} };
6
7        return donations;
8    }
```

MD-Array Example

```
1   public int[][] sampleDonations()
2   {
3       int[][] donations;
4
5       donations = new int[3][]; // 3 rows, no columns
6
7       donations[0] = new int[2];
8       donations[0][0] = 10;
9       donations[0][1] = 75;
10
11      donations[1] = new int[3];
12      donations[1][0] = 15;
13      donations[1][1] = 20;
14      donations[1][2] = 20;
15
16      donations[2] = new int[3];
17      donations[2][0] = 100;
18      donations[2][1] = 63;
19      donations[2][2] = 10;
20
21      return donations;
22  }
```

Rectangular MD-Array

```
1   public int[][] sampleDonationsRectangular()
2   {
3       int rows;
4       int columns;
5       int[][] donations;
6
7       rows = 3;
8       columns = 4;
9       donations = new int[rows][columns];
10
11      for (int row = 0; row < rows; row++)
12      {
13          for (int col = 0; col < columns; col++)
14          {
15              donations[row][col] = row * col;
16          }
17      }
18
19      return donations;
20  }
```

Finding the Maximum Value...

Finding the Maximum Value

```
1
2       public static int maximum(int[][] values, int rows, int columns)
3       {
4           int maxVal = Integer.MIN_VALUE;
5
6           for (int row = 0; row < rows; row++)
7           {
8               for (int col = 0; col < columns; col++)
9               {
10                  if (values[row][col] > maxVal)
11                  {
12                      maxVal = values[row][col];
13                  }
14              }
15          }
16          return maxVal;
17      }
```

Finding the Maximum Value

```
1
2       public static int maximum(int[][] values, int rows, int columns)
3       {
4           int maxVal = Integer.MIN_VALUE;
5
6           for (int row = 0; row < rows; row++)
7           {
8               for (int col = 0; col < columns; col++)
9               {
10                  if (values[row][col] > maxVal)
11                  {
12                      maxVal = values[row][col];
13                  }
14              }
15          }
16          return maxVal;
17      }
```

Only works for rectangular array. Why require the caller to provide the array sizes?

Finding the Maximum Value

```
1   public static int maximum1(int[][] values)
2    {
3        int maxVal = Integer.MIN_VALUE;
4
5        for (int row = 0; row < values.length; row++)
6        {
7            for (int col = 0; col < values[row].length; col++)
8            {
9                if (values[row][col] > maxVal)
10               {
11                   maxVal = values[row][col];
12               }
13           }
14       }
15       return maxVal;
16   }
```

Finding the Maximum Value

```
1   public static int maximum1(int[][] values)
2    {
3        int maxVal = Integer.MIN_VALUE;
4
5        for (int row = 0; row < values.length; row++)
6        {
7            for (int col = 0; col < values[row].length; col++)
8            {
9                if (values[row][col] > maxVal)
10               {
11                   maxVal = values[row][col];
12               }
13           }
14       }
15       return maxVal;
16   }
```

Better! What if we are worried about unitialized arrays?

Finding the Maximum Value

```
1   public static int maximum3(int[][] values)
2   {
3       int maxVal = Integer.MIN_VALUE;
4
5       if (values != null) // Make sure values is initialized.
6       {
7           for (int row = 0; row < values.length; row++)
8           {
9               if (values[row] != null) // Make sure the row is initialized.
10              {
11                  for (int col = 0; col < values[row].length; col++)
12                  {
13                      if (values[row][col] > maxVal)
14                      {
15                          maxVal = values[row][col];
16                      }
17                  }
18              }
19          }
20      }
21
22      return maxVal;
23  }
```

Finding the Maximum Value

One more:

```
1   public static int maximum4(int[][] values)
2   {
3       int maxVal = Integer.MIN_VALUE;
4
5       for (int[] row : values)
6       {
7           for (int value : row)
8           {
9               if (value > maxVal)
10              {
11                  maxVal = value;
12              }
13          }
14      }
15      return maxVal;
16  }
```

Finding the Maximum Value

One more:

```
1     public static int maximum4(int[][] values)
2     {
3         int maxVal = Integer.MIN_VALUE;
4
5         for (int[] row : values)
6         {
7             for (int value : row)
8             {
9                 if (value > maxVal)
10                {
11                    maxVal = value;
12                }
13            }
14        }
15        return maxVal;
16    }
```

I think this is less clear than the indexed approach, but it works.