CS100J 10 April 2005
Rectangular arrays and ragged arrays. Secs. 9.1 – 9.3

Do as many of the exercises on pp. 311-312 as you can to get familiar with concepts and develop a skill. Practice in DrJava! Test your methods, both by hand and on computer!

A Billion. The next time you hear someone in government rather casually use a number that includes the word "billion", think about it.

- A billion seconds ago was 1976.
- A billion minutes ago Jesus was alive.
- A billion hours ago our ancestors were living in the Stone Age.
- A billion days ago no creature walked the earth on two feet.
- A billion dollars lasts less than 8 hours at the rate our government spends it.

1,000,000,000

1

---

one-dimensional array

b.length

b	0	1	2	3
	5	4	7	3

rectangular array: 5 rows and 4 columns

d	0	1	2	3
0	5	4	7	3
1	4	8	9	7
2	5	1	2	3
3	4	1	2	9
4	6	7	8	0

Type of d is **int**[][] (“**int** array array”, “an array of **int** arrays”)

To declare variable d:

int d[][].

To create a new array and assign it to d:

d= **new int**[5][4];

To reference element at row r column c:

d[r][c]

number of rows

number of cols

2

---

d	0	1	2	3
0	5	4	7	3
1	4	8	9	7
2	5	1	2	3
3	4	1	2	9
4	6	7	8	0

Type of d is **int**[][] (“**int** array array”, “an array of **int** arrays”)

To declare variable d:

int d[][].

To create a new array and assign it to d:

d= **new int**[5][4];

To reference element at row r column c:

d[r][c]

number of rows

number of cols

Number of rows: d.length

Number of columns in row r: d[r].length

“Length of one array in array of arrays”

Using an array initializer:

int[][] d= **new int**[][]{ {5,4,7,3}, {4,8,9,7}, {5,1,2,3}, {4,1,2,9}, {6,7,8,0} };

3

```
/** = sum of first elements of rows of d. e.g. for array to
      right, it's 5 + 4 + 5 + 4 + 6. */
public static int sum0(int[][] d) {
    int x= 0;
    // inv: x = sum of first element of rows d[0..r–1]
    for (int r= 0;  r != d.length;  r= r+1) {
        x= x + d[r][0];
    }

    // x = sum of first element of rows d[0..d.length–1]
    return x;

}
```

d	0	1	2	3
0	5	4	7	3
1	4	8	9	7
2	5	1	2	3
3	4	1	2	9
4	6	7	8	0

4

Pattern for processing all the elements of an array

Row-major order (first row 1, then row 2, etc.)

```
// Process elements of b[][] in row-major order
// inv: rows 0..r-1 have been processed.
//      In row r, b[r, 0..c-1] have been processed
  for (int r= 0; r != b.length; r= r + 1)
     for (int c= 0; c != b[r].length; c= c+1) }
         Process b[r][c]
}
```

5

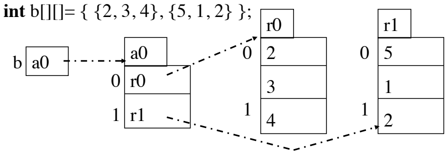
```
/** = a String rep of b[][] (as in an array initializer) */
public static String toString(int b[][]) {
    String s= "{"
    // inv: Rows 0..r–1 have been appended to s */
    for (int r= 0; r != b.length; r= r + 1) {
        // Add row r to s
            if (r != 0) s= s + ", ";   s= s + "{";
            // inv: the partial row b[r][0..c–1] has been added to s
            for (int c= 0; c != b[r].length; c= c + 1) {
                    if (c != 0)  s= s + ", ";
                    s= s + b[r][c];
            }
            s= s + "}";

    return s + "}";
} }
```

6

How multi-dimensional arrays are stored: ragged arrays

int b[][]= { {2, 3, 4}, {5, 1, 2} };



b is a one-dimensional array of b.length elements

Its elements are one-dimensional arrays.

b[0] is a one-dimensional array of **int**s of length b[0].length.
Must all these arrays have the same length? No!

7

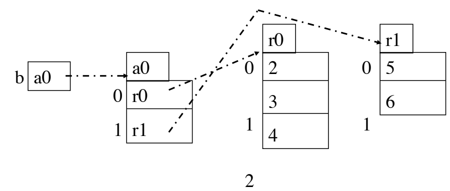
How multi-dimensional arrays are stored: ragged arrays

int[][] b; Declare variable b of type **int** [][]

b= **new int**[2][] Create a one-dim. array of length 2 and store its name in b. Its elements are **null**, have type **int**[]

b[0]= **new int**[] {2, 3, 4}; Create **int** array, store its name in b[0].

b[1]= **new int**[] {5, 6}; Create **int** array, store its name in b[1].



8

Pascal's Triangle

1	0
1 1	1
1 2 1	2
1 3 3 1	3
1 4 6 4 1	4
1 5 10 10 5 1	5
	…

The first and last entries on each row are 1.

Each other entry is the sum of the two entries above it

row r has r+1 values.

9

Pascal's Triangle

1	0
1 1	1
1 2 1	2
1 3 3 1	3
1 4 6 4 1	4
1 5 10 10 5 1	5
	…

Entry p[i][j] is the number of ways j elements can be chosen from a set of size i !

p[i][j] = "i choose j" = $\binom{i}{j}$

recursive formula:
for $0 < i < j$, $p[i][j] = p[i-1][j-1] + p[i-1][j]$

10

Pascal's Triangle

1	0
1 1	1
1 2 1	2
1 3 3 1	3
1 4 6 4 1	4
1 5 10 10 5 1	5

Binomial theorem: Row r gives the coefficients of $(x + y)^r$

$(x + y)^2 = 1x^2 + 2xy + 1y^2$

$(x + y)^3 = 1x^3 + 3x^2y + 3xy^2 + 1y^3$

$(x + y)^r = \sum_{0 \le k \le r} (k \text{ choose } r)\, x^k y^{r-k}$

11

Method to compute first r rows of Pascal's Triangle in a ragged array

```
/** Return ragged array of first n rows of Pascal's triangle.
    Precondition: 0 ≤ n */
public static int[][] pascalTriangle(int n) {
    int[][] b= new int[n][];    // First n rows of Pascal's triangle
    // invariant: rows 0..i-1 have been created
    for (int i= 0; i != b.length; i= i+1) {
      // Create row i of Pascal's triangle
        b[i]= new int[i+1];
      // Calculate row i of Pascal's triangle
        b[i][0]= 1;
        // invariant b[i][0..j-1] have been created
        for (int j= 1; j < i; j= j+1) {
          b[i][j]= b[i-1][j-1] + b[i-1][j];
        }
        b[i][i]= 1;
    }
    return b;
  }
```

12