гр. София, пл. Славейков 11 ет.5 тел. : +359 897 91 93 96

office@progressbg.net www.progressbg.net

Компютърно образователен център Progress Палитра от възможности!

Лекция 9

Многомерни масиви в Java– 1-ва част

- **Деклариране и обработка на многомерни масиви**
- **Четене на аргументи от командния ред**
- **Примери**
- **Задачи**

гр. София, пл. Славейков 11 ет.5 тел. : +359 897 91 93 96

office@progressbg.net www.progressbg.net

Многомерни масиви

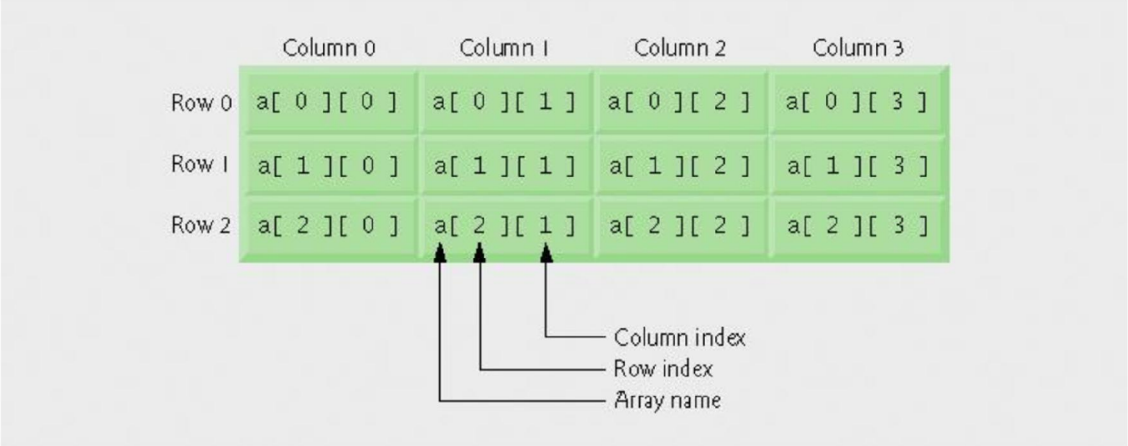
- Многомерни масиви с две размерности се използват често за представяне на таблица от стойности състояща се от данни подредени в редове и колони.В математиката се използват двумерни матрици за същите цели.

- За идентифициране на даден елемент от такава таблица се задават два индекса.По подразбиране, първият индекс определя реда, а втория индекс определя колоната в която се намира въпросния елемент.

- Може да се дефинират и масиви с повече от две размерности

- Java не поддържа многомерни масиви директно.За целта многомерен масив се представя като едноморен масив, чиито елементи са също едномерни масиви.

Диаграма на двумерен масив с 3 реда и 4 колони :



гр. София, пл. Славейков 11 ет.5 тел. : +359 897 91 93 96

office@progressbg.net www.progressbg.net

Двумерни масиви представени, чрез едномерни

- Декларация на двумерен масив b[2][2] с инициализиращ списък

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

- 1 и 2 инициализират съответно b[0][0] и b[0][1]
- 3 и 4 инициализират съответно b[1][0] и б[1][1]

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

- ред 0 е масив с елементи 1 и 2
- ред 1 е масив с елементи 3,4 и 5

✓ Всеки ред на многомерен масив може да е с различна дължина.

Двумерни масиви, имащи редове с различна дължина

- Деклариране с инициализиращ списък

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

- Обща форма на декларация – инициализиране на всеки ред по отделно

```
int b[] [] = new int [ 2 ][];//създаваме масив с 2 реда
b[0] = new int[ 2 ];// създаваме масив с 2 елемента за първи ред(индекс 0)
b[1] = new int[ 3 ];// създаваме масив с 3 елемента за втори ред(индекс 1)
```

Обща форма на декларация

- Деклариране на масив с 2 реда съдържащи по 3 колони

```
int b[][];
b = new int[ 2 ] [ 3];
```

- Деклариране на масив с 2 реда съдържащи по 3 колони

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

Пример с двумерен int array :

```
1  // Fig. 7.17: InitArray.java
2  // Initializing two-dimensional arrays.
3
4  public class InitArray
5  {
6     // create and output two-dimensional arrays
7     public static void main( String args[] )
8     {
9        int array1[][] = { { 1, 2, 3 }, { 4, 5, 6 } };
10       int array2[][] = { { 1, 2 }, { 3 }, { 4, 5, 6 } };
11
12       System.out.println( "Values in array1 by row are" );
13       outputArray( array1 ); // displays array1 by row
14
15       System.out.println( "\nValues in array2 by row are" );
16       outputArray( array2 ); // displays array2 by row
17    } // end main
18
```

Използване на вложени инициализиращи списъци за инициализиране на array1

Използване на вложени инициализиращи списъци за инициализиране на array2

```
19    // output rows and columns of a two-dimensional array
20    public static void outputArray( int array[][] )
21    {
22       // loop through array's rows
23       for ( int row = 0; row < array.length; row++ )
24       {
25          // loop through columns of current row
26          for ( int column = 0; column < array[ row ].length; column++ )
27             System.out.printf( "%d  ", array[ row ][ column ] );
28
29          System.out.println(); // start new line of output
30       } // end outer for
31    } // end method outputArray
32 } // end class InitArray
```

```
Values in array1 by row are
1  2  3
4  5  6

Values in array2 by row are
1  2
3
4  5  6
```

Използване на двойни скоби за достъп до стойността на елемент от двумерния масив

`array[row].length` връща дължината на масив позициониран като елемент с индекс `row` на едномерния масив от най- горно ниво

- Често срещани конструкции за обработка на елементите на многомерен масив с **for** команди

Нека int **a**[ ] [ ] ; a = **new int [3] [4];**

```
    //  инициализация на елементите на ред  2
for ( int column = 0; column < a[ 2 ].length; column++ )
       a[ 2 ][ column ] = 0;
Еквивалентно на
a[ 2 ][ 0 ] = 0;
a[ 2 ][ 1 ] = 0;
a[ 2 ][ 2 ] = 0;
a[ 2 ][ 3 ] = 0;
```

Class GradeBook използващ двумерен масив

- Използва двумерен масив в който се съхраняват оценките на всеки студент и дадена група студенти като цяло

```
1  // Fig. 7.18: GradeBook.java
2  // Grade book using a two-dimensional array to store grades.
3
4  public class GradeBook
5  {
6     private String courseName; // name of course this grade book represents
7     private int grades[][]; // two-dimensional array of student grades
8
9     // two-argument constructor initializes courseName and grades array
10    public GradeBook( String name, int gradesArray[][] )
11    {
12       courseName = name; // initialize courseName
13       grades = gradesArray; // store grades
14    } // end two-argument GradeBook constructor
15
16    // method to set the course name
17    public void setCourseName( String name )
18    {
19       courseName = name; // store the course name
20    } // end method setCourseName
21
22    // method to retrieve the course name
23    public String getCourseName()
24    {
25       return courseName;
26    } // end method getCourseName
27
```

Декларира двумерен масив за съхраняване на оценките

Контруктор на class GradeBook взима за аргумент String и двумерен масив- инициализира всички клас данни

Този команда също е писана набързо- да се промени, както в предишната редакция на class GradeBook

```
28    // display a welcome message to the GradeBook user
29    public void displayMessage()
30    {
31       // getCourseName gets the name of the course
32       System.out.printf( "Welcome to the grade book for\n%s!\n\n",
33          getCourseName() );
34    } // end method displayMessage
35
36    // perform various operations on the data
37    public void processGrades()
38    {
39       // output grades array
40       outputGrades();
41
42       // call methods getMinimum and getMaximum
43       System.out.printf( "\n%s %d\n%s %d\n\n",
44          "Lowest grade in the grade book is", getMinimum(),
45          "Highest grade in the grade book is", getMaximum() );
46
47       // output grade distribution chart of all grades on all tests
48       outputBarChart();
49    } // end method processGrades
50
51    // find minimum grade
52    public int getMinimum()
53    {
54       // assume first element of grades array is smallest
55       int lowGrade = grades[ 0 ][ 0 ];
56
57       // loop through rows of grades array
58       for ( int studentGrades[] : grades )
59       {
60          // loop through columns of current row
61          for ( int grade : studentGrades
62          {
63             // if grade less than lowGra
64             if ( grade < lowGrade )
65                lowGrade = grade;
66          } // end inner for
67       } // end outer for
68
69       return lowGrade; // return lowest grade
70    } // end method getMinimum
71
72    // find maximum grade
73    public int getMaximum()
74    {
75       // assume first element of grades array is largest
76       int highGrade = grades[ 0 ][ 0 ];
77
```

Специализиран цикъл по редовете на масив `grades` за намиране на най- ниската оценка

Специализиран цикъл по колоните на масив `grades` за намиране на най- ниската оценка


```
78       // loop through rows of grades array
79       for ( int studentGrades[] : grades )
80       {
81          // loop through columns of current row
82          for ( int grade : studentGrades )
83          {
84             // if grade greater than highGrade, assign it to highGrade
85             if ( grade > highGrade )
86                highGrade = grade;
87          } // end inner for
88       } // end outer for
89
90       return highGrade; // return highest grade
91    } // end method getMaximum
92
93    // determine average grade for particular set of grades
94    public double getAverage( int setOfGrades[] )
95    {
96       int total = 0; // initialize total
97
98       // sum grades for one student
99       for ( int grade : setOfGrades )
100          total += grade;
101
102       // return average of grades
103       return (double) total / setOfGrades.length;
104    } // end method getAverage
105
```

Специализиран цикъл по редовете на масив grades за намиране на най- високата оценка

Специализиран цикъл по колоните на масив grades за намиране на най- високата оценка

Пресмятане на средна оценка за студент. Подава се за аргумент **масив с само оценките на избрания студент**

GradeBook.java

(4 от 7)

Редове 79-88

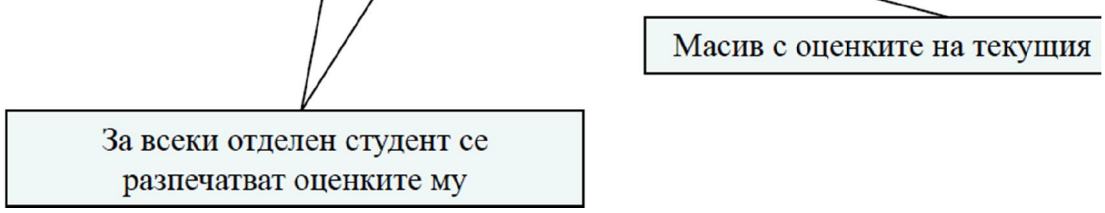
Редове 94-104

Outline

```
106    // output bar chart displaying overall grade distribution
107    public void outputBarChart()
108    {
109       System.out.println( "Overall grade distribution:" );
110
111       // stores frequency of grades in each range of 10 grades
112       int frequency[] = new int[ 11 ];
113
114       // for each grade in GradeBook, increment the appropriate frequency
115       for ( int studentGrades[] : grades )
116       {
117          for ( int grade : studentGrades )
118             ++frequency[ grade / 10 ];
119       } // end outer for
120
121       // for each grade frequency, print bar in chart
122       for ( int count = 0; count < frequency.length; count++ )
123       {
124          // output bar label ( "00-09: ", ..., "90-99: ", "100: " )
125          if ( count == 10 )
126             System.out.printf( "%5d: ", 100 );
127          else
128             System.out.printf( "%02d-%02d: ",
129                count * 10, count * 10 + 9  );
130
131          // print bar of asterisks
132          for ( int stars = 0; stars < frequency[ count ]; stars++ )
133             System.out.print( "*" );
```

Пресмята разпределението на оценките по интервали

```
134
135             System.out.println(); // start a new line of output
136          } // end outer for
137       } // end method outputBarChart
138
139       // output the contents of the grades array
140       public void outputGrades()
141       {
142          System.out.println( "The grades are:\n" );
143          System.out.print( "            " ); // align column heads
144
145          // create a column heading for each of the tests
146          for ( int test = 0; test < grades[ 0 ].length; test++ )
147             System.out.printf( "Test %d  ", test + 1 );
148
149          System.out.println( "Average" ); // student average column heading
150
151          // create rows/columns of text representing array grades
152          for ( int student = 0; student < grades.length; student++ )
153          {
154             System.out.printf( "Student %2d", student + 1 );
155
156             for ( int test : grades[ student ] ) // output student's grades
157                System.out.printf( "%8d", test );
158
```



За всеки отделен студент се разпечатват оценките му


```
Welcome to the grade book for
CS101 Introduction to Java Programming!

The grades are:

            Test 1  Test 2  Test 3  Average
Student  1      87      96      70    84.33
Student  2      68      87      90    81.67
Student  3      94     100      90    94.67
Student  4     100      81      82    87.67
Student  5      83      65      85    77.67
Student  6      78      87      65    76.67
Student  7      85      75      83    81.00
Student  8      91      94     100    95.00
Student  9      76      72      84    77.33
Student 10      87      93      73    84.33

Lowest grade in the grade book is 65
Highest grade in the grade book is 100

Overall grade distribution:
00-09:
10-19:
20-29:
```

```
30-39:
40-49:
50-59:
60-69: ***
70-79: ******
80-89: ***********
90-99: *******
  100: ***
```

гр. София, пл. Славейков 11 ет.5 тел. : +359 897 91 93 96

office@progressbg.net www.progressbg.net