Chapter 8 Multidimensional Arrays

创建和声明二维数组

```
// 二维数组声明
dataType[][] refVar;

// 创建二维数组
refVar = new dataType[10][10];

// 可以把上面的两步合成一步
dataType[][] refVar = new dataType[10][10];

// 另一种写法，不推荐
dataType refVar[][] = new dataType[10][10];
```

二维数组图解

	0	1	2	3	4
0					
1					
2					
3					
4					

```
matrix = new int[5][5];
```

	0	1	2	3	4
0					
1					
2		7			
3					
4					

```
matrix[2][1] = 7;
```

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9
3	10	11	12

```
int[][] array = {
   {1, 2, 3},
   {4, 5, 6},
   {7, 8, 9},
   {10, 11, 12}
};
```

matrix.length? 5

matrix[0].length? 5

array.length? 4

array[0].length? 3

声明、创建并初始化二维数组

以上三步可以合成一步，例如：

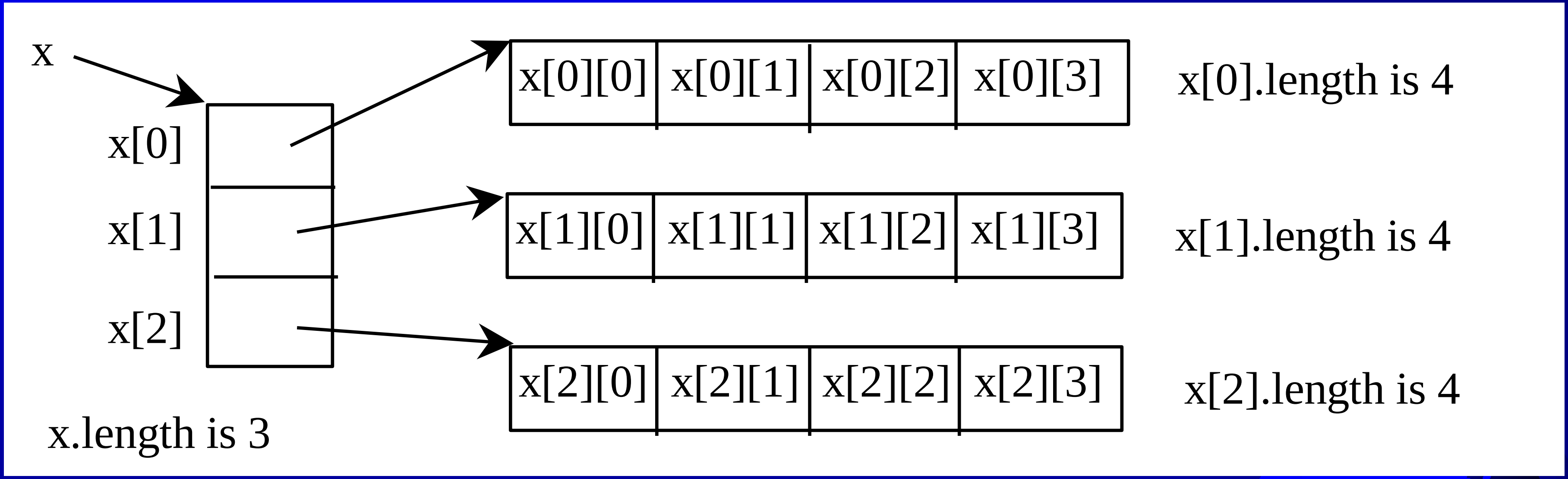
```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

相当于

```
int[][] array = new int[4][3];
array[0][0] = 1; array[0][1] = 2; array[0][2] = 3;
array[1][0] = 4; array[1][1] = 5; array[1][2] = 6;
array[2][0] = 7; array[2][1] = 8; array[2][2] = 9;
array[3][0] = 10; array[3][1] = 11; array[3][2] = 12;
```

二维数组的长度属性

```
int[][] x = new int[3][4];
```


x
x[0]
x[1]
x[2]
x.length is 3
x[0][0] x[0][1] x[0][2] x[0][3]
x[1][0] x[1][1] x[1][2] x[1][3]
x[2][0] x[2][1] x[2][2] x[2][3]
x[0].length is 4
x[1].length is 4
x[2].length is 4

再看个例子

```
int[][] array = {
  {1, 2, 3},
  {4, 5, 6},
  {7, 8, 9},
  {10, 11, 12}
};
```

```
array.length
array[0].length
array[1].length
array[2].length
array[3].length
```

array[4].length？越界了，下标最大是3。Java会引发异常 ArrayIndexOutOfBoundsException

锯齿数组

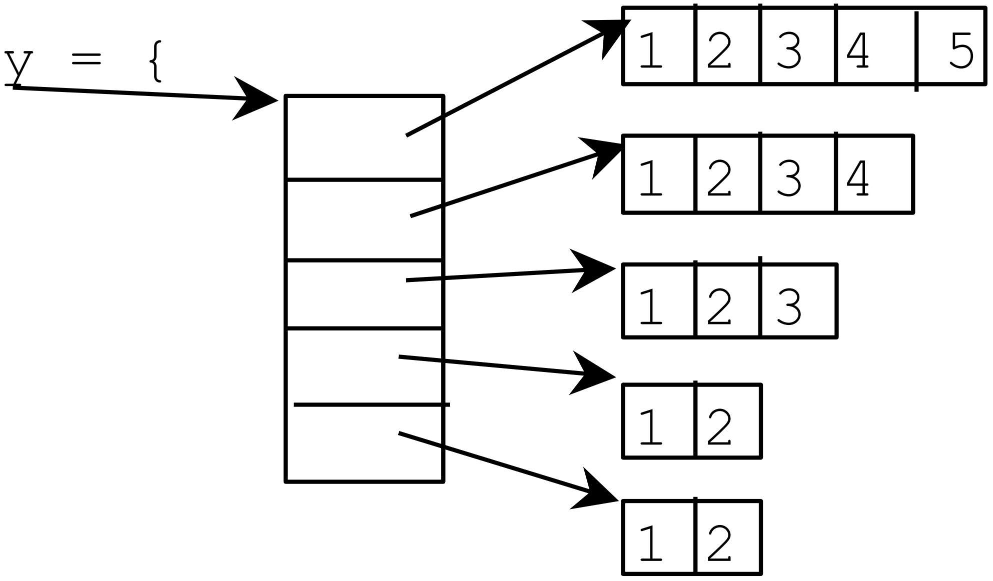
Java把二维数组设计为数组的数组，每一行的数组是独立的，并不要求长度相同，所以Java允许你构造一个锯齿数组：

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

```
  matrix.length is 5
matrix[0].length is 5
matrix[1].length is 4
matrix[2].length is 3
matrix[3].length is 2
matrix[4].length is 1
```

锯齿数组图解

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



二维数组常用例程

以下是6个常用例程，欢迎抄袭，鼓励模仿：

1. (用输入值初始化数组)
2. (用随机值初始化数组)
3. (打印数组)
4. (数组求和)
5. (数组按列求和)
6. (打散数组)

1. 用输入值初始化数组

```
java.util.Scanner input = new Scanner(System.in);
for (int row = 0; row < matrix.length; row++) {
  for (int column = 0; column < matrix[row].length;
      column++) {
    matrix[row][column] = input.nextInt();
  }
}
```

2. 用随机值初始化数组

```
for (int row = 0; row < matrix.length; row++) {
  for (int column = 0; column < matrix[row].length;
      column++) {
    matrix[row][column] = (int)(Math.random() *
      100);
  }
}
```

3. 打印数组

```
for (int row = 0; row < matrix.length; row++) {
  for (int column = 0; column < matrix[row].length;
      column++) {
    System.out.print(matrix[row][column] + " ");
  }

  System.out.println();
}
```

4. 数组求和

```
int total = 0;
for (int row = 0; row < matrix.length; row++) {
  for (int column = 0; column < matrix[row].length;
      column++) {
    total += matrix[row][column];
  }
}
```

5. 数组按列求和

```
for (int column = 0; column < matrix[0].length; column++) {
  int total = 0;
  for (int row = 0; row < matrix.length; row++)
    total += matrix[row][column];
  System.out.println("Sum for column " + column + " is "
    + total);
}
```

6. 打散数组

```
for (int i = 0; i < matrix.length; i++) {
  for (int j = 0; j < matrix[i].length; j++) {
    int i1 = (int)(Math.random() * matrix.length);
    int j1 = (int)(Math.random() * matrix[i].length);
    // Swap matrix[i][j] with matrix[i1][j1]
    int temp = matrix[i][j];
    matrix[i][j] = matrix[i1][j1];
    matrix[i1][j1] = temp;
  }
}
```

二维数组作形参

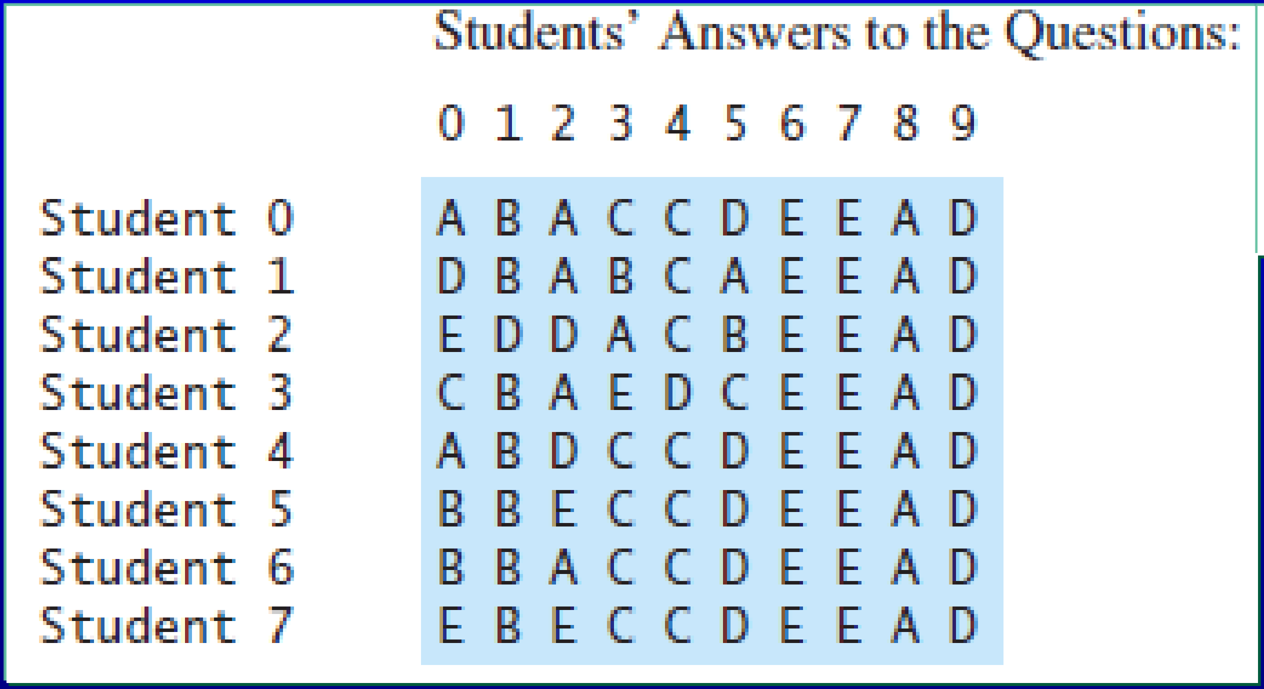
```
public static int sum(int[][] m) {
  int total = 0;
  for (int row = 0; row < m.length; row++)
    for (int column = 0; column < m[row].length; column++)
       total += m[row][column];
   return total;
}
```

上述方法的形参是一个二维数组，因此调用此方法需要一个二维数组的实参，例如：

```
int[][] m = new int[3][4]; int total = sum(m);
```

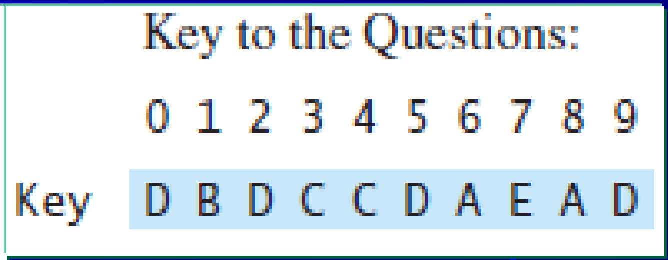
例题：自动评卷

- 一份试卷有10题，8个学生参加考试，所有答卷存储在一个二维数组中，标准答案存储在一个一维数组中，写一个程序自动评卷。



Students' Answers to the Questions:

	0	1	2	3	4	5	6	7	8	9
Student 0	A	B	A	C	C	D	E	E	A	D
Student 1	D	B	A	B	C	A	E	E	A	D
Student 2	E	D	D	A	C	B	E	E	A	D
Student 3	C	B	A	E	D	C	E	E	A	D
Student 4	A	B	D	C	C	D	E	E	A	D
Student 5	B	B	E	C	C	D	E	E	A	D
Student 6	B	B	A	C	C	D	E	E	A	D
Student 7	E	B	E	C	C	D	E	E	A	D



Key to the Questions:

	0	1	2	3	4	5	6	7	8	9
Key	D	B	D	C	C	D	A	E	A	D

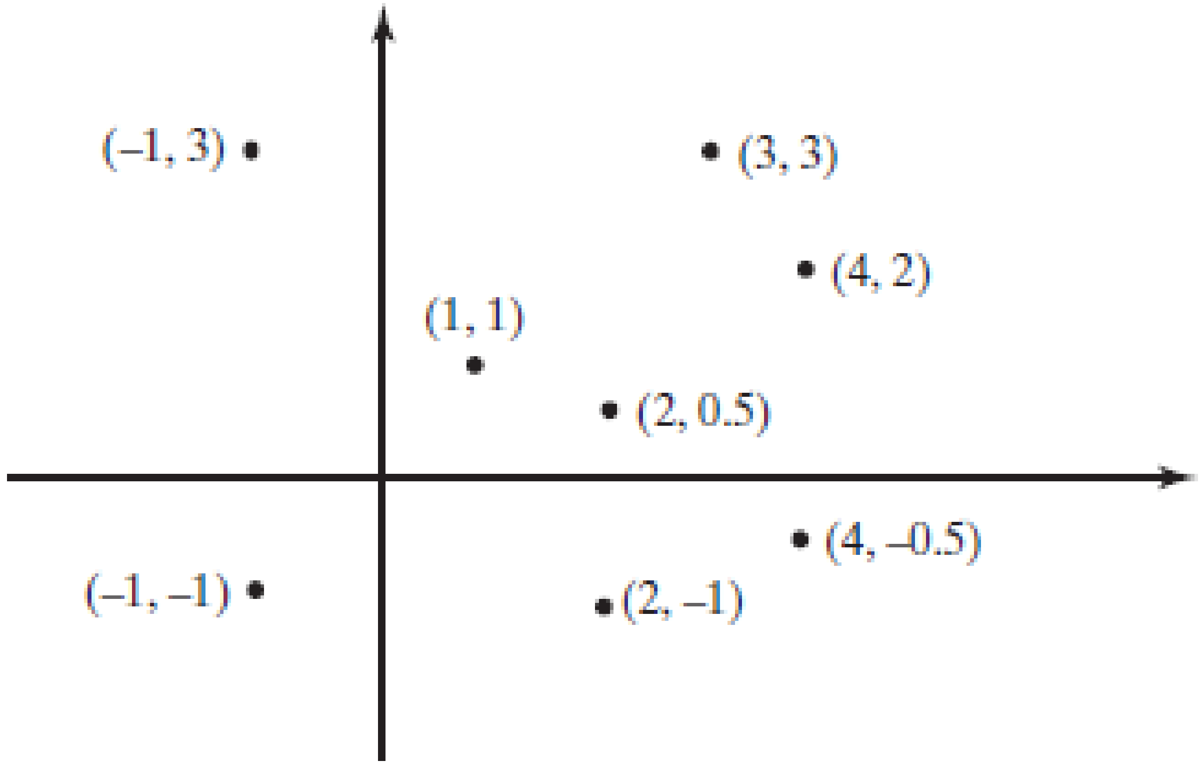
源代码—1/2

```
public class GradeExam {
/** Main method */
public static void main(String[] args) {
// Students' answers to the questions
char[][] answers = {
{ 'A', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D' },
{ 'D', 'B', 'A', 'B', 'C', 'A', 'E', 'E', 'A', 'D' },
{ 'E', 'D', 'D', 'A', 'C', 'B', 'E', 'E', 'A', 'D' },
{ 'C', 'B', 'A', 'E', 'D', 'C', 'E', 'E', 'A', 'D' },
{ 'A', 'B', 'D', 'C', 'C', 'D', 'E', 'E', 'A', 'D' },
{ 'B', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D' },
{ 'B', 'B', 'A', 'C', 'C', 'D', 'E', 'E', 'A', 'D' },
{ 'E', 'B', 'E', 'C', 'C', 'D', 'E', 'E', 'A', 'D' } };

// Key to the questions
char[] keys = { 'D', 'B', 'D', 'C', 'C', 'D', 'A', 'E', 'A', 'D'
};
```

源代码—2/2

```
// Grade all answers
for (int i = 0; i < answers.length; i++) {
  // Grade one student
  int correctCount = 0;
  for (int j = 0; j < answers[i].length; j++) {
    if (answers[i][j] == keys[j])
      correctCount++;
  }

  System.out.println("Student " + i + "'s correct
count is "
+ correctCount);
}
}
}
```

例题：找距离最近的两点



	x	y
0	–1	3
1	–1	–1
2	1	1
3	2	0.5
4	2	–1
5	3	3
6	4	2
7	4	–0.5

谈谈解题思路

- 首先，用一个二维数组存储所有点的坐标：**double**[][] points = **new double**[N][**2**];
- 使用穷举法，计算每两个点的距离，然后找出它们的最小值。
- 穷举可以用二重循环，外循环从p1,p2,…pN，内循环从p1,p2,…pN，即可穷尽所有组合。考虑到距离的对称性，内循环只需要从外循环的下一个点开始计算。

关键代码

```
for (int i = 0; i < points.length; i++) {
 for (int j = i + 1; j < points.length; j++) {
  double distance = distance(points[i][0], points[i][1],
             points[j][0], points[j][1]); // Find distance
  if (shortestDistance > distance) {
   p1 = i; // Update p1
   p2 = j; // Update p2
   shortestDistance = distance; // Update shortestDistance
  }
}
```

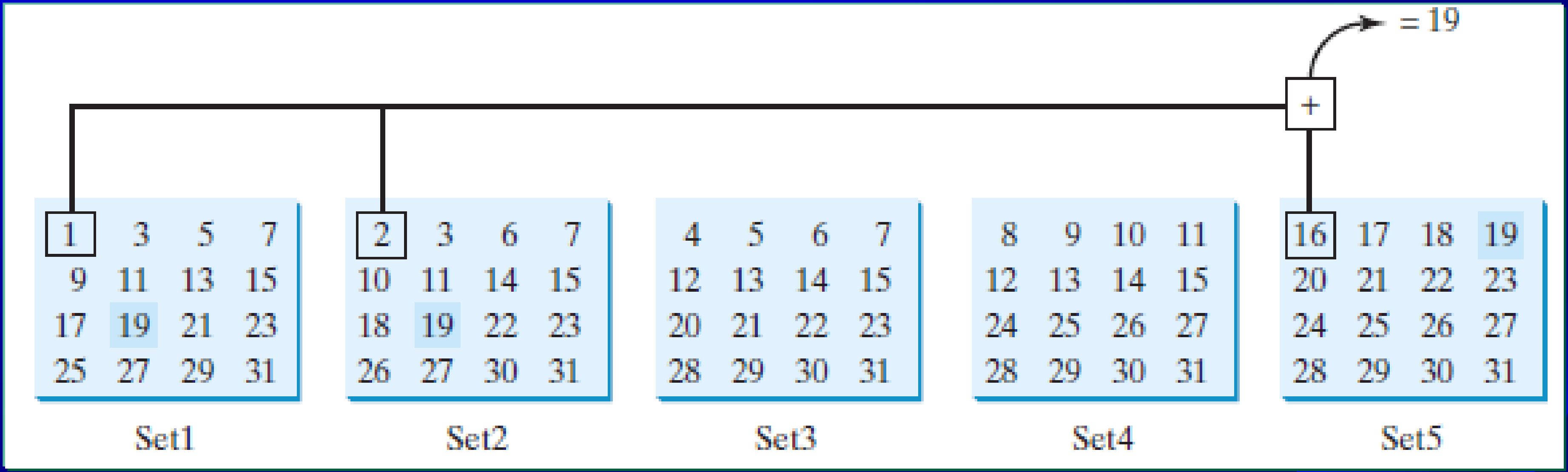
多维数组

Java中多维数组的用法和二维数组差不多，依然是采用数组的数组这种技术来构造。例如一个三维数组可以这样声明和创建：

```
double[][][] scores = new double[10][5][2];
```

例题：猜生日

- 还记得那些天我们一起猜过的生日吗？5张表格，依次提问，最后求和。现在，我们把表格存储在三维数组里，然后再猜一次。



Set1

1	3	5	7
9	11	13	15
17	19	21	23
25	27	29	31

Set2

2	3	6	7
10	11	14	15
18	19	22	23
26	27	30	31

Set3

4	5	6	7
12	13	14	15
20	21	22	23
28	29	30	31

Set4

8	9	10	11
12	13	14	15
24	25	26	27
28	29	30	31

Set5

16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31

源代码—1/2

```
import java.util.Scanner;
public class GuessBirthdayUsingArray {
public static void main(String[] args) {
int day = 0; // Day to be determined
int answer;
int[][][] dates = {
{{ 1, 3, 5, 7 }, { 9, 11, 13, 15 }, { 17, 19, 21, 23 }, { 25, 27, 29, 31 }},
{{ 2, 3, 6, 7 }, { 10, 11, 14, 15 }, { 18, 19, 22, 23 }, { 26, 27, 30, 31 }},
{{ 4, 5, 6, 7 }, { 12, 13, 14, 15 }, { 20, 21, 22, 23 }, { 28, 29, 30, 31 }},
{{ 8, 9, 10, 11 }, { 12, 13, 14, 15 }, { 24, 25, 26, 27 }, { 28, 29, 30, 31 }},
{{ 16, 17, 18, 19 }, { 20, 21, 22, 23 }, { 24, 25, 26, 27 }, { 28, 29, 30, 31 }}
};
```

源代码—2/2

```
Scanner input = new Scanner(System.in);
for (int i = 0; i < 5; i++) {
  System.out.println("Is your birthday in Set" + (i + 1) + "?");
  for (int j = 0; j < 4; j++) {
    for (int k = 0; k < 4; k++)
      System.out.printf("%4d", dates[i][j][k]);
    System.out.println();
  }
  System.out.print("\nEnter 0 for No and 1 for Yes: ");
  answer = input.nextInt();
  if (answer == 1)
    day += dates[i][0][0];
}
System.out.println("Your birthday is " + day);
}}
```

THE END