Array 2

Programming Fundamentals Teaching Team 2023

Objectives

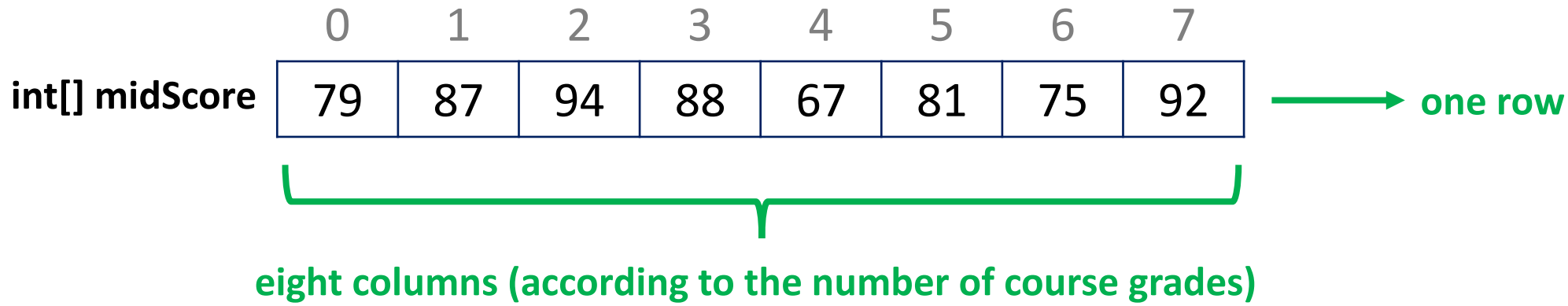
After studying this material, students should be able to:

- Understand the concept of 2-dimensional arrays
- Provide examples of the use of 2-dimensional arrays
- Solve case studies with 2-dimensional arrays

Preface

- In the previous material, a one-dimensional array can be used to store several values in a variable. The array consists of **only one row** and **several columns**
- Example:

  A student's Mid score in 8 courses is stored in an array variable



Preface

- How to store the Mid scores of 5 students in 8 courses into an array variable?

	Course 1	Course 2	Course 3	Course 4	Course 5	Course 6	Course 7	Course 8
Student 1	79	87	94	88	67	81	75	92
Student 2	63	83	58	80	86	69	98	87
Student 3	84	88	60	82	80	74	84	75
Student 4	70	91	65	94	80	91	85	60
Student 5	93	84	77	97	76	82	73	91

Preface

- One-dimensional arrays cannot be used because the value data to be stored has more than one row
- Do we need to create 5 array variables to store the values for each student?

	0	1	2	3	4	5	6	7
int[] student1	79	87	94	88	67	81	75	92
int[] student2	63	83	58	80	86	69	98	87
int[] student3	84	88	60	82	80	74	84	75
int[] student4	70	91	65	94	80	91	85	60
int[] student5	93	84	77	97	76	82	73	91

2-Dimensional Array

- 2-dimensional arrays can be used to store data consisting of **several rows** and **several columns** in an array variable
- Similar to one-dimensional arrays, 2-dimensional arrays also have index numbers, but the index number consists of 2 numbers

Students (row) \ Courses (column)	0	1	2	3	4	5	6	7
0	79	87	94	88	67	81	75	92
1	63	83	58	80	86	69	98	87
2	84	88	60	82	80	74	84	75
3	70	91	65	94	80	91	85	60
4	93	84	77	97	76	82	73	91

Row index

```
nilaiUTS[3][5] = 91
```

Column index

2-Dimensional Array Declaration

- A 2-dimensional array can be illustrated as a matrix or table of size x rows and y columns
- To declare a 2-dimensional array variable, the method is the same as for a one-dimensional array, but the number of square brackets [ ] is different
- General form of array declaration:

```
dataType[][] arrayName;
```

- Another form of array declaration:

```
dataType [][]arrayName;
dataType arrayName[][];
dataType []arrayName[];
```

- Example:

```
int[][] midScore;
double [][]landArea;
char gender[][];
int []age[];
```

2-Dimensional Array Instantiation

- In order to be used, the 2-dimensional array that has been declared must first be instantiated with the keyword new and determine the **number of row and column elements**.
- Array instantiation:

```
arrayName = new dataType[numberOfRows][numberOfColumns];
```

- Example:

```
midScore = new int[5][8];
landArea = new double[10][3];
gender = new char[7][30];
age = new int[2][10];
```

Declaration & Instantiation of 2-Dimensional Array

- Declarations and instantiations can also be written in one statement line
- Array declaration and instantiation:

```
dataType[][] arrayName = new dataType[numberOfRows][numberOfColumns];
```

- Example:

```
int[][] midScore = new int[5][8];
double [][]landArea = new double[10][3];
```

2-Dimensional Array with Different Lengths for Each Row

- Declaration and instantiation of a 2-dimensional array with a different length for each row can be done in the following way:

```
dataType[][] arrayName = new dataType[numberOfRows][];
dataType arrayName[i] = new dataType[numberOfColumns];
```

- Example:

```
int[][] stocks = new int[3][];
stocks[0] = new int[2];
stocks[1] = new int[5];
stocks[2] = new int[3];
```

Default Value

- Just like a one-dimensional array, instantiating a 2-dimensional array (with the new keyword) provides a default value for each element
  - String → null
  - int, double → 0
  - boolean → false

```
int[][] x = new int[3][5];
```

	0	1	2	3	4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0

```
boolean[][] y = new boolean[2][3];
```

	0	1	2
0	False	False	False
1	False	False	False

```
String[][] z = new String[3][2];
```

	0	1
0	*null*	*null*
1	*null*	*null*
2	*null*	*null*

Initializing 2-Dimensional Arrays

- Just like with a one-dimensional array, initialization of a 2-dimensional array can be done with curly braces

```
int[][] score = new int[][]{
      {84, 57, 93},
      {76, 71, 82, 88, 90},
      {97}
};
```

```
int[][] score = {
      {84, 57, 93},
      {76, 71, 82, 88, 90},
      {97}
};
```

	0	1	2	3	4
0	84	57	93		
1	76	71	82	88	90
2	97				

2-Dimensional Array Size

- Each array, both one-dimensional and 2-dimensional arrays, has a size
- The size of the array can be known through the length attribute
- Example:

```
int[][] x = new int[3][5];
```

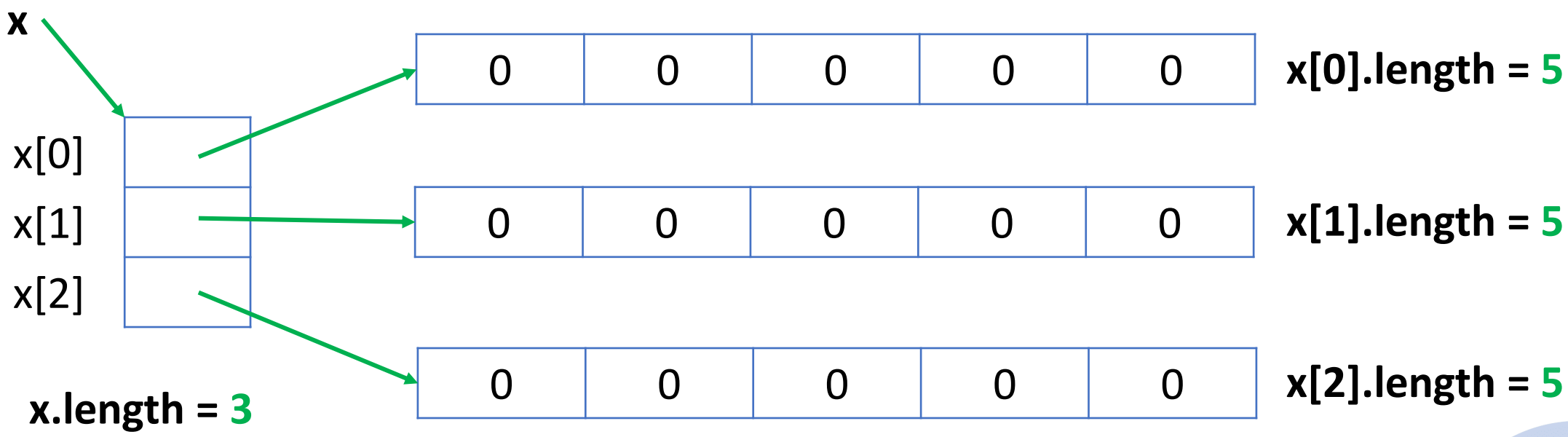
x.length returns 3, or its row (the first dimension)

x[0].length returns 5, or its column (second dimension)

- When using attribute / variable **length**, the advantage is that when the array size changes, we don't need to change the code to input / display the array.

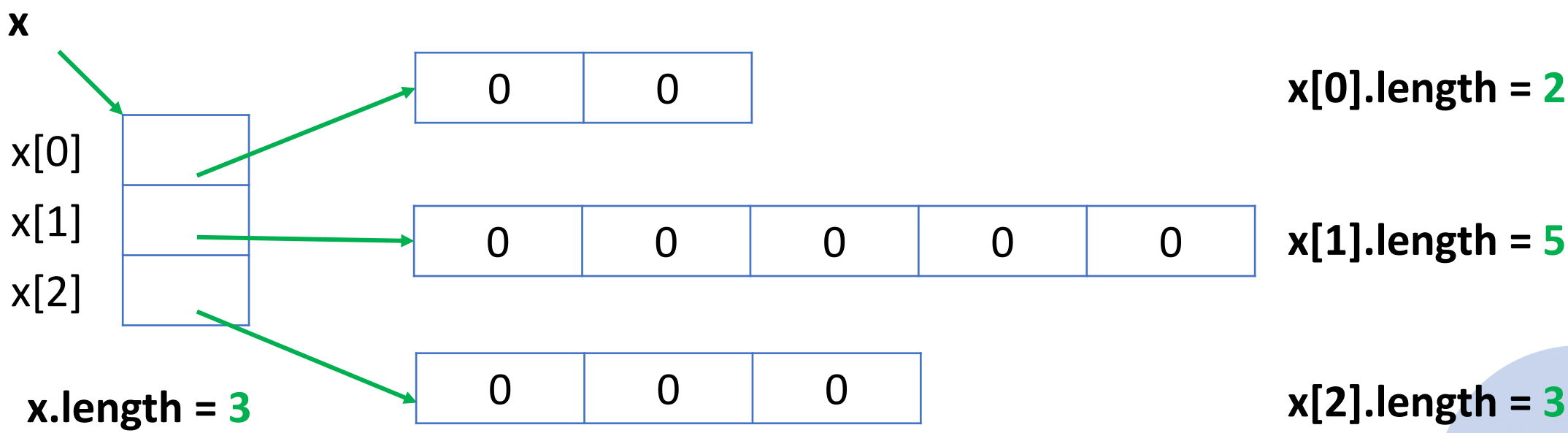
2-Dimensional Array Size (2)

```
int[][] x = new int[3][5];
```



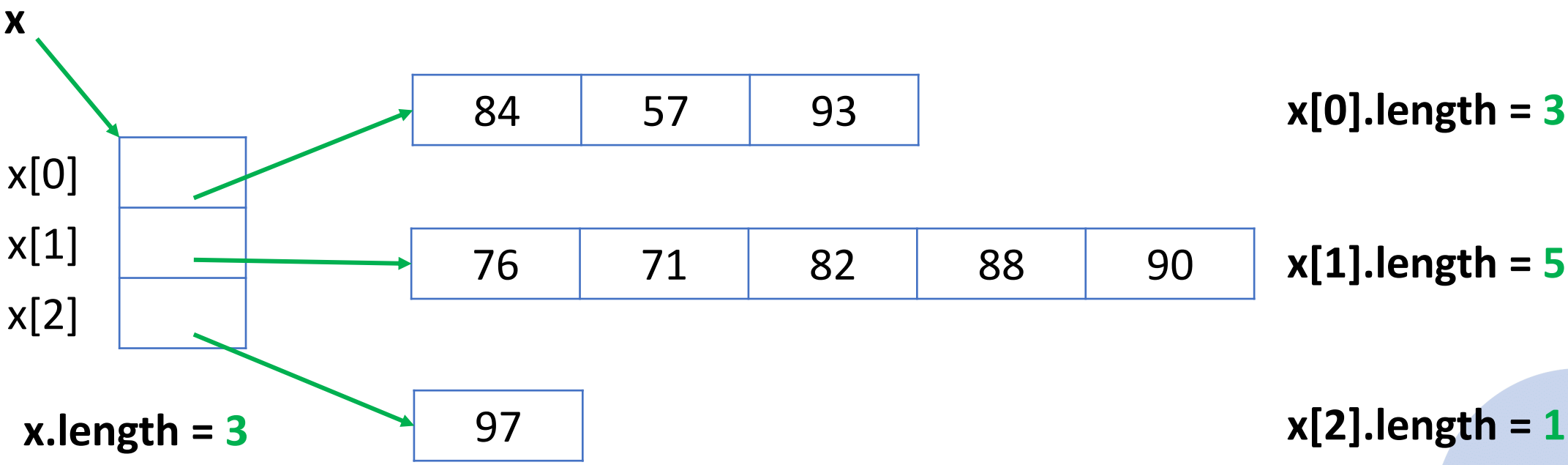
2-Dimensional Array Size (3)

```
int[][] x = new int[3][];
x[0] = new int[2];
x[1] = new int[5];
x[2] = new int[3];
```



2-Dimensional Array Size (4)

```
int[][] x = {
    {84, 57, 93},
    {76, 71, 82, 88, 90},
    {97}
};
```



Accessing 2-Dimensional Array Elements

- Accessing one of the elements of a 2-dimensional array can be done by writing the row and column indices of the array variable

	0	1	2	3	4
0	84	57	93	7	7
1	76	71	82	88	90
2	97	0	3	0	9

```
System.out.print(value[1][2]);    //82
System.out.print(value[0][1]);    //57
```

Filling 2-Dimensional Array Elements

- Filling in the elements of a 2-dimensional array can be done by accessing the row and column indices of the array variable
- Values are filled in using the assignment operator

	0	1	2	3	4
0	84	57	93	7	7
1	76	71	82	88	90
2	97	0	3	0	9

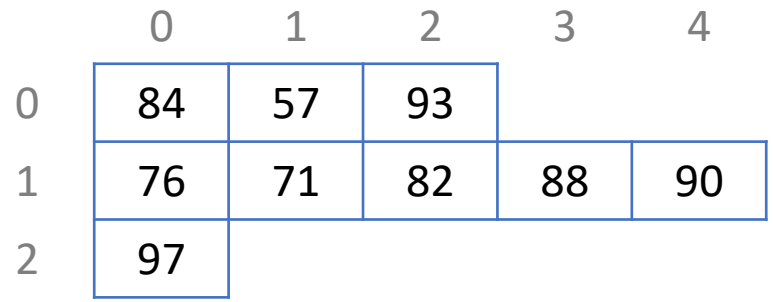
```
value[2][3] = 77;
```

	0	1	2	3	4
0	84	57	93	7	7
1	76	71	82	88	90
2	97	0	3	77	9

ArrayIndexOutOfBoundsException

- Note that the length for each row in the array is not necessarily the same

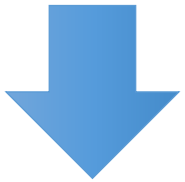
```
int[][] value = {
        {84, 57, 93},
        {76, 71, 82, 88, 90},
        {97}
};
```



```
int[][] value = {
      {84, 57, 93},
      {76, 71, 82, 88, 90},
      {97}
};
```

	0	1	2	3	4
0	84	57	93		
1	76	71	82	88	90
2	97				

```
value[2][2] = 1
System.out.print(nilai[2][3]);
```



ArrayIndexOutOfBoundsException

array accessed with illegal index

Case Study

Example 1

Togamas has three branch stores in Malang. There are 6 encyclopedias sold at the Dieng and Soehat branches. The Sengkaling branch can sell 4, 6 and 5 novels, comics and encyclopedias respectively. The Dieng branch can only sell 2 novels, but 8 comics have been sold. On the other hand, the Soehat branch can only sell 2 novels, but 8 comics have been sold. On the other hand, the Soehat branch can sell 7 novels, but unfortunately only 3 comics are sold. How to store sales data in a 2-dimensional array?

Example 1 - Answer

Togamas has three branch stores in Malang. There are 6 encyclopedias sold at the Dieng and Soehat branches. The Sengkaling branch can sell 4, 6 and 5 novels, comics and encyclopedias respectively. The Dieng branch can only sell 2 novels, but 8 comics have been sold. On the other hand, the Soehat branch can only sell 2 novels, but 8 comics have been sold. On the other hand, the Soehat branch can sell 7 novels, but unfortunately only 3 comics are sold. How to store sales data in a 2-dimensional array?

Book Category (Column)

Branch (Row)

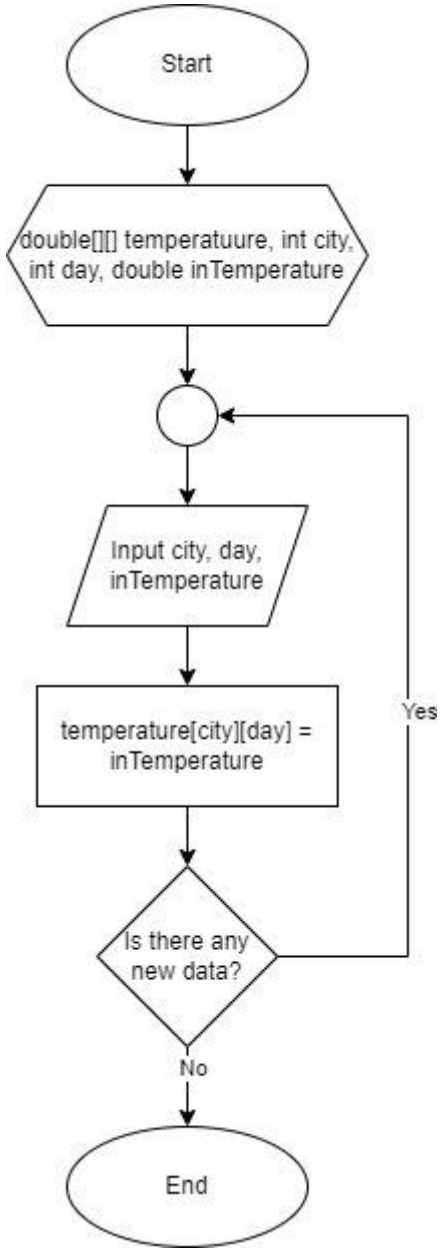
		Novel	Comic	Encyclopedia
		0	1	2
Dieng	0	2	8	6
Soehat	1	7	3	6
Sengkaling	2	4	6	5

Example 2

Temperature measurements were taken for 7 consecutive days in five cities in Japan during the summer, namely Tokyo, Osaka, Sapporo, Fukuoka and Naha. Create a flowchart to get temperature data from the user and store it in a 2-dimensional array.

Example 2 - Answer

Temperature measurements were taken for 7 consecutive days in five cities in Japan during the summer, namely Tokyo, Osaka, Sapporo, Fukuoka and Naha. Create a flowchart to get temperature data from the user and store it in a 2-dimensional array.



Cities → rows
Days → columns

Can dimensions for columns and rows be reversed?

Assignment

- Identify according to each project group, the features that require the use of 2-dimensional arrays
- Create a flowchart to manipulate and display array elements based on user input