CS1101 Programming and Problem Solving

Dr. Gina Bai

Spring 2023

Logistics

- **ZY-6** on zyBook > Assignments
  - Due: **Monday, March 20**, at 11:59pm
- **PA08 – W, A, B** on zyBook > Chap 11
  - Due: **Saturday, March 25**, at 11:59pm

How would we solve this problem?

Write a program that

- prompts user for number of students
- prompts user for Exam 1 grades for each student
- prints average grade for Exam 1
- prints number of students with Exam 1 grade higher than average

```
How many students? 5
Student 1's Exam 1 Grade: 96
Student 2's Exam 1 Grade: 92.5
Student 3's Exam 1 Grade: 80.5
Student 4's Exam 1 Grade: 99
Student 5's Exam 1 Grade: 87
Average Exam 1 Grade: 91.0
3 students were above average.
```

Why is this problem difficult?

- We need each input value twice:
  - First to compute the average (a cumulative sum)
  - Then count how many were above average
- We could read each value into a variable... but we:
  - do not know how many students are needed until the program runs
  - do not know how many variables to declare

```
How many students? 5
Student 1's Exam 1 Grade: 97
Student 2's Exam 1 Grade: 92.5
Student 3's Exam 1 Grade: 80.5
Student 4's Exam 1 Grade: 99
Student 5's Exam 1 Grade: 87
Average Exam 1 Grade: 91.0
3 students were above average.
```

Array Basics

zyBook Chap 7.1, 7.2, 7.3, 7.4

Array Terminologies

An array object is an **indexed** (possibly large) **collection of data** of the **same type**

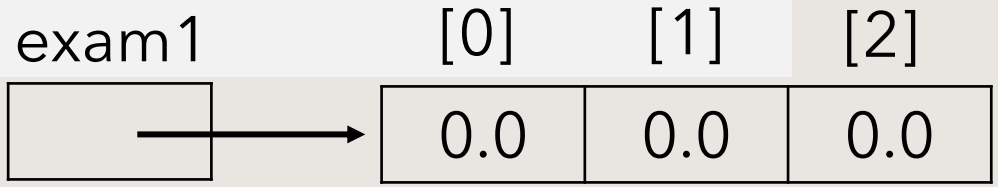
- Elements → variables stored in an array
- Index → an integer indicating the position of an element in an array
  - **Zero-based** indexing

Arrays Construction

```
<type>[] <arrayName> = new <type>[<arraySize>];
```

- For example, construct a double type array to store the exam 1 grades for three students.

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = new double[3];
        ...
    }
}
```



Array Auto-initialization

The initialization of variables to a default value.

Type	Default Value
int	`0`
double	`0.0`
char	`'\u0000'`
boolean	`false`
Objects (e.g., String, Array)	**`null (Java keyword)`**

`null` is a special value that shows that an **object** is **referring to nothing**.

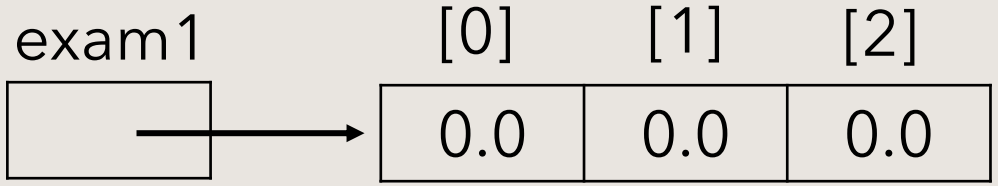
Access Array Elements

- Accessing an array element with **<arrayName>[<index>]**

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = new double[3];

        System.out.println("The array elements are: " + exam1[0] + ", " +
                           exam1[1] + ", and " + exam1[2] + ".");
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
The array elements are: 0.0, 0.0, and 0.0.
```



Use Array Elements

- Array elements can be treated as a variable of a given type

Q: What is the resulting array after each of the following statement?

```
int[] list = new int[3];   // [0, 0, 0]
list[0] = 3;               // [3, 0, 0]
list[2]++;                 // [3, 0, 1]
list[0]--;                 // [2, 0, 1]
list[0] *= 4;              // [8, 0, 1]
```

Length of an Array

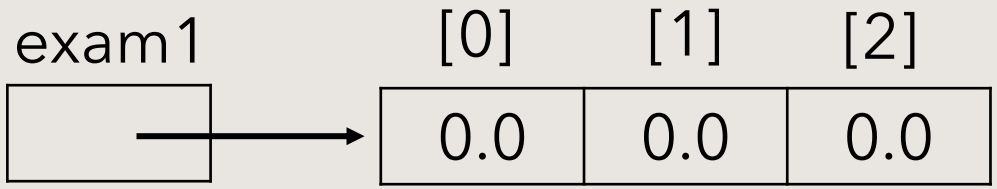
IMPORTANT: **Length** is a **read-only property of an array**, not a method

- Getting the length of an array with **<arrayName>.length**

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = new double[3];

        System.out.println("The array contains " + exam1.length + " elements: " +
                           exam1[0] + ", " + exam1[1] + ", and " + exam1[2] + ".");
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
The array contains 3 elements: 0.0, 0.0, and 0.0.
```

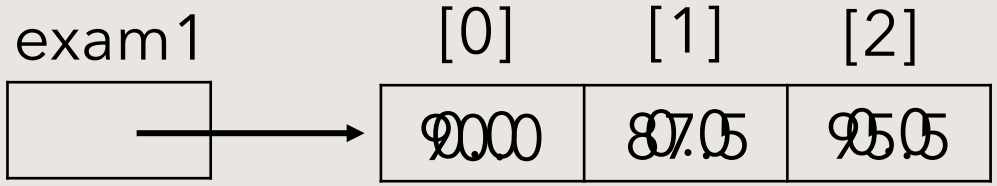


Array Initialization (Not Preferred Approach)

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = new double[3];
        exam1[0] = 90.0;
        exam1[1] = 87.5;
        exam1[2] = 95.5;

        System.out.println("The array contains " + exam1.length + " elements: " +
                           exam1[0] + ", " + exam1[1] + ", and " + exam1[2] + ".");
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
The array contains 3 elements: 90.0, 87.5, and 95.5.
```



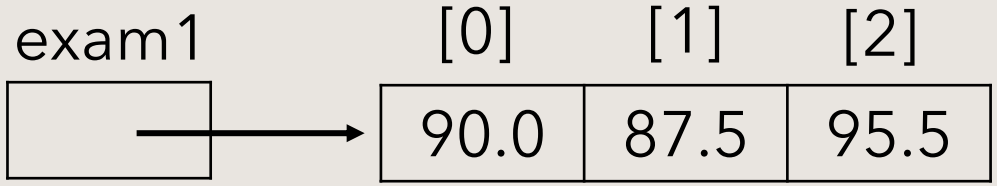
Array Initialization (Shorthand)

- Initializing an array with
  `<type>[] <arrayName> = { val1, val2, …, valN };`

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = {90.0, 87.5, 95.5};

        System.out.println("The array contains " + exam1.length + " elements: " +
                           exam1[0] + ", " + exam1[1] + ", and " + exam1[2] + ".");
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
The array contains 3 elements: 90.0, 87.5, and 95.5.
```



Array Initialization (via loop, usually preferred)

```
for (int i = 0; i < arrayName.length; ++i) {
    // access and initialize each element via arrayName[i] = <val>;
}
```

Arrays are usually used to store a collection of data from

- console
  - e.g., prompts the user for array size, and then for each element
- an input file
  - e.g., student1 has 7 PA scores, student2 has 7 PA scores, …

Array Traversal

- Traversing an array with a for loop

```
for (int i = 0; i < arrayName.length; ++i) {
    // access each element via arrayName[i]
}
```

For example,

```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = {90.0, 87.5, 95.5};

        for (int i = 0; i < exam1.length; ++i) {
            System.out.println("Student " + (i + 1) + ": " + exam1[i]);
        }
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
Student 1: 90.0
Student 2: 87.5
Student 3: 95.5
```

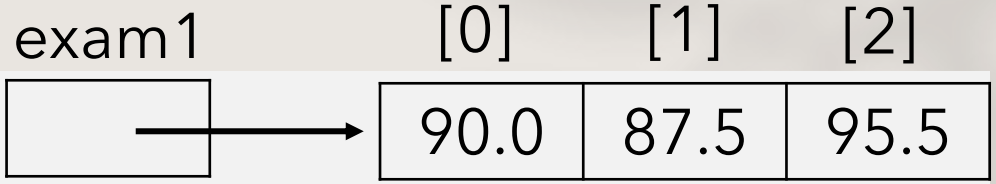
Error Handling – Array Index Out Of Bounds

Accessing an element of an array at an index

- **less than `0`** or
- **greater than `length - 1`**

will result in an `ArrayIndexOutOfBoundsException` during runtime (unchecked exception).

Error Handling – Array Index Out Of Bounds




```
public class Gradebook {
    public static void main (String[] args) {
        double[] exam1 = {90.0, 87.5, 95.5};

        for (int i = 0; i <= exam1.length; ++i) {
            System.out.println("Student " + (i + 1) + ": " + exam1[i]);
        }
    }
}
```

```
$ javac Gradebook.java
$ java Gradebook
Student 1: 90.0
Student 2: 87.5
Student 3: 95.5
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:
Index 3 out of bounds for length 3
at Gradebook.main(Gradebook.java:6)
```

Coding Practice

Write a program that

- prompts user for number of students
- prompts user for Exam 1 grades for each student
- prints average grade for Exam 1
- prints number of students with Exam 1 grade higher than average

Sample Solution

Try to improve the program so it

- validates if the number of students is a positive integer,
- validates if the exam grade is a double from 0 – 100.

```java
import java.util.Scanner;

public class Gradebook {
    public static void main (String[] args) {
        // See the starter code for the main method
    }

    public static void getGrades (double[] exam1, Scanner input) {
        // Prompt user for Exam 1 grades for each student
        for (int i = 0; i < exam1.length; i++) {
            System.out.print("Student " + (i + 1) + "\'s Exam 1 Grade: ");
            exam1[i] = input.nextDouble();
        }
    }

    public static double calcAvg (double[] exam1) {
        double average = 0;
        for (int i = 0; i < exam1.length; i++) {
            average += exam1[i];
        }
        return average /= exam1.length;
    }

    public static int countAbove (double[] exam1, double average){
        int numAbove = 0;
        for (int i = 0; i < exam1.length; i++) {
            if (exam1[i] > average) {
                numAbove++;
            }
        }
        return numAbove;
    }
}
```