CS1101 Programming and Problem Solving

Dr. Gina Bai

Spring 2023

Logistics

- **ZY-7B** and **ZY-8A**on zyBook > Assignments
  - Due: **Wednesday, April 5**, at 11:59pm
- **PA10 – A, B** on zyBook > Chap 11
  - Due: **Thursday, April 6**, at 11:59pm
- Midterm Exam 2 Regrade Request
  - Due: Tuesday, April 11

Start Early!!!

Recap

The reference of **a** is reassigned, it now refers to the array of **[f, f]**.

Q: What's the output of the following code?

The reference of **a** is never reassigned, and hence **a** still refers to the array of **[f, f, t, f, f]**.

```
boolean[] a = new boolean[5];

System.out.println( a.length );          // 5
System.out.println( Arrays.toString(a) );   // [false, false, false, false, false]

a[2] = true;
System.out.println( Arrays.toString(a) );   // [false, false, true, false, false]

System.out.println( Arrays.toString(Arrays.copyOf(a, 2)) );   // [false, false]
System.out.println( Arrays.toString(Arrays.copyOf(a, 3)) );   // [false, false, true]

a = Arrays.copyOf(a, 2);
System.out.println( Arrays.toString(a) );   // [false, false]
System.out.println( Arrays.toString(Arrays.copyOf(a, 3)) );   // [false, false, false]
```

Modifying Arrays

zyBook Chap 7.6, 7.7, 7.8

Commonly Seen Manipulation

- Replace Array Elements
- Swap Array Elements
  - Swap the elements at two indices
  - Reverse the array
  - Shift the array elements towards left/right

Replace Array Elements

- Replace all the instances of x in the array with y

Q: Fill in the blanks

```java
import java.util.Arrays;

public class ReplaceElement {
    public static void main(String[] args) {
        int[] dice = {1, 3, 3, 5, 3 };
        System.out.println("Before replaceAll: " + Arrays.toString(dice));
        replaceAll(dice, 3, 4);
        System.out.println("After replaceAll: " + Arrays.toString(dice));
    }

    /**
     * Replace all instances of target in array with replacement
     *
     * @param array int array use for replacement
     * @param target int to search for in array
     * @param replacement int to replace target with within array
     */
    public static void replaceAll (int[] array, int target, int replacement) {
        for (_____________________________________) {
            if(_____________________________) {
                _____________________________;
            }
        }
    }
}
```

```
$ javac ReplaceElement.java
$ java ReplaceElement
Before replaceAll: [1, 3, 3, 5, 3]
After replaceAll: [1, 4, 4, 5, 4]
```

```java
import java.util.Arrays;

public class ReplaceElement {
    public static void main(String[] args) {
        int[] dice = {1, 3, 3, 5, 3 };
        System.out.println("Before replaceAll: " + Arrays.toString(dice));
        replaceAll(dice, 3, 4);
        System.out.println("After replaceAll: " + Arrays.toString(dice));
    }

    /**
     * Replace all instances of target in array with replacement
     *
     * @param array int array use for replacement
     * @param target int to search for in array
     * @param replacement int to replace target with within array
     */
    public static void replaceAll (int[] array, int target, int replacement) {
        for (int i = 0; i < array.length; ++i) {
            if(array[i] == target) {
                array[i] = replacement;
            }
        }
    }
}
```

```
$ javac ReplaceElement.java
$ java ReplaceElement
Before replaceAll: [1, 3, 3, 5, 3]
After replaceAll: [1, 4, 4, 5, 4]
```

Brainstorm: How to swap two integer values?

```
int a = 1;
int b = 2;
```

Q: How to swap the value of `a` and `b` so that `a` has the value of 2 and `b` has the value of 1? (Hint: in 3 steps)

```
int temp = a;
a = b;
b = temp;
```

Swap Array Elements

Q: Fill in the blanks.

```
/**
 * Swaps the elements at indices x and y in the array
 * @param  a array of integers
 * @param  x index of the first element to be swapped
 * @param  y index of the second element to be swapped
 */
public static void swap (int[] a, int x, int y) {
    System.out.println("--Before swap (swap): " + Arrays.toString(a));
    ____________________;
    ____________________;
    ____________________;
    System.out.println("--After swap (swap): " + Arrays.toString(a));
}
```

Swap Array Elements

Q: Fill in the blanks.

```
/**
 * Swaps the elements at indices x and y in the array
 * @param  a array of integers
 * @param  x index of the first element to be swapped
 * @param  y index of the second element to be swapped
 */
public static void swap (int[] a, int x, int y) {
    System.out.println("--Before swap (swap): " + Arrays.toString(a));
    int temp = a[x];
    a[x] = a[y];
    a[y] = temp;
    System.out.println("--After swap (swap): " + Arrays.toString(a));
}
```

```java
import java.util.Arrays;

public class SwapArray {
    public static void main (String[] args) {
        int[] a = {1, 2, 3, 4};
        System.out.println("Before swap (main): " + Arrays.toString(a));
        swap(a, 2, 3);
        System.out.println("After swap (main): " + Arrays.toString(a));
    }

    /**
     * Swaps the elements at indices x and y in the array
     * @param  a array of integers
     * @param  x index of the first element to be swapped
     * @param  y index of the second element to be swapped
     */
    public static void swap (int[] a, int x, int y) {
        System.out.println("--Before swap (swap): " + Arrays.toString(a));
        int temp = a[x];
        a[x] = a[y];
        a[y] = temp;
        System.out.println("--After swap (swap): " + Arrays.toString(a));
    }
}
```

```
$ javac SwapArray.java
$ java SwapArray
Before swap (main): [1, 2, 3, 4]
--Before swap (swap): [1, 2, 3, 4]
--After swap (swap): [1, 2, 4, 3]
After swap (main): [1, 2, 4, 3]
```

Q: What's the exact output of the following code?

```
public class SwapInt {
    public static void main (String[] args) {
        int x = 1;
        int y = 16;
        System.out.println("Before swap (main): " + x + ", " + y);
        swap(x, y);
        System.out.println("After swap (main): " + x + ", " + y);
    }

    public static void swap (int x, int y) {
        System.out.println("--Before swap (swap): " + x + ", " + y);
        int temp = x;
        x = y;
        y = temp;
        System.out.println("--After swap (swap): " + x + ", " + y);
    }
}
```

In the **scope of swap()**, the value of **x** and **y** are swapped. Not in main(). NOTE: "pass by value"

```
$ javac SwapInt.java
$ java SwapInt
Before swap (main): 1, 16
--Before swap (swap): 1, 16
--After swap (swap): 16, 1
After swap (main): 1, 16
```

Brainstorm: How to reverse an array?

Reverse the order of the elements in an array. For example,

[1, 2, 3, 4, 5] becomes [5, 4, 3, 2, 1]

Approach 1:

- Create an array `tempArr` of the **same size** of `a1`
- Copy the value from `a1` to `tempArr` in a reversed order
- Reassign the reference of `a1`

Reverse Array Elements

Q: Fill in the blanks.

```java
import java.util.Arrays;

public class ReverseArray {
    public static void main (String[] args) {
        int[] a1 = { 1, 2, 3, 4 };
        int[] a2 = { 1, 2, 3, 4, 5 };
        a1 = reverse(a1);
        System.out.println("Reversed a1: " + Arrays.toString(a1));
        a2 = reverse(a2);
        System.out.println("Reversed a2: " + Arrays.toString(a2));
    }

    /**
     * Reverses the elements in an array
     * @param a array of integer
     */
    public static int[] reverse (int[] a) {
        _____________________________; // Construct the tempArr
        // Initialize the tempArr
        for (_____________________________) {
            _____________________________;
        }

        return tempArr;
    }
}
```

Reverse Array Elements

Q: Fill in the blanks.

```java
import java.util.Arrays;

public class ReverseArray {
    public static void main (String[] args) {
        int[] a1 = { 1, 2, 3, 4 };
        int[] a2 = { 1, 2, 3, 4, 5 };
        a1 = reverse(a1);
        System.out.println("Reversed a1: " + Arrays.toString(a1));
        a2 = reverse(a2);
        System.out.println("Reversed a2: " + Arrays.toString(a2));
    }

    /**
     * Reverses the elements in an array
     * @param a array of integer
     */
    public static int[] reverse (int[] a) {
        int[] tempArr = new int[a.length]; // Construct the tempArr
        // Initialize the tempArr
        for (int i = 0; i < a.length; ++i) {
            tempArr[i] = a[a.length - 1 - i];
        }

        return tempArr;
    }
}
```

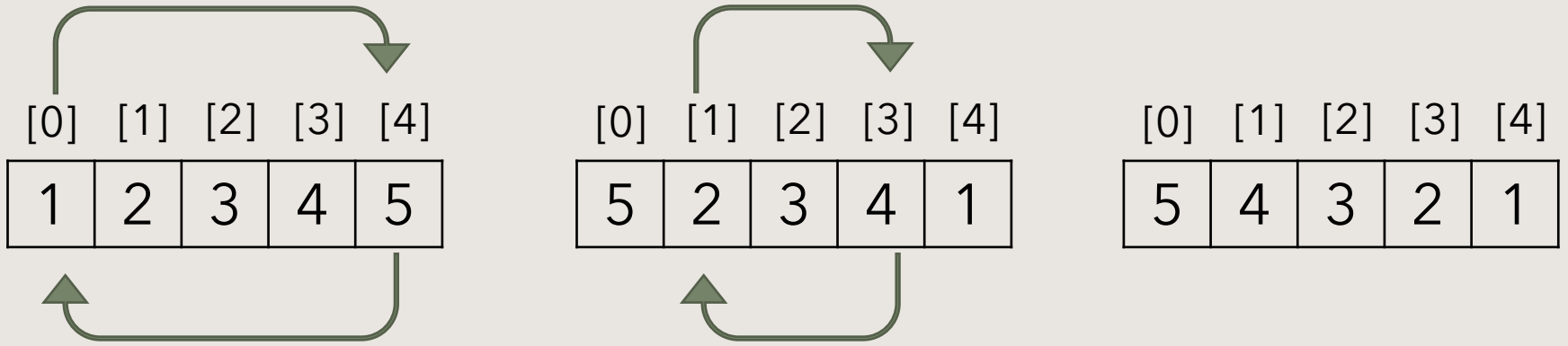
```
$ javac ReverseArray.java
$ java ReverseArray
Reversed a1: [4, 3, 2, 1]
Reversed a2: [5, 4, 3, 2, 1]
```

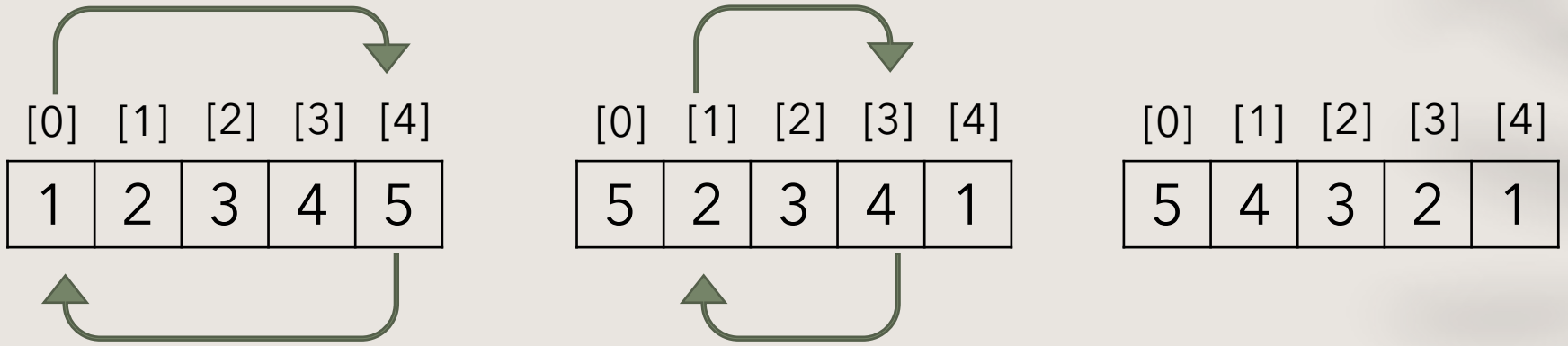
Brainstorm: How to reverse an array?

Reverse the order of the elements in an array. For example,

[1, 2, 3, 4, 5] becomes [5, 4, 3, 2, 1]

Approach 2: Swap the elements within the array





- Swap a[0] with a[a.length – 1] (use expression of <arrName>.length)
- Swap a[1] with a[a.length – 2]
- Repeat till the array is completely reversed
  - When?

Reverse Array Elements

Q: Fill in the blanks.

```java
import java.util.Arrays;

public class ReverseArray {
    public static void main (String[] args) {
        int[] a1 = { 1, 2, 3, 4 };
        int[] a2 = { 1, 2, 3, 4, 5 };
        reverse(a1);
        System.out.println("Reversed a1: " + Arrays.toString(a1));
        reverse(a2);
        System.out.println("Reversed a2: " + Arrays.toString(a2));
    }

    /**
     * Reverses the elements in an array
     * @param a array of integer
     */
    public static void reverse (int[] a) {
        for (_____________________________________) {
            _____________________________________; // expression for j
            swap(a, i, j); // swap elements at indices i and j
        }
    }

    public static void swap (int[] a, int i, int j) {
        int temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
}
```

Reverse Array Elements

Q: Fill in the blanks.

```java
import java.util.Arrays;

public class ReverseArray {
    public static void main (String[] args) {
        int[] a1 = { 1, 2, 3, 4 };
        int[] a2 = { 1, 2, 3, 4, 5 };
        reverse(a1);
        System.out.println("Reversed a1: " + Arrays.toString(a1));
        reverse(a2);
        System.out.println("Reversed a2: " + Arrays.toString(a2));
    }

    /**
     * Reverses the elements in an array
     * @param a array of integer
     */
    public static void reverse (int[] a) {
        for (int i = 0; i <= (a.length - 1)/2; ++i) {// OR: i < a.length / 2
            int j = a.length - 1 - i; // expression for j
            swap(a, i, j); // swap elements at indices i and j
        }
    }

    public static void swap (int[] a, int i, int j) {
        int temp = a[i];
        a[i] = a[j];
        a[j] = temp;
    }
}
```

```
$ javac ReverseArray.java
$ java ReverseArray
Reversed arr1: [4, 3, 2, 1]
Reversed arr2: [5, 4, 3, 2, 1]
```

Coding Practice


```
$ java SwapArray
Enter String: Go Commodore
Before Swapping: [G,o, ,C,o,m,m,o,d,o,r,e]
After Swapping: [o,G,C, ,m,o,o,m,o,d,e,r]

$ java SwapArray
Enter String: 12345
Before Swapping: [1,2,3,4,5]
After Swapping: [2,1,4,3,5]
```

Write a program called **SwapArray** that

- Prompt the user for a string input
- Create an array of characters to store the user input
- Implement a swapPairs method to swap the elements at adjacent indices
  - e.g., elements 0 and 1 are swapped, elements 2 and 3 are swapped, and so on
  - If the array has an odd length, the final element should be left unmodified.
- Output the array before swap
- Output the array after swap

Sample Solution

```java
import java.util.Scanner;
import java.util.Arrays;

public class SwapArrayCoding {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter String: ");
        String userString = in.nextLine();

        char[] userArray = new char[userString.length()];
        for (int i = 0; i < userString.length(); ++i) {
            userArray[i] = userString.charAt(i);
        }

        System.out.println("Before swap: " + Arrays.toString(userArray));
        swapPairs(userArray);
        System.out.println("After swap: " + Arrays.toString(userArray));
    }

    public static void swapPairs(char[] userArray) {
        for (int i = 0; i < userArray.length - 1; i += 2) {
            char temp = userArray[i];
            userArray[i] = userArray[i + 1];
            userArray[i + 1] = temp;
        }
    }
}
```