lesson Plan

Java Array-2

PW SKILLS

Pre-requisites:

- Java Syntax
- Basics of Array

List of concepts involved:

1. Passing array to Methods
2. ArrayList in Java
3. Operations on ArrayList
4. Problem on arrays and Two pointers

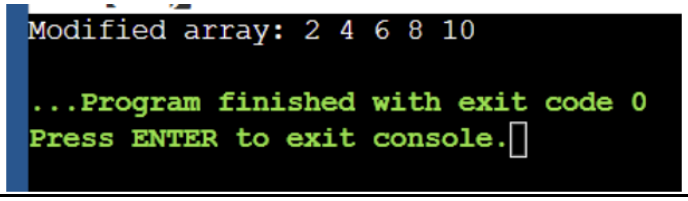
Passing array to methods

In Java, you can pass arrays to methods just like any other data type. When you pass an array to a method, you are actually passing a reference to the array, so any changes made to the array within the method will affect the original array outside the method.

Example:

```
public class Main {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, 5};

        // Passing the array to a method
        modifyArray(numbers);

        // Printing the modified array
        System.out.print("Modified array: ");
        for (int num : numbers) {
            System.out.print(num + " ");
        }
    }

    // Method that modifies the array
    static void modifyArray(int[] arr) {
        for (int i = 0; i < arr.length; i++) {
            arr[i] = arr[i] * 2;
        }
    }
}
```

Output:



In this example, the modifyArray method takes an array as a parameter and doubles each element in the array. The changes made to the array within the method are reflected in the original array outside the method.

Remember that arrays are passed by reference in Java, so modifications to the array within the method affect the original array.

MCQ : When you pass an array as an argument to a Method, what actually gets passed?

1. address of the array
2. values of the elements of the array
3. address of the first element of the array
4. number of elements of the array

Ans: 3. address of the first element of the array

ArrayList

An ArrayList in Java is a part of the Java Collections Framework and is used to dynamically resize the array. It is similar to an array but provides more flexibility and functionality.

Let's recap array first. The Array is a fixed number of groups of Similar kinds of objects placed at a continuous memory location. Similarly, ArrayList is a class which holds the object of the same kind in the order of insertion without any limit to the number of objects to store, i.e. we can say that Array is of fixed size and List is of dynamic sizing.

ArrayList class extends AbstractList class and implements List interface.

Syntax to create ArrayList

```
List<AnyClass> list = new ArrayList<AnyClass>();
```

Features of ArrayList:

1. It uses a dynamic array data structure to store objects and elements.
2. It allows duplicate objects and elements.
3. It maintains the insertion order.
4. It is non-synchronized.
5. Its elements/objects can be accessed randomly.

Basic Operations on ArrayList

1. **Creating an ArrayList:**

```
import java.util.ArrayList;
ArrayList<String> list = new ArrayList<>();
```

2. Adding Elements:
list.add("Apple");
list.add("Banana");
list.add("Orange");

3. Accessing Elements:
String fruit = list.get(1); // Retrieves the element at index 1 (Banana)

4. Updating Elements:
list.set(0, "Cherry"); // Replaces the element at index 0 with "Cherry"

5. Removing Elements:
list.remove("Banana"); // Removes the first occurrence of "Banana"
// OR
list.remove(2); // Removes the element at index 2

6. Checking if an Element Exists:
boolean containsOrange = list.contains("Orange"); // Returns true if "Orange" is in the list

7. Getting the Size:
int size = list.size(); // Returns the number of elements in the list

8. Checking if the ArrayList is Empty:
boolean isEmpty = list.isEmpty(); // Returns true if the list is empty

9. Iterating through the ArrayList:
```
for (String item : list) {
   System.out.println(item);
}
```

10. Clearing the ArrayList:
list.clear(); // Removes all elements from the list

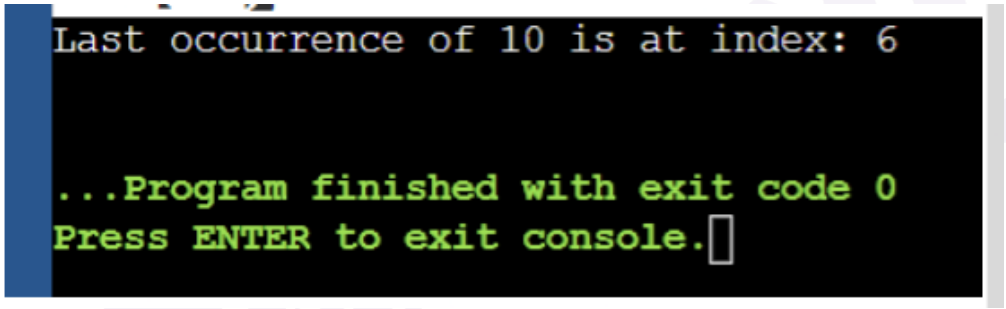
Looping in ArrayList

Ques : Find the last occurrence of x in the array.
Input:
A = [5, 10, 15, 10, 20, 25, 10, 30], 10
Output:
Index = 6

Code:
```
import java.util.ArrayList;
import java.util.List;

public class Main {
    public static void main(String[] args) {
        ArrayList<Integer> arrayList = new ArrayList<>(List.of(5, 10, 15, 10, 20, 25, 10,
30));
```

```
  int x = 10;

        int lastOccurrence = -1;

        for (int i = arrayList.size() - 1; i ⩾ 0; i--) {
            if (arrayList.get(i) == x) {
                lastOccurrence = i;
                break;
            }
        }

        if (lastOccurrence ≠ -1) {
            System.out.println("Last occurrence of " + x + " is at index: " +
lastOccurrence);
        } else {
            System.out.println(x + " not found in the ArrayList.");
        }
    }
}
```

Output:



Ques: Find the doublet in the Array whose sum is equal to the given value x.
Input:
A = [2, 5, 8, 10, 15]
X = 18
Output:
8 10

Code:

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {2, 5, 8, 10, 15};
        int x = 18;

        findDoubletWithSum(arr, x);
    }
```

```
 static void findDoubletWithSum(int[] arr, int targetSum) {
        for (int i = 0; i < arr.length - 1; i++) {
            for (int j = i + 1; j < arr.length; j++) {
                if (arr[i] + arr[j] == targetSum) {
                    System.out.println("Doublet with sum " + targetSum + ": (" + arr[i] + ",
" + arr[j] + ")");
                    return;
                }
            }
        }
        System.out.println("No doublet found with sum " + targetSum);
    }
}
```

Output:

```
Doublet with sum 18: (8, 10)


...Program finished with exit code 0
Press ENTER to exit console.
```

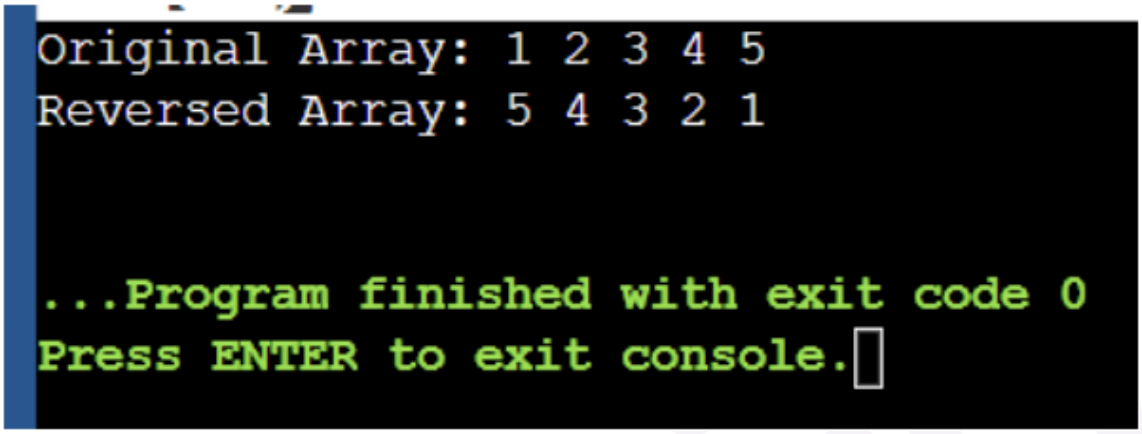
Ques : Write a program to copy the contents of one array into another in the reverse order.
Input:
A = [1,2,3,4,5]
Output:
B = [5,4,3,2,1]

Code:

```
public class Main {
    public static void main(String[] args) {
        int[] originalArray = {1, 2, 3, 4, 5};
        int[] reversedArray = new int[originalArray.length];

        // Copying contents in reverse order
        for (int i = 0; i < originalArray.length; i++) {
            reversedArray[i] = originalArray[originalArray.length - 1 - i];
        }

        System.out.print("Original Array: ");
        displayArray(originalArray);

        System.out.print("Reversed Array: ");
        displayArray(reversedArray);
    }
```

```
static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

Output:



Two Pointers

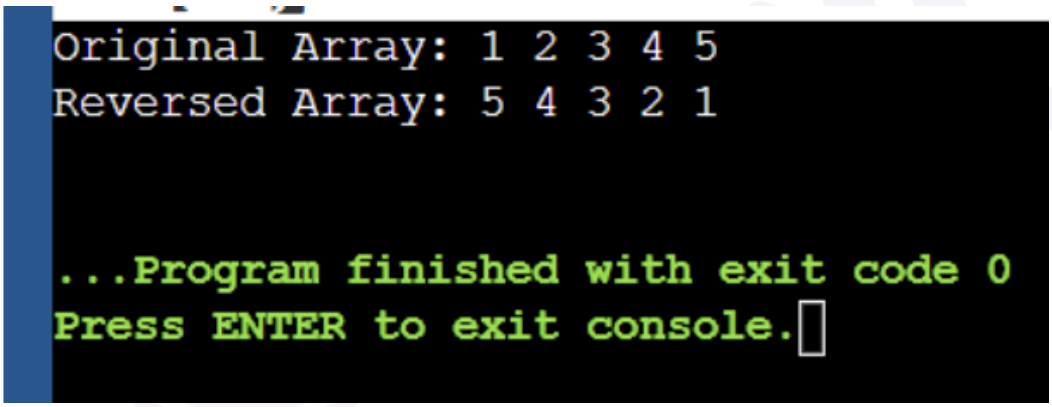
Ques : Write a program to reverse the array without using any extra array.
Input:
A = [1,2,3,4,5]
Output:
B = [5,4,3,2,1]

Code:

```
public class Main {
    public static void main(String[] args) {
        int[] array = {1, 2, 3, 4, 5};

        System.out.print("Original Array: ");
        displayArray(array);

        reverseArray(array);

        System.out.print("Reversed Array: ");
        displayArray(array);
    }

    static void reverseArray(int[] arr) {
        int start = 0;
        int end = arr.length - 1;
```

```
while (start < end) {
            // Swap elements at start and end indices
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;

            // Move pointers towards the center
            start++;
            end--;
        }
    }

    static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

Output:



Ques : Rotate the given array 'a' by k steps, where k is non-negative.
Note : k can be greater than n as well where n is the size of array 'a'.
Input:
A = [1, 2, 3, 4, 5]
K = 3
Output:
Rotated array = [3, 4, 5, 1, 2]

Code:

```
public class Main {
    public static void main(String[] args) {
        int[] array = {1, 2, 3, 4, 5};
        int k = 3;

        System.out.print("Original Array: ");
        displayArray(array);
```

```
System.out.print("Rotated Array: ");
        displayArray(array);
    }

    static void rotateArray(int[] arr, int k) {
        int n = arr.length;
        k = k % n; // Effective rotation

        // Reverse the entire array
        reverseArray(arr, 0, n - 1);

        // Reverse the first k elements
        reverseArray(arr, 0, k - 1);

        // Reverse the remaining n - k elements
        reverseArray(arr, k, n - 1);
    }

    static void reverseArray(int[] arr, int start, int end) {
        while (start < end) {
            // Swap elements at start and end indices
            int temp = arr[start];
            arr[start] = arr[end];
            arr[end] = temp;

            // Move pointers towards the center
            start++;
            end--;
        }
    }

    static void displayArray(int[] arr) {
        for (int num : arr) {
            System.out.print(num + " ");
        }
        System.out.println();
    }
}
```

Output: