Exercises and Homework

java.util Methods for Arrays

fill(A, x)

copyOf(A, n)

copyOfRange(A, s, t):

toString(A)

sort(A):

binarySearch(A, x)

1	R-3.1	Give the next five pseudorandom numbers generated by the process described on page 113, with a = 12, b = 5, and n = 100, and 92 as the seed for cur. See page 113 9 13 61 37 49

2	R-3.2	Write a Java method that repeatedly selects and removes a random entry from an array until the array holds no more entries.
		`public static void removeRandomEntries(int[] array) {`
		`  List<Integer> list = new ArrayList<>();`
		`  for (int num : array) {`
		`  list.add(num);`
		`  }`
		`  Random random = new Random();`
		`  while (!list.isEmpty()) {`
		`  int index = random.nextInt(list.size());`
		`  System.out.println("Removed: " + list.remove(index));`
		`  }`
		`}`

3	R-3.3	Explain the changes that would have to be made to the program of Code Fragment 3.8 so that it could perform the Caesar cipher for messages that are written in an alphabet-based language other than English, such as Greek, Russian, or Hebrew. تحديد الأبجدية المستخدمة بدلاً من الاعتماد على الحروف الإنجليزية فقط (A-Z). تخزين الأبجدية الجديدة كسلسلة نصية أو مصفوفة، مثل: • اليونانية: "ΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩ" • الروسية: "АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ" حساب الإزاحة بناءً على طول الأبجدية. تعديل طريقة استرجاع الحروف بحيث تعتمد على الأبجدية المحددة بدلاً من الحروف الإنجليزية.

4	R-3.4	The TicTacToe class of Code Fragments 3.9 and 3.10 has a flaw, in that it allows a player to place a mark even after the game has already been won by someone. Modify the class so that the putMark method throws an IllegalStateException in that case
		`public class TicTacToe {`
		`  private static final int X = 1, O = -1; // Players`
		`  private static final int EMPTY = 0;  // Empty cell`
		`  private int[][] board = new int[3][3]; // Game board`
		`  private int currentPlayer = X;  // Current player`
		`  private boolean gameWon = false;  // Flag for game status`
		
		`  public void putMark(int row, int col) {`
		`  if (gameWon) {`
		`  throw new IllegalStateException("The game has already been won!");`
		`  }`
		`  if (row < 0 || row >= 3 || col < 0 || col >= 3 || board[row][col] != EMPTY) {`
		`  throw new IllegalArgumentException("Invalid move");`
		`  }`
		`  board[row][col] = currentPlayer;`
		`  if (isWinner(row, col)) {`
		`  gameWon = true;`
		`  }`
		`  currentPlayer = -currentPlayer;`
		`  }`

		private boolean isWinner(int row, int col) {
		return (board[row][0] + board[row][1] + board[row][2] == 3 * currentPlayer
		|| board[0][col] + board[1][col] + board[2][col] == 3 * currentPlayer
		|| row == col && board[0][0] + board[1][1] + board[2][2] == 3 * currentPlayer
		|| row + col == 2 && board[0][2] + board[1][1] + board[2][0] == 3 * currentPlayer);
		}
		
		public String toString() {
		StringBuilder sb = new StringBuilder();
		for (int[] row : board) {
		for (int cell : row) {
		char mark = (cell == X ? 'X' : (cell == O ? 'O' : ' '));
		sb.append(mark).append("|");
		}
		sb.append("\n");
		}
		return sb.toString();
		}
		}

5	R-3.13	What is the difference between a shallow equality test and a deep equality test between two Java arrays, A and B, if they are one-dimensional arrays of type int? What if the arrays are two-dimensional arrays of type int? **(Shallow Equality)**: • تتحقق من أن المصفوفتين تشير إلى نفس الموقع في الذاكرة. • باستخدام ==. **(Deep Equality)**: • تتحقق من أن العناصر داخل المصفوفتين متطابقة. • باستخدامArrays.equals() للمصفوفات أحادية البعد، وArrays.deepEquals() للمصفوفات متعددة الأبعاد.
6	R-3.14	Give three different examples of a single Java statement that assigns variable, backup, to a new array with copies of all int entries of an existing array, original. 1. backup = original.clone();

		2. backup = Arrays.copyOf(original, original.length); 3. System.arraycopy(original, 0, backup, 0, original.length);

7	C-3.17	Let A be an array of size $n \geq 2$ containing integers from 1 to $n-1$ inclusive, one of which is repeated. Describe an algorithm for finding the integer in A that is repeated.
		def find_repeated_element(B):
		distinct_elements = set()
		
		for b in B:
		if b in distinct_elements:
		return b
		else:
		distinct_elements.add(b)
		
		return None

8	C-3.18	Let B be an array of size $n \geq 6$ containing integers from 1 to $n-5$ inclusive, five of which are repeated. Describe an algorithm for finding the five integers in B that are repeated.
		الفكرة الأساسية:
		• لتتبع العناصر التي تمت رؤيتها (Set) استخدام هيكل بيانات مثل مجموعة.
		• كل مرة نواجه عنصرًا مكررًا (أي أنه موجود بالفعل في المجموعة)، نضيفه إلى قائمة العناصر المكررة.
		• نستمر حتى نجد العناصر الخمسة المكررة.
		الكود:
		public static List<Integer> findFiveRepeatedElements(int[] B) {
		Set<Integer> seen = new HashSet<>();
		List<Integer> repeated = new ArrayList<>();
		
		for (int num : B) {
		if (seen.contains(num)) {
		if (!repeated.contains(num)) {
		repeated.add(num);
		}
		if (repeated.size() == 5) {
		break;
		}
		} else {
		seen.add(num);
		}

		}
		
		return repeated;
		}
		
		public static void main(String[] args) {
		int[] B = {1, 2, 3, 4, 5, 6, 1, 2, 3, 4, 5};
		System.out.println("Repeated elements: " +
		findFiveRepeatedElements(B));
		}
		
		□ **من التنفيذ**: O(n)O(n)O(n)لأننا نمر على المصفوفة مرة واحدة فقط، .
		□ **المساحة**: O(n−5)O(n-5)O(n−5) بسبب استخدام مجموعة للعناصر الفريدة.

9	C-3.19	Give Java code for performing add(e) and remove(i) methods for the Scoreboard class, as in Code Fragments 3.3 and 3.4, except this time, don't maintain the game entries in order. Assume that we still need to keep n entries stored in indices 0 to n−1. You should be able to implement the methods without using any loops, so that the number of steps they perform does not depend on n.
		public class Scoreboard {
		private int[] scores;
		private int size;
		
		public Scoreboard(int capacity) {
		scores = new int[capacity];
		size = 0;
		}
		
		public void add(int e) {
		if (size < scores.length) {
		scores[size] = e;
		size++;
		} else {
		throw new IllegalStateException("Scoreboard is full");
		}
		}
		
		public void remove(int i) {
		if (i < 0 || i >= size) {
		throw new IllegalArgumentException("Invalid index");

```
        }
        scores[i] = scores[size - 1]; // Replace with the last element
        scores[size - 1] = 0;         // Clear the last element
        size--;
    }

    public String toString() {
        StringBuilder sb = new StringBuilder("[");
        for (int j = 0; j < size; j++) {
            sb.append(scores[j]);
            if (j < size - 1) sb.append(", ");
        }
        sb.append("]");
        return sb.toString();
    }

    public static void main(String[] args) {
        Scoreboard scoreboard = new Scoreboard(5);
        scoreboard.add(10);
        scoreboard.add(20);
        scoreboard.add(30);
        System.out.println("Before removal: " + scoreboard);
        scoreboard.remove(1);
        System.out.println("After removal: " + scoreboard);
    }
}
```

10	C-3.20	Give examples of values for a and b in the pseudorandom generator given on page 113 of this chapter such that the result is not very random looking, for n = 1000. قيم سيئة: • a=1a = 1a=1 ،b=0b = 0b=0: التكرار سيكون دائمًا نفس القيمة. • aaa وnnn كعدد زوجي: سيؤدي إلى توزيع غير منتظم كعدد زوجي. يجب أن يكونا أوليين نسبيًا مع bbb وaaa لتحسين العشوائية، nnn.

11	C-3.21	Suppose you are given an array, A, containing 100 integers that were generated using the method r.nextInt(10), where r is an object of type java.util.Random. Let x denote the product of the integers in A. There is a single number that x will equal with probability at least 0.99. What is that number and what is a formula describing the probability that x is equal to that number? □ يولد القيم من ٠ إلى ٩، r.nextInt(10)r.nextInt(10)r.nextInt(10) إذا كان x=0x = 0x=0 فإن أي وجود لـ ٠ في المصفوفة يجعل الناتج. □ الاحتمالية: • احتمال أن يكون x=0x = 0x=0: 1−(0.9100)≈0.999991 - (0.9^{100}) \approx 0.999991−(0.9100)≈0.99999. • الاحتمالية لأن تكون القيم الأخرى مختلفة صغيرة جدًا.

12	C-3.22	Write a method, shuffle(A), that rearranges the elements of array A so that every possible ordering is equally likely. You may rely on the nextInt(n) method of the java.util.Random class, which returns a random number between 0 and n−1 inclusive.
		`public static void shuffle(int[] A) {`
		`  Random rnd = new Random();`
		
		`  for (int i = A.length - 1; i > 0; i--) {`
		`  // Swap the current element with a randomly chosen element from the remaining array`
		`  int j = rnd.nextInt(i + 1);`
		`  int temp = A[i];`
		`  A[i] = A[j];`
		`  A[j] = temp;`
		`  }`
		`}`

13	C-3.23	Suppose you are designing a multiplayer game that has n ≥ 1000 players, numbered 1 to n, interacting in an enchanted forest. The winner of this game is the first player who can meet all the other players at least once (ties are allowed). Assuming that there is a method meet(i, j), which is called each time a player i meets a player j (with i 6= j), describe a way to keep track of the pairs of meeting players and who is the winner. **الفكرة الأساسية:** • استخدام مصفوفة ثنائية الأبعاد boolean[][] لتتبع اللقاءات بين اللاعبين. • كلما التقى لاعبان، يتم تحديث القيم في المصفوفة وإضافة اللقاء.

- نحتفظ بعداد لكل لاعب لتسجيل عدد اللقاءات.

الكود:

```
public class MultiplayerGame {
    private boolean[][] meetings;
    private int[] meetingCount;
    private int n;

    public MultiplayerGame(int n) {
        this.n = n;
        meetings = new boolean[n][n];
        meetingCount = new int[n];
    }

    public void meet(int i, int j) {
        if (!meetings[i][j]) {
            meetings[i][j] = meetings[j][i] = true; // تحديث اللقاء
            meetingCount[i]++;
            meetingCount[j]++;
        }
    }

    public int checkWinner() {
        for (int i = 0; i < n; i++) {
            if (meetingCount[i] == n - 1) {
                return i;
            }
        }
        return -1; // لم يتم تحديد فائز بعد
    }

    public static void main(String[] args) {
        MultiplayerGame game = new MultiplayerGame(5);

        game.meet(0, 1);
        game.meet(0, 2);
        game.meet(0, 3);
        game.meet(0, 4);

        int winner = game.checkWinner();
```

		System.out.println("Winner: " + (winner != -1 ? "Player " + winner : "No winner yet")); } }

14	C-3.24	Write a Java method that takes two three-dimensional integer arrays and adds them componentwise.
		public static int[][][] addThreeDimensionalArrays(int[][][] array1, int[][][] array2) {
		if (array1.length != array2.length || array1[0].length != array2[0].length || array1[0][0].length != array2[0][0].length) {
		throw new IllegalArgumentException("Arrays must have the same dimensions");
		}
		
		int[][][] result = new int[array1.length][array1[0].length][array1[0][0].length];
		
		for (int i = 0; i < result.length; i++) {
		for (int j = 0; j < result[0].length; j++) {
		for (int k = 0; k < result[0][0].length; k++) {
		result[i][j][k] = array1[i][j][k] + array2[i][j][k];
		}
		}
		}
		
		return result;
		}