1. Write the code for method with the following header

**public dataSet(int size, int min, int max)**

The method creates the ArrayList of integers that contains ‘size’ randomly chosen values in the range ‘min’ to ‘max’ inclusive. **No duplicates are allowed**. If the difference between ‘min’ and ‘max’ is less than or equal to the value of ‘size’ the method should return the ‘null’ value.

**public ArrayList<Integer> dataSet(int dataSetSize, int min, int max) {**

**// Check if min > max and swap them if necessary**

**if(min > max) {**

**int temp = min;**

**min = max;**

**max = temp;**

**}**

**// Check if size > (max-min). If it is we cannot create a data set so return null**

**if(dataSetSize > (max-min) ) {**

**return null;**

**}**

**// Now populate the arraylist with 'size' values between 'min' and 'max'**

**int diff = max - min + 1;**

**ArrayList<Integer> dataSet = new ArrayList<Integer>() ;**

**while(dataSet.size() < dataSetSize) {**

**int newValue = (int)(Math.random()\*diff + min);**

**if(!dataSet.contains(newValue)){**

**dataSet.add(newValue);**

**}**

**}**

**// Alternatively**

**// ArrayList<Integer> dataSet = new ArrayList<Integer>(dataSetSize) ;**

**// for(int i=0; i< dataSetSize; i++) {**

**// int newValue = (int)(Math.random()\*diff + min);**

**// if(!dataSet.contains(newValue)){**

**// dataSet.add(newValue);**

**// }**

**// }**

**return dataSet;**

**}**

1. Write the code for a method with the following header

**public ArrayList<Integer> lottoQuickPick()**

The method returns an ArrayList that contains 6 randomly chosen integers in the range 1 to 47. The numbers in the ArrayList should be in ascending order.

**public ArrayList<Integer> lottoQuickPick() {**

**// Create ArrayList for chosen numbers (i.e. 6 numbers between 1 and 50)**

**ArrayList<Integer> chosenNumbers = dataSet(6,1,47);**

**Collections.sort(chosenNumbers);**

**System.out.println(chosenNumbers); // Included for testing purposes ONLY**

**return chosenNumbers;**

**}**

1. Write the code for a method with the following header

**public ArrayList<Integer> euroMillionsQuickPick()**

The method returns an ArrayList that contains 5 randomly chosen integers in the range 1 to 50 sorted in ascending sequence FOLLOWED BY 2 randomly chosen integers in the range 1 to 12 sorted in ascending sequence. NOTE: The first 5 values should be in ascending sequence and then the final two numbers should be in ascending sequence.

**public ArrayList<Integer> euroMillionsQuickPick() {**

**ArrayList<Integer> chosenNumbers = new ArrayList<Integer>();**

**// Choose quick pick numbers (i.e. 5 numbers between 1 and 50)**

**ArrayList<Integer> numbers = dataSet(5,1,50);**

**Collections.sort(numbers);**

**// Choose Lucky Star numbers (i.e. 2 numbers between 1 and 12)**

**ArrayList<Integer> luckyStar = dataSet(2,1,12);**

**Collections.sort(luckyStar);**

**// Create list that contains both sets of values - chosen numbers first then lucky star**

**chosenNumbers.addAll(numbers);**

**chosenNumbers.addAll(luckyStar);**

**System.out.println(chosenNumbers); //Included for testing purposes ONLY.**

**return chosenNumbers;**

**}**

1. Write the code for a method with the following header

**public ArrayList<Integer> lowerHalfOf(ArrayList<Integer> dataSet)**

The method is passed an ArrayList of integers. If the list is empty or null or it contains just one value the method should return null. Otherwise the method should return an ArrayList that contains the values in the first half of the parameter list.

**public ArrayList<Integer> lowerHalfOf(ArrayList<Integer> dataSet) {**

**// If the dataSet is empty of has only one value then return it unchanged**

**if(dataSet == null || dataSet.size() < 2) {**

**return dataSet;**

**}**

**int midPoint = dataSet.size() / 2;**

**// Return ArrayList**

**ArrayList<Integer> bottomHalf = new ArrayList<Integer>(dataSet.subList(0,midPoint));**

**System.out.println(bottomHalf); //Included for testing purposes ONLY.**

**return bottomHalf;**

**// Alternatively**

**// return new ArrayList<Integer>(dataSet.subList(0,midPoint));**

**}**

1. Write the code for a method with the following header

**public ArrayList<Integer> upperHalfOf(ArrayList<Integer> dataSet)**

The method is passed an ArrayList of integers. If the list is empty or null or it contains just one value the method should return null. Otherwise the method should return an ArrayList that contains the values in the second half of the parameter list.

**public ArrayList<Integer> upperHalfOf(ArrayList<Integer> dataSet) {**

**// If the dataSet is empty of has only one value then return it unchanged**

**if(dataSet == null || dataSet.size() == 1) {**

**return dataSet;**

**}**

**int midPoint = dataSet.size() / 2;**

**// Return ArrayList**

**ArrayList<Integer> topHalf = new ArrayList<Integer>(dataSet.subList(midPoint,dataSet.size()));**

**System.out.println(topHalf); //Included for testing purposes ONLY.**

**return topHalf;**

**// Alternatively**

**// return new ArrayList<Integer>(dataSet.subList(midPoint,dataSet.size()));**

**}**