```
import java.util.ArrayList;
import java.util.Random;
import java.util.Scanner;

public class ArrayListExercises {

        public static void main(String[] args) {

                // You do not need to handle the User Interface (UI).
                // Instead you can run the JUnit test cases found in
                // ArrayListExercisesTests.java

                Scanner scan = new Scanner(System.in);

                System.out.println("Please enter the number of cards
you want to start with: ");

                int numCards = scan.nextInt();

                bulgarianSolitaire(numCards);

                scan.close();

        }

        /**
         * Removes all of the strings of even length from the given
list
         *
         * @param listOfStrings the list of Strings (list can be
empty)
         * @return the given list with all even length strings removed
         */
        public static ArrayList<String>
removeEvenLength(ArrayList<String> listOfStrings) {

                System.out.println(listOfStrings);

                for (int i = 0; i < listOfStrings.size(); i++) {

                        String length = listOfStrings.get(i);

                        if (length.length() % 2 == 0) {

                                listOfStrings.remove(i);

                                i--;

                        }
                }

                System.out.println(listOfStrings);

                return listOfStrings; // This return statement should
be last
        }

        /**
         * Moves the minimum value in the list to the front, otherwise
preserving the
         * order of the elements
         *
         * @param listOfIntegers the list of Integers (list cannot be
empty)
         * @return the given list with the minimum value in the front
(zeroth element)
         */
        public static ArrayList<Integer>
minimumToFront(ArrayList<Integer> listOfInts) {

                int low = listOfInts.get(0);

                for (int i = 0; i < listOfInts.size(); i++) {

                        int num = listOfInts.get(i);

                        if (num < low) {

                                low = num;

                                i--;

                        }

                }

                int position = listOfInts.indexOf(low);

                listOfInts.remove(position);

                listOfInts.add(0, low);

                return listOfInts; // This return statement should be
last
        }

        /**
         * Removes all elements from the given list whose values are
in the range min
```

```
	 * through max (inclusive). If no elements in range min-max
are found in the
	 * list, the list's contents are unchanged. If an empty list
is passed, the list
	 * remains empty. Assume min < max.
	 *
	 * @param listOfInts the list of Integers (list can be empty)
	 * @param min        the minimum value in the range
	 * @param max        the maximum value in the range
	 * @return the given list with the range min-max removed
	 */
	public static ArrayList<Integer>
filterRange(ArrayList<Integer> listOfInts, int min, int max) {

		if (listOfInts.size() != 0) {

			for (int i = 0; i < listOfInts.size(); i++) {

				int num1 = listOfInts.get(i);

				if (num1 >= min && num1 <= max) {

					listOfInts.remove(i);

					i--;

				}

			}
		}

		return listOfInts; // This return statement should be
last
	}

	/**
	 * Models/simulates the game of Bulgarian Solitaire.
	 *
	 * @param numCards the number of cards to start with; n must
be a triangular
	 *                 number (a triangular number is a number
that can be written
	 *                 as the sum of the first n positive
integers).
	 */
	public static void bulgarianSolitaire(int numCards) {

		// Check if given number of cards is triangular
		int n = (int) Math.sqrt(2 * numCards);
		if (n * (n + 1) / 2 != numCards) {
```

```
                        System.out.println(numCards + " is not
triangular");

                        return;
                }

                int numberofcards = numCards;

                ArrayList<Integer> finallist = new
ArrayList<Integer>();

                int i = 1;

                while (numCards != 0) {

                        finallist.add(i);

                        numCards -= i;

                        i++;

                }

                System.out.println(finallist);

                ArrayList<Integer> remove = new ArrayList<Integer>();

                remove.add(0);

                ArrayList<Integer> solitaire = new
ArrayList<Integer>();

                Random rand = new Random();

                while (numberofcards > 0) {

                        int randompile = rand.nextInt(numberofcards) +
1;
                        solitaire.add(randompile);
                        numberofcards -= randompile;
                }

                // test if numCards is 28
                // solitaire.add(0, 1);
                // solitaire.add(1, 6);
                // solitaire.add(2, 2);
                // solitaire.add(3, 1);
                // solitaire.add(4, 1);
                // solitaire.add(5, 2);
                // solitaire.add(6, 7);
                // solitaire.add(7, 1);
                // solitaire.add(8, 5);
                // solitaire.add(9, 2);

                while (!solitaire.containsAll(finallist)) {

                        for (i = 0; i < solitaire.size(); i++) {

                                solitaire.set(i, solitaire.get(i) -
1);

                        }

                        solitaire.add(i);
                        solitaire.removeAll(remove);

                        System.out.println(solitaire);

                }

        }

}
```