```java
package player;

import java.math.BigInteger;

/**
 * Generates combination indices.
 *
 * (http://www.merriampark.com/comb.htm)
 */
public class Combinations {
    private int[] a;
    private int n;
    private int r;
    private BigInteger numLeft;
    private BigInteger total;

    public Combinations(int n, int r) {
        this.n = n;
        this.r = r;
        a = new int[r];
        BigInteger nFact = getFactorial(n);
        BigInteger rFact = getFactorial(r);
        BigInteger nminusrFact = getFactorial(n - r);
        total = nFact.divide(rFact.multiply(nminusrFact));
        reset();
    }

    public void reset() {
        for (int i = 0; i < a.length; i++) {
            a[i] = i;
        }

        numLeft = new BigInteger(total.toString());
    }

    public BigInteger getNumLeft() {
        return numLeft;
    }

    public boolean hasMore() {
        return numLeft.compareTo(BigInteger.ZERO) == 1;
    }

    public BigInteger getTotal() {
        return total;
    }

    private static BigInteger getFactorial(int n) {
        BigInteger fact = BigInteger.ONE;

        for (int i = n; i > 1; i--) {
            fact = fact.multiply(new BigInteger(Integer.toString(i)));
        }

        return fact;
    }

    public int[] getNext() {
        if (numLeft.equals(total)) {
            numLeft = numLeft.subtract(BigInteger.ONE);
            return a;
        }

        int i = r - 1;

        while (a[i] == n - r + i) {
            i--;
        }

        a[i] = a[i] + 1;

        for (int j = i + 1; j < r; j++) {
            a[j] = a[i] + j - i;
        }

        numLeft = numLeft.subtract(BigInteger.ONE);

        return a;

    }
}
```