Exercise NMCGJ:
Revisiting Text Patterns

The aim of this exercise is to enhance the class `TextPattern` which was introduced in the previous exercise *Drawing Text Patterns*. First, some of the existing methods are revised, and then new functionality is added to the class by providing a text visualization of the famous triangle of Pascal.

New version of TextPattern

The class `TextPattern` can be revised in the following two ways.

1. Using a container class

Instead of holding the instances of `TextPrinter` in a standard array, it is more convenient to use a container class. For example, make use of the class `ArrayList`, and

- update the method `addPrinter`;
- update the methods `print` and `println`.

Try different ways to iterate through the list:

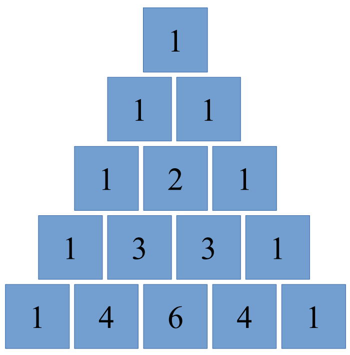
- with an Iterator;
- with a standard for loop;
- with a foreach loop.

2. Using string formatting

The implementation of the method `printPyramid` can be updated by using the string formatting functionality (`String.format`).

Pascal's triangle

Pascal's triangle is a set of numbers arranged in the form of a triangle (actually, it looks like a pyramid according to our previous exercise). Each number in a row is the sum of the left number and right number on the above row. If a number is missing in the above row, it is assumed to be 0. The first row starts with number 1. The following is such a triangle with 5 rows,



These numbers form the binomial coefficients. The triangle is named after the French mathematician Blaise Pascal who organized detailed information on the triangle in a book. However, this triangle was already known in many ancient civilizations.

Implementation

Write a method `printPascal` that generates a (text-based) visualization of Pascal's triangle for a given height. The numbers in the triangle should be properly formatted according to the height of the triangle. Some examples are:

```
Pascal's triangle (height = 5):

    1
   1 1
  1 2 1
 1 3 3 1
1 4 6 4 1

Pascal's triangle (height = 9):

                 1
               1   1
             1   2   1
           1   3   3   1
         1   4   6   4   1
       1   5  10  10   5   1
     1   6  15  20  15   6   1
   1   7  21  35  35  21   7   1
 1   8  28  56  70  56  28   8   1
```

The method should look like

```java
public void printPascal(int height, int nwidth)
```

where `height` stands for the number of rows of the triangle and `nwidth` for the number of characters reserved for each number. The above examples were generated by the calls `printPascal(5, 1)` and `printPascal(9, 2)`.

Notes

Besides the formula described before, another way to generate the numbers in the triangle of Pascal is by means of the explicit expression of the binomial coeffcients, or a variation. The class `Math` does not provide any method to compute a factorial of a number. In case you want to use factorials, you must implement it by yourself. Be careful with such an implementation because factorials grow very fast and it is easy to get an arithmetic overflow.
As a check of your implementation, try to compute `trianglePascal(25, 5)`. The output of the final row should start with the following numbers:

```
     1       24       276      2024     10626     42504    134596
346104   735471   1307504   1961256   2496144   2704156       ...
```