

## Triângulo de Pascal

### Exercício 1 (Ex1)

#### Informações

- Trabalho em duplas;
- Entrega pelo Moodle em <<http://trab.dc.unifil.br/moodle/>>.

Leia atentamente a definição do Triângulo de Pascal<sup>1</sup>:

O triângulo [a seguir], dividido em linhas e colunas, é composto de números binomiais<sup>2</sup>:

$$\begin{array}{c}
 \binom{0}{0} \\
 \binom{0}{0} \binom{0}{0} \\
 \binom{2}{0} \binom{2}{1} \binom{2}{2} \\
 \binom{3}{0} \binom{3}{1} \binom{3}{2} \binom{3}{3} \\
 \binom{4}{0} \binom{4}{1} \binom{4}{2} \binom{4}{3} \binom{4}{4} \\
 \binom{5}{0} \binom{5}{1} \binom{5}{2} \binom{5}{3} \binom{5}{4} \binom{5}{5}
 \end{array}$$

Em cada número binomial  $\binom{n}{k}$ ,  $n$ , o numerador, está relacionado ao número da linha e  $k$ , o denominador, ao número da coluna. Observe que na quinta linha temos 5 números binomiais, todos eles com numerador igual a 4. Veja também que na terceira coluna todos os números binomiais possuem 2 como denominador.

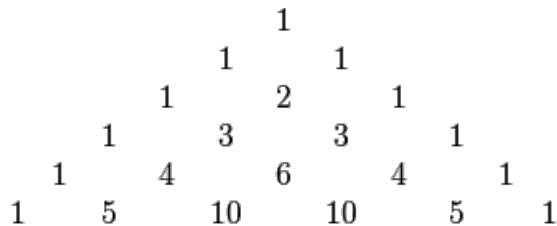
Resumindo, o numerador de todos os números binomiais de uma determinada linha é o mesmo, assim como o denominador de todos os números binomiais de uma certa coluna é igual ao número da coluna. Linhas e colunas começam em 0.

As linhas de um Triângulo de Pascal possuem uma quantidade finita de elementos, que é igual ao número da linha mais 1. Por exemplo, a quinta linha, que é a de número 4, possui 5 elementos. Já a quantidade de elementos por coluna é infinita, pois o número de linhas do Triângulo de Pascal também é infinito. Portanto, na figura acima temos apenas um fragmento do Triângulo de Pascal.

Agora vejamos este mesmo fragmento do triângulo já com os números binomiais substituídos por seus respectivos valores:

<sup>1</sup>Retirado do site <<http://www.matematicadidatica.com.br/TrianguloDePascal.aspx>>.

<sup>2</sup>Definição disponível em: <<http://www.matematicadidatica.com.br/NumeroBinomial.aspx>>.



Repare que todas as linhas começam e terminam com o número 1. Isto já era de se esperar, pois como vimos no estudo dos números binomiais,  $\binom{n}{0} = 1$  e  $\binom{n}{n} = 1$ .

A explicação sobre como definir um número qualquer de qualquer posição do triângulo está na sequência do texto:

Para construirmos um triângulo como este podemos calcular os números binomiais, um a um. Vejamos como exemplo os números [a seguir]:

$$\binom{4}{1} = \frac{4!}{1!(4-1)!} \Rightarrow \binom{4}{1} = 4$$

$$\binom{4}{2} = \frac{4!}{2!(4-2)!} \Rightarrow \binom{4}{2} = 6$$

$$\binom{5}{2} = \frac{5!}{2!(5-2)!} \Rightarrow \binom{5}{2} = 10$$

Embora simples, o processo é bastante trabalhoso, principalmente à medida que o número da linha vai crescendo, mas felizmente há uma alternativa mais simples para realizarmos a montagem de um triângulo destes.

Vamos analisar os mesmos três números acima. Repare que o número 10 pode ser obtido se somarmos o número que está na linha imediatamente acima (6) com o vizinho da esquerda deste número (4). Veja que este raciocínio serve para todos os números do triângulo, com exceção do primeiro e do último número de cada linha.

Com base nessas informações, faça o que se pede:

1. Programe um método que calcule o valor da função fatorial  $n!$ , com a seguinte assinatura:

```
public static int fatorial(int n)
```

*Atenção! Não utilize recursividade para esta implementação!*

2. Implemente um método que calcula o valor de um binômio qualquer com base na equação de Newton, conforme descrito no início desse roteiro. A assinatura do método deverá ser

```
public static int resolverBinomioNewton(int n, int k)
```

Para calcular os valores dos fatoriais, utilize o método programado na questão 1.

3. Agora, programe um método que receba como parâmetro um valor  $n$  e escreva no console um Triângulo de Pascal com  $n$  linhas, utilizando o método programado na questão 2. Experimente executar o método para vários valores de  $n$  e responda a seguinte questão:

- (a) Até qual linha o método de escrita do triângulo de pascal consegue calcular os números sem apresentar erros de execução ou erros numéricos?
- 4. Programe o método `resolverBinomioPascal`, que possui mesma assinatura do método `resolverBinomioNewton` porém resolve o valor do binômio utilizando o método da soma dos dois valores adjacentes da linha anterior. Adapte o método da questão 3 para utilizar este método e responda:
  - (a) Agora, até qual linha o método `resolverBinomioPascal` consegue calcular os números sem apresentar erros de execução?
  - (b) Porque este método funciona melhor que o método anterior?
- 5. Programe o método `resolverBinomioPascalRecursivo`, que implementa o mesmo algoritmo de `resolverBinomioPascal` porém recursivamente, sem utilizar nenhuma estrutura explícita de laço.

Não se esqueça de documentar os métodos com *javadocs*, e colocar as respostas e os nomes da dupla na folha de projeto.