

[ED185] TAD Números Arbitrariamente Grandes (BigInteger)

Neste problema deverá apenas submeter uma classe **BigInteger** (e não um programa completo).

O problema

A sua tarefa é criar uma classe **BigInteger**, que implementa um número arbitrariamente grande (com tantos dígitos quanto o necessário). A classe deverá suportar quaisquer números inteiros positivos, e deverá implementar os seguintes métodos:

- **Construtores:**
 - **BigInteger(String n)** - Inicializa o número a partir da String *n* (supondo que a String só contém dígitos e não tem zeros à esquerda)
- **Métodos padrão:**
 - **public boolean equals(BigInteger n)** - devolve *true* caso o número seja igual a n, ou *false* caso contrário
 - **public String toString()** - devolve uma String representando o número (os dígitos em si)
- **Operações:**
 - **public BigInteger add(BigInteger n)** - devolve um novo BigInteger igual à soma de *n* com o próprio objecto (*this* + *n*).
 - **public BigInteger multiply(BigInteger n)** - devolve um novo BigInteger igual à multiplicação de *n* com o próprio objecto (*this* * *n*).



Um exemplo de utilização seria:

```
class TestBigInteger {
    public static void main(String[] args) {
        BigInteger n1 = new BigInteger("1234567890");
        System.out.println(n1); // Escreve "1234567890"

        BigInteger n2 = new BigInteger("42");
        BigInteger n3 = new BigInteger("1234567890");
        System.out.println(n1.equals(n2)); // Escreve "false"
        System.out.println(n1.equals(n3)); // Escreve "true"

        BigInteger n4 = new BigInteger("46711237126582920746212");
        BigInteger n5 = new BigInteger("8765432110");
        BigInteger n6 = n1.add(n3);
        System.out.println(n6); // Escreve "2469135780"
        BigInteger n7 = n1.add(n4);
        System.out.println(n7); // Escreve "46711237126584155314102"
        BigInteger n8 = n1.add(n5);
        System.out.println(n8); // Escreve "10000000000"

        BigInteger n9 = n1.multiply(n3);
        System.out.println(n9); // Escreve "1524157875019052100"
        BigInteger n10 = n1.multiply(n4);
        System.out.println(n10); // Escreve "57668193458655139375688174332680"
    }
}
```

Input e Output

Deverá apenas submeter a classe **BigInteger**. O Mooshak irá chamar criar várias instâncias da sua classe usando o construtores definido e irá fazer uma série de testes aos métodos por si implementados (como mostrado no exemplo de utilização).

É garantido que o números guardados serão sempre positivos e que nunca terão mais do que 1000 dígitos.