



---

# Desafios

---

# Orientação a Objetos

---

## Fundamentos de Java





# Sistema de Biblioteca

A Biblioteca da Universidade está procurando sistematizar sua operação de empréstimos e devoluções. Para isso ela chamou você como programador de uma aplicação para empréstimos e devoluções de livros.

Cada livro tem com características seu título, autor, ano de publicação, categoria, posição na biblioteca (numero da estante, numero da prateleira) e uma informação se está emprestado ou não.

Seu sistema de bibliotecas deverá contar já com alguns livros (5 no caso) nos quais o usuário poderá solicitar o empréstimo ou verificar se ele já foi emprestado ou não.



# Livro Eletrônico (Ebook)

Parecido com o exercício anterior? Claro que não! Imagine um Livro Eletrônico cujos atributos e métodos estão descritos no diagrama abaixo.

<b>Ebook</b>
Titulo: String Autor: String totalPaginas: int paginaAtual: int zoom: int
avancarPagina() retrocederPagina() irParaPagina() exibirPagina() mostrarCapa()

Considere, nos métodos descritos, funcionalidades que exibam o status do livro (em qual página está, para qual página foi, etc).

Implemente esta classe, bem como um aplicativo JAVA que instancie um ou mais livros e faça a sua navegação através dos métodos.



# Número Fracionário

Um número fracionário é definido como uma estrutura que armazena 2 valores inteiros (numerador e denominador). Dessa forma, um número fracionário pode, inclusive realizar operações de soma, subtração, multiplicação e divisão, dadas abaixo.

$$\frac{a}{b} + \frac{c}{d} = \frac{a*d + b*c}{b*d}$$

$$\frac{a}{b} - \frac{c}{d} = \frac{a*d - b*c}{b*d}$$

$$\frac{a}{b} * \frac{c}{d} = \frac{a*c}{b*d}$$

$$\frac{a}{b} / \frac{c}{d} = \frac{a*d}{b*c}$$

Neste sentido, um número fracionário  $a/b$  pode se somar ao número fracionário  $c/d$  seguindo-se a fórmula descrita, onde "a" representa o numerador do primeiro elemento e "b" seu denominador.

Uma operação em Java é sempre denotada por métodos. Portanto se quisermos, por exemplo, somar 2 números fracionários, é necessário que haja um método que calcule isso. De que forma?

Supondo que N1 e N2 sejam objetos da classe NumeroFracionario, obteremos um número N3 através do seguinte comando

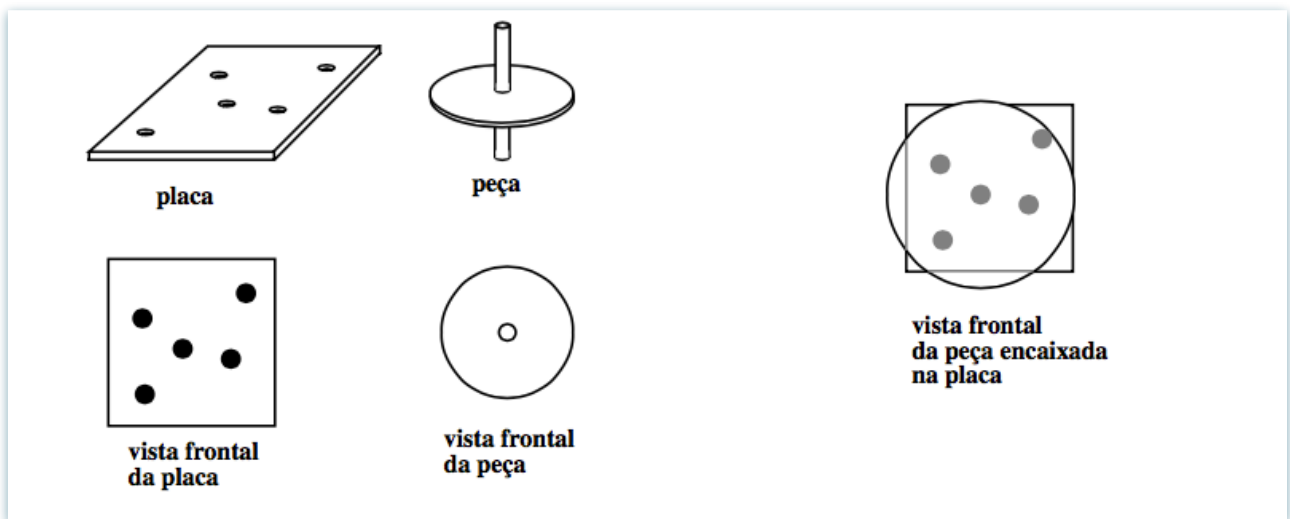
`N3 = N1.soma(N2);`



# Cubra os Furos

Uma placa de aço retangular contém  $N$  furos circulares de 5 mm de diâmetro, localizados em pontos distintos, não sobrepostos – ou seja, o centro de cada furo está a uma distância maior ou igual a 5 mm do centro de todos os outros furos.

Uma peça de forma circular, tendo em seu centro um eixo de 5 mm de diâmetro, deve ser colocada sobre a placa, de modo que o eixo encaixe-se em um de seus furos.



## 1. Tarefa

Você deve escrever um programa para determinar o diâmetro mínimo que a peça deve ter de tal forma que, com seu eixo encaixado em um dos furos da placa, a parte circular cubra completamente todos os outros furos da placa.



## 2. Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um inteiro  $N$ , que indica o número de furos na placa de aço ( $1 \leq N \leq 1000$ ). As  $N$  linhas seguintes contêm cada uma dois inteiros  $X$  e  $Y$ , separados por um espaço em branco, que descrevem a posição do centro de um furo. A unidade de medida das coordenadas dos furos é 1 mm. O final da entrada é indicado por  $N = 0$ .

### Exemplo de Entrada

```
3
20 25
10 5
10 10
3
05 10
0 0
10 0
```

## 3. Saída

Para cada conjunto de teste da entrada seu programa deve produzir três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato "Teste # $n$ ", onde  $n$  é numerado seqüencialmente a partir de 1. A segunda linha deve conter o diâmetro mínimo que a peça deve ter, como um número inteiro. A terceira linha em deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

### Exemplo de Saída

```
Teste #1
41

Teste #2
27
```

(esta saída corresponde ao exemplo de entrada acima)



# Numerão

Dependendo das aplicações científicas, um número com 1000 casas é insuficiente (por exemplo para armazenar a massa de um planeta ou a energia dele em órbita).

Para isso, os cientistas do ITO (Institute of Tecnology of Oz-asco) decidiram contratar você para implementar uma classe Java chamada NumeroGrande, capaz de ser criado com uma precisão definida pelo usuário. Exemplo:

```
NumeroGrande ng = new NumeroGrande(1000);
```

Significa que o usuário cria um número inteiro grande com 1000 casas.

Para que seja possível implementar esse número, ele deve receber uma String como valor (e obviamente essa String deve ser decodificada para que o número possa ser manipulado. Exemplo:

```
ng.setValorString("1827232318390128309128301938102938129038");
```

Além disso, na operação de soma (a única que ele implementa), um número grande recebe como argumento outro número grande para ser calculado, como no exemplo abaixo.

```
NumeroGrande n3 = n1.soma(n2); // n3 = n1 + n2;
```

A implementação da soma deve ser feita da seguinte maneira (vamos somar 350 com 971):

	8	7	6	5	4	3	2	1	0
						+1	+1		
N1							3	5	0
N2							9	7	1
N3						1	3	2	1

Vai-um      Vai-um



# Máquina de Refrigerante



Nos tempos de forte calor, claro que é sempre uma boa opção beber um delicioso refrigerante. Melhor ainda se você puder vendê-los. E ainda mais legal se o software que será instalado em todas as máquinas de vendas de refrigerante da Empresa ACME Refrigerante Machines for desenvolvido em Java e por você.

Toda máquina de refrigerante possui sempre uma lista (de tamanho fixo) de refrigerantes. Neste caso um refrigerante é caracterizado por seu nome, seu preço e a quantidade em estoque disponível na máquina.

O processo de compra é bem simples. Uma vez o usuário inserindo créditos (na forma de moedas ou mesmo cédulas), ele seleciona o número do refrigerante. Havendo crédito suficiente e quantidade em estoque, o usuário pode efetuar quantas compras forem necessárias.

Caso o usuário já tenha concluído suas compras e ainda houver crédito na máquina, ele solicita seu troco e recebe o valor restante também mostrado no display da máquina.





# Posto de Combustível

A IsidroCorp está expandindo suas atuações. Agora o prof. Isidro lançou a IsidroCorp Gas & Energy, uma rede de postos de combustíveis. A parte legal é que todas as bombas de Combustível serão implementadas em Java. Uma bomba de combustível é algo parecido com a figura abaixo:



Nela podemos ver algumas informações, como: nome do combustível, valor do litro, quantidade de litros abastecidas e valor total. Muito bem, diante disso, toda vez que uma pessoa chegar no posto e o frentista disser "Que vai ser patrão/madame?". As prováveis respostas seriam: "abastece com X litros" ou "abastece Y reais aí", correto?

Pois bem, modele e implemente uma bomba de combustível e um aplicação "Posto" que irá conter várias bombas de combustível.