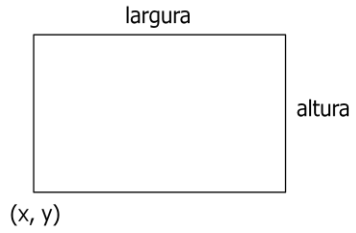
3ª lista de exercícios
Criação de Classes e Métodos Básicos

Crie ou altere as classes conforme definido nos itens abaixo e crie um programa para testar essas classes:

1. Crie a classe `AlunoAcademia` que representa o aluno de uma academia. Para esse aluno devem ser registrados, obrigatoriamente, o `peso` e a `altura`. Nessa classe, implemente os métodos:
   - `flot imc()` que calcula o IMC baseado na fórmula peso / altura^2
   - `float pesoMinimo()` que calcula o peso mínimo baseado na fórmula 18.5 * altura^2
   - `float pesoMaximo()` que calcula o peso máximo baseado na fórmula 24.9 * altura^2
   - `float pesoMedio()` que calcula o peso médio baseado na fórmula (peso mínimo + peso máximo) / 2

2. Implemente a classe `Retangulo` com os atributos x, y, largura e altura. Crie um construtor para receber esses valores e o método:
   - `float perimetro()`



3. Crie a classe `Data` para representar uma data. Para criar uma data é obrigatório informar dia, mês e ano. Crie, também, três métodos:

- `ehValida()` que deverá retornar true se a data for válida ou false caso contrário.
- `ehBissexto()` que deverá retornar true se a data for válida e o ano for bissexto ou false caso contrário.
- `imprimir() e imprimir(char sep)` que deverá imprimir a data com o separador default "/" ou com um separador definido pelo usuário. Caso a data seja inválida, o método deverá imprimir "INVÁLIDA".

  Dica: ano bissexto é aquele que é múltiplo de 4 e não é múltiplo de 100 OU aquele que é múltiplo de 400.

4. Altere a classe `Data` e implemente os métodos:

   - `void avanca()` que avança a data para o dia seguinte.
   - `void retrocede()` que retrocede a data para o dia anterior.

5. Na classe Sequencia, vista em sala de aula, implemente os métodos:

   - `int getQuantidade()` que retorna a quantidade de números existentes na sequência
   - `int somatorio()` que retorna a soma dos elementos da sequência
   - `float media()` que retorna a média aritmética da sequência

6. Na classe `Retangulo` implemente o método:

   - `boolean pertence(int x, int y)`

   que retorna *true* se o ponto (x,y) está dentro do retângulo, ou *false* se está fora.

7. Na classe `Retangulo` implemente o método:

   - `boolean intersecao(Retangulo r)`

   que retorna *true* se o retângulo em questão possui alguma interseção com o retangulo r, ou *false* caso contrário. Dica: os dois retângulos terão uma área de interseção se algum dos vértices de r estiver dentro do retângulo.

8. Crie a classe `Hora` que deverá ter, obrigatoriamente, os atributos `hora`, `minuto` e `segundo`. Em seguida implemente o método:

   - `Hora diferenca(Hora outraHora)`

   que deverá retornar um novo objeto da classe `Hora` que é a diferença entre a hora e `outraHora`.

9. Crie a classe `Fibonacci` com os métodos estáticos:

   - `int numero(int n)` que retorna o número da série de Fibonacci na enésima posição. Se `n` for inválido retorne -1.
   - `bool pertence(int num)` que retorna *true* se o número `num` pertence à série de Fibinacci, ou *false* caso contrário.
   - `void imprime(int n)` que imprime os n primeiros números da série de Fibonacci.

   Lembre-se que a série de Fibonacci começa com 0 e 1 e o próximo elemento é calculado pela soma dos dois últimos.