Trabalho de Programação Web, a ser entregue até o dia 02/09 pelo e-mail [ivan@spei.br](mailto:ivan@spei.br).

Devem ser enviados os fontes dos exercícios apenas.

1. Crie um conjunto (Set) de números do tipo double e insira 20 números randômicos até 100. Após, calcule a média aritmética dos números inseridos.
2. Crie uma classe *EquacaoSegundoGrau* que contenha somente um método estático que calcula as raízes da equação. Considere que os valores de a, b e c serão passados como parâmetro neste método.
3. Escreva em Java a classe *NumeroComplexo* que represente um número complexo. A classe deverá ter os seguintes métodos:
   1. *inicializaNúmero*, que recebe dois valores como argumentos para inicializar os campos da classe (parte real e imaginária);
   2. *imprimeNúmero*, que deve imprimir o número complexo encapsulado usando a notação a + bi onde a é a parte real e b a imaginária;
   3. *eIgual*, que recebe outra instância da classe *NumeroComplexo* e retorna true se os valores dos campos encapsulados forem iguais aos da instância passada como argumento;
   4. *soma*, que recebe outra instância da classe *NumeroComplexo* e soma este número complexo com o encapsulado usando a fórmula ***(a+bi) + (c+di) = (a+c) + (b+d)i***;
   5. *subtrai*, que recebe outra instância da classe *NumeroComplexo* e subtrai o argumento do número complexo encapsulado usando a fórmula ***(a+bi)−(c+di) = (a−c) + (b−d)i***;
   6. *multiplica*, que recebe outra instância da classe *NumeroComplexo* e multiplica este número complexo com o encapsulado usando a fórmula ***(a+bi) ∗ (c+di) = (ac−bd) + (ad +bc)i***;
   7. *divide*, que recebe outra instância da classe *NumeroComplexo* e divide o número encapsulado pelo passado como argumento usando a fórmula ***(a+bi)/(c+di) = (ac+bd)/(c2+d)2+(bc−ad)/(c2+d)2i***;
4. Crie uma classe chamada *Ponto*, que contenha um método que calcule a distância para outra instância da classe *Ponto* e retorna um valor do tipo double. Dica: A distância euclidiana entre um ponto com coordenadas (x1,y1) e outro ponto com coordenadas (x2,y2) é calculada por
5. Escreva uma classe *ConversaoDeTemperatura* que contenha métodos estáticos para calcular a conversão entre diferentes escalas de temperatura. Considere as fórmulas de conversão abaixo:
   1. De graus Celsius (C) para graus Fahrenheit (F): F = (9×C/5) +32)
   2. De graus Fahrenheit (F) para graus Celsius (C): C = (F −32)×5/9
   3. De graus Celsius (C) para graus Kelvin (K): K = C +273.15
   4. De graus Kelvin (K) para graus Celsius (C): C = K −273.15
   5. De graus Celsius (C) para graus Réaumur (Re): Re = C∗ 4/5
   6. De graus Réaumur (Re) para graus Celsius (C): C = Re ∗ 5/4
   7. De graus Kelvin (K) para graus Rankine (R): R = K ∗ 1.8
   8. De graus Rankine (R) para graus Kelvin (K): K = R/1.8

Veja que já que existem cinco sistemas de medidas de temperatura, devem haver 20 diferentes métodos de conversão de temperatura. Alguns podem ser escritos indiretamente, por exemplo, para converter de Celsius para Rankine, podemos converter de Celsius para Kelvin e converter esse resultado para Rankine.

1. Escreva um programa em Java que, usando a classe *ConversaoDeTemperatura* (exercício acima), mostre quantos graus Kelvin e Fahrenheit correspondem a zero e cem graus Celsius.
2. Crie um programa Java que faça as operações aritméticas básica, baseado na entrada de uma equação básica pelo usuário. Por exemplo: se o usuário entrar com a equação 1+2, o programa deve retorna a soma dos valores 1 e 2.

Inclua no programa uma verificação para divisão por 0(zero).

1. Escreva um programa em Java que calcule a série com *N* termos, sendo que o valor de *N* pode ser entrado via teclado. Note que *N* só pode ser um número ímpar, ou seja, caso seja entrado um número par o programa deve retornar um erro.
2. Crie um programa em Java que calcule a soma de duas matrizes 3x3.
3. Crie uma classe que represente um jogo da velha, usando uma matriz de duas dimensões para representar as posições do jogo. A matriz deve ser alocada no construtor da classe, ter o tamanho 3 × 3 e ser de um tipo que suporte três estados possíveis: vazio, preenchido com ’O’ e preenchido com ’X’. A classe deve poder ser usada para jogos com dois jogadores. Dica: A classe deve ter os seguintes métodos:
   1. *jogaO*, que aceita dois valores que são as coordenadas onde um ’O’ será jogado, e marca na matriz a posição somente se esta estiver livre.
   2. *jogaX*, que aceita dois valores que são as coordenadas onde um ’X’ será jogado, e marca na matriz a posição somente se esta estiver livre.
   3. *verifica*, que verifica a matriz para ver se existe algum ganhador (esse método deve verificar se existem três marcas iguais que não sejam vazias em uma horizontal, vertical ou diagonal da matriz).
   4. *toString*, que retornará uma string com a representação gráfica do jogo com as posições atuais.

Escreva também um programa que use a classe. Este programa deve executar um laço no qual fica perguntando as posições para os jogadores alternadamente, enquanto não houver vitória, desistência ou acabarem as posições vazias da matriz.