**ESTATICATOR   
TABAJARA**

**por Pedro Lucas Nascimento Christensen**

**Aplicação java para cálculos simples de estatística  
a partir da entrada de dados indiscriminados.**

**SPECIALISTERNE  
Outubro de 2017**

**Sumário**

**Introdução3**

**Execução3**

**Modo de Uso3**

**Implementação3**

Maior3

Menor3

Moda3

Mediana3

Média3

Variância3

Desvio Padrão3

**Screenshots4**

**Código Fonte5**

**1. Introdução**

Estaticator Tabajara é uma aplicação de cálculo de média, mediana, máximo, mínimo, moda, variância e desvio padrão, programado em Java SE, para uso no console.

**2. Execução**

Por ser um programa feito em Java, é necessário ter instalado a máquina virtual *java* no computador, assim como as variáveis de ambiente do sistema devidamente configuradas.

Para rodar no Windows, abra o prompt de comando e navegue até a pasta onde se encontra o EstaticatorTabajara.class usando do comando “cd” (que significa Change Directory) seguido do caminho onde se encontra o arquivo.

Em seguida, rodar “java EstaticatorTabajara” (sem a extensão, importante).

**3. Modo de Uso**

Inserir os valores confirmando cada entrada com *Enter* e ao terminar, apertar *Enter* mais uma vez sem inserir nenhum valor.

A entrada de dados será encerrada, os resultados calculados e mostrados, terminando a execução do programa.

**4. Implementação**

**4.1** Maior  
Primeiro eu atribui o valor mínimo possível pra uma variável do tipo float, e iterei a lista inteira, sempre atribuindo o item atual para essa variável quando encontrando algum número maior do que o valor na variável atualmente

**4.2** Menor  
Atribui o maior valor encontrado anteriormente, em uma variável. Fiz o mesmo processo descrito no item acima, porém verificando se o item atual era MENOR que o valor na variável atualmente. Foi necessário uma segunda iteração pela lista.

**4.3** Moda  
O mais complicado de calcular. Fiz uma segunda lista contendo cada número da lista de trabalho de forma individual, junto com a quantidade de vezes que cada item aparece. Fiz uma iteração por essa segunda lista pra encontrar o maior número que mais aparece, determinando assim a moda.

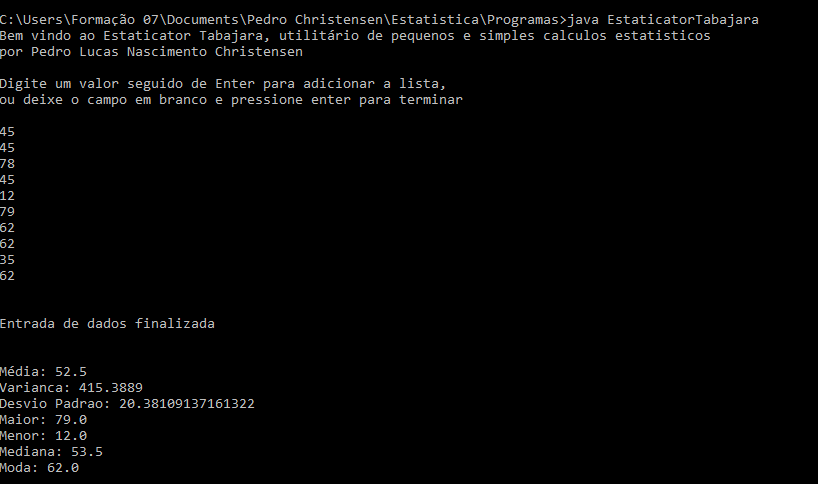
**4.4** Mediana  
Primeiro verifiquei se o número de itens da lista é par ou impar. Caso fosse impar, atribui o valor do meio de forma absoluta para a variável Mediana. Caso fosse impar calculei a média dos dois itens do meio.

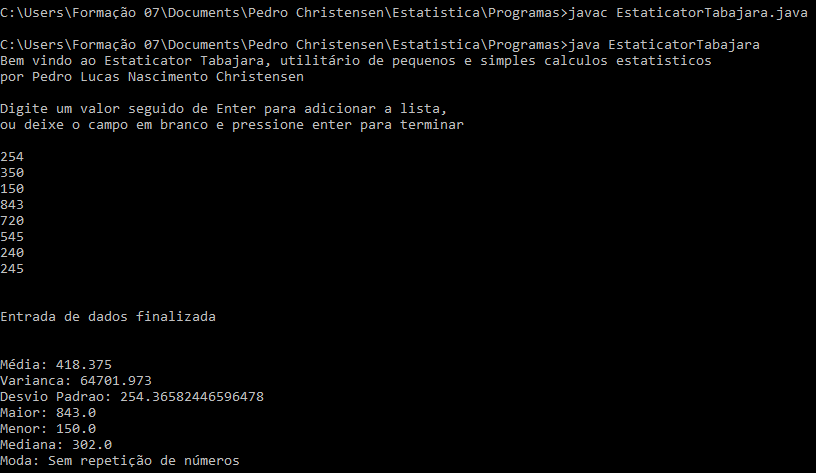
**4.5** Média  
Iterei a lista uma vez somando tudo e dividi essa soma pela quantidade de itens da lista

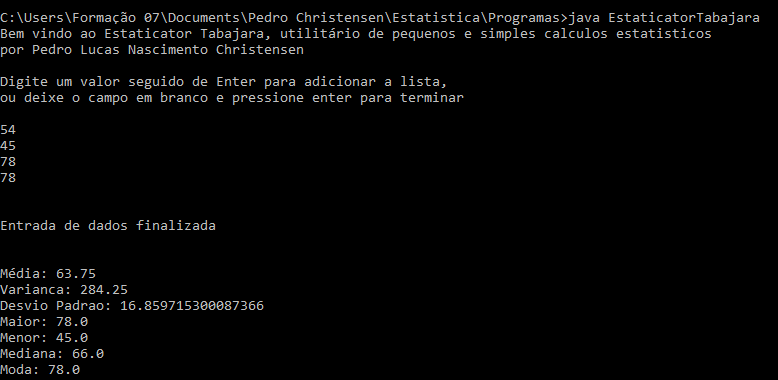
**4.6** Variância  
Iterei a lista mais uma vez e no processo somei o quadrado de cada item da lista subtraído pela média. No final dividi pela quantidade de itens da lista menos um.

**4.7** Desvio Padrão  
Calculei a raiz quadrada da variância.

**5. Screenshots**



****

****

**6. Código Fonte**

O código fonte original pode ser encontrado em *EstaticatorTabajara.java* presente no mesmo diretório desse documento. Abaixo você pode acompanhar a lógica, no código junto aos seus comentários.

1. **import** java.util.\*;
3. **public** **class** EstaticatorTabajara {
5. **public** **static** **void** main (String[] args) {
6. //REFERENTE A INICIALIZACAO E APRESENTACAO
7. Scanner entrada = **new** Scanner(System.in); //Inicializa o objeto scanner, que será utilizado para inserir textos
9. System.out.println("Bem vindo ao Estaticator Tabajara, utilitário de pequenos e simples calculos estatisticos\npor Pedro Lucas Nascimento Christensen\n\nDigite um valor seguido de Enter para adicionar a lista,\nou deixe o campo em branco e pressione enter para terminar\n");
11. List<Float> amostra = **new** ArrayList<Float>(); //Inicializa a lista com os números com que trabalharemos
13. String texto = **null**; //String de entrada
15. //REFERENTE A ENTRADA DE DADOS
16. **while**(!(texto = entrada.nextLine()).equals("")) { //Loop de entrada de texto que será  
    executado até o usuario apertar enter sem digitar nada, o que caracteriza uma string vazia
17. amostra.add(Float.valueOf(texto)); //Converte o texto que foi entrado para Float  
    e insere na lista
18. }
20. System.out.println("\nEntrada de dados finalizada\n\n"); //Avisa o término do loop e  
    consequentemente da entrada de dados
22. //Inicializa algumas variaveis
23. **float** media = 0; //Para armazenar a média
24. **float** somaTudo = 0; //Soma de toda a lista, para calculo da média
25. **float** maior = Float.MIN\_VALUE \* -1; //Maior item da lista. Atribuido o menor valor possivel para variavel do tipo float, para  
    permitir achar o maior valor entre valores muito negativos

28. //REFERENTE AO CALCULO DE MEDIA
29. **for**(Float numero : amostra) { //Primeiro loop da lista
30. somaTudo += numero; //Vai somando, para permitir o calculo da média
31. **if**(numero > maior) { maior = numero; } //Vai substituindo o valor da variavel maior,  
    pelo item atual, caso for maior que o ultimo valor
32. }
34. media = somaTudo / amostra.size(); //Calcula a média de todos os itens da lista
36. //REFERENTE AO CALCULO DE VARIANCIA
37. somaTudo = 0; //Zera a variavel para uso posterior no calculo da variancia
38. **float** variancia = 0; //Declara a variavel pra armazenar a Variancia
39. **float** menor = maior; //Menor item da lista, com o maior associado para funcionar em todos os tipos de entradas
40. **double** desvioPadrao = 0; //Declara a variavel para armazenar o Desvio Padrao
42. System.out.println("Média: " + media); //Exibe o resultado da média
44. **for**(Float numero : amostra) { //Segundo loop pela lista
45. somaTudo += (numero - media) \* (numero - media); //Reusa o somaTudo (que fora  
    utilizado pra calcular a média) para somar o quadrado de cada numero da lista menos a media
46. **if**(numero < menor) { menor = numero; } //Aproveita e já tira o menor número da lista
47. }
49. Collections.sort(amostra); //Organiza a lista em ordem crescente para calculo da mediana e para facilitar a compreensão do calculo da moda, ao fazer o teste de mesa
51. //REFERENTE A MODA
52. List<Float[]> numerosUnicos = **new** ArrayList<Float[]>(); //Cria uma lista de vetores que   
    vai conter os numeros da lista de trabalho, sem repetição. Os vetores dentro da lista terão duas  
    casas. Uma para o número em sí, e outro pra pontuação (quantidade de vezes em que já apareceu)
53. Float[] score = **null**; //Esse é o vetor que vai ser armazenado dentro de cada entrada da   
    lista de números unicos.
55. **for**(Float numero : amostra) { //Terceiro loop pela lista, agora em ordem crescente
56. **boolean** jaApareceu = **false**; //Variavel para poder verificar se o atual número já   
    apareceu na lista de números unicos
57. **for**(Float[] numeroUnico : numerosUnicos) { //Primeiro loop na lista de numeros unicos
58. **if**((**float**) numeroUnico[0] == (**float**) numero) { jaApareceu = **true**; **break**; }   
    //Efetivamente verifica se já apareceu ou não na lista, se sim, marca a flag jaApareceu como   
    positivo e termina o loop
59. }
60. **if**(!jaApareceu) { //Se nunca apareceu
61. score = **new** Float[2]; //Instancia um novo vetor
62. score[0] = numero; //Armazena qual o número em questão
63. score[1] = 1f; //Na posição de quantidade de aparições, armazena 1, por ser a   
    primeira aparição
64. numerosUnicos.add(score); //Adiciona na lista de numeros unicos
65. } **else** { //Se já apareceu
66. **for**(Float[] numeroUnico : numerosUnicos) { //Segundo loop na lista de numeros   
    unicos
67. **if**((**float**) numeroUnico[0] == (**float**) numero) { //Serve para caçar o numero com qual estamos trabalhando. Caso for positivo..
68. numeroUnico[1]++; //Soma mais um na posição de quantidade de aparições
69. **break**; //E sai do loop
70. }
71. }
72. }
73. }
75. **float** maiorContagem = 1; //Se tiver uma entrada na lista, a maior contagem vai ser ao   
    menos 1
76. **float** moda = menor; //Isso aqui é para assegurar que se tiver duas modas quanto a  
    pontuação, a de maior número será a exibida
78. **for**(Float[] numeroUnico : numerosUnicos) { //Loop final pela lista de números unicos
79. **if**(numeroUnico[1] >= maiorContagem && numeroUnico[0] >= moda) { maiorContagem =   
    numeroUnico[1]; moda = numeroUnico[0]; } //Caso o numero de aparicoes seja o maior até então, E, o  
    numero seja o maior também até então, atribuir os valores correspondentes
80. }
82. //REFERENTE A MEDIANA
83. **float** mediana = 0; //Declara a variavel para armazenar a mediana
85. **if**(amostra.size() % 2 == 0) { //Se o número for par, executa o seguinte
86. mediana = (amostra.get((**int**) ((amostra.size() / 2) - 0.5)) + amostra.get((**int**)   
    ((amostra.size() / 2) + 0.5))) / 2;  //Pega o valor do meio, calculando a média no processo
87. } **else** { //Caso contrário, o número é impar e se deve executar o seguinte
88. mediana = amostra.get((**int**) ((amostra.size() / 2) + 0.5));  //Pega o valor do meio,  
    absoluto
89. }
91. //REFERENTE A VARIANCIA E DESVIO PADRAO
92. variancia = somaTudo / (amostra.size() - 1); //Efetivamente calcula a variancia
93. desvioPadrao = Math.sqrt(variancia); //Faz a raiz quadrada da variancia e atribui a  
    variavel desvioPadrao
95. //REFERENTE A EXIBICAO DOS RESULTADOS
96. System.out.println("Variancia: " + variancia + "\nDesvio Padrao: " + desvioPadrao); //  
    Exibe a variancia e o desvio padrão
97. System.out.print("Maior: " + maior + "\nMenor: " + menor + "\nMediana: " + mediana +   
    "\nModa: "); //Exibe o Maior, o menor e a Mediana
98. **if**(maiorContagem > 1) { System.out.println(moda); } **else** { System.out.println("Sem   
    repetição de números"); } //Exibe a moda ou avisa se não houver
100. }
101. }