QUESTÕES

⚠EQUIPE: 2 discentes - Envio: Classroom - Somente 1 envio por equipe ⚠

⚠Nome da equipe deve ser adicionado no momento do envio⚠

⚠Um único .c/.h por questão⚠

⚠Não coloque nenhum endereço relativo ao seu computador⚠

⚠NOTA $\in [0, 10]$ $\wedge$ NOTA corresponde a $\frac{1}{3} * \frac{1}{3}$ da média da disciplina⚠

1. Considere as seguintes entrada e saída abaixo:

L0Q1.in

```
points (−2,−1) (4,2) (4,0) (2,2) (8,6)
```

L0Q1.out

```
points (−2,−1) (2,2) (4,0) (4,2) (8,6) distance 18.75 shortcut 12.21
```

Sobre a entrada
A cada linha um `points` marca o início de uma lista de pontos no espaço bidimensional; veja a figura abaixo.
Sobre a saída
Na saída é preciso ordenar todos os pontos `points` conforme suas distâncias Euclidianas em relação origem, ou seja, em relação ao ponto (0,0).
Em seguida deve-se exibir a distância total Euclidiana `distance` considerando os pontos na mesma ordem em que aparecem na entrada.
Por fim, deve-se exibir a distância total Euclidiana `shortcut` entre o primeiro e último ponto na mesma ordem em que aparecem na entrada.
Observações

- Não faça Ctrl+C/Ctrl+V deste arquivo PDF no seu arquivo de entrada pois podem ocorrer erros.
- Cada ponto aparece separado por **apenas um espaço** em branco dos demais e entre (), números com parte decimal são representados usando o ponto final e não a vírgula.
- Sua resposta deve estar em um único arquivo chamado RL0Q1.[java, py, js, c, cpp, etc]
- Você deve ler o arquivo L0Q1.in e escrever o arquivo L0Q1.out no mesmo diretório onde estiver rodando seu código RL0Q1.[java, py, js, c, cpp, etc]

jose.dihego@ifba.edu.br

- Não deve ser usado nenhum recurso de ordenação da linguagem de programação de sua escolha
- Pode e vai haver mais de uma linha na estrada
- O número de linhas na entrada deve ser o mesmo da saída, ou seja cada linha da entrada produzirá uma linha na saída
- Os pontos podem ter coordenadas reais, ou seja o ponto (13.34,10.2) é um ponto válido para a entrada
- **não existem** dois espaços em branco seguidos

jose.dihego@ifba.edu.br

2. Considere as seguintes entrada e saída abaixo:

L0Q2.in

```
maria 3.15 jose 4 8 -1 12.7 (-1,-1) julia (-0.5,-0.5)
74.5 3.15 jose 4 8 -1 12.7 (8,2) carlos (-0.5,-0.5)
```

L0Q2.out

```
str:jose julia maria int:-1 4 8 float:3.15 12.7 p:(-0.5,-0.5) (-1,-1)
str:carlos jose int:-1 4 8 float:3.15 12.7 74.5 p:(-0.5,-0.5) (8,2)
```

Esta questão consiste em ler uma lista de valores contendo strings, inteiros, reais e pontos no espaço bidimensional. A saída consiste em ordenar de maneira crescente, as strings por ordem alfabética, os inteiros, os reais e por fim os pontos por ordem crescente em relação a suas distância Euclidianas em relação a origem, o ponto (0,0).

Observações

- Não faça Ctrl+C/Ctrl+V deste arquivo PDF no seu arquivo de entrada pois podem ocorrer erros.
- Cada ponto aparece entre (), números com parte decimal são representados usando o ponto final e não a vírgula.
- Sua resposta deve estar em um único arquivo chamado RL0Q2.[java, py, js, c, cpp, etc]
- Você deve ler o arquivo L0Q2.in e escrever o arquivo L0Q2.out no mesmo diretório onde estiver rodando seu código RL0Q2.[java, py, js, c, cpp, etc]
- Não deve ser usado nenhum recurso de ordenação da linguagem de programação de sua escolha
- Pode e vai haver mais de uma linha na estrada
- O número de linhas na entrada deve ser o mesmo da saída, ou seja cada linha da entrada produzirá uma linha na saída
- Os pontos podem ter coordenadas reais, ou seja o ponto (13.34,10.2) é um ponto válido para a entrada
- **não existem** dois espaços em branco seguidos

jose.dihego@ifba.edu.br