Dredd - Juiz Online

Minutos Restantes: 565

Usuário: Lucas Gomes Colombo

Notas:
Q1: 100
Q2: 75.8
Q3: 100
Q4: ?
Q5: ?
Q6: ?
Q7: 100
Q8: 100
Total: 59

Lista de estudos Ordenação - Bublesort, InsertionSort, SelectionSort

Prova Aberta Até: 26/04/2022 09:30:00

Número Máximo de Tentativas: 5

Atenuação da Nota por Tentativa: 0%

Instruções para a prova: Lista avaliativa do conteúdo Métodos de ordenação - parte 1 para todas as turmas de IAlg. Deve ser feita individualmente e está sujeita à verificação de plágio.

Questão 1: Entrega de Pizzas (Registro)(Ordenação) - 2

Uma pizzaria precisa entregar mais pizzas que estava planejando. Entretanto, para agilizar as entregas, seu motoboy atenderá somente os N clientes mais próximos da pizzaria. Como saída, o programa deverá retornar a soma do valor das pizzas nos N clientes mais próximos da pizzaria.

Lembre: N é somente uma porção de clientes de um total de clientes que serão fornecidos na entrada.

Para cada cliente, a pizzaria possui os seguintes dados (usar um registro):

Nome do cliente (string).
Coodenada x do cliente (real),
Coordenada y do cliente (real).
Valor da pizza (real).

Para cada cliente, deverá ser criado um registro com os dados acima, e a ordenação deverá ser feita usando o algoritmo **InsertionSort**. Não haverá clientes com distância repetida.

Dica: A distância entre dois pontos pode ser calculada pela forma: sqrt [(x1-x2)^2 + (y1-y2)^2]

Entradas:

1. Número de clientes que serão atendidos
2. Coordenada x da pizzaria
3. Coordenada y da pizzaria
4. Número total de clientes
5. Nome cliente 1
6. Coordenada x do cliente 1

7. Coordenada y do cliente 1
8. Valor da pizza do cliente 1
9. Nome cliente 2
10. Coordenada x do cliente 2
11. Coordenada y do cliente 2
12. Valor da pizza do cliente 2
13. ...

Saídas:

1. Soma do valor das pizzas dos N clientes mais próximos da pizzaria

Exemplo de Entrada:

```
2
5
5
3
Joao
10
5
22.50
Lucas
15
30
32.00
Maria
20
10
21.00
```

Exemplo de Saída:

```
43.50
```

Peso: 1

Última tentativa realizada em: 25/04/2022 11:27:43

Tentativas: 1 de 5

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo Nenhum arquivo escolhido Enviar Resposta

Minutos Restantes: 565

Usuário: Lucas Gomes Colombo

Notas:
Q1: 100
Q2: 75.8
Q3: 100
Q4: ?
Q5: ?
Q6: ?
Q7: 100
Q8: 100
Total: 59

Questão 2: Ordenação - Ordenar Produtos (registros, módulos, Insertion Sort)

Faça um programa que lê dados de vários produtos, ordena-os e escreve aqueles que estiverem dentro de uma faixa de preços.

Inicialmente o programa deverá ler a quantidade de produtos que deve ser lida. Cada produto deve ser representado por um registro que tem identificador (número inteiro), descrição (sequência de caracteres - 79 caracteres são suficientes) e preço (número real). Os campos dos produtos devem ser lidos na ordem mencionada anteriormente, sendo que podem haver espaços na descrição do produto. Considere que cada um dos campos será digitado numa linha e que **pode** haver uma linha em branco entre os produtos.

Após a leitura, os produtos deverão ser ordenados por preço. Depois o programa deverá ler um preço mínimo e um máximo. Por fim, o programa deverá escrever em ordem crescente de preço todos os produtos cujo preço está no intervalo determinado. O intervalo é fechado em ambos os limites. Se não houver produtos no intervalo informado o programa deverá escrever na saída padrão **Inexistente**.

O programa deverá fazer uso de modularização. **É obrigatório o uso do InsertionSort para fazer a ordenação.**

Dica: para descartar um final de linha antes da leitura de uma linha, evitando assim a leitura de uma string vazia para uma linha, na linguagem C++, recomenda-se o uso do comando `ignore`, como em `cin.ignore(2, '\n')` que ignora até dois caracteres na tentativa de encontrar um final de linha.

Entradas:

1. O número de produtos
2. Os dados de cada produto
    1. O identificador do produto (número inteiro)
    2. A descrição do produto (pode conter espaços - não terá mais que 79 caracteres)
    3. O preço do produto (número real)
3. O limite inferior da faixa de preços (número real)
4. O limite superior da faixa de preços (número real)

Saídas:

1. Os dados de cada produto cujo preço está no intervalo determinado, do produto de menor preço para o produto de maior preço. Todos os campos do produto deverão ser escritos na ordem mencionada acima, numa mesma linha separados por espaços. O programa deverá escrever aspas em volta da descrição do produto.

Exemplo de Entrada:

```
3

4352
Lancheira Sestini
99.90
```

```
3641
Mochila 14 Xeryus
149.90

3556
Compasso 559 Basic Flex
33.90

40
200
```

Exemplo de Saída:

```
4352 "Lancheira Sestini" 99.9
3641 "Mochila 14 Xeryus" 149.9
```

Peso: 1

Última tentativa realizada em: 25/04/2022 10:43:06

Tentativas: 2 de 5

Nota (0 a 100): 75.8

Status ou Justificativa de Nota: O programa não resolve todas as instâncias do problema.

Ver Código da Última Tentativa

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo Nenhum arquivo escolhido Enviar Resposta

Questão 3: Ordenação - Ordena até posição (v1)

Nem sempre deseja-se ordenar um vetor inteiro. Faça um programa que receba um conjunto de inteiros e a posição (índice) do vetor até onde deseja-se ordenar. A ordenação deverá ser por ordem decrescente, usando **selectionsort**, de tal forma que somente os primeiros elementos participam da ordenação. Os últimos elementos devem ser desconsiderados.

Você deverá adaptar o trecho de código a seguir do algoritmo selectionsort com ordenação crescente. Atenção com a implementação de algoritmos clássicos: adicionar testes ou variáveis, introduzir processamento desnecessário resultará em perda de pontos na avaliação manual. Os nomes das variáveis devem ser significativos: altere os nomes conforme necessário para que continuem sendo significativos após as modificações realizadas.

Use modularização. Não misture interface e processamento.

```
for (int i=0; i < tam-1; ++i) {
    posMenor = i;
    for (int j=i+1; j < tam; ++j) {
        if (vetor[j] < vetor[posMenor])
            posMenor = j;
    }
    aux = vetor[i];
    vetor[i] = vetor[posMenor];
    vetor[posMenor] = aux;
}
```

Entradas:

1. Tamanho do vetor
2. Os elementos do vetor (números inteiros)
3. Posição até onde o vetor deve ser ordenado.

Saídas:

1. Todo o vetor resultante.

Exemplo de Entrada :

```
5
1 4 3 5 2
2
```

Exemplo de Saída:

```
4 3 1 5 2
```

Peso: 1

Última tentativa realizada em: 25/04/2022 18:29:53

Tentativas: 2 de 5

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo Nenhum arquivo escolhido Enviar Resposta

Questão 4: Modularização - Ordenação Selection Sort e busca/impressão em vetor de registros - Memes

Com a proximidade das férias, você resolveu fazer um programa para organizar seus memes. Para isso, você verifica que precisará implementar um algoritmo para cadastrar, ordenar e buscar valores num vetor de registros. Cada registro conterá os seguintes dados: número do meme, nome do arquivo (sem espaços), assunto (sem espaços) e site. Como você pretende futuramente expandir o aplicativo, o programa será construído utilizando módulos.

Assim, o programa deve ter módulos distintos para:

- ler valores, preenchendo o vetor de registros;
- ordenar os elementos de um vetor, utilizando o **número do meme como chave**, utilizando o algoritmo Selection Sort;
- imprimir todos os memes de um **dado site** passado como parâmetro. A impressão deverá ser realizada no vetor já ordenado.

Outros módulos são opcionais.

O módulo principal deve:

1. ler a quantidade de elementos a ser processada;
2. ativar o módulo que lê os elementos do vetor;
3. ativar o módulo que ordena o vetor pelo número do meme;
4. ler o site para busca;
5. ativar o módulo que imprime os memes que pertençam a um dado site. Caso não exista nenhum meme daquele site, deverá ser impresso a mensagem **Inexistente**. Os dados do memes do site devem ser impressos seguindo a ordem do vetor já ordenado. Os campos deverão ser impressos na mesma ordem de leitura.

A ordem de entrada dos dados dos registros é a seguinte:

```
num_do_meme   nome_do_arquivo   assunto   site
```

Exemplo de entrada:

```
5
5 xxx.jpg minions 9gag
3 yyy.jpg funk nuintendo
2 zzz.jpg minions nuintendo
1 ppp.jpg funk 9gag
6 qqq.jpg politica facebook
9gag
```

Exemplo de saída:

```
1 ppp.jpg funk 9gag
5 xxx.jpg minions 9gag
```

Exemplo de entrada:

```
8
91 1.jpg minions reddit
18 2.jpg minions 9gag
21 3.jpg minions facebook
45 4.jpg minions nuintendo
15 5.jpg temer capinaremos
55 6.jpg temer capinaremos
78 7.jpg temer facebook
9  8.jpg temer 9gag
4chan
```

Exemplo de saída:

```
Inexistente
```

Exemplo de entrada:

```
4
8 a.jpg trump 9gag
6 b.jpg trump 9gag
3 c.jpg trump 9gag
1 d.jpg trump 9gag
9gag
```

Exemplo de saída:

```
1 d.jpg trump 9gag
3 c.jpg trump 9gag
6 b.jpg trump 9gag
8 a.jpg trump 9gag
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo | Nenhum arquivo escolhido | Enviar Resposta

Questão 5: Modularização - Pagamento de pessoal (registros, ordenação com Bubble Sort) (tamanho definido)

Faça um programa que controla salários de funcionários. Ele deve ler dados a respeito de vários funcionários, calcular o salário de cada um e escrever os dados ordenados. A quantidade de funcionários que terão seu salário calculado deve ser informada pelo usuário.

Os dados importantes na representação de um funcionário são (utilizar registros):

1. identificador do funcionário (número inteiro),
2. número de horas trabalhadas (número real),
3. valor de uma hora trabalhada (número real) e
4. salário a receber (número real).

Deve existir um subprograma que realiza a leitura dos dados, na mesma ordem vista acima. Ele deve ser usado sempre que outro subprograma precisar fazer tal leitura. Entretanto, o **salário a receber** não deve ser realmente lido, ele deve ser calculado a partir dos valores lidos para o **número de horas trabalhadas** e o **valor de uma hora trabalhada**. No seu programa, este subprograma deve ser usado pelo subprograma principal, responsável por ler todos os funcionários num laço.

Também deve existir um subprograma que ordena funcionários (ordem crescente) pelo **salário a receber**, utilizando o algoritmo de ordenação Bubble Sort. Ele deve servir para ordenar qualquer quantidade de funcionários. Depois deve escrever os dados de todos funcionários em ordem.

Entradas:

- Quantidade de funcionários.
- Dados de vários funcionários. Os dados não incluem o **salário a receber**.

Saídas:

- O identificador e o **salário a receber** de cada funcionário (**nesta ordem**), ordenados pelo **salário a receber** (ordem crescente).

Exemplo de entrada:

```
3
12 80.1 14.9
11 76.1 12.2
14 82.6 11.8
```

Exemplo de saída:

```
11 928.42
14 974.68
12 1193.49
```

Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo | Nenhum arquivo escolhido | Enviar Resposta

Questão 6: Ordenação - Bubble Sort detalhado

Como forma de facilitar o ensino dos métodos de ordenação, um professor resolveu fazer um programa que mostrasse o funcionamento do **Bubble Sort** passo a passo. A ideia é que a cada comparação realizada pelo método, o vetor seja impresso destacando-se os elementos que estão sendo comparados e, quando ocorre uma troca, o vetor é exibido novamente mostrando os elementos que serão trocados. Para isso será necessário criar um programa que tenha:

- Um subprograma que exibe um vetor de inteiros com os elementos separados por tabulações. Além disso, ele deve receber por parâmetro as posições de dois elementos e um booleano indicando se eles serão trocados ou não. Se eles serão trocados deve ser impresso " t" logo após os dois elementos, caso contrário, deve ser impresso " *". Para a última impressão (sem marcas), você pode usar um índice inválido, evitando imprimir tanto os asteriscos quanto os " t".
- Um subprograma com a implementação do Bubble Sort (para vetor de inteiros), que utilize adequadamente o subprograma anterior. Obs: as chamadas devem ser feitas antes dos elementos serem trocados, mas já indicando se serão trocados ou não.
- No subprograma principal deve ser pedido ao usuário a quantidade de elementos e os elementos do vetor.
- Não deve haver nenhuma operação de escrita fora do subprograma de exibição do vetor.

Dica: faça a implementação de forma incremental. Ou seja, primeiro implemente o método Bubble Sort comum e faça um subprograma para apenas exibir o vetor normalmente. Depois altere o subprograma conforme o enunciado e o utilize no método de ordenação.

Entradas:

1. Quantidade de números do vetor.
2. Os números do vetor a ser ordenado.

Saídas:

1. Os passos de ordenação do Bubble Sort conforme o enunciado.
2. Uma última exibição do vetor sem destacar elementos (mas ainda assim usando o mesmo subprograma).

Exemplo de Entrada:

```
6
4 8 2 9 5 0
```

Exemplo de Saída:

```
4 *     8 *     2       9       5       0
4       8 t     2 t     9       5       0
4       2       8 *     9 *     5       0
4       2       8       9 t     5 t     0
4       2       8       5       9 t     0 t
4 t     2 t     8       5       0       9
2       4 *     8 *     5       0       9
2       4       8 t     5 t     0       9
2       4       5       8 t     0 t     9
```

```
2 *     4 *     5       0       8       9
2       4 *     5 *     0       8       9
2       4       5 t     0 t     8       9
2 *     4 *     0       5       8       9
2       4 t     0 t     5       8       9
2 t     0 t     4       5       8       9
0       2       4       5       8       9
```


Peso: 1

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo | Nenhum arquivo escolhido | Enviar Resposta

Questão 7: Ordenação - Soma dos maiores elementos

Faça um algoritmo para somar os K maiores elementos de um vetor de inteiros, que deverá ser ordenado, com tamanho N. Sendo que o vetor pode possuir valores repetidos e K pode ir de 0 até N.

Os valores devem ser ordenados pelo método **insertion sort**.

Entradas:

1. Tamanho do vetor
2. Elementos do vetor
3. Quantidade dos maiores elementos a serem somados

Saídas:

1. Soma dos K maiores elementos.

Exemplo de Entrada:

```
6
12 1 45 8 4 9
2
```

Exemplo de Saída:

```
57
```

Exemplo de Entrada:

```
6
12 1 45 8 4 9
0
```

Exemplo de Saída:

```
0
```

Peso: 1

Última tentativa realizada em: 25/04/2022 21:43:49

Tentativas: 1 de 5

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo Nenhum arquivo escolhido Enviar Resposta

Questão 8: Ordenação - Selection Sort Inverso

Implemente uma variação do algoritmo de ordenação Selection Sort para que os elementos do vetor sejam ordenados em **ordem decrescente**. Dessa forma, deve-se procurar o maior valor de um vetor e trocá-lo com o que estiver na primeira posição do vetor. Em seguida, procurar o segundo maior valor e trocá-lo com o que estiver na segunda posição do vetor. Assim sucessivamente, até que o vetor esteja ordenado.

Os elementos do vetor devem ser impressos após cada troca de valor.

Entradas:

1. Tamanho do vetor que será ordenado.
2. Vários números inteiros que serão ordenados.

Saídas:

1. Os elementos do vetor a cada troca de valor.

Exemplo de Entrada:

```
5
2 3 1 5 7
```

Exemplo de Saída:

```
7 3 1 5 2
7 5 1 3 2
7 5 3 1 2
7 5 3 2 1
```

Peso: 1

Última tentativa realizada em: 25/04/2022 20:25:05

Tentativas: 1 de 5

Nota (0 a 100): 100

Status ou Justificativa de Nota: Nenhum erro encontrado.

Ver Código da Última Tentativa

Nova Resposta:

Selecione o arquivo com o código fonte do programa que resolve o problema para enviá-lo.

Escolher arquivo Nenhum arquivo escolhido Enviar Resposta

Minutos Restantes: 565

Usuário: Lucas Gomes Colombo

Notas:
Q1: 100
Q2: 75.8
Q3: 100
Q4: ?
Q5: ?
Q6: ?
Q7: 100
Q8: 100
Total: 59