

Aula Prática V - QuickSort

- Procedimento para a entrega:

1. Submissão: via **Moodle**.
2. Os nomes dos arquivos e das funções devem ser especificados considerando boas práticas de programação.
3. Funções auxiliares, complementares aquelas definidas, podem ser especificadas e implementadas, se necessário.
4. A solução deve ser devidamente modularizada e separar a especificação da implementação em arquivos *.h* e *.c* sempre que cabível.
5. Os arquivos a serem entregues, incluindo aquele que contém *main()*, devem ser compactados (*.zip*), sendo o arquivo resultante submetido via **Moodle**.
6. Caracteres como acento, cedilha e afins não devem ser utilizados para especificar nomes de arquivos ou comentários no código.
7. Siga atentamente quanto ao formato da entrada e saída de seu programa, exemplificados no enunciado.
8. Durante a correção, os programas serão submetidos a vários casos de testes, com características variadas.
9. A avaliação considerará o tempo de execução e o percentual de respostas corretas.
10. Eventualmente, serão realizadas entrevistas sobre os estudos dirigidos para complementar a avaliação.
11. Considere que os dados serão fornecidos pela entrada padrão. Não utilize abertura de arquivos pelo seu programa. Se necessário, utilize o redirecionamento de entrada.
12. Os códigos fonte serão submetidos a uma ferramenta de detecção de plágios em software.
13. Códigos cuja autoria não seja do aluno, com alto nível de similaridade em relação a outros trabalhos, ou que não puder ser explicado, acarretará na perda da nota.
14. Códigos ou funções prontas específicos de algoritmos para solução dos problemas elencados não são aceitos.
15. Não serão considerados algoritmos parcialmente implementados.

- **Bom trabalho!**

Ordenação de estoque

Você foi contratado para criar um programa que ordene as peças recém compradas no estoque de uma fábrica. A fábrica compra três tipos de peças diferentes: A, B e C. As peças de tipo A têm tamanhos variados e, portanto, são ordenadas pelo peso. As peças de tipo B podem ser caras ou baratas, devendo ser ordenadas pelo seu preço. Já as peças do tipo C precisam apenas seguir a ordem do lote. Em caso de empate na ordenação para qualquer tipo de peça, a ordem de desempate deve ser: lote, peso e preço.

Você deve utilizar o algoritmo de QuickSort para realizar a ordenação de cada tipo de peça de acordo com seu critério de ordenação. **Toda ordenação deve ser feita de forma crescente.**

Especificação da Entrada e da Saída

A entrada irá consistir na indicação da quantidade de peças que chegaram e nas descrições dessas peças nas linhas subsequentes. Cada linha de descrição indica, respectivamente, o tipo de peça, o ID, o número do lote (sempre um inteiro), o preço e o peso da peça em gramas.

A saída deve imprimir a lista de peças ordenadas de forma crescente, listadas por tipo.

Entrada	Saída
8 A 010 10 0.15 200 B 105 78 1.88 15 A 001 44 0.27 540 C 110 75 1.00 205 C 108 14 0.98 201 A 123 35 0.10 78 A 333 25 0.35 874 C 144 15 0.95 198	Tipo A: 123 010 001 333 Tipo B: 105 Tipo C: 108 144 110

Siga o protótipo fornecido, sem alterar os nomes dos arquivos e das funções principais e execute sua implementação com os testes disponibilizados, comparando a saída esperada da sua saída.

A saída e a entrada devem OBRIGATORIAMENTE seguir o padrão fornecido no trabalho e permitir redirecionamento por arquivo.

Diretivas de Compilação

As seguintes diretivas de compilação devem ser usadas (essas são as mesmas usadas no run.codes).

```
$ gcc -c aluno.c -Wall  
$ gcc -c pratica.c -Wall  
$ gcc aluno.o pratica.o -o exe
```

Antes de enviar TESTE o seu código com:

```
$ ./exe < teste1.in
```

Avaliação de *leaks* de memória

Uma forma de avaliar se não há *leaks* de memória é usando a ferramenta *valgrind*. Um exemplo de uso é:

```
1 gcc -g -o exe *.c -Wall  
2 valgrind --leak-check=full -s ./exe < casoteste.in
```

Espera-se uma saída com o fim semelhante a:

```
1 ==xxxxx== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Para instalar no Linux, basta usar: `sudo apt install valgrind`.