Este é o CS50

Introdução à Ciência da Computação (CS50)

OpenCourseWare

Doar (https://cs50.harvard.edu/donate)

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan)
(https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/)
(https://www.reddit.com/user/davidjmalan)
(https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Organizar

Problema a resolver

Relembrando a aula, vimos alguns algoritmos para ordenar uma sequência de números: ordenação por seleção, ordenação por bolha e ordenação por intercalação.

- O algoritmo de ordenação por seleção percorre as partes não ordenadas de uma lista, selecionando o menor elemento a cada iteração e movendo-o para a posição correta.
- O algoritmo de ordenação por bolha compara pares de valores adjacentes um de cada vez e os troca de lugar se estiverem na ordem incorreta. Isso continua até que a lista esteja ordenada.
- O algoritmo de ordenação por intercalação (merge sort) divide recursivamente a lista em duas partes repetidamente e, em seguida, mescla as listas menores de volta em uma lista maior na ordem correta.

Neste problema, você analisará três programas de ordenação (compilados!) para determinar quais algoritmos eles utilizam. Em um arquivo chamado `<nome_do_arquivo>` `answers.txt` em uma pasta chamada `<nome_da_pasta>` `sort`, registre suas respostas, juntamente com uma explicação para cada programa, preenchendo os espaços em branco marcados com `<br>` `TODO`.

Código de Distribuição

Para este problema, você precisará de algum "código de distribuição" — ou seja, código escrito pela equipe do CS50. Foram fornecidos três programas em C já compilados: `c` `sort1`, `sort2` `c` e `sort3` `c`, além de vários `.txt` arquivos de entrada e outro arquivo, `c` `answers.txt`, no qual você deverá escrever suas respostas. Cada um dos programas `c` `sort1`, `c` `sort2` e `sort3` `c` implementa um algoritmo de ordenação diferente: ordenação por seleção, ordenação por bolha ou ordenação por intercalação (embora não necessariamente nessa ordem!). Sua tarefa é determinar qual algoritmo de ordenação é usado por cada arquivo. Comece baixando esses arquivos.

▼ **Baixe os arquivos de distribuição.**

Abra o VS Code (https://cs50.dev/) .

Comece clicando dentro da janela do terminal e, em seguida, execute `cd` o comando. Você verá que o "prompt" será semelhante ao abaixo.

```
$
```

Clique dentro da janela do terminal e execute o seguinte comando.

```
wget https://cdn.cs50.net/2024/fall/psets/3/sort.zip
```

Em seguida, pressione Enter para baixar um arquivo ZIP chamado [nome do arquivo] `sort.zip` em seu espaço de código. Preste atenção ao espaço entre [nome do arquivo] `wget` e o URL a seguir, ou qualquer outro caractere!

Agora execute

```
unzip sort.zip
```

para criar uma pasta chamada `sort`. Você não precisa mais do arquivo ZIP, então pode executar

```
rm sort.zip
```

e responda com “y” seguido de Enter quando solicitado para remover o arquivo ZIP que você baixou.

Dicas

Clique nos botões abaixo para ler algumas dicas!

▼ **Explore os `.txt`arquivos**

- `.txt` Você receberá vários arquivos. Esses arquivos contêm `n` linhas de valores, que podem estar invertidas, embaralhadas ou ordenadas.

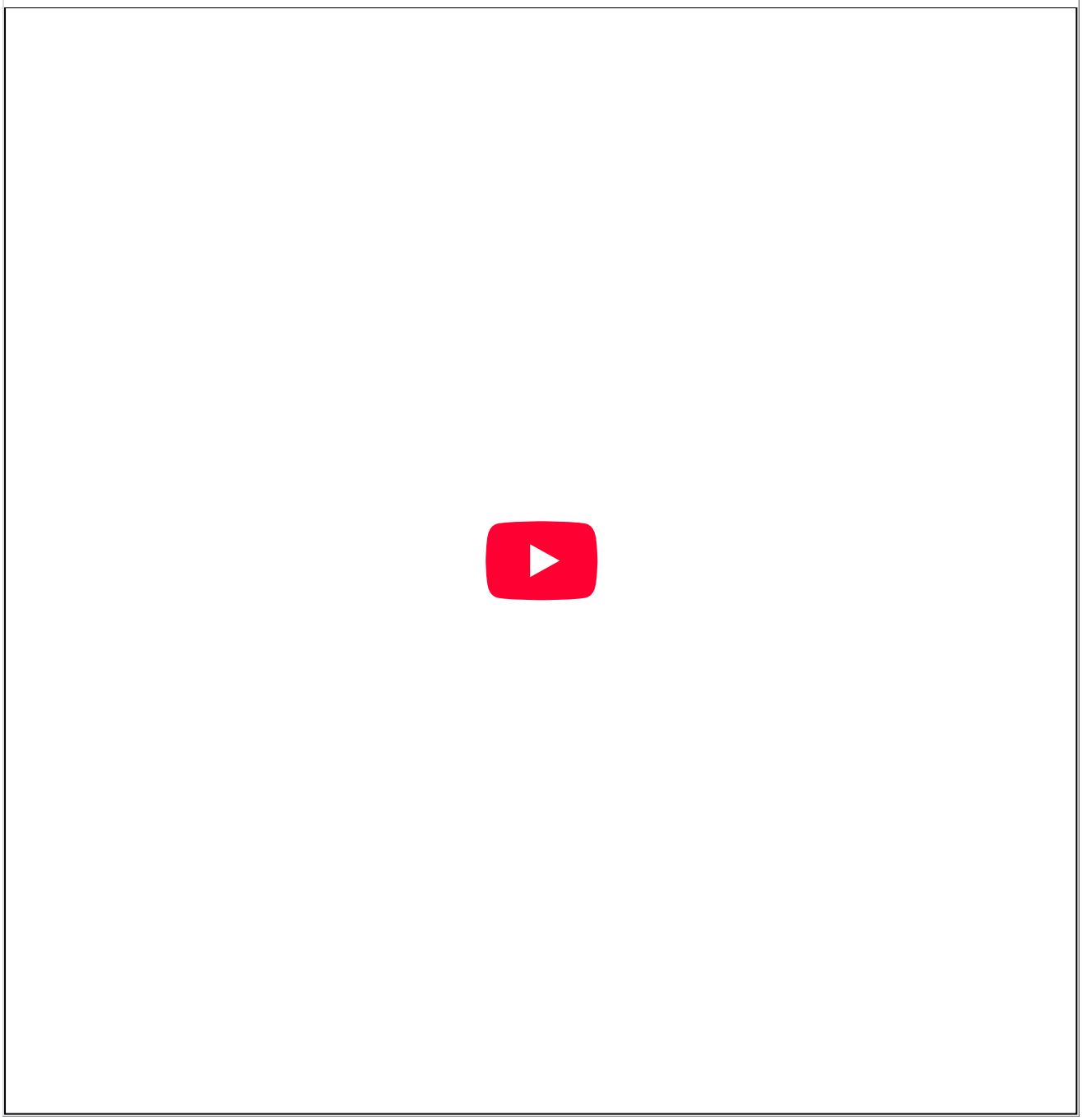
    - Por exemplo, `reversed10000.txt` contém 10.000 linhas de números que estão invertidos em relação a `10000`, enquanto `random50000.txt` contém 50.000 linhas de números que estão em ordem aleatória.
- Os diferentes tipos de `.txt` arquivos podem ajudar a determinar qual tipo de ordenação corresponde a qual lista. Considere o desempenho de cada algoritmo com uma lista já ordenada. E com uma lista invertida? Ou uma lista embaralhada? Pode ser útil trabalhar com uma lista menor de cada tipo e analisar cada processo de ordenação.

▼ **Cronometre cada classificação com diferentes entradas.**

- Para executar as operações de classificação nos arquivos de texto, execute o comando no terminal `./[program_name] [text_file.txt]`. Certifique-se de ter utilizado o comando `cd` para entrar no `sort` diretório!
    - Por exemplo, para classificar `reversed10000.txt` com `sort1`, execute `./sort1 reversed10000.txt`.
- Pode ser útil cronometrar suas classificações. Para isso, execute o comando `time ./[sort_file] [text_file.txt]`.
    - Por exemplo, você pode executar `time ./sort1 reversed10000.txt` o programa `sort1` em 10.000 números invertidos. Ao final da saída do terminal, você pode verificar o `real` tempo para ver quanto tempo realmente decorreu durante a execução do programa.

Passo a passo

CS50



▶ **Não sabe como resolver?**

Como testar

Correção

```
check50 cs50/problems/2025/x/sort
```

Como enviar

```
submit50 cs50/problems/2025/x/sort
```