












# Isso é CS50

Introdução do CS50 à Ciência da Computação

OpenCourseWare

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

David J. Malan (<https://cs.harvard.edu/malan/>)  
malan@harvard.edu

 (<https://www.clubhouse.com/@davidjmalan>)  (<https://www.facebook.com/dmalan>)   
(<https://github.com/dmalan>)  (<https://www.instagram.com/davidjmalan/>)   
(<https://www.linkedin.com/in/malan/>)  (<https://orcid.org/0000-0001-5338-2522>)   
(<https://www.quora.com/profile/David-J-Malan>)  (<https://www.reddit.com/user/davidjmalan>)   
(<https://www.tiktok.com/@davidjmalan>)  (<https://davidjmalan.t.me/>)   
(<https://twitter.com/davidjmalan>)

## Laboratório 3: classificar

Você está convidado a colaborar com um ou dois colegas neste laboratório, embora seja esperado que todos os alunos em qualquer grupo contribuam igualmente para o laboratório.

Analise três programas de classificação para determinar quais algoritmos eles usam.

## Fundo

Lembre-se da palestra em que vimos alguns algoritmos para classificar uma sequência de números: classificação por seleção, classificação por bolha e classificação por mesclagem.

- A classificação por seleção itera pelas partes não classificadas de uma lista, selecionando o menor elemento a cada vez e movendo-o para o local correto.
- A classificação por bolhas compara pares de valores adjacentes um de cada vez e os troca se estiverem na ordem incorreta. Isso continua até que a lista seja classificada.
- A classificação por mesclagem divide recursivamente a lista em duas repetidamente e, em seguida, mescla as listas menores de volta em uma maior na ordem correta.

## Começando

Abra o **VS Code** (<https://code.cs50.io/>).

Comece clicando dentro da janela do seu terminal e, em seguida, execute-o `cd` sozinho. Você deve descobrir que seu “prompt” se parece com o abaixo.

```
$
```

Clique dentro dessa janela de terminal e execute

```
wget https://cdn.cs50.net/2022/fall/labs/3/sort.zip
```

seguido de Enter para baixar um ZIP chamado `sort.zip` em seu codespace. Tome cuidado para não ignorar o espaço entre `wget` e o seguinte URL ou qualquer outro caractere!

Agora execute

```
unzip sort.zip
```

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

```
rm sort.zip
```

e responda com “y” seguido de Enter no prompt para remover o arquivo ZIP que você baixou.

Agora digite

```
cd sort
```

seguido de Enter para entrar (ou seja, abrir) nesse diretório. Seu prompt agora deve se parecer com o abaixo.

```
sort/ $
```

Se tudo foi bem sucedido, você deve executar

```
ls
```

e você deve ver uma coleção de `.txt` arquivos ao lado de programas executáveis `sort1`, `sort2` e `sort3`.

Se você tiver algum problema, siga estas mesmas etapas novamente e veja se consegue determinar onde errou!

## Instruções

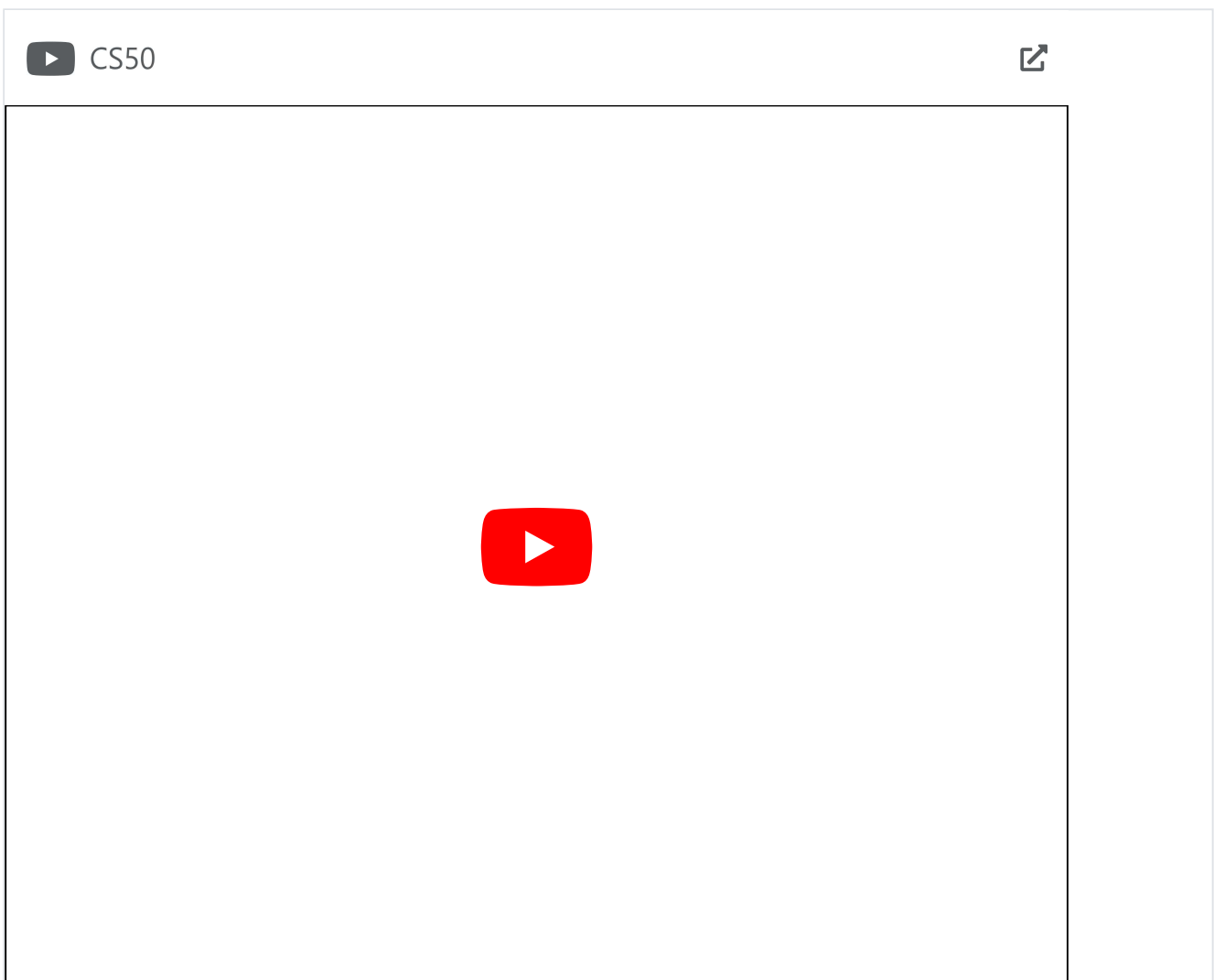
São fornecidos a você três programas C já compilados, `sort1`, `sort2` e `sort3`. Cada um desses programas implementa um algoritmo de classificação diferente: classificação por seleção, classificação por bolha ou classificação por mesclagem (embora não necessariamente nessa ordem!). Sua tarefa é determinar qual algoritmo de classificação é usado por cada arquivo.

- `sort1`, `sort2` e `sort3` são arquivos binários, então você não poderá visualizar o código-fonte C de cada um. Para avaliar qual classificação implementa qual algoritmo, execute as classificações em diferentes listas de valores.

- Vários `.txt` arquivos são fornecidos a você. Esses arquivos contêm `n` linhas de valores, invertidos, embaralhados ou classificados.
  - For example, `reversed10000.txt` contains 10000 lines of numbers that are reversed from `10000`, while `random10000.txt` contains 10000 lines of numbers that are in random order.
- To run the sorts on the text files, in the terminal, run `./[program_name] [text_file.txt]`. Make sure you have made use of `cd` to move into the `sort` directory!
  - For example, to sort `reversed10000.txt` with `sort1`, run `./sort1 reversed10000.txt`.
- You may find it helpful to time your sorts. To do so, run `time ./[sort_file] [text_file.txt]`.
  - For example, you could run `time ./sort1 reversed10000.txt` to run `sort1` on 10,000 reversed numbers. At the end of your terminal's output, you can look at the `real` time to see how much time actually elapsed while running the program.
- Record your answers in `answers.txt`, along with an explanation for each program, by filling in the blanks marked `TODO`.

## Walkthrough

This video was recorded when the course was still using CS50 IDE for writing code. Though the interface may look different from your codespace, the behavior of the two environments should be largely similar!



## Hints

- The different types of `.txt` files may help you determine which sort is which. Consider how each algorithm performs with an already sorted list. How about a reversed list? Or shuffled list? It may help to work through a smaller list of each type and walk through each sorting process.

► **Not sure how to solve?**

## How to Check Your Answers

Execute o abaixo para avaliar a exatidão de suas respostas usando `check50`. Mas certifique-se de preencher suas explicações também, que `check50` não serão verificadas aqui!

```
check50 cs50/labs/2023/x/sort
```

## Como enviar

Em seu terminal, execute o abaixo para enviar seu trabalho.

```
submit50 cs50/labs/2023/x/sort
```