

This is CS50

Week 4

Scan your HUID for attendance at the back table.
Open code.cs50.io and log in!

carterzenke.me/section

Think, Pair, Share

- What are you excited about from this week's lecture?
- What do you want to learn more about?

<https://carterzenke.me/section>

Today

- What are **pointers**? Why use them?
- How can we read from (and write to) **files**?
- Problem Set 4

Pointers

(And why to use them)

calls



3

A **variable** is a name for some value that can change.

Address

Value

0x50000000

...

0x50000004

...

0x50000008

...

0x5000000C

...

Address	Value
0x50000000	3
0x50000004	...
0x50000008	...
0x5000000C	...

Address	Value
0x50000000	3
0x50000004	...
0x50000008	...
0x5000000C	...

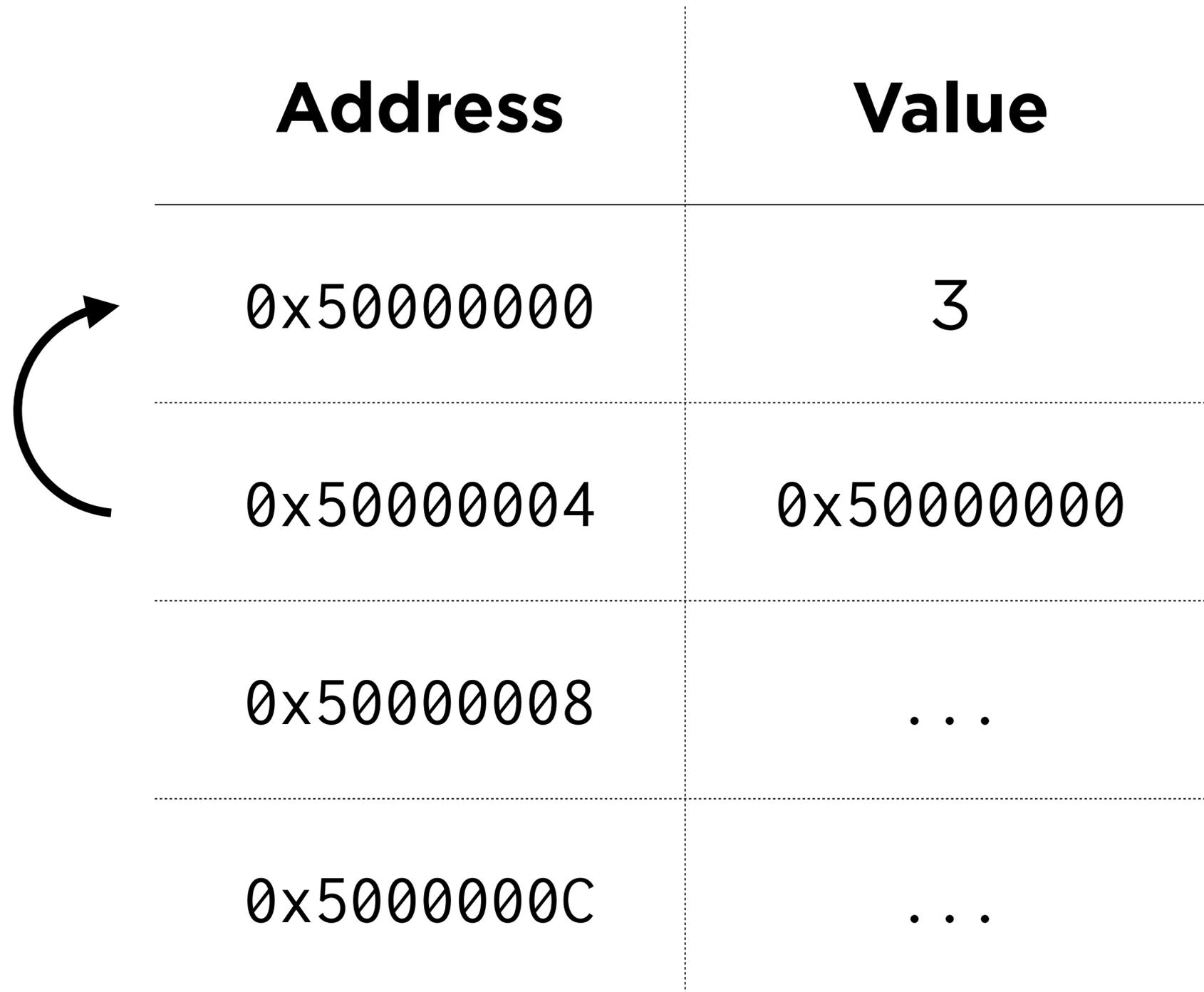
```
int calls = 3;
```

Address	Value
0x50000000	3
0x50000004	0x50000000
0x50000008	...
0x5000000C	...

```
int calls = 3;
```

Address	Value
0x50000000	3
0x50000004	0x50000000
0x50000008	...
0x5000000C	...

```
int calls = 3;
```



Address	Value
0x50000000	3
0x50000004	0x50000000
0x50000008	...
0x5000000C	...

```
int calls = 3;
```

```
int *pCalls = &calls;
```

```
int *pCalls = &calls;
```

pCalls

0x50000000

```
int *pCalls = &calls;
```

name

pCalls

0x50000000

```
int *pCalls = &calls;
```

type

pCalls

0x50000000

```
int *pCalls = &calls;
```

value

pCalls

0x50000000

Key Syntax

- **type *** is a pointer that stores the address of a **type**
- ***x** takes a pointer **x** and gets the value stored at that address.
- **&x** takes **x** and gets its address.

Reasons to Use Pointers

- You can pass variables to functions **by reference**, not just **by copy**. *The code you write is cleaner as a result.*
- You can use **dynamic memory** (e.g., with malloc). *Your programs can now scale their usage of memory according to user behavior.*

Passing by Copy
Passing by Reference

```
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

main

a

b

10

50

```
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

main

a

b

10

50

swap

a

b

10

50

```
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

main

a

b

10

50

swap

a

b

50

10

```
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b)
{
    int temp = a;
    a = b;
    b = temp;
}
```

main

a

b

10

50

```
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

main

a

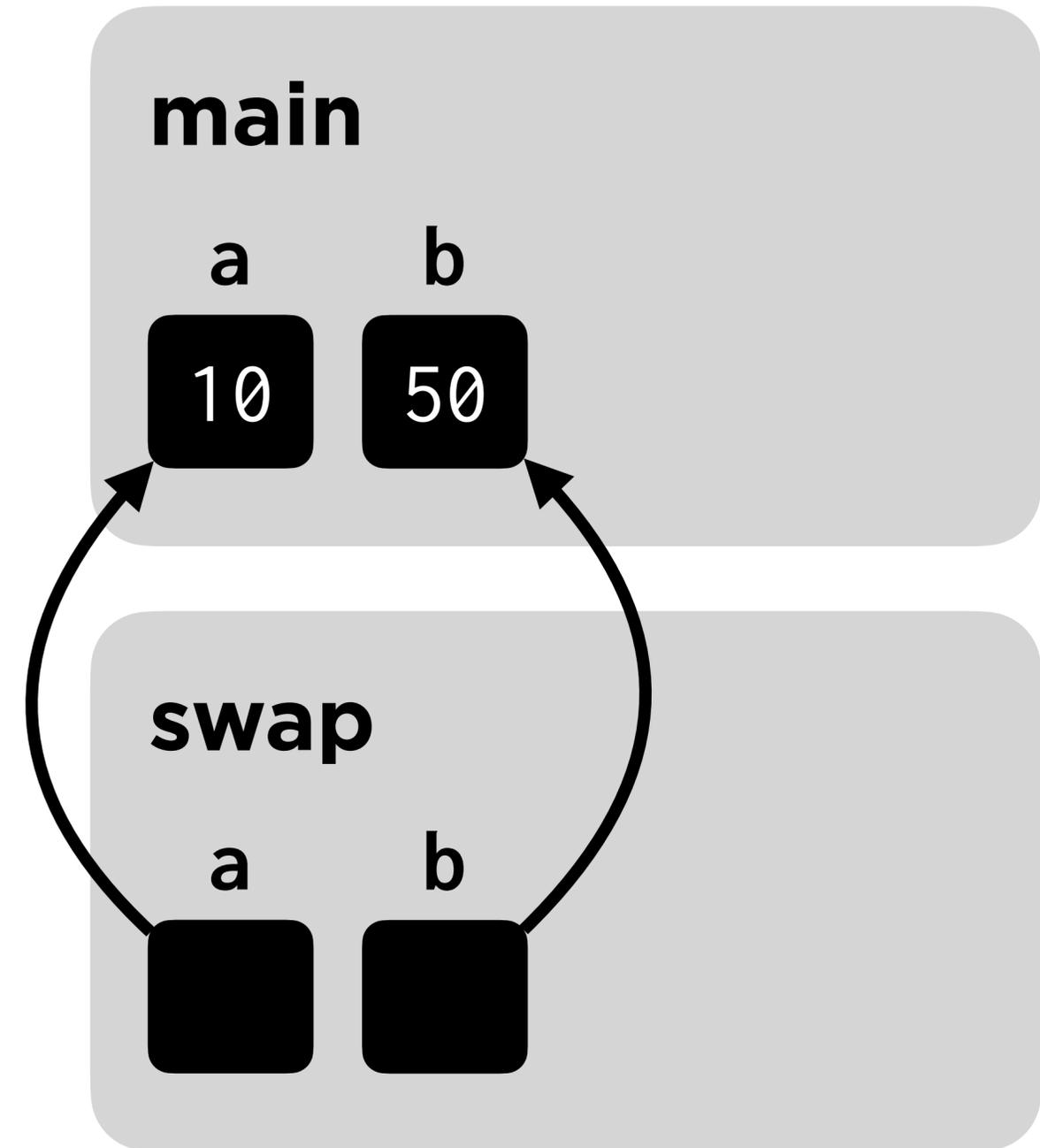
b

10

50

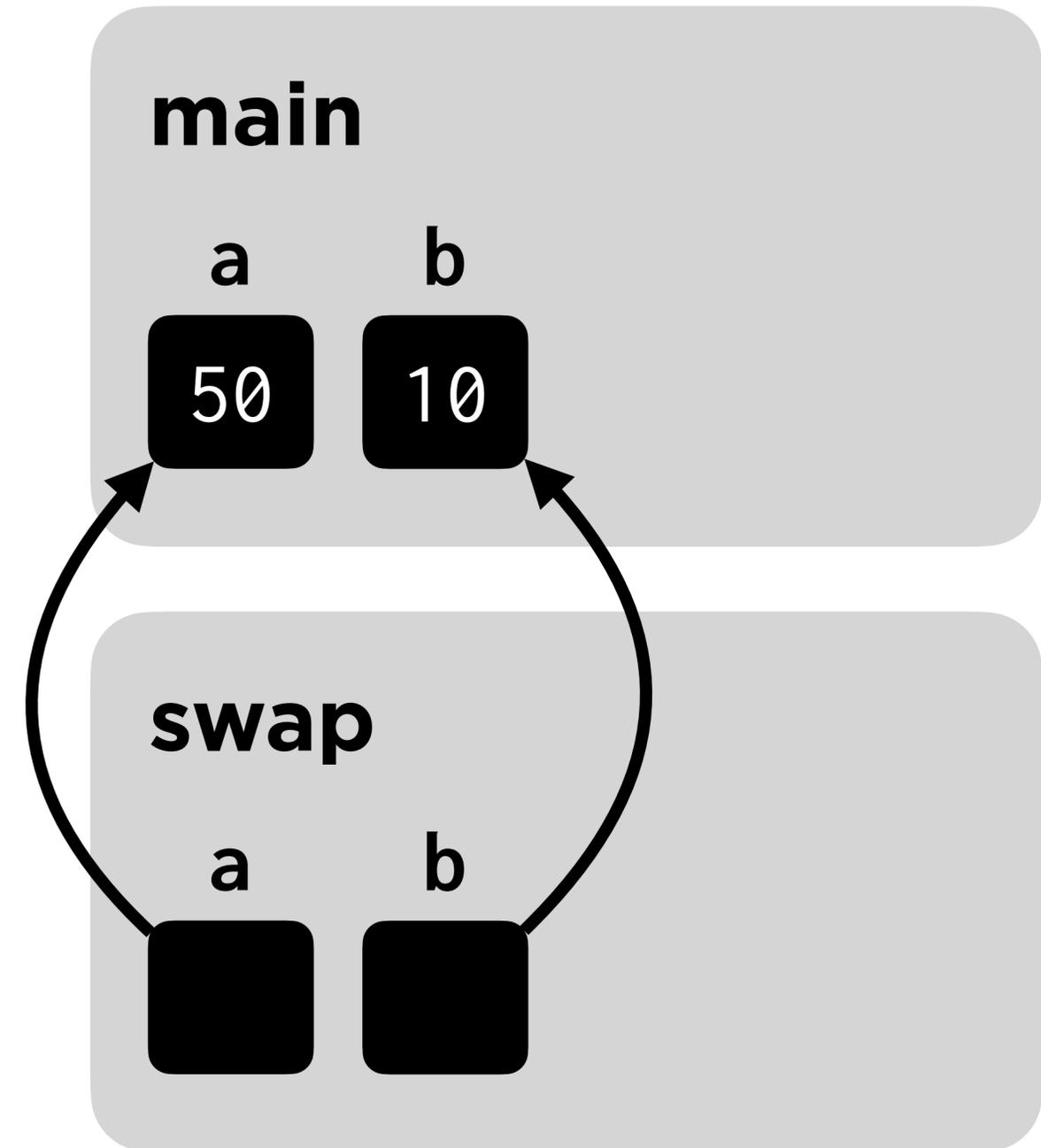
```
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```



```
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```



Pointer Prediction

- Read **pointers.c** and predict the final values of **a** and **b**.
- Once you've predicted, run **pointers.c** to check your answer!

File I/O

Reading data from (and writing data to) files

Opening and Closing Files

Key Functions

- **fopen** opens a file for future reading/writing.
- **fclose** closes a file.

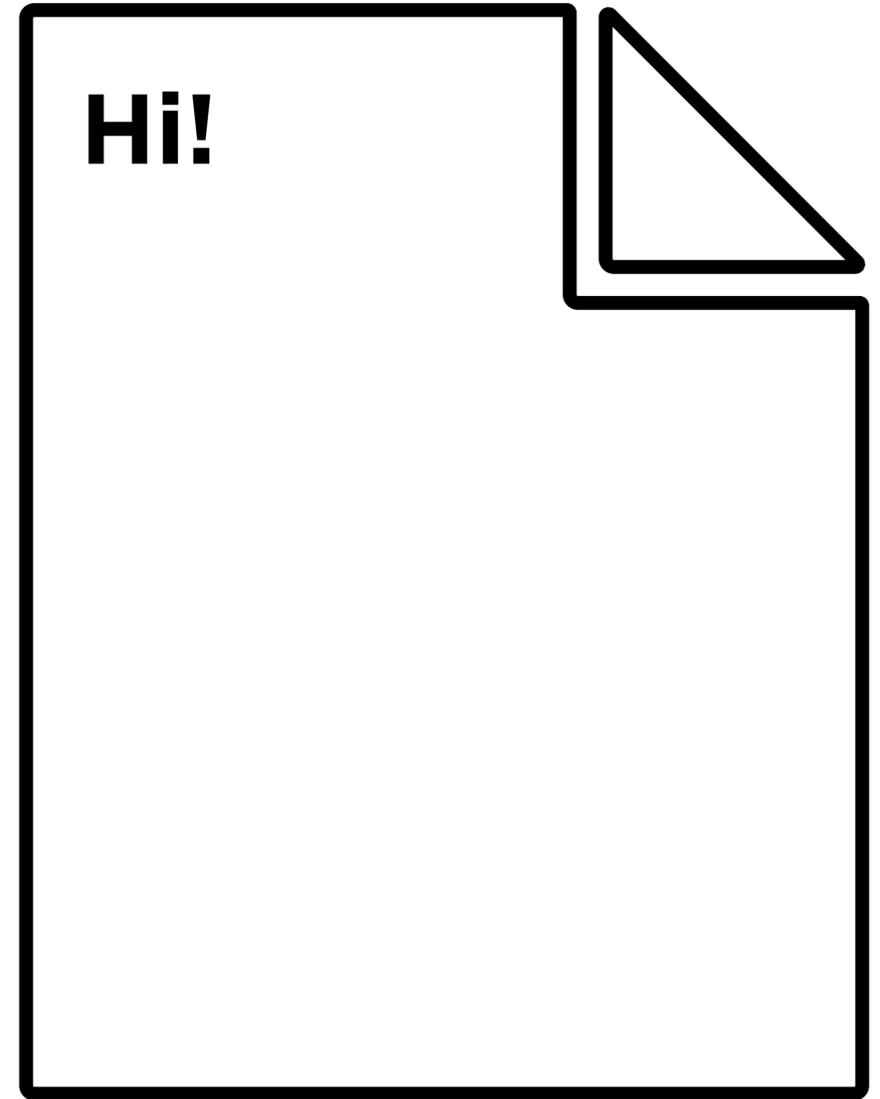
Always **fclose** all files you **fopen**!

```
FILE *f = fopen("hi.txt", "r");
```

f

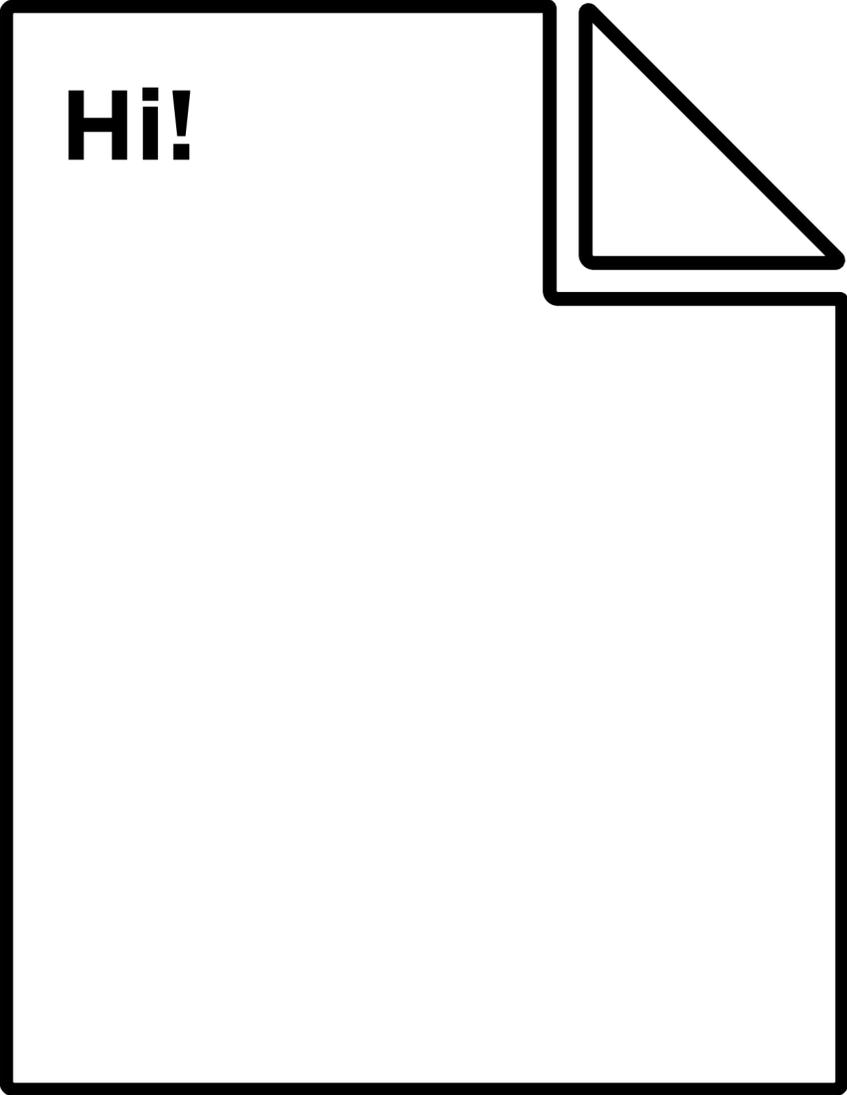
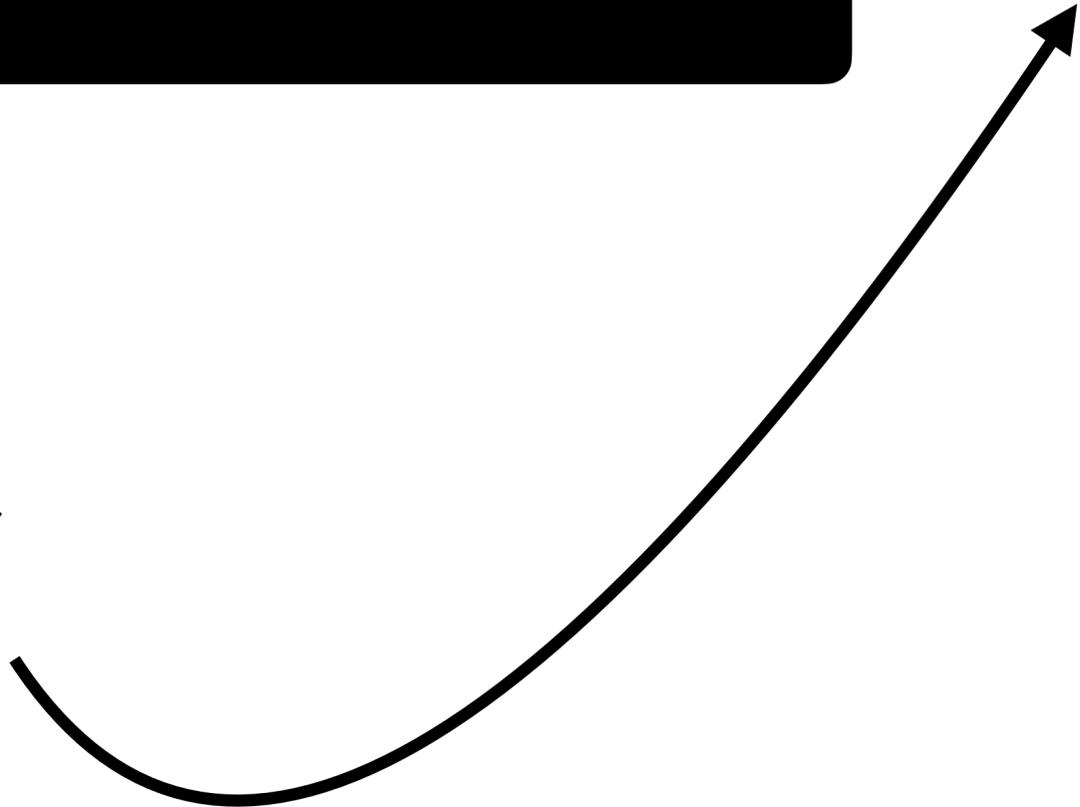
Hi!

hi.txt



```
fclose(f);
```

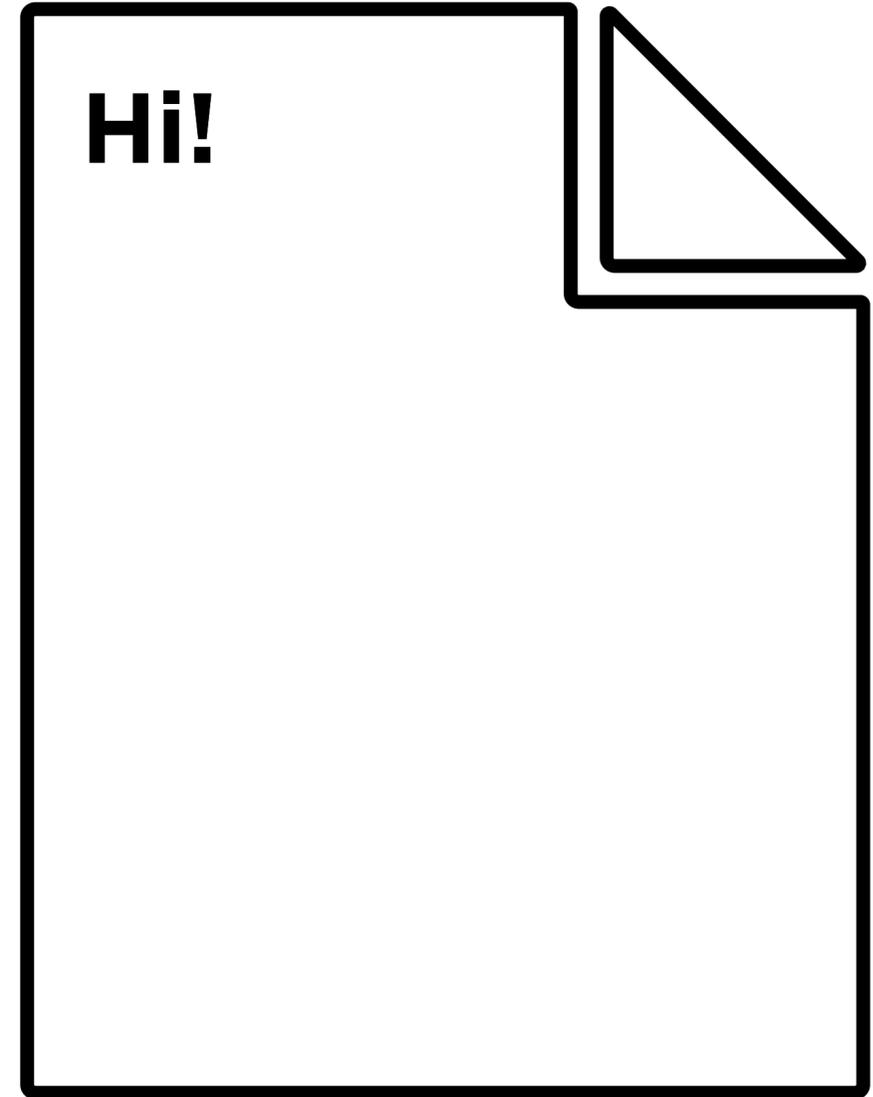
f



hi.txt

Reading and Writing

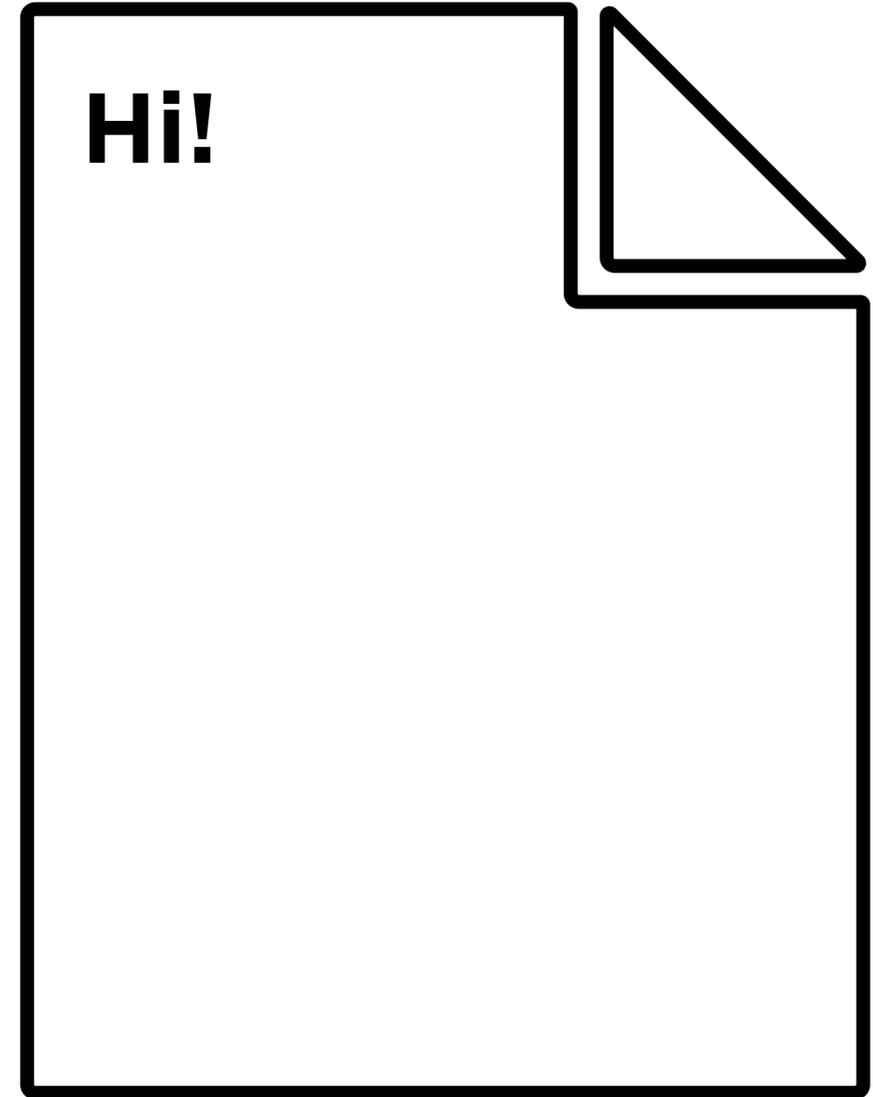
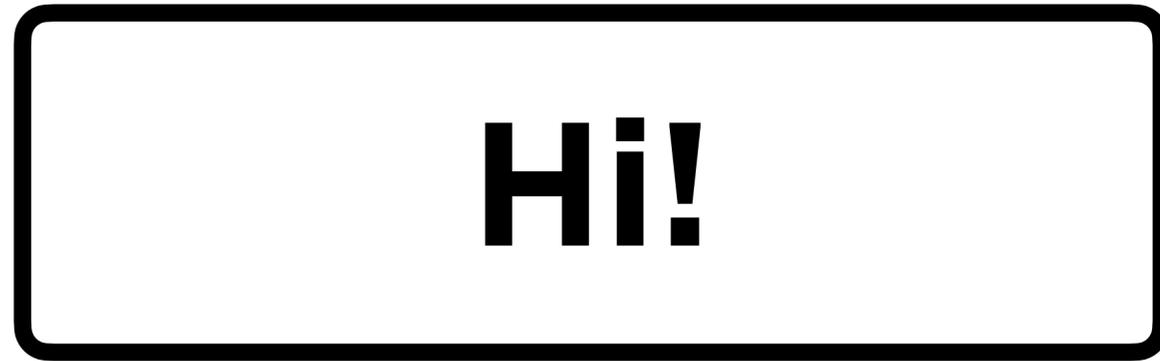
text



hi.txt

Reading data

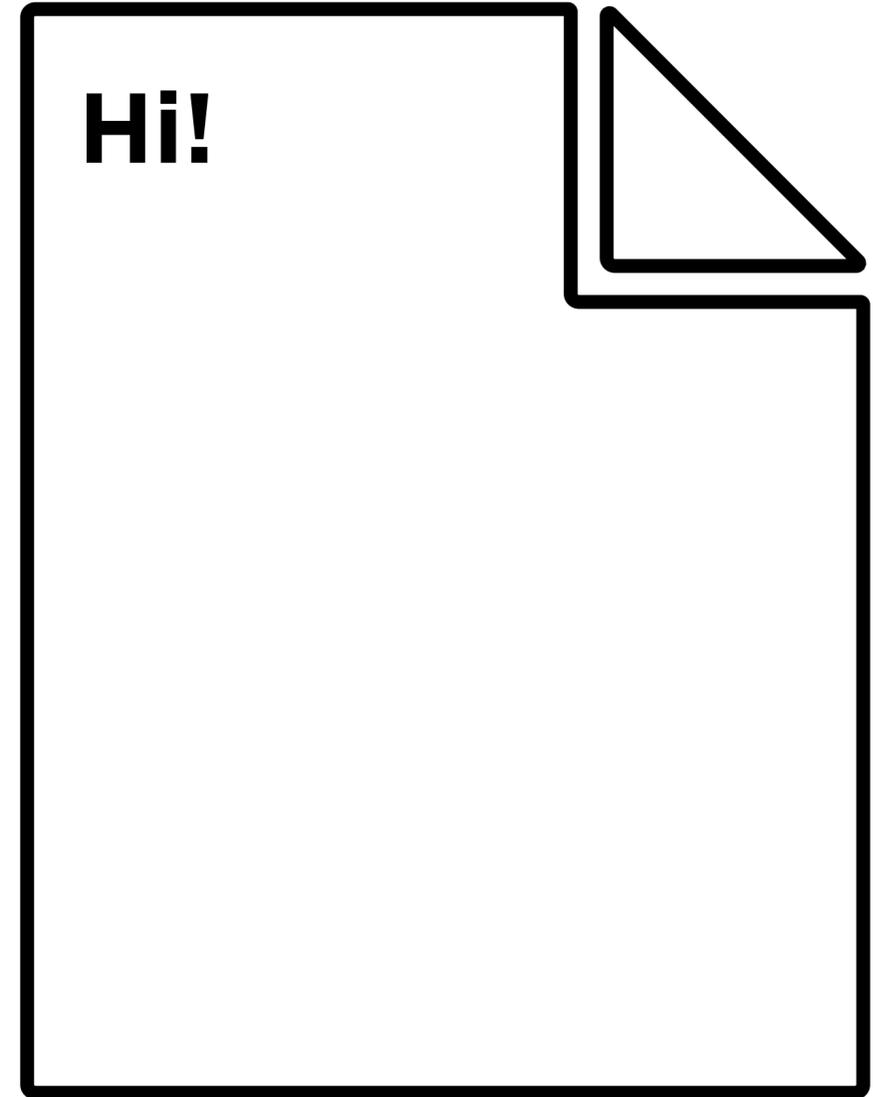
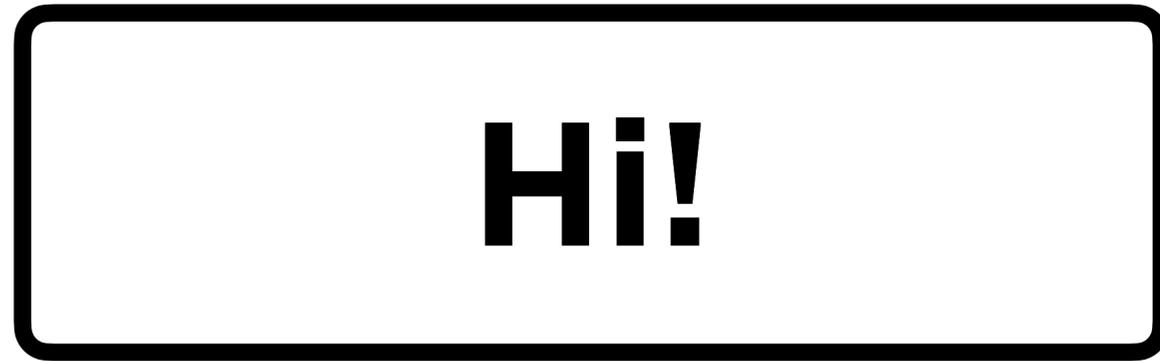
text



hi.txt

Reading data

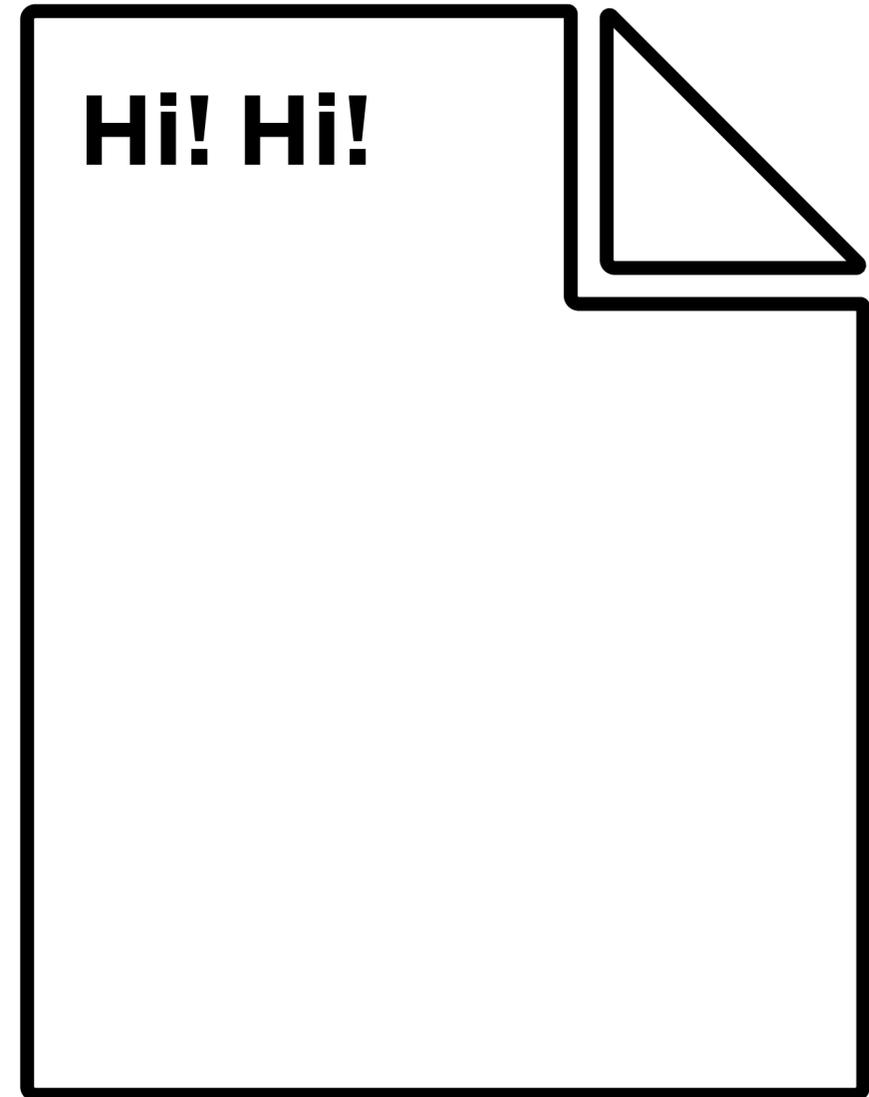
text



hi.txt

Writing data

text



hi.txt

Writing data

Key Functions

- **fread** reads data from a file into a buffer*.
- **fwrite** writes data from a buffer* to a file.

*a buffer is a variable that can temporarily store some data from the file.

Thought Question

- If we want to read an entire file, why use a buffer?
- Or, why might you *not* want to read the entire file into memory at once?

Reading from a File

Questions to Answer

- **From where** are you reading?
- **To where** are you reading?
- **What size** is each block of data you want to read?
- **How many** blocks do you want to read?

Questions to Answer

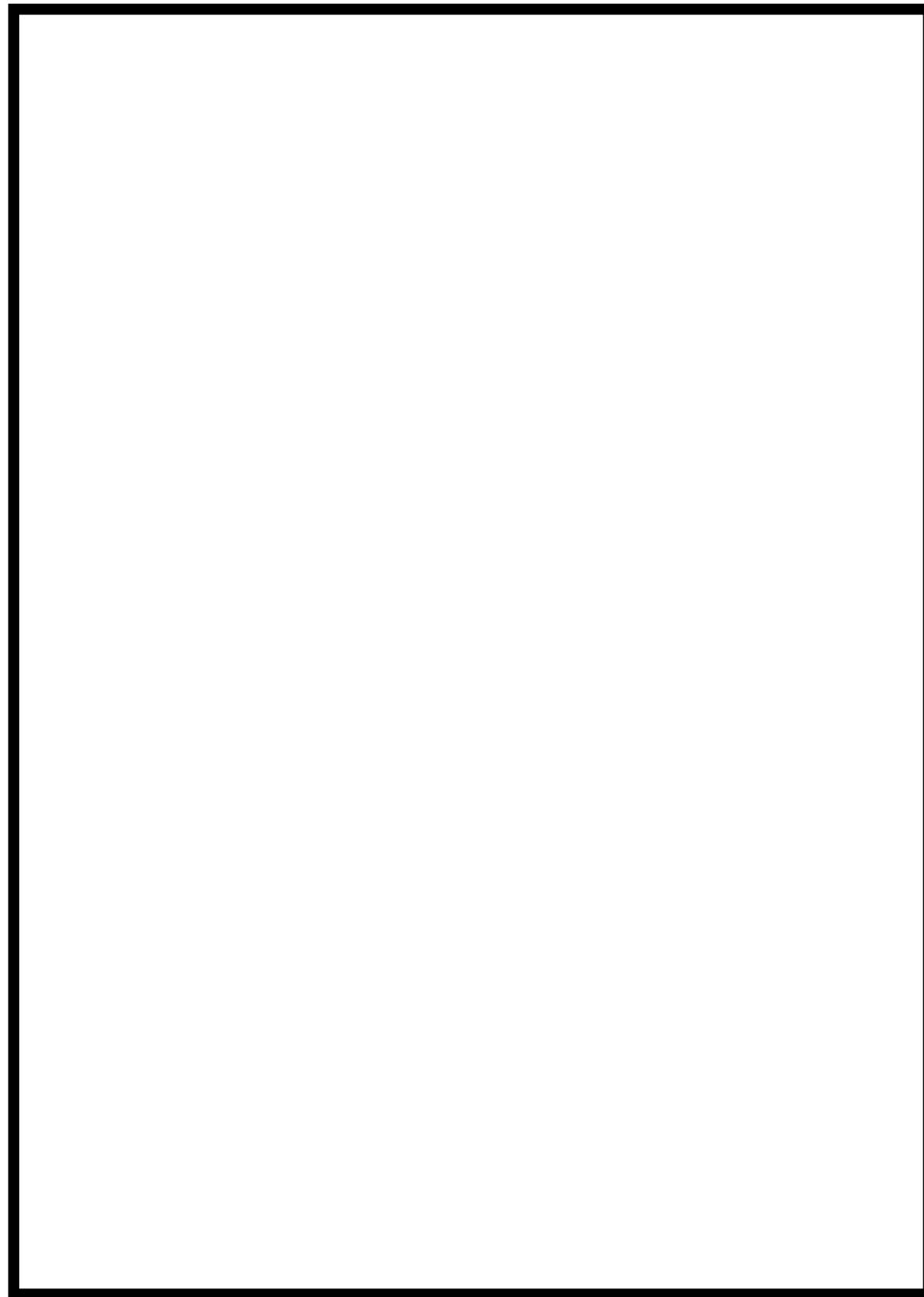
- **From where?**
- **To where?**
- **What size?**
- **How many?**

```
fread(..., ..., ..., ...);
```

```
fread(..., ..., ..., ...);
```

From where?

f →



```
fread(..., ..., ..., f);
```

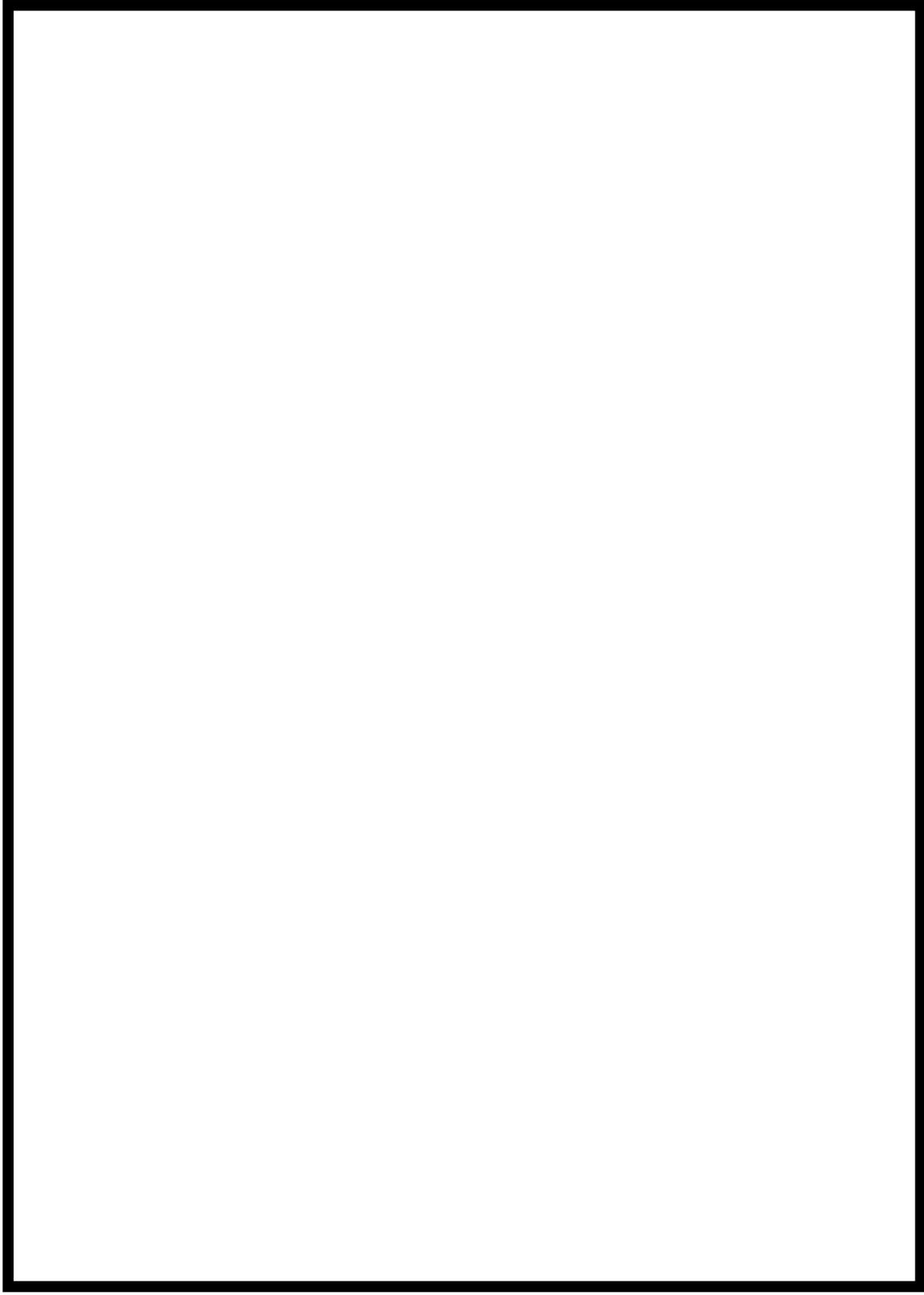
```
fread(..., ..., ..., f);
```

To where?

buffer



f →



```
fread(buffer, ..., ..., f);
```

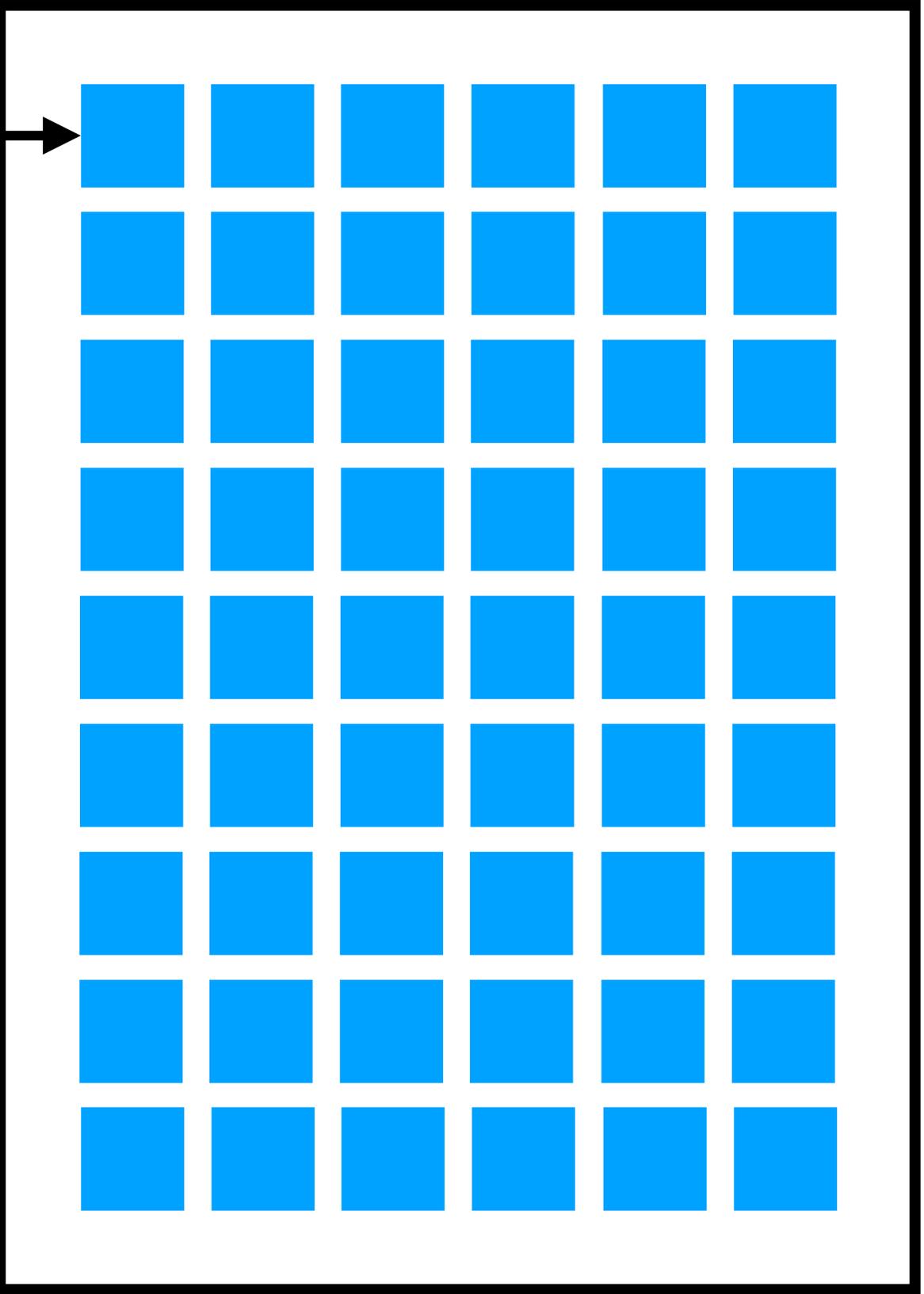
```
fread(buffer, ..., ..., f);
```

What size?

buffer



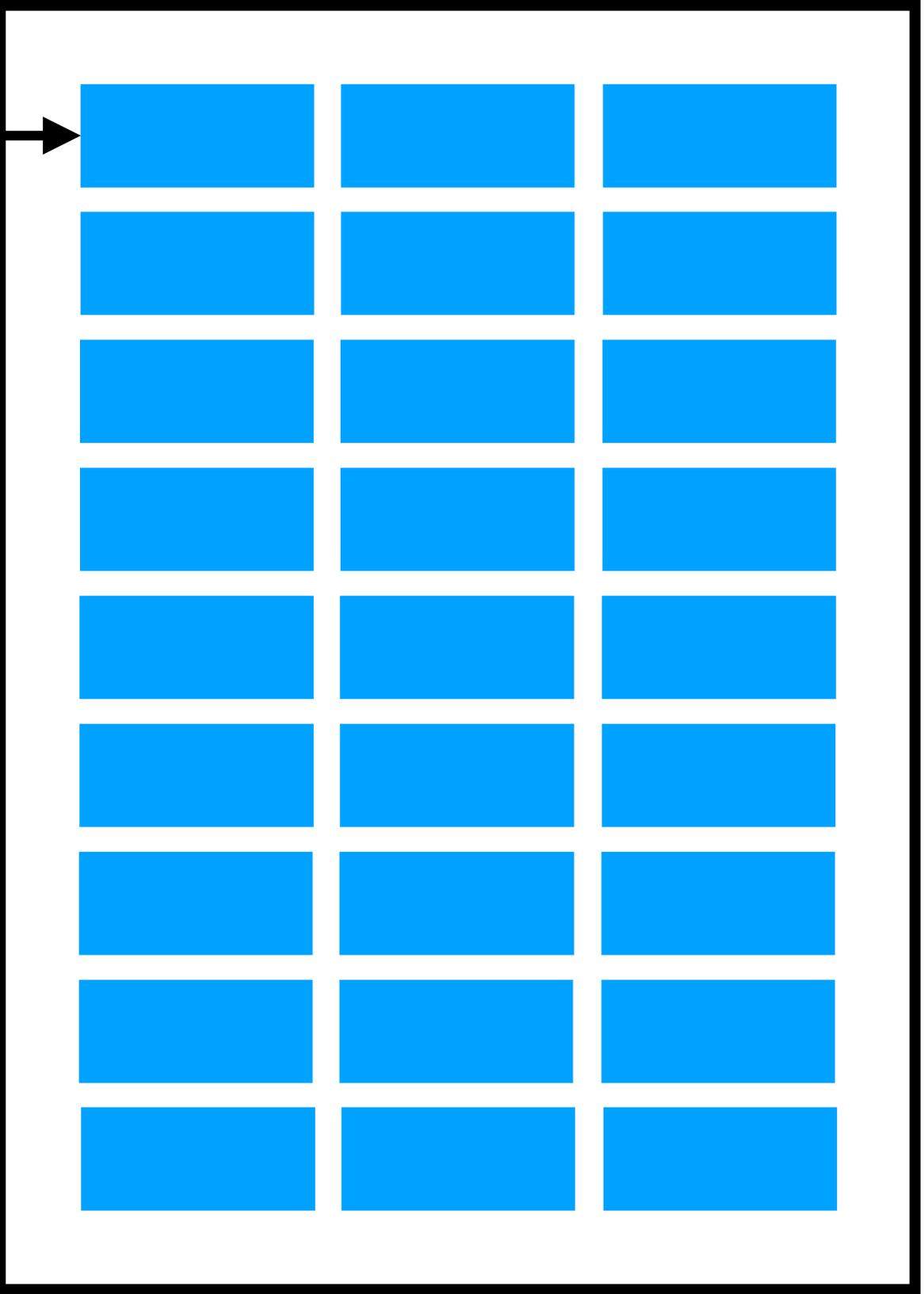
f



buffer



f



```
fread(buffer, 1, ..., f);
```

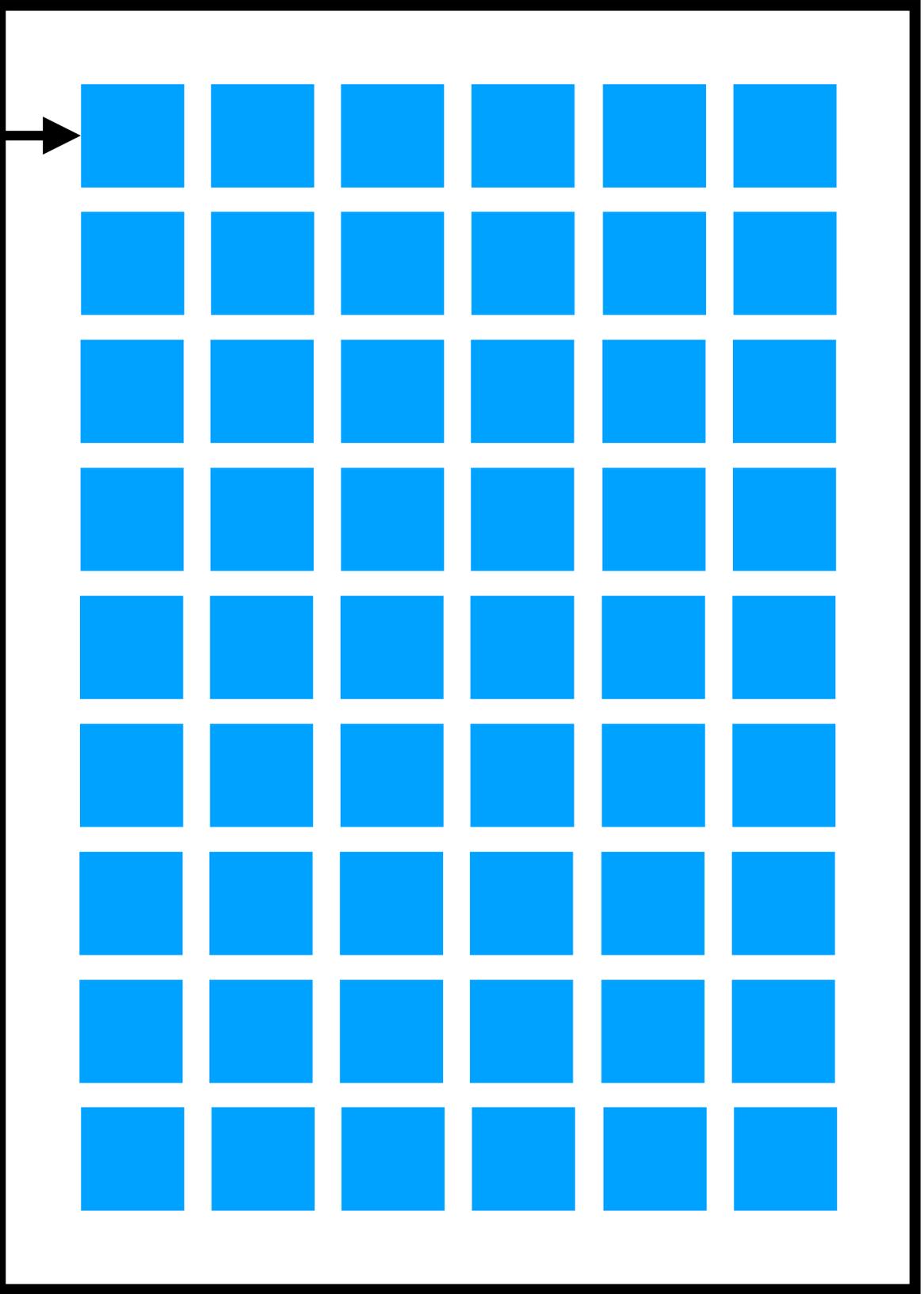
```
fread(buffer, 1, ..., f);
```

How many?

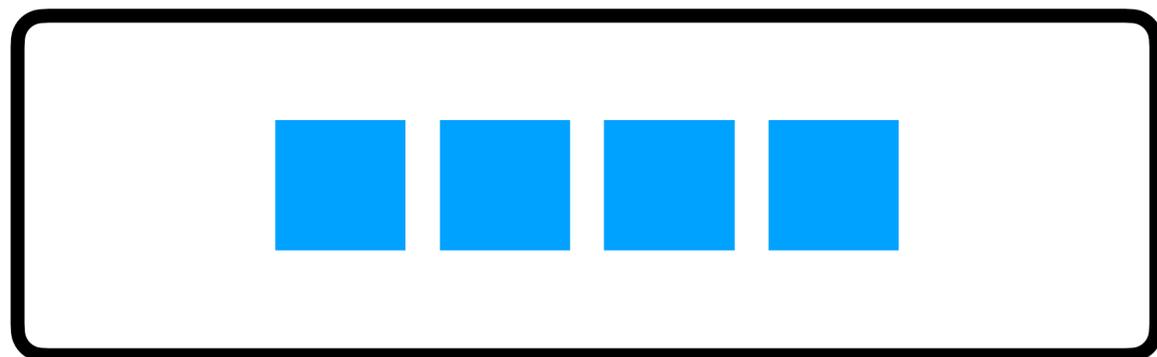
buffer



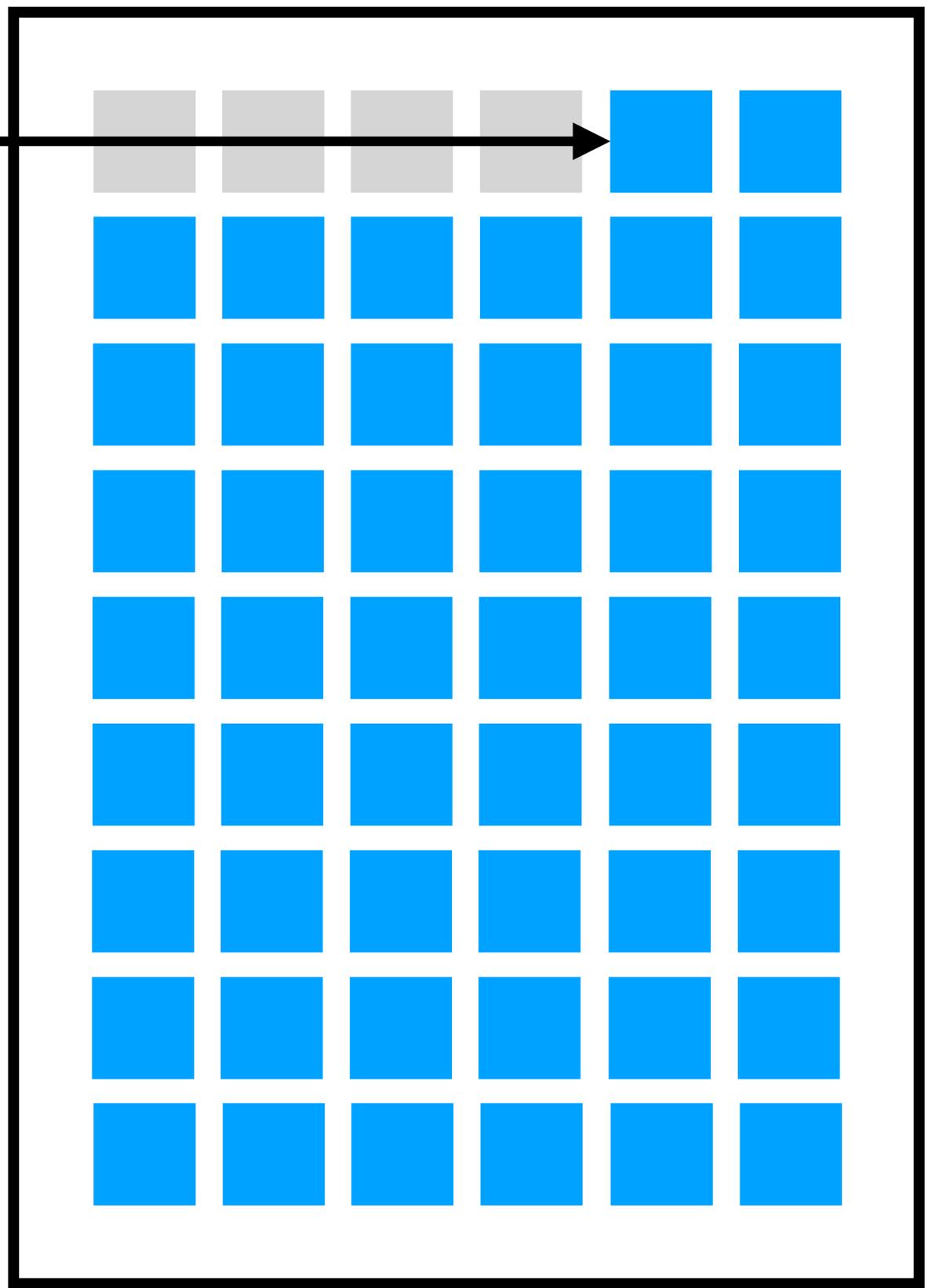
f



buffer



f



```
fread(buffer, 1, 4, f);
```

Writing to a File

Questions to Answer

- **From where** are you reading?
- **To where** are you reading?
- **What size** is each block of data you want to read?
- **How many** blocks do you want to read?

Questions to Answer

- **From where?**
- **To where?**
- **What size?**
- **How many?**

```
fwrite(buffer, 1, 4, f);
```

Practice with Reading

- Create a program, **pdf.c**, that opens a file given as a command-line argument.
- Check if that file is a PDF. A PDF always begins with a four-byte sequence, corresponding to these integers:
 - 37, 80, 68, 70

Volume

Filter

The week ahead

- **Submit Problem Set 4** by Sunday, February 26, 11:59 PM.
- Attend **office hours**.
- Complete <https://cs50.ly/studybuddy> to be paired with a classmate if you'd like!