This is CS50

Week 4
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)
A **variable** is a name for some value that can change.
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x50000000</td>
<td>...</td>
</tr>
<tr>
<td>0x50000004</td>
<td>...</td>
</tr>
<tr>
<td>0x50000008</td>
<td>...</td>
</tr>
<tr>
<td>0x5000000C</td>
<td>...</td>
</tr>
<tr>
<td>Address</td>
<td>Value</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>0x50000000</td>
<td>3</td>
</tr>
<tr>
<td>0x50000004</td>
<td>...</td>
</tr>
<tr>
<td>0x50000008</td>
<td>...</td>
</tr>
<tr>
<td>0x5000000C</td>
<td>...</td>
</tr>
<tr>
<td>Address</td>
<td>Value</td>
</tr>
<tr>
<td>-------------</td>
<td>-------</td>
</tr>
<tr>
<td>0x500000000</td>
<td>3</td>
</tr>
<tr>
<td>0x500000004</td>
<td>...</td>
</tr>
<tr>
<td>0x500000008</td>
<td>...</td>
</tr>
<tr>
<td>0x50000000C</td>
<td>...</td>
</tr>
</tbody>
</table>

int calls = 3;
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x50000000</td>
<td>3</td>
</tr>
<tr>
<td>0x50000004</td>
<td>0x50000000</td>
</tr>
<tr>
<td>0x50000008</td>
<td>...</td>
</tr>
<tr>
<td>0x5000000C</td>
<td>...</td>
</tr>
</tbody>
</table>

int calls = 3;
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x50000000</td>
<td>3</td>
</tr>
<tr>
<td>0x50000004</td>
<td>0x50000000</td>
</tr>
<tr>
<td>0x50000008</td>
<td>...</td>
</tr>
<tr>
<td>0x5000000C</td>
<td>...</td>
</tr>
</tbody>
</table>

```
int calls = 3;
```
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x50000000</td>
<td>3</td>
<td>int calls = 3;</td>
</tr>
<tr>
<td>0x50000004</td>
<td>0x50000000</td>
<td>int *p = &amp;calls;</td>
</tr>
<tr>
<td>0x50000008</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>0x5000000C</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
int *p = &calls;
int *p = &calls;
int *p = &calls;

type

p

0x5000000000
int *p = &calls;

value

p

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.

• You can use dynamic memory (e.g., with malloc).
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 vs. Passing by Reference
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}
```c
#include <cs50.h>
#include <stdio.h>

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

main

10 50

swap

10 50
```
```c
#include <cs50.h>
#include <stdio.h>

void swap(int a, int b) {
    int temp = a;
    a = b;
    b = temp;
}
```
```c
#include <cs50.h>
#include <stdio.h>

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

**main**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>50</td>
</tr>
</tbody>
</table>
```c
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
main
    a   b
    10  50
```
```c
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```
```c
#include <cs50.h>
#include <stdio.h>

void swap(int *a, int *b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
}
```
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");
fclose(f);
Reading and Writing
Hi!

Reading data
Writing data
Hi!

Writing data
Key Functions

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

* a buffer is a chunk of memory that can temporarily store some data from the file.
Thought Question

- If we want to read an entire file, why use a buffer?
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?
fread(..., ..., ..., ...);
fread(..., ..., ..., ...);

From where?
fread(..., ..., ..., ..., f);
fread(..., ..., ..., f);

To where?
buffer
fread(buffer, ..., ..., f);
Questions to Answer

• What size is each block of data you want to read?
• How many blocks do you want to read?
fread(buffer, ..., ..., f);

What size?
fread(buffer, 1, ..., f);
fread(buffer, 1, ..., f);

How many?
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?
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
This is CS50

Week 4