Welcome to CS50 section!
As you get settled, please **write this all down**—it will be helpful:

<table>
<thead>
<tr>
<th><strong>My name</strong></th>
<th>Brandon Wang</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>My email</strong></td>
<td><a href="mailto:cs50@brandon.wang">cs50@brandon.wang</a></td>
</tr>
<tr>
<td></td>
<td><a href="mailto:brandonwang@college.harvard.edu">brandonwang@college.harvard.edu</a> for non-CS50 related</td>
</tr>
<tr>
<td><strong>Section materials</strong></td>
<td>brandon.wang/cs50</td>
</tr>
<tr>
<td></td>
<td>Please bookmark me now</td>
</tr>
<tr>
<td><strong>Office hours</strong></td>
<td>Mondays 4-5pm</td>
</tr>
<tr>
<td></td>
<td>CS50 at HSA, 67 Mt. Auburn Street #400</td>
</tr>
<tr>
<td><strong>Then, open your IDE and run:</strong></td>
<td>cd ~/workspace</td>
</tr>
<tr>
<td></td>
<td>git clone <a href="https://github.com/bw/cs50-section.git">https://github.com/bw/cs50-section.git</a></td>
</tr>
</tbody>
</table>
Today’s agenda

- About me and about CS50
- Resources you can use
- Keys for success in CS50
  - AKA “norms”

- Quick introductions

- Grading guidelines
  - Easy ways to raise your grade
  - Pet peeves of graders

- New material
  - Debugging
  - Arrays
  - Functions
  - Command line arguments

- Pset 2 review

- Questions
About your shiny new TF

- Brandon Wang
- Sophomore, Lowell House
- Statistics, Government, and Computer Science
- Houston, TX and New England
- brandonwang@college.harvard.edu
My background

- I took CS50 too
- Background in web and full-stack development (What does that even mean?)
- Primary background in HTML/CSS/JavaScript/jQuery, PHP, SQL
- Some background in C, Python, and a whole bunch of other tech
- (Most programming languages are very similar!)

- Talk to me about startups, edtech, ideas, and more
- Happy to grab meals with any of you
CS50 overview

Newly designed this year to be more approachable:

- **Starting with C**
  - Foundations of programming

- **Adding in some Python** (2-3 weeks)
  - Useful programming language for variety of needs
  - Data science applications

- **Ending with JavaScript** (1 week)
  - The dynamic web
  - A fairly different language, but very necessary to know
CS50 overview

CS50 is also about much more:

- **Fundamentals of modern computing**
  (How does the Internet work?)

- **Logical, quantitative, and procedural thinking**
  (How should I approach a problem?)

- **A broad overview to programming as a whole**
  (How is coding for the Internet different? How are they the same?)
My personal opinions

- CS50 as an overview to software
- CS50 as a gateway to computer science
- CS50 as an introduction to programming
- CS50 as an introduction to software engineering
- CS50 as a quintessential Harvard experience
  - Relax and try to enjoy it
  - But understand it will be stressful at times
    - And be okay with that. You’re taking CS50 to learn something
About getting help

- You should always feel comfortable getting help.

- **Course-wide resources:**
  - Big office hours (Wed, Thurs, Sun) (Widener, Northwest)
  - Small office hours (Every day) (HSA)
  - Online resources
  - Google

- **Resources from me:**
  - Section--first line of attack
  - Email and office hours
Section with me/others

● Should you go to section? YES!

(Please clap come to section)

● Section is better for everyone when more people attend

● I promise to make it as helpful as I can... This is not lecture

● This is a class you can fall behind on; don’t let that happen

● Shows initiative to me (your grader)
Office hours with me

Every TF grades slightly differently

- **Small office hours (HSA)**
  - With me: Monday 4-5pm
    - Students in this section receive priority during this slot
  - Other TFs: Most of the day, 7 days a week

- **Big office hours**
  - For everyone in the course. Come work on psets with others
  - Do this earlier rather than later in the week
Emailing me

Questions, comments, compliments, complaints:

- **TO:** cs50@brandon.wang
- **FROM:** Your Harvard email address (or the email you used to register for CS50)

- **Why?**
  - Keep track of emails with you easily
  - Keep track of which emails are about CS50
  - Helps me refer back to our conversations at end of term
Succeeding in this course

- TFs are students too-- please don’t overwhelm us

- Course-wide resources are better staffed this year, but they may still be frustrating.
  - Take advantage of your friends and dorm/entry mates
  - Work on psets together
  - Work on psets in office hours and in public places
  - A lot of people take this class
Today’s agenda

- About me and about CS50
- Resources you can use
- Keys for success in CS50
  - AKA “norms”

- Quick introductions

- Grading guidelines
  - Easy ways to raise your grade
  - Pet peeves of graders

- New material
  - Debugging
  - Arrays
  - Functions
  - Command line arguments

- Pset 2 review

- Questions
Grading guidelines
Turning in your pset

● Never turn in a pset late
  ○ 1 min late = 0 credit, not even partial (CS50 policy, not mine)
  ○ Give yourself ample buffer time
    ■ You will still get feedback from me for late psets

● Must not: Have excuses as comments in psets
  ■ Always OK: “I didn’t quite understand this part of the pset”
  ■ Annoying: “I had a ton of work last night so sorry about this whole file!”

○ Desperate to include an excuse? Email to me instead
  ■ No obligation to take pity on you

○ Optional: Include a joke or pun at beginning of pset
Style

- Style takes so little time and is so important. **Get this right.**
- For most engineers (including me), a **massive deal-breaker**

- You should ALWAYS follow these rules (more on them now)
  - Indentation
  - Proper commenting
  - Intuitive naming
  - Logical flow
  - Abstraction
    - We will cover functions today
Style → Always indent properly

Either is OK, but be consistent:

```c
if (fruit = "Apple") {
    printf("You're healthy\n");
}

if (fruit = "Apple") {
    printf("You're healthy\n");
}
```

Please don’t do this:

```c
if
    (fruit = "Apple") {
        printf("You're healthy\n");
    }

if (fruit = "Apple") {
    printf("You're healthy\n");
}

if (fruit = "Apple") {
    printf("You're healthy\n");
}
```
Style → Always indent properly

- Indentations help us understand the structure of your code
- In C, indentations are for humans, not computers
  - (In Python, later in the term, they will matter for computers too!)
- Not indenting things consistently is frustrating for everyone
- **Must**: Always indent your code properly
  - It takes 2 minutes and your grade will go up
Style → Always indent properly

Either is OK, but be consistent:

```c
if (fruit = "Apple") {
    printf("You're healthy\n");
}

if (fruit = "Apple") {
    printf("You're healthy\n");
}
```

Please don’t do this:

```c
if
    (fruit = "Apple") {
        printf("You're healthy\n");
    }

if (fruit = "Apple") {
    printf("You're healthy\n");
}
```
int main(int argc, char *argv[]) {
  ...

  while (x == y) {
    something();
    something_else();

    if (some_error)
      do_correct();
    else
      continue_as_usual();
  }

  finalthing();
  ...
}

Style → Always indent properly

- Official CS50 style guide: https://manual.cs50.net/style/

- I prefer start brackets on the same line as the control
  - You will not lose points if you do not do this

```c
if (fruit == "Apple") {
  // Doing something here
  printf("You're healthy\n");

  return true;
}
```
Style → Comments

- **Helpful in context**
  - Better to explain with variable naming and clear code, rather than to write a comment

- **Err on side of more comments** if unsure
  - (Especially for less comfortable coders)
  - Comments might make the difference between partial credit and no credit at all

- Don’t be excessive, don’t comment every line
Style → Comments

Good to have:

- Commenting tricky bits
- Magic numbers
- Unfamiliar libraries
- Clever logic
  - Don’t be clever!

Unnecessary:

- Control structures
- Basic definitions
- Comments for the sake of commenting
// Convert Fahrenheit to Celsius.
float c = 5.0 / 9.0 * (f - 32.0);

// Define the num_apples variable
int num_apples = 4;

// I didn’t put a space at the beginning!

/**
 * I am a multiline comment!
 * Hi!
 */
Style → Intuitive naming

- Variable names should make sense
- Contextually identify its type
- Long variable names are generally okay--nobody cares!

- **Integers**
  - num_apples, num_people_in_line

- **Strings**
  - first_name, last_name, address

- **Booleans (true/false)**
  - is_turned_on, has_activated_account

- **Lists and arrays**
  - apples, people_in_line
Style → Logical flow

- We will learn more about this as the course progresses.

In general:
- Be intuitive about the ordering of your code
- Organize things into visual blocks
- Limit the number of loops you do
Style → Abstraction

- Functions, functions, functions
- You should ideally never copy-paste code
- This will become increasingly important
Key takeaways for style

You will earn points if--

●  Your code is easy to understand and read through
●  You segment it intuitively
●  You abstract out sections and utilize functions and loops

You will lose points if you do not--

●  Indent properly and consistently (inexcusable!)
●  Comment your code properly (inexcusable!)
●  Name your variables confusingly
Today’s agenda

- About me and about CS50
- Resources you can use
- Keys for success in CS50
  - AKA “norms”
- Quick introductions
- Grading guidelines
  - Easy ways to raise your grade
  - Pet peeves of graders
- New material
  - Debugging
  - Arrays
  - Functions
  - Command line arguments
- Pset 2 review
- Questions
This week’s content
(New material)
Today in section

You should understand these concepts before starting pset 2:

- Debugging
- Arrays
- Functions
- Command line arguments
- ASCII
- Modulo (%)
Debugging

Tools at your disposal:

- Simple debugging
  - printf
  - eprintf

- Smart debugging
  - Debuggers
  - (Later in term, other tools)

- CS50 tools (help50)
  - So extremely and ridiculously unrepresentative of real coding
Simple debugging

- Is this code being run?
- Is this even working?
- What is this number?

- Low hassle and easy

- eprintf
Smart debugging

● debug50
  ○ Step into
  ○ Step over
  ○ Display variables
  ○ Change variables’ values

● Let’s give it a shot together
Arrays

How do you make an array?
Arrays

How do you make an array?

<datatype> <name>[<size>];

- char alpha[26];
- int score[5];
Arrays

How do you initialize an array?
Arrays

How do you initialize an array?

```c
int score[0] = 0; // zero index all arrays!
int score[1] = 1;
int score[2] = 2;

int score[] = {0, 1, 2}; // size based on the number of entries
```
Arrays

What are strings?
Arrays

What are strings?

- Without getting into the complexities...
  
  - **Strings** = **Arrays of characters**

- For now:
  
  - You can index into strings like any other array
  - `string s = "brandon wang"
  - `s[0]?`
  - `s[4]?`
  - `s[7]?`
  - `s[500]?`
Arrays

- How many things in an array?
  - You pick; you remember
  - To get it back: `int size = sizeof array / sizeof array[0];`
Arrays

Section exercise:

1. Create an array that has your name
2. Iterate over its members
3. Give me the corresponding ASCII integer for the letter
Functions

- Functions are black boxes
- Think math
Functions

- Functions are black boxes
- Think math
- By definition, functions:
  - (1) take something in [parameters],
  - (2) do something [methods], and finally
  - (3) spit out an answer [return value].
Functions

Why use functions?

● Simplification
  ○ Easier to write smaller pieces of code
  ○ Easier to use smaller pieces of code

● Organization
  ○ Breaking code into subparts is helpful

● Reusability
Functions

One function everyone has seen already is `int main(void)`
Functions

One function everyone has seen already is `int main(void)`

- `int` is the return type
- `main` is the name of the function
  - Every program needs a `main()` function: it signifies to the computer where to start running your code
- `void` is the parameter, which, in this case, is nothing
Functions

Variable scope?

- Bring along things you need
- Keep your workspace clean
Command line arguments

- argc
- argv
Command line arguments

- argc
- argv

If, for example, in my terminal window I type in:
  - ./mario 8
  - ./luigi 82 carrot bob
Pset 2 review