CS 113: Computer Science I

Instructor: Kelly Shiptoski

Week: 1, Lecture: 1

Agenda

Introductions

Admin

What is Computer Science?

Linux Directories & Files

Terminal / Command Line

Prof. Kelly Shiptoski

PhD from UPenn.

Worked at multiple startups post graduation.

Research areas: operating systems, containers / virtual machines, determinism, reproducibility.

Class Resources

Textbook: Think Java (don’t buy it, I will post a link to a free version in our class slack / website).

Slack: Announcements, links, etc.

Website: Policies, syllabus, schedule, office hours info, etc.

Gradescope: For submitting homework assignments.

Lab: Park 231.

Links

Textbook: https://greenteapress.com/thinkjava7/thinkjava2.pdf

Course website: https://brynmawr-cs113-f26.github.io/website

Slack: https://brynmawr-cs113-f26.slack.com

Look Out for Invites!

Watch for invites to:

- Our class slack.
- Gradescope.

Course Format

Lectures

- Combination of slides + live coding / debugging

Labs

- Code practice
- Worksheets
- Occasional quizzes

Assignments

- Weekly due on Wednesdays (No homework due week 1)

Exams

- Midterm exam
- Final exam

Workload

At least 10 hours a week.

Weekly homework assignments.

Weekly labs.

100-level class **does not equal** a lighter workload.

Grade Breakdown

Final Exam: 35%

Midterm Exam: 35%

Homework: 10%

Lab Practice / Quizzes: 15%

Lab / Lecture Attendance & Participation: 5%

Office Hours

Prof. Shiptoski: Tuesdays 4-5pm & by appointment

Proposed TA Office Hour Schedule:

- Every day M-F 3-5pm

Late Policy

No extensions.

Assignments will only be accepted for 48 hours after the deadline.

Each day the assignment is late incurs a penalty of 20%.

Sick Policy

Prof Shiptoski is on **immunosuppressing medications**.

PLEASE wear a **mask** if you are feeling **sick** or experiencing symptoms.

Office hours will be **flexible**. If you are sick and need help, we can set up a **Zoom** call.

If you are sick and need to miss a quiz or lab, contact me to **reschedule it**.

How to Succeed

- Do the work.
- Do not procrastinate.
- Attend lectures and labs.
- **ASK QUESTIONS!**
  - Labs / Lectures / Office Hours / By appointment
  - Slack (we will try to respond within 24 hours, M-F)
  - **Seriously, ask me anything. I am here to help!**

Let’s Get Started

Computer Science (in this class)

We will learn to:

- Break down problems into solvable components.
- Learn how to instruct and command a computer to solve a complex problem.

Algorithms vs Programs

Algorithm: step-by-step set of rules to solve a problem or complete a task.

Program: implementation of an algorithm that the computer understands.

- Unambiguous
- Expressive (communicate many ideas)

Simplest Java Program

```
class Hello {
  public static void main(String[] args) {
    System.out.println(x: "Hello World!");
  }
}
```

But how can the computer understand Java code?

As it exists in your file on your computer... it can't.

To run a Java program, we first must **compile** it.

Compiling a Java Program

Converts java file (`.java`) to a file that the computer understands (`.class`)

`javac filename.java → filename.class`

Compiling a Java Program

The compiler is your friend. It will show you errors before you even run your code.

But it can’t show you **every** error.

Some errors can only be discovered when a program runs (at **runtime**).

Compiling a Java Program

Converts java file (`.java`) to a file that the computer understands (`.class`).

```
javac filename.java
```

`javac` is the **command line program**.

Compiling a Java Program

Converts java file (`.java`) to a file that the computer understands (`.class`).

```
javac filename.java
```

`filename.java` is the **argument**.

Running a Java Program

`java filename`

(Don’t include the `*.class` file extension).

But where do we actually call `java filename`?

Command Line Applications

No graphical component, all text.

Often expect the user to pass data to the application.

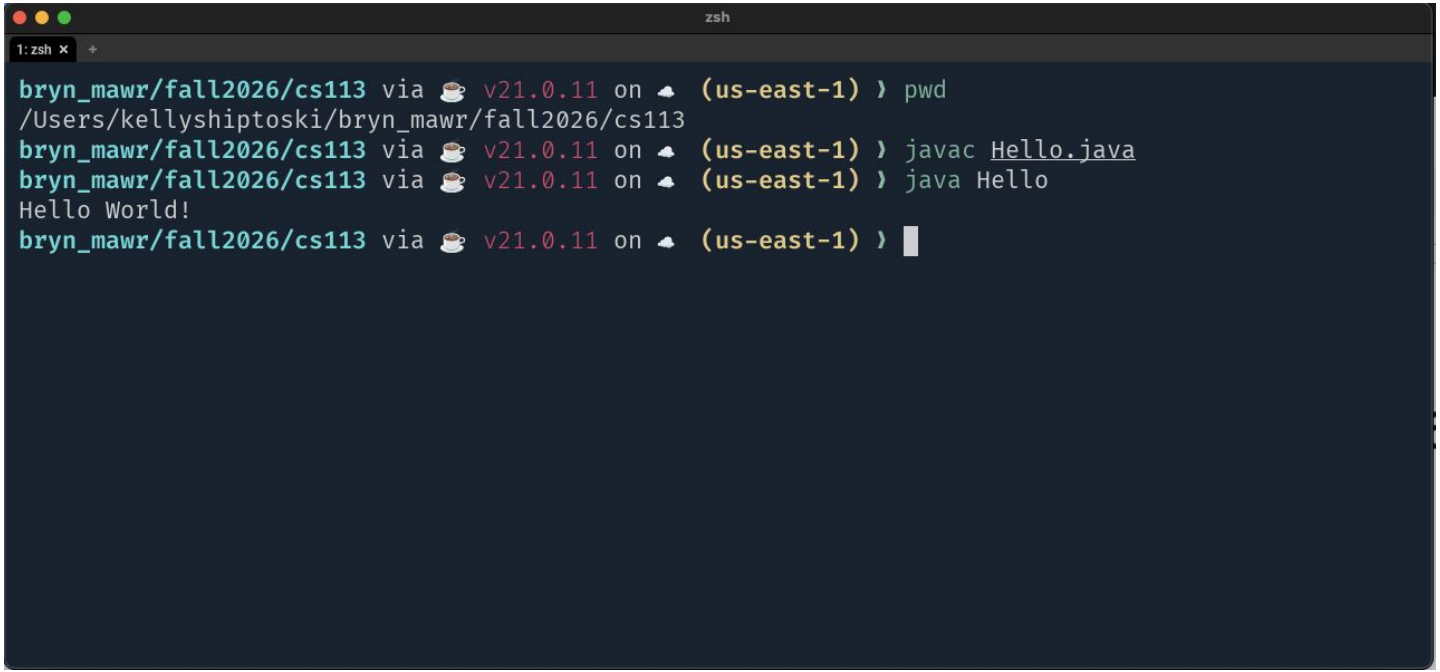
The **terminal** is a command line application.

Terminal & Command Line

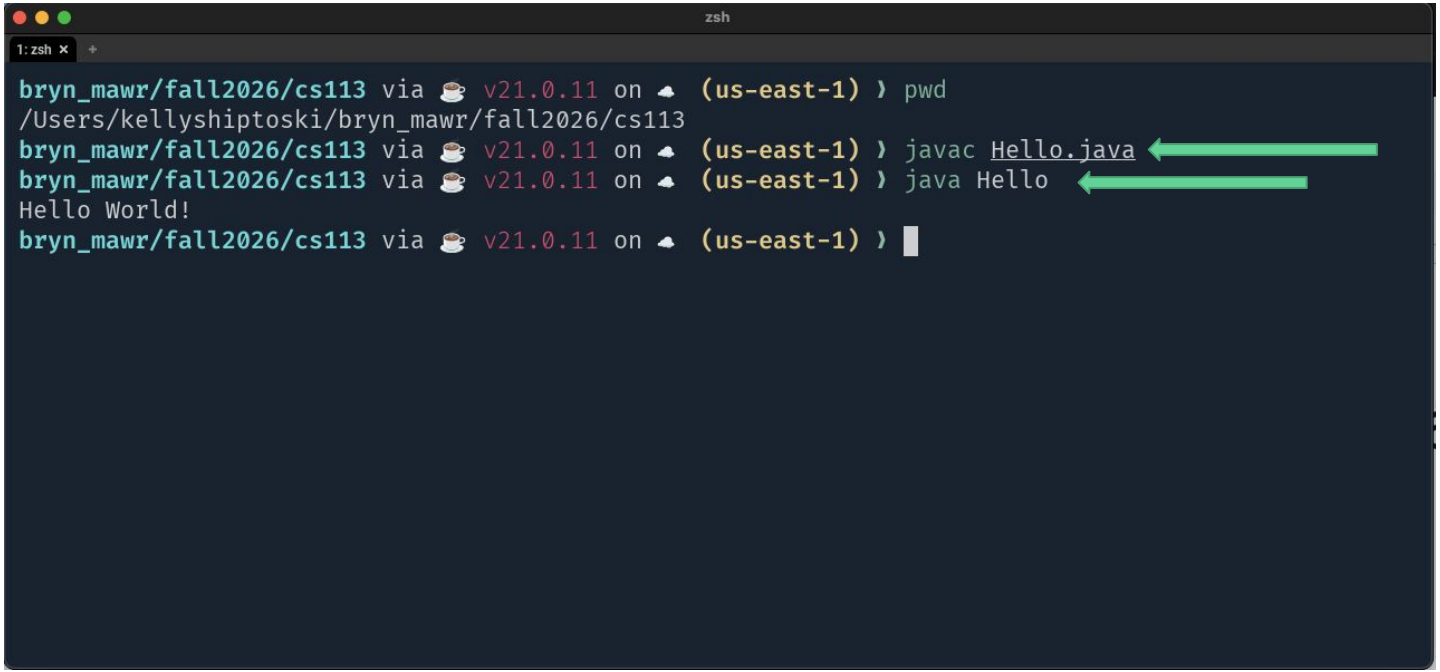
Essential for success in this course:

- Learning terminal commands.
- Getting comfortable and confident with using the terminal.
- We'll do LOTS of practice in labs!

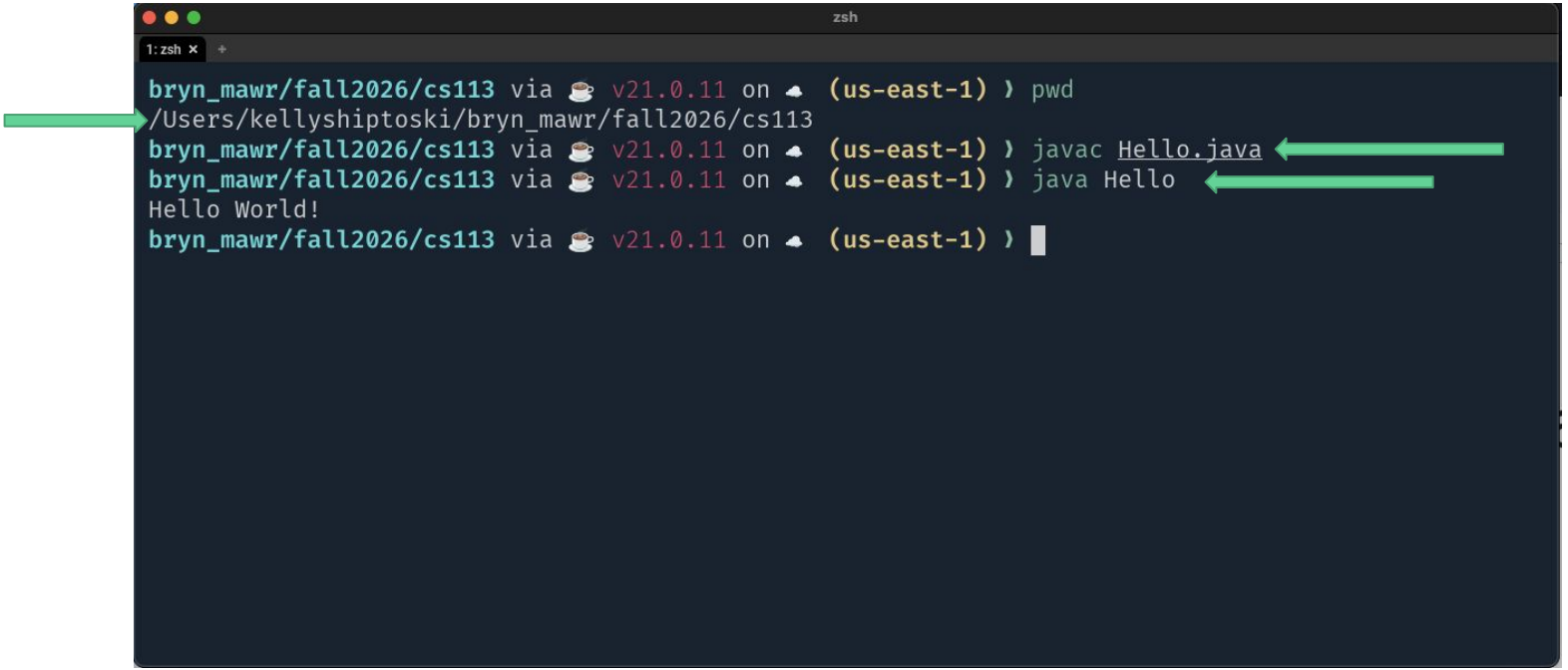
Look Closely...



Look Closely...



Look Closely...



The terminal has no graphical interface.

Instead, we issue it text-based commands.

```
fall2026/cs223/lecture2 via C v21.0.0-clang via ☕ v21.0.11 on ☁  (us-east-1) took 1m26s ❯ vim hello.c
fall2026/cs223/lecture2 via C v21.0.0-clang via ☕ v21.0.11 on ☁  (us-east-1) took 2s ❯ pwd
/Users/kellyshiptoski/bryn_mawr/fall2026/cs223/lecture2
fall2026/cs223/lecture2 via C v21.0.0-clang via ☕ v21.0.11 on ☁  (us-east-1) ❯ cd
~ on ☁  (us-east-1) ❯ cd bryn_mawr
~/bryn_mawr via ☕ v21.0.11 on ☁  (us-east-1) ❯ pwd
/Users/kellyshiptoski/bryn_mawr
~/bryn_mawr via ☕ v21.0.11 on ☁  (us-east-1) ❯
```

Command Line Arguments

Many programs expect the user to provide data.

Example: commands in terminal.

```
javac Hello.java
```

```
java Hello
```

Command Line Arguments

```
public static void main(String args[]) {
    String firstArg = args[0];
    System.out.println(firstArg);
}
```

Note: indices start at **0** not **1**.

Directories & Files

Computer data is structured as a hierarchy of files. Everything’s a file!

Directories (folders) can contain files and other directories.

Programs are organized in directories (many programs include > 1 file).

Navigating a Directory

`ls`: list files in current directory.

`cd`: change current directory.

`pwd`: print path to the current directory.

`mkdir`: make a directory.

Special Directories

`..`	Parent directory
`.`	Current working directory
`/`	Root directory
`/home/username`	Your home directory
`~`	Your home directory

Working with Paths

Two kinds of paths:

- Absolute paths
- Relative paths

Absolute: `/Users/kellyshiptoski/bryn_mawr`

Relative: `../../Users`

Path Exercise

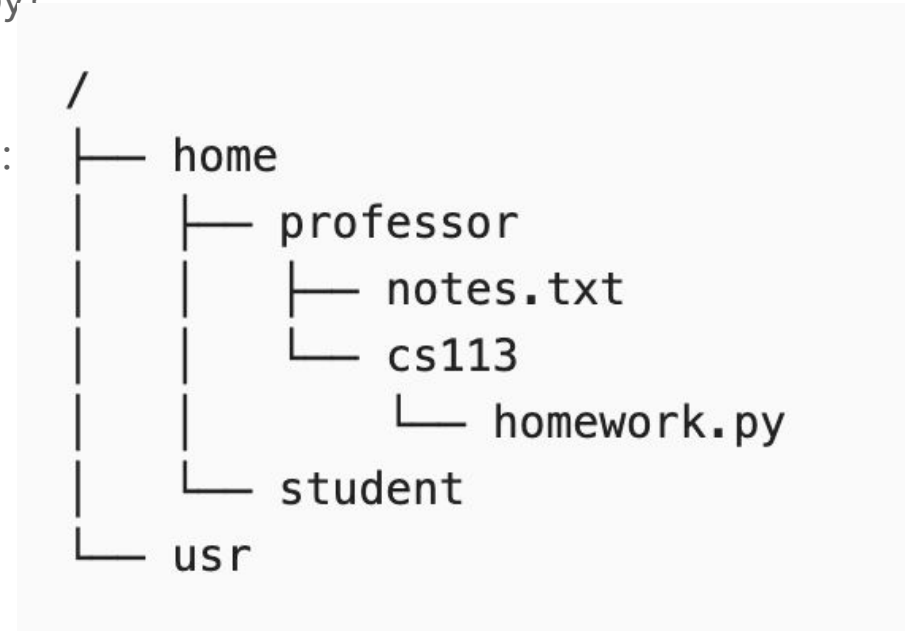
What is the absolute path of homework.py?

What is the relative path of notes.txt from:

- The */cs113* directory?
- The */usr* directory?

What is the relative path of the /usr directory from:

- The */cs113* directory?
- The */professor* directory?



When Things Go Wrong

When code doesn't work correctly, this is called a **bug**.

Syntax bug: some part of the code doesn't follow the language's syntax rules.

Semantic bug: some logic in the code isn't behaving the way we expect.

What’s wrong here?

```
public clas SyntaxErrors {

   public static void main(String args) {
      System.out.println(“Hello World);

   }
```

Storing Data

Data Types

Way to store information in programs.

Tells the compiler how the programmer intends to use a particular piece of data.

Data Type Examples

`int` - whole numbers (ex: `7`)

`double` - decimal numbers (ex: `1.25`)

`String` - represents text (ex: `"Kelly"`)

Variables

Allows us to give a piece of data a **name**.

`String greeting;`	Creates a variable called `greeting` that can store a `String`.
`int a, b, c;`	Creates variables `a`, `b`, and `c`, each of which can store an `int`.
`a = 3;`	Assigns the value of the existing variable `a` to 3.
`int d = 10;`	Creates an `int` variable called `d` and assigns its value to 10.

Variables

Allows us to give a piece of data a **name**.

`String greeting;`	Creates a variable called `greeting` that can store a `String`.
`int a, b, c;`	Creates variables `a`, `b`, and `c`, each of which can store an `int`.
`a = 3;`	Assigns the value of the existing variable `a` to 3.
`int d = 10;`	Creates an `int` variable called `d` and assigns its value to 10.

These are called **literals**.

Variables

Variables can be created and changed by:

- Declaration ➜ ex: int x;
- Assignment ➜ ex: x = 4;
- Declaration & Assignment ➜ ex: int x = 4;

Variables

`String greeting;`	Creates a variable called `greeting` that can store a `String`.
`int a, b, c;`	Creates variables `a`, `b`, and `c`, each of which can store an `int`.
`a = 3;`	Assigns the value of the existing variable `a` to 3.
`int d = 10;`	Creates an `int` variable called `d` and assigns its value to 10.

Best practice!

Properties of Variables

- Name
- Data Type
- Location
  - Where on the computer is the variable stored?
  - More on this later...

Example: `String greeting;`

Properties of Variables

- Name
- Data Type
- Location
  - Where on the computer is the variable stored?
  - More on this later...

Example: `String greeting;`

Data type

Properties of Variables

- Name
- Data Type
- Location
  - Where on the computer is the variable stored?
  - More on this later...

Example: `String greeting;`

Printing Variables

```
class Hello {
  public static void main(String[] args) {
    int a = 1;
    double b = 2.0;
    String blah = "thisisastring";

    System.out.println("a is " + a + ", b is " + b + ", and blah is " + blah);
  }
}
```

```
a is 1, b is 2.0, and blah is thisisastring
```

Printing Variables

```
class Hello {
  public static void main(String[] args) {
    int a = 1;
    double b = 2.0;
    String blah = "thisisastring";

    System.out.println(a);
    System.out.println(b);
    System.out.println(blah);
  }
}
```

```
1
2.0
thisisastring
```

Variable Examples

a	b	c

Variable Examples

```
int a, b;
```

a	b	c

Variable Examples

```
int a, b;
```

a	b	c
undefined	undefined	-

Variable Examples

```
int a, b;

String c = “Kelly”;
```

a	b	c
undefined	undefined	-

Variable Examples

```
int a, b;

String c = “Kelly”;
```

a	b	c
undefined	undefined	-
undefined	undefined	“Kelly”

Variable Examples

```
int a, b;
String c = “Kelly”;
a = 3;
```

a	b	c
undefined	undefined	-
undefined	undefined	“Kelly”

Variable Examples

```
int a, b;
String c = “Kelly”;
a = 3;
```

a	b	c
undefined	undefined	-
undefined	undefined	“Kelly”
3	undefined	“Kelly”

Variable Examples

```
int a, b;
String c = “Kelly”;
a = 3;
b = a;
```

a	b	c
undefined	undefined	-
undefined	undefined	“Kelly”
3	undefined	“Kelly”

Variable Examples

```
int a, b;
String c = “Kelly”;
a = 3;
b = a;
```

a	b	c
undefined	undefined	-
undefined	undefined	“Kelly”
3	undefined	“Kelly”
3	3	“Kelly”

Before Next Lecture

Complete Lab 1 (right after this lecture!).

Sign up on:

- Slack.
- Gradescope.

Homework 1 due next Wednesday night.