Week 1: Agenda

- ✔ Syllabus
- ✔ Data, Variables, Expressions, Functions
- ✔ Quiz#1
  - Examples

Announcements

- OH start today
  - See the calendar on the course website
- You should be able to access:
  - CS Academy
  - Gradescope
  - Discord

Data

Data Types

- Numbers (e.g., int, float): Whole numbers and decimals.
- Text (str): Strings of characters (e.g., "Hello").
- Boolean (bool): True or False.
- Others: Lists, tuples, dictionaries, etc.

How to know the type?

- type()

Example: Final Grade Policy

You must receive a 60% or higher (calculating using the appropriate weightings above) on in-class assessments in order to receive a D or higher in the course.

You must receive a 65% or higher (calculating using the appropriate weightings above) on in-class assessments in order to receive a C or higher in the course.

I used ChatGPT for my homeworks, and I scored

- An average of 12.5 on quizzes (after dropping lowest two)
- 61 on Exam 1
- 58 on Exam 2
- 62 on the final

Can I get a C? Can I get a D?

Variables

What are they?

Named storage for data

Why do we need them?

- Makes code readable and reusable
  - Easier to read and modify later if needed
- Enable Dynamic Input
    - Programs can adapt to different inputs without code changes
- Reusability and Flexibility
  - Update a value once instead of modifying multiple lines
  - It will be obvious later

Operations and Expressions

Expressions: Code that produces a value (e.g., 2 + 3).

Operators: Symbols for calculations or logic.

- Arithmetic: + - * / % ** //
- Comparison: == != > < >= <=
- Logical: and, or, not
- Precedence Rules:

```
result = 2 + 3 * 4
```

Important to know:

- Integer division (//)
- Modulo (%)
- Float and equality

Nice Tricks:

- Concatenation: 'Hello' + ' World' → 'Hello World'
- Repeat Strings: 'Hi' * 3 → 'HiHiHi'
- Playing with digits

Functions

What? Reusable blocks of code that perform tasks.

- Defining vs Using Functions

Why? Avoid repetition, organize code.

Data and Expressions: Types of problems

- Math Problems
  - Examples: `isRightTriangle`

Data and Expressions: Types of problems

- Math Problems
  - Examples: `isRightTriangle`
- Extracting Digits
  - Examples: `isSmallFair`, `isSmallPal (Quiz 1, Pitt S22)`

Data and Expressions: Types of problems

- Math Problems
  - Examples: `isRightTriangle`
- Extracting Digits
  - Examples: `isSmallFair`, `isSmallPal (Quiz 1, Pitt S22)`
- …
- Mixed

`isSmallFair(n)` (Pittsburgh F21)

- We will say that a value **n** is *"fair"* if it is an integer and it has the same number of even digits as odd digits (ignoring leading 0's). A *"small fair"* number is a fair number with exactly 4 digits.
- For example, 1083, 1081, and -1092 are each small fair numbers because each have two odds and two evens.
- With this in mind, and without using strings or loops, write the function `isSmallFair(n)` that takes a value **n**, that may or may not be an integer, and returns **True** if **n** is a small fair number, and **False** otherwise. **Do not crash if n is not an integer!**

`clockHour(currentHour, difference)`

- It's 10:00 (am/pm doesn't matter). Where will the hour hand be in 7 hours? In practice the hour will be 17, but we are restricted to show 5 since the hour hand only goes up to 12!
- Create a function called `clockHour(currentHour, difference)` that calculates the hour hand's position on a 12-hour clock after a certain number of hours have passed. The function takes two parameters: currentHour, an integer in the range 1–12 indicating the current hour hand position, and `difference`, an integer representing how many hours have passed since the current hour. The function should return the new hour hand position, ensuring that the result stays within 1–12. For example, if currentHour = 11 and difference = 2, the hour hand moves from 11 to 12, and then to 1, so the function returns 1.
- Example:
  - clockHour(1, 3) == 4
  - clockHour(8, 11) == 7
  - clockHour(4, 13) == 5
  - clockHour(4, 24) == 4
  - clockHour(12, 1) == 1
  - clockHour(11, 2) == 1
  - clockHour(6, 25) == 7
  - clockHour(12, 12) == 12
  - clockHour(3, 24) == 3
  - clockHour(1, 24) == 1

`timeInterval(t1,t2)` Quiz, Qatar F21

Write the function `timeInterval(t1, t2)` which, given two non-negative integers `t1`, `t2`, that encode two 24-hour times in the format `hhmm`, returns the time interval, in minutes, between those two times. If `t2 < t1`, you should assume that `t2` refers to a next day time. You can assume that `0 < hh < 24` is the hour, and `0 < mm < 60` are the minutes. If `hh > 0`, then mm is always a two-digit number. If `hh == 0`, then mm can be either a one or a two-digit number, depending on its value.

- 1503 is 15 hours, 3 minutes, or 3:03pm.
- 849 is 8 hours, 49 minutes, or 8:49am.
- 0 is 0 hours, 0 minutes, or 12:00am midnight.
- 59 is 0 hours, 59 minutes, or 12:59am.
- 101 is 1 hour, 1 minute, or 1:01am.

For example...

- `timeInterval(1400, 1545)` returns the time interval between 14 o'clock and 15:45 (same day) which is 105 minutes.
- `timeInterval(2359, 31)` returns the time interval between 23:59 and 00:31 (next day), which is 32 minutes.
- `timeInterval(31, 2359)` returns the time interval between 00:31 and 23:59 (same day), which is 1408 minutes.
- `timeInterval(1200, 0)` returns the time interval between noon and midnight (next day), which is 720 minutes.
- Hint: There are 1440 minutes in a day.