Objectives

- String Formatting
- Broader Issue: Autonomous Vehicles

FORMATTING STRINGS

Usability

- Users want out that is easy to read and understand
- Achieving that in a variety of circumstances requires a lot of customizability

Solution: `format` Method

- How to use:
  - `"templatestring".format(<tobeformatted>)`
- `templatestring` allow us to control how output is displayed to user
  - Examples:
    - Right, left justification
    - Number of decimals to display

Solution: `format` Method

- How to use:
  - `"templatestring".format(<tobeformatted>)`
- Semantics: creates a **formatted string**
  - Means "format the `templatestring`, using the format(s) specified by ***format specifiers*** on the corresponding replacement values"
  - Evaluates to/returns a `str` data type
- Typically used with print statements

Formatting Strings: `format` Method

- **`templatestring`** is a template for the resulting string with format specifiers instead of the values
  - For each format specifier in templatestring, should have a ***replacement value***

```
"{:.2f}".format(3.14159)
```

Evaluates to `"3.14"`

One format specifier in template string

Corresponding replacement value

  - Throws **`IndexError`** if not enough replacements for specifiers in `templatestring`

Format Specifiers

[ ] mean optional

- General format: {[field_name]:conversion}

index number of the argument,
i.e., which field in the template string

- conversion
  - conversion code of the data type

Code	Type
s	string
d	integer
f	float
e	floating point with exponent

Format Specifiers

[ ] mean optional

Conversion options

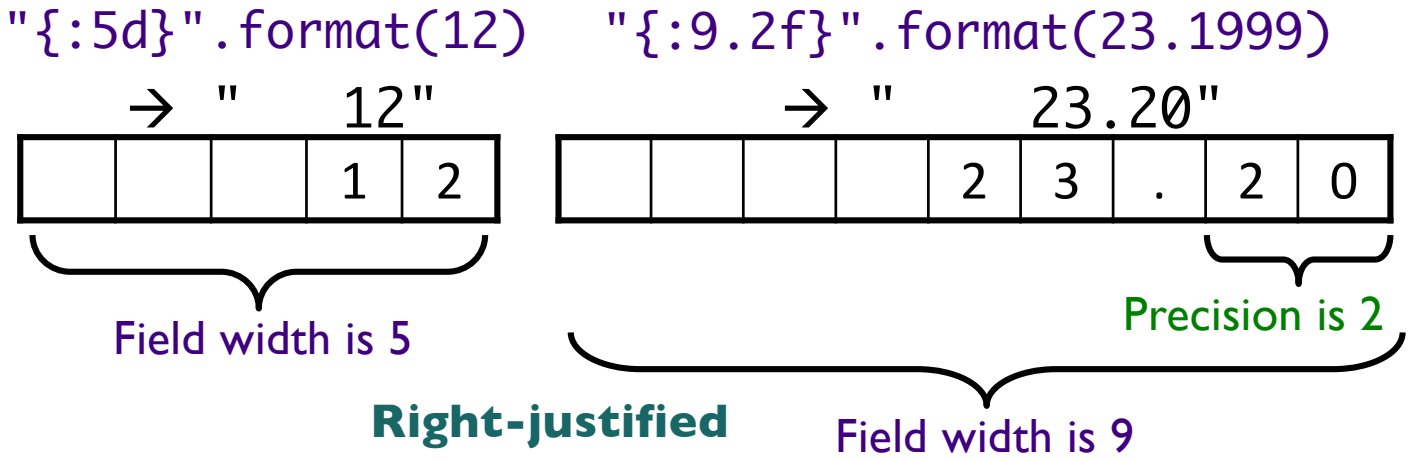
```
:[flags][width][.precision][code]
```

- flags:

Flag	Meaning
0	Zero fill to width
+	Adds a + sign before positive values
<	Left justify (default for strings)
>	Right justify (default for numbers)
^	center

- **width**:
  - *Minimum* number of character spaces reserved to display the entire value
  - Includes decimal point, digits before and after the decimal point and the sign
- **precision**:
  - Number of digits after the decimal point for **floating point** values

Example Format Specifiers

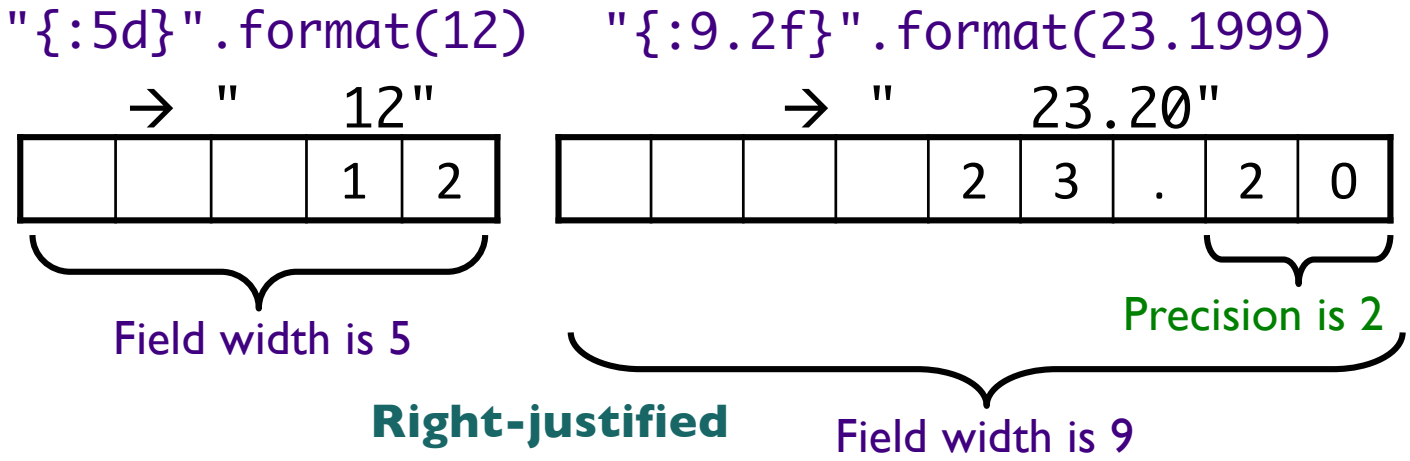


- What if precision is bigger than the decimal places?

- What if field width is smaller than the length of the value?

Any guesses? Try out in interpreter.

Example Format Specifiers



- What if precision is bigger than the decimal places?
  - Fills decimal with 0s
- What if field width is smaller than the length of the value?
  - String contains entire value

Formatting Practice

What is the resulting string?

```
x = 10
y = 3.5
z = "apple"
1. "{:6d}".format(x)
2. "{:6.2f}".format(y)
3. "{:06.2f}".format(y)
4. "{:6.2f}".format(x)
5. "{:^11s}".format(z)
6. "{:5d} {:<7.3f}".format(x, y)
```

String Formatting

- There is a lot more you can do with string formatting
  - This is a subset of the most commonly used functionality
- When formatting strings, consider
  - What is the data type of your data?
    - If a float, how many decimal places do you want?
  - How wide do you want the data to be?
  - What justification? Zero fill? Other flags?
- The answer to these questions help guide your creation of format specifiers

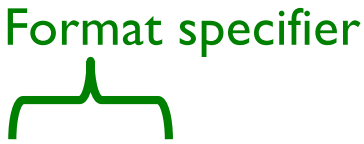
Using `format` Method in `print`

- You often want to format data within a broader context.

- Example: printing out money values
  - How do you want that data formatted?

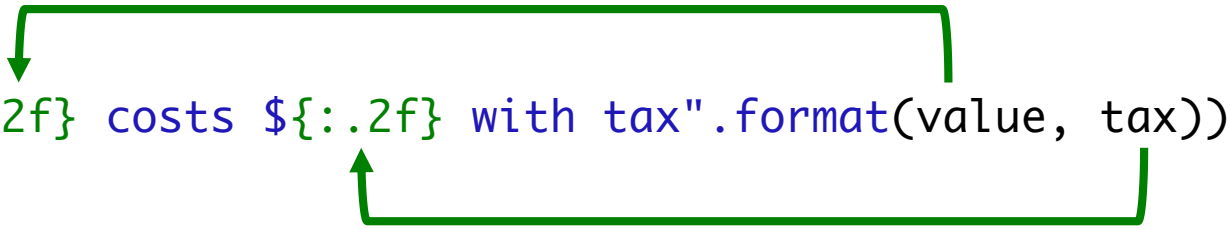
Using format Method in print

- Printing money values




```
print("Your item that cost ${:.2f}".format(value))
print("costs ${:.2f} with tax".format(tax))
```

Alternative:



```
print(
 "Your item that cost ${:.2f} costs ${:.2f} with tax".format(value, tax))
```

Using `format` Method in `print`

- Printing money values

Format specifier

```
print("Your item that cost ${:.2f}".format(value))
print("costs ${:.2f} with tax".format(tax))
```

Alternative:

```
print(
 "Your item that cost ${:.2f} costs ${:.2f} with tax".format(value, tax))
```

How is this different from using the round function?

Example: Printing Out Tables

- A table of temperature conversions

```
Temp F      Temp C      Temp K
------      ------      ------
-459.7      -273.1        0.0
   0.0       -17.8      255.2
  32.0         0.0      273.1
```

- If we want to print data in rows, what is the template for what a row looks like?
  - How do we make the column labels line up?
  - For above, not as simple as using tabs. Why not?

String Formatting Conclusion

- There is a lot more you can do with string formatting
  - This is a subset of the most commonly used functionality
- When formatting strings, create the format specifiers by asking:
  - What is the data's type?
  - How do I want it to look?

Broader Issue: Autonomous Vehicles

Pod 1	Pod 2	Pod 3	Pod 4	Pod 5
Aidan Chris Sam Sanil	Ben Charlotte Hollins James	Ethan Ryan Sophie Zuhaira	Aiden Georgia Jack Lizzie	Adhip John Matthew Thomas

Broader Issue: Autonomous Vehicles

- Why do I still assign an article from 2007?
- Autonomous Vehicles: love ‘em or loathe ‘em
  - As a passenger? As a driver (or passenger) in another car? As a pedestrian?
- What are the tradeoffs of autonomous vehicles?
  - What guarantees about the cars would you want from a company/government?
  - Are there situations that would be particularly difficult for software to handle that a person would be better equipped to handle?
    - What about the ethics of tough decisions?
- Compare the approach of AI that we’ve been reading about with social media with autonomous vehicles
  - What are the challenges/tradeoffs/consequences?

Autonomous Vehicles vs AI

- Autonomous vehicles: See the “hype” phase to the 20 years later
  - Now: emphasis on testing and smaller, correct steps, considering safety first