Outline Sept 7:

• Recap last time (informal quiz)
• Continue: type conversion and input
• First programs in atom (using print and main)
• Mathematical operations and math library

Reminders
• Make sure to handin21 for Lab 0 by Sat night!
• Office hours today 3-5pm (Sci. Center 249)
Informal quiz: discuss with a partner

1) Given the code below, what is the value of x?
   ```python
   x = 5
   y = 3.14
   x = x + y
   int(x)
   ```

2) What is the assignment operator? What does it do?

3) What do we call constructs like `print(..)`, `type(..)`, `input(..)`?

4) BONUS: consider the code below:
   ```python
   name = input("Enter your name: ")
   year = input("Enter the current year: ")
   ```
   what is the type of `name`?
   what is the type of `year`?
Informal quiz: discuss with a partner

• 1) Given the code below, what is the value of x?

```
x = 5
y = 3.14
x = x + y
int(x)
```

The value of x is 8.14

• 2) What is the assignment operator? What does it do?

• 3) What do we call constructs like `print(..), type(..), input(..)`?

• 4) BONUS: consider the code below:

```
name = input("Enter your name: ")
year = input("Enter the current year: ")
```

what is the type of `name`?  
what is the type of `year`?
Informal quiz: discuss with a partner

1) Given the code below, what is the value of x?

```python
x = 5
y = 3.14
x = x + y
int(x)
```

The value of x is 8.14

2) What is the assignment operator? What does it do?

The assignment operator (=) gives a variable a value

3) What do we call constructs like `print(..), type(..), input(..)`?

4) BONUS: consider the code below:

```python
name = input("Enter your name: ")
year = input("Enter the current year: ")
```

what is the type of `name`?
what is the type of `year`?
1) Given the code below, what is the value of x?

\begin{verbatim}
x = 5
y = 3.14
x = x + y
int(x)
\end{verbatim}

The value of x is 8.14

2) What is the assignment operator? What does it do?

The assignment operator (=) gives a variable a value

3) What do we call constructs like `print(..), type(..), input(..)`?

These are called functions. Their arguments go inside parenthesis.

4) BONUS: consider the code below:

- `name = input(“Enter your name: ”)`
- `year = input(“Enter the current year: ”)`

what is the type of `name`?
what is the type of `year`?
Informal quiz: discuss with a partner

1) Given the code below, what is the value of x?

```python
x = 5
y = 3.14
x = x + y
int(x)
```

The value of x is 8.14

2) What is the assignment operator? What does it do?

The assignment operator (=) gives a variable a value

3) What do we call constructs like `print(..), type(..), input(..)`?

These are called functions. Their arguments go inside parenthesis.

4) BONUS: consider the code below:

```python
name = input("Enter your name: ")
year = input("Enter the current year: ")
```

what is the type of `name`?
what is the type of `year`?

We didn’t explicitly talk about this last time, but input(..) always returns a string <str>. 
Variables, Types, Functions
Recap key concepts

- *Variables* as a way to store *values*
Recap key concepts

• *Variables* as a way to store *values*

• *Assignment operator* (=) is a way to change the value of a variable (not symmetric like equals operator in math!)
  • Variable name on the left, expression on the right
Recap key concepts

• *Variables* as a way to store *values*

• *Assignment operator* (=) is a way to change the value of a variable (not symmetric like equals operator in math!)
  • Variable name on the left, expression on the right

• The *type* of a variable is the type of the value it refers to
Recap key concepts

• *Variables* as a way to store *values*

• *Assignment operator* (=) is a way to change the value of a variable (not symmetric like equals operator in math!)
  • Variable name on the left, expression on the right

• The *type* of a variable is the type of the value it refers to

• We can *convert* a variable to a different type, but it does not change the value of the original variable
Types for the first few weeks
Types for the first few weeks

- **int**, examples: 3, 0, -10

- **float**, examples: 7.32, 21.1, -30.0

- **str**, examples: “hello”, ‘hello’, “what is your name?”
Functions so far

- **type(...)**
  - *input*: variable or value
  - *output*: the type (*int, str, float*…)

- **input(...)**
  - *input*: *str* (usually a query)
  - *output*: *str* (whatever the user typed in response to the query)

- **int(...)**
  - *input*: variable or value
  - *output*: input converted to an *int* if possible (error if not possible)

- **str(...)**
  - *input*: variable or value
  - *output*: input converted to a *str* if possible (error if not possible)

- **float(...)**
  - *input*: variable or value
  - *output*: input converted to a *float* if possible (error if not possible)

- **print(...)**
  - *input*: variable or value (separate multiple by a comma)
  - *output*: None! (prints to terminal)
Demo + first programs
• Run `update21`

• In `/home/[username]/cs21/inclass/w01` there should be a file `welcome.py`

• Run `atom welcome.py` then edit the code

• Run `python3 welcome.py` in the terminal
Other program to try with a partner

1) Ask the user for their graduation year and the current year, then print out how many years until they graduate.
Key ideas

• We will always use `def main():` and then write main indented
Key ideas

- We will always use `def main():` and then write `main` indented

- *Expressions* vs. *statements*
  - In the interpreter, the results of expressions are shown
  - In the editor (i.e. in our code) we need to write full statements
Key ideas

• We will always use `def main():` and then write main indented

• *Expressions* vs. *statements*
  • In the interpreter, the results of expressions are shown
  • In the editor (i.e. in our code) we need to write full statements

• *Comments*: use hashtag symbol (`#`)
Key ideas

• We will always use `def main():` and then write main indented

• **Expressions** vs. **statements**
  • In the interpreter, the results of expressions are shown
  • In the editor (i.e. in our code) we need to write full statements

• **Comments**: use hashtag symbol (`#`)

• User variable names that implicitly show type
Key ideas

• We will always use `def main():` and then write main indented

• *Expressions* vs. *statements*
  • In the interpreter, the results of expressions are shown
  • In the editor (i.e. in our code) we need to write full statements

• *Comments*: use hashtag symbol (`#`)

• User variable names that implicitly show type

• `print` is very powerful! A way to see what is going on and to give the user valuable information
Arithmetic operations and math library demo

Next time!