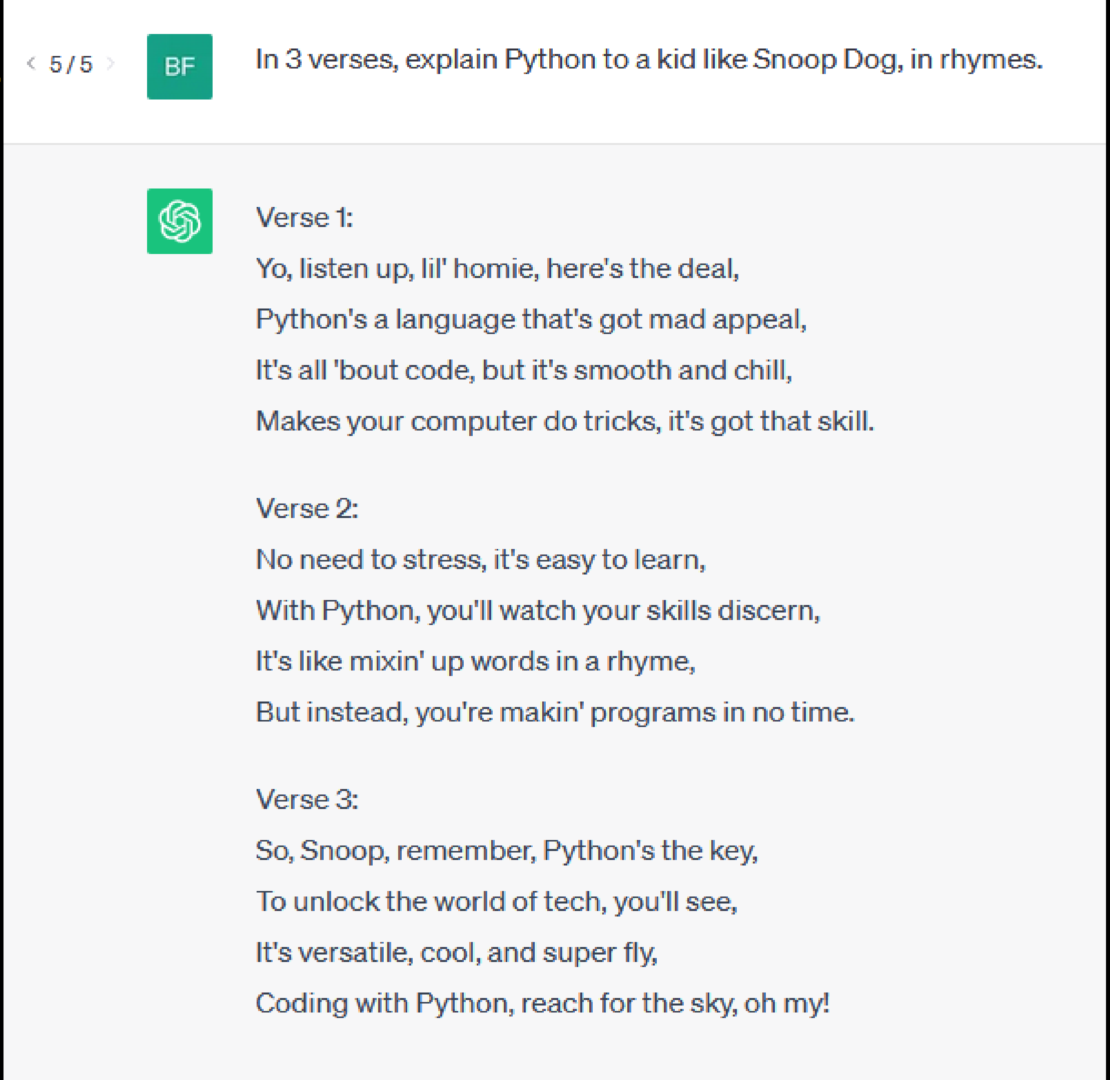
Chatbots with Personality

By Dr. Angelica Lim
Modified by Dr. Brian Fraser



5 / 5

BF

In 3 verses, explain Python to a kid like Snoop Dog, in rhymes.

Verse 1:

Yo, listen up, lil' homie, here's the deal,

Python's a language that's got mad appeal,

It's all 'bout code, but it's smooth and chill,

Makes your computer do tricks, it's got that skill.

Verse 2:

No need to stress, it's easy to learn,

With Python, you'll watch your skills discern,

It's like mixin' up words in a rhyme,

But instead, you're makin' programs in no time.

Verse 3:

So, Snoop, remember, Python's the key,

To unlock the world of tech, you'll see,

It's versatile, cool, and super fly,

Coding with Python, reach for the sky, oh my!

Review from last class

- What is it called when a high-level programming language is translated into **machine code** *while it's running*?
- How about when the high-level programming language is translated into **machine code** completely *first*, and then run?

What are some **chatbots** today?

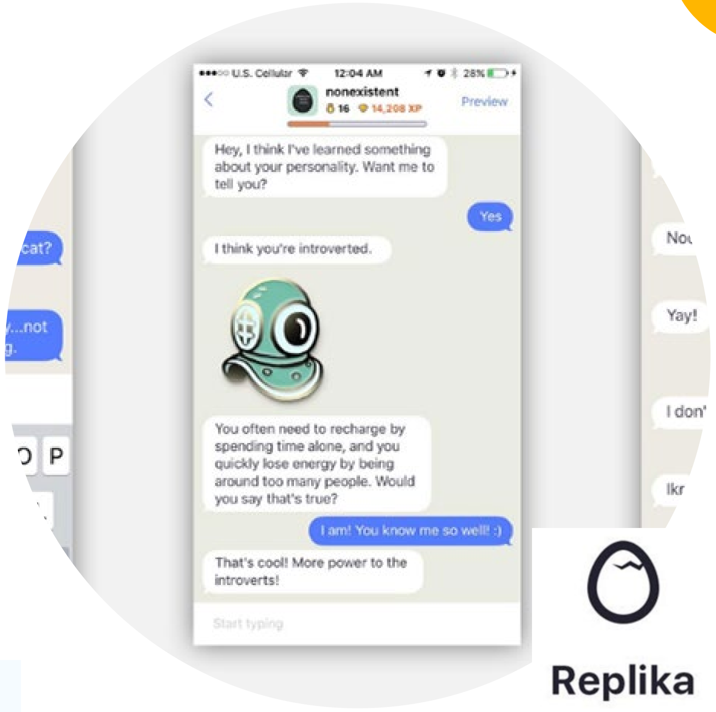
The study of **natural language processing** processes the text of natural languages like English and Japanese.

- ELIZA
- Woebot
- ChatGPT
- Bard

Google Translate

BOTLER AI

ChatGPT



U.S. Cellular 12:04 AM 28%

nonexistent

16 14,208 XP

Preview

Hey, I think I've learned something about your personality. Want me to tell you?

Yes

I think you're introverted.

You often need to recharge by spending time alone, and you quickly lose energy by being around too many people. Would you say that's true?

I am! You know me so well! :)

That's cool! More power to the introverts!

Start typing

Replika

Unit 1 Chatbots

APPLICATIONS

In this unit, we'll learn about the computing science field of **natural language processing**, and its applications such as chat bots.

PROGRAMMING LANGUAGE

In Python 3, we'll be learning the **syntax** and **keywords** to implement our algorithms.

ALGORITHMS

We'll learn about **input, output, strings, conditionals, concatenation** and other concepts to build a simple chatbot algorithm.

TESTING

We will learn how to tell if our program is any **good or not**!

Let's start!

First challenge: write a chatbot that can say hi and learn your name.

Design an Algorithm — Write it in Python — Test and Deploy — Change the World

Design an algorithm

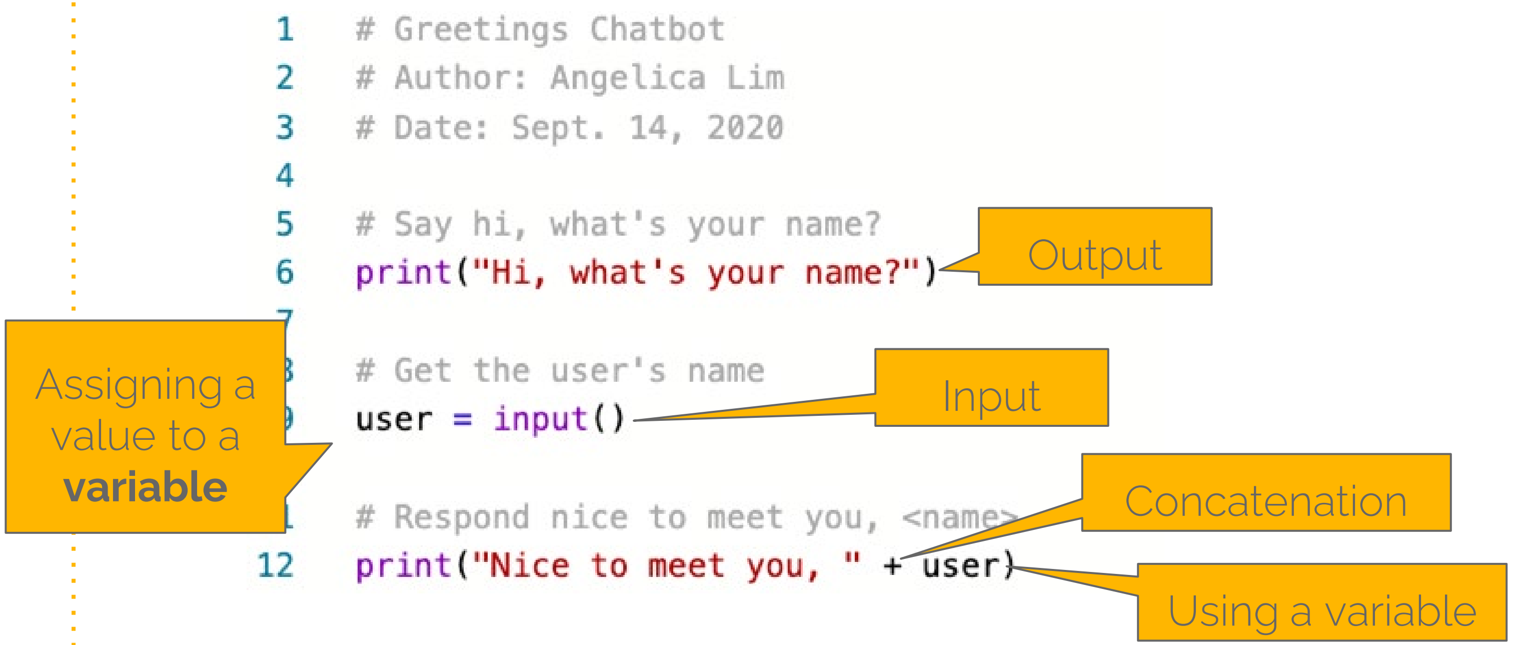
```
# Greetings Chatbot
# Author: Angelica Lim
# Date: Sept. 14, 2020

# Say hi, what's your name?
# Get the user's name
# Respond nice to meet you, <name>
```

"Header block" containing comments with title, author and date

Algorithm steps

https://runestone.academy/runestone/books/published/thinkcspy/GeneralIntro/Comments.html

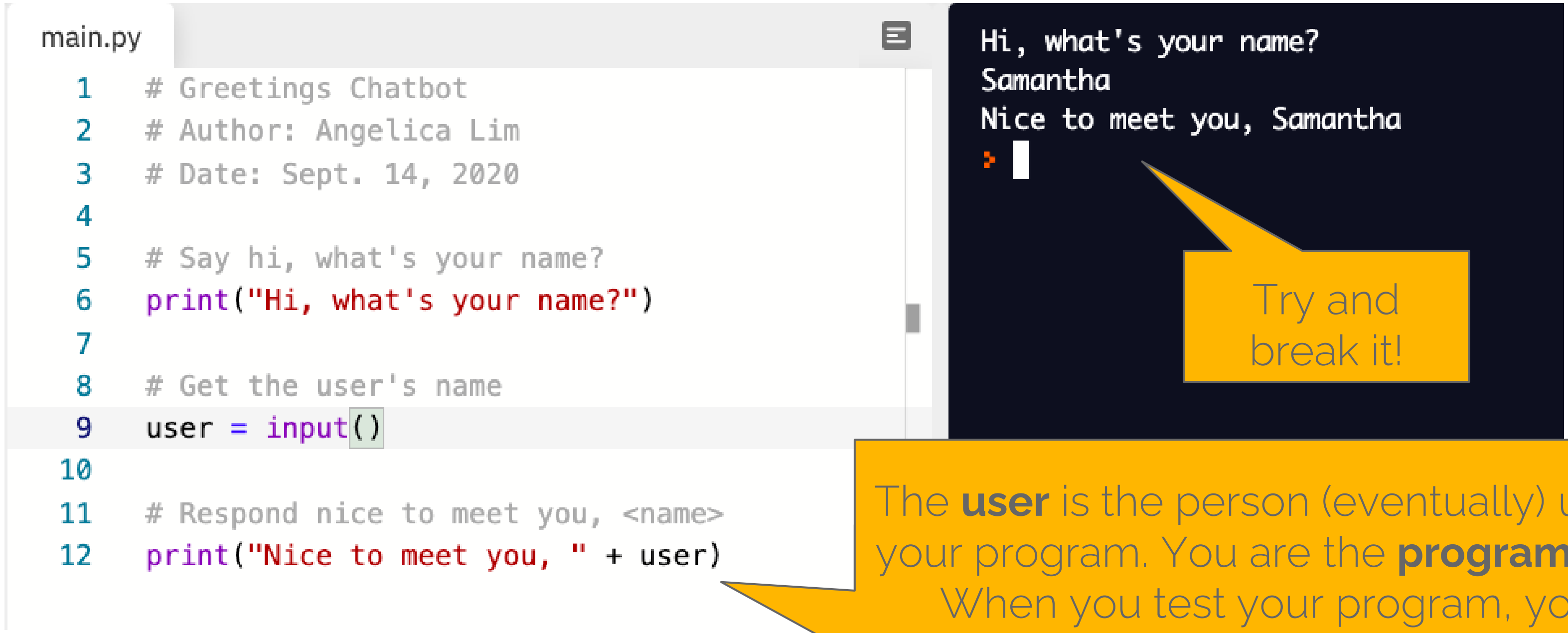
Translate it to Python 3



```
# Greetings Chatbot
# Author: Angelica Lim
# Date: Sept. 14, 2020

# Say hi, what's your name?
print("Hi, what's your name?")

# Get the user's name
user = input()

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)
```

https://runestone.academy/runestone/books/published/thinkcspy/SimplePythonData/Variables.html
https://runestone.academy/runestone/books/published/thinkcspy/SimplePythonData/Input.html
https://runestone.academy/runestone/books/published/thinkcspy/Strings/OperationsonStrings.html

Test your algorithm in Python



main.py

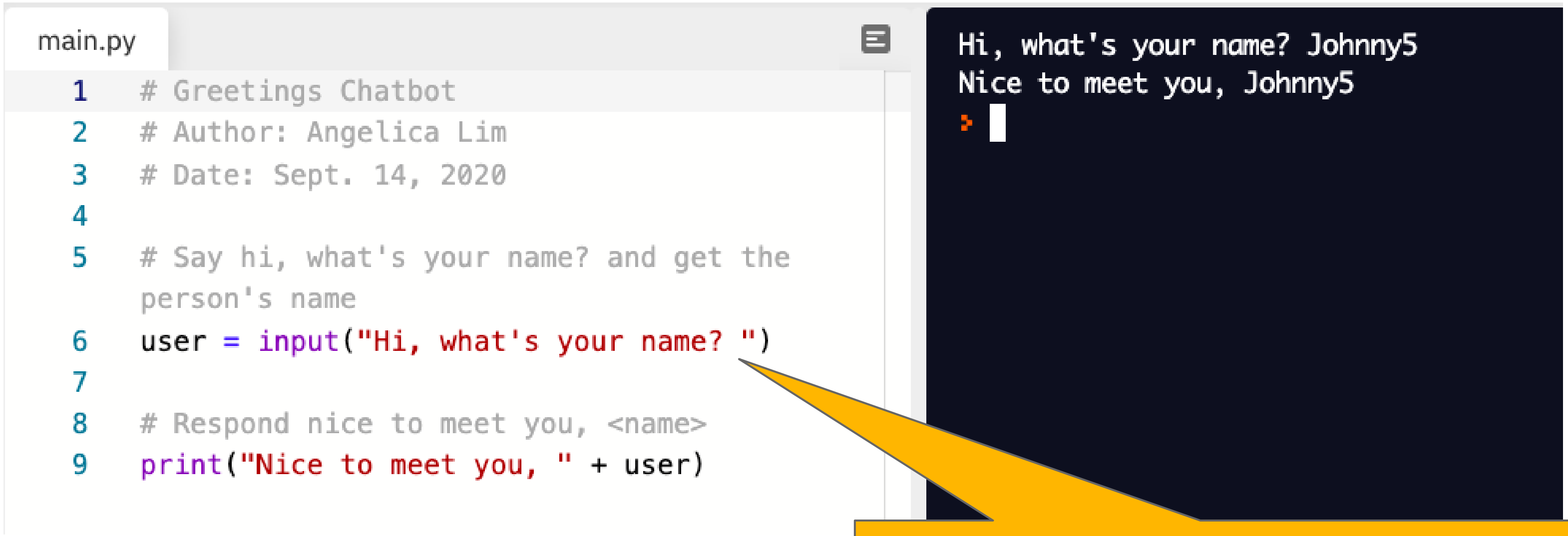
```python
# Greetings Chatbot
# Author: Angelica Lim
# Date: Sept. 14, 2020

# Say hi, what's your name?
print("Hi, what's your name?")

# Get the user's name
user = input()

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)
```

```
Hi, what's your name?
Samantha
Nice to meet you, Samantha
> 
```

Try and break it!

The **user** is the person (eventually) using your program. You are the **programmer**. When you test your program, you pretend to be a user.

Another way to input


main.py

```python
# Greetings Chatbot
# Author: Angelica Lim
# Date: Sept. 14, 2020

# Say hi, what's your name? and get the person's name
user = input("Hi, what's your name? ")

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)
```

```
Hi, what's your name? Johnny5
Nice to meet you, Johnny5
>
```

You can also put the message inside the **input** parentheses. Note the space needed to produce the example.

About variables

- A variables is a **location in memory** that we can store **some information**.
- It's like a post-it note:
  - We **write** a value
  - Later, we can **change** the value

About variables

There are some constraints to how you can name your variables, e.g.

✓

- Can contain letters, numbers, and underscores
- Should start with a letter (lowercase, by convention)

✗

- They can't contain spaces or symbols
- They can't be one of the reserved keywords (see below for list)

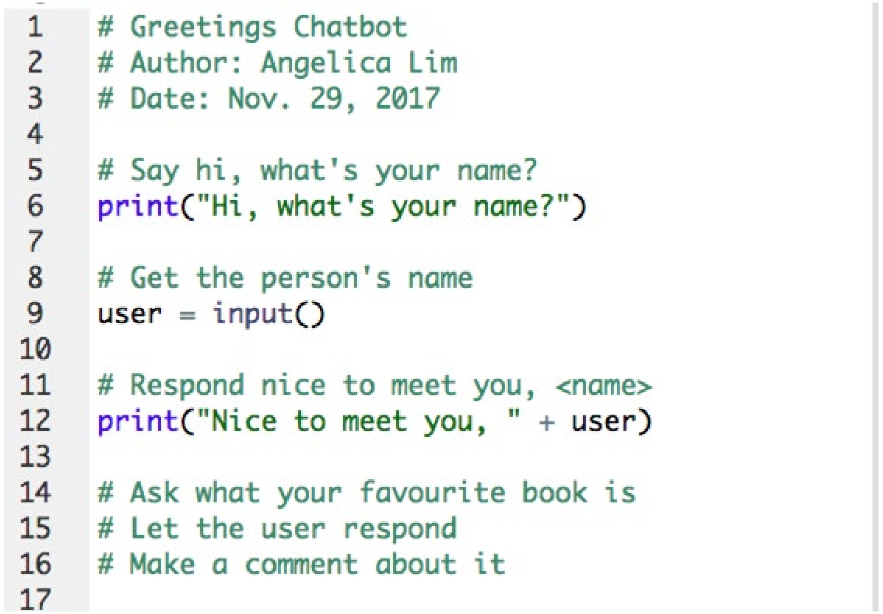
https://runestone.academy/runestone/books/published/thinkcspy/SimplePythonData/VariableNamesandKeywords.html

Add more features

After the greetings, make the chatbot **ask what is your favourite book**, and **let you respond**.

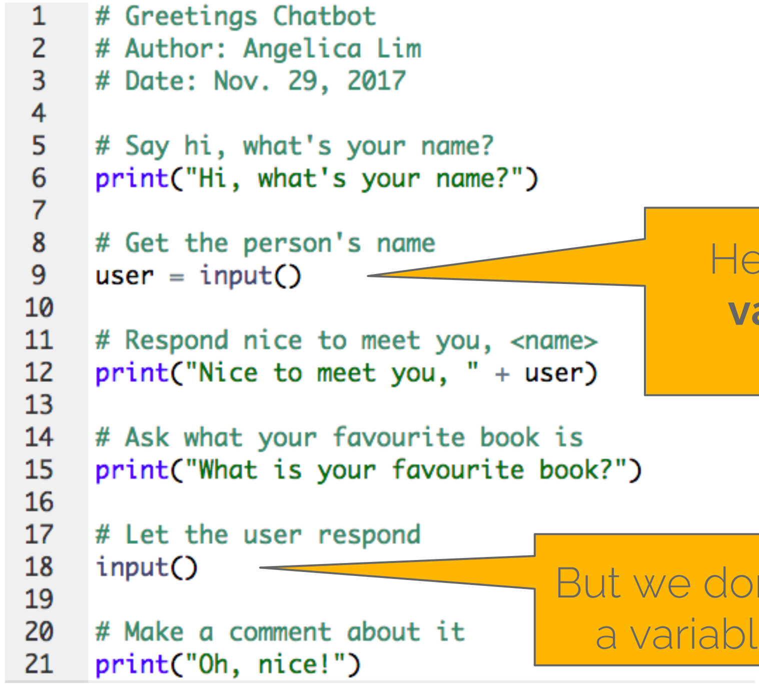
The chatbot should then **make a comment** about your response.

Update your algorithm



```
# Greetings Chatbot
# Author: Angelica Lim
# Date: Nov. 29, 2017

# Say hi, what's your name?
print("Hi, what's your name?")

# Get the person's name
user = input()

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)

# Ask what your favourite book is
# Let the user respond
# Make a comment about it

```

Translate to Python 3

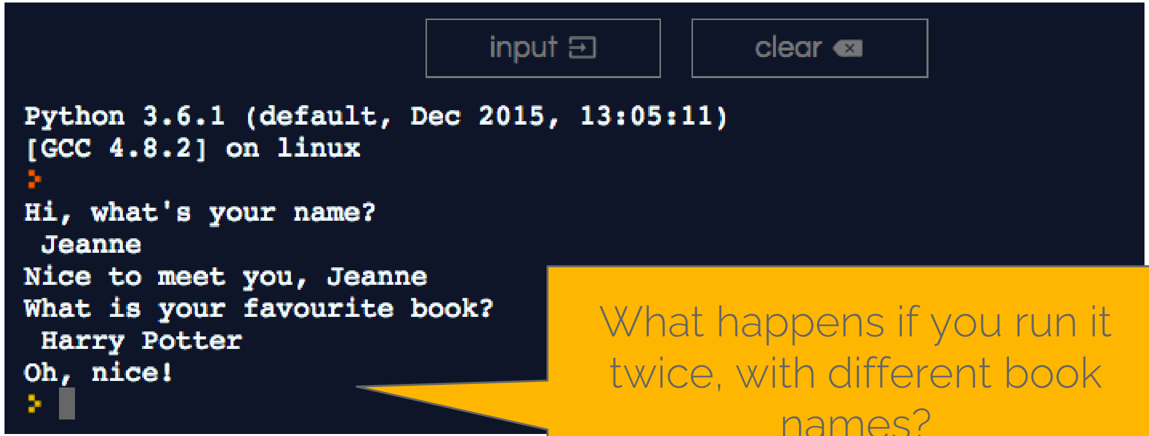


```
# Greetings Chatbot
# Author: Angelica Lim
# Date: Nov. 29, 2017

# Say hi, what's your name?
print("Hi, what's your name?")

# Get the person's name
user = input()

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)

# Ask what your favourite book is
print("What is your favourite book?")

# Let the user respond
input()

# Make a comment about it
print("Oh, nice!")
```

Here we stored the user's name in a **variable** (called user) because we wanted to use it later.

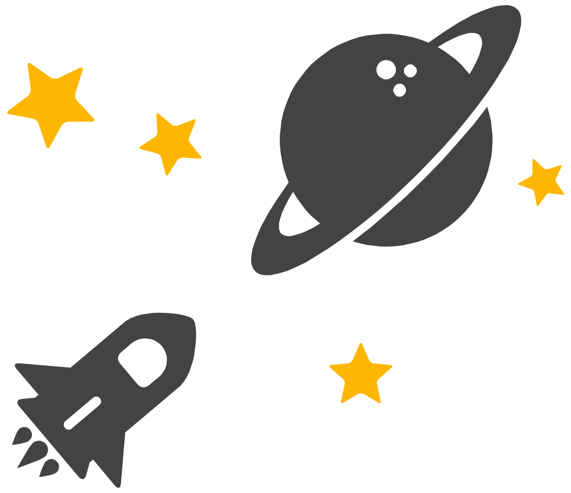
But we don't need to store the answer in a variable if we're not going to use it.

Test how it works



```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
>
Hi, what's your name?
 Jeanne
Nice to meet you, Jeanne
What is your favourite book?
 Harry Potter
Oh, nice!
>
```

What happens if you run it twice, with different book names?



Lists

This is where content can come from.

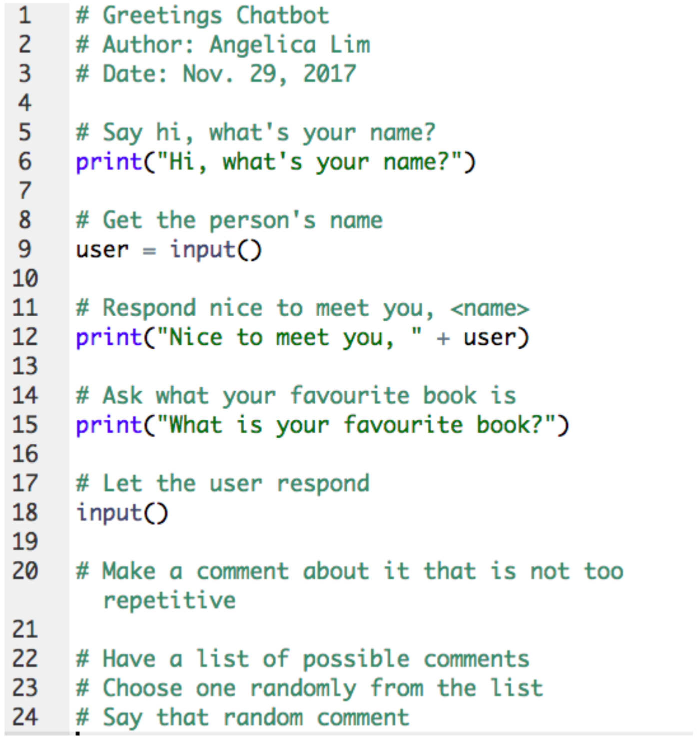
https://runestone.academy/runestone/books/published/thinkcspy/Lists/ListValues.html

Add more features

The first challenge was to **write a chatbot** that can **learn your name**.
Secondly, it **asks what your favourite book is** and **makes a comment** about it.

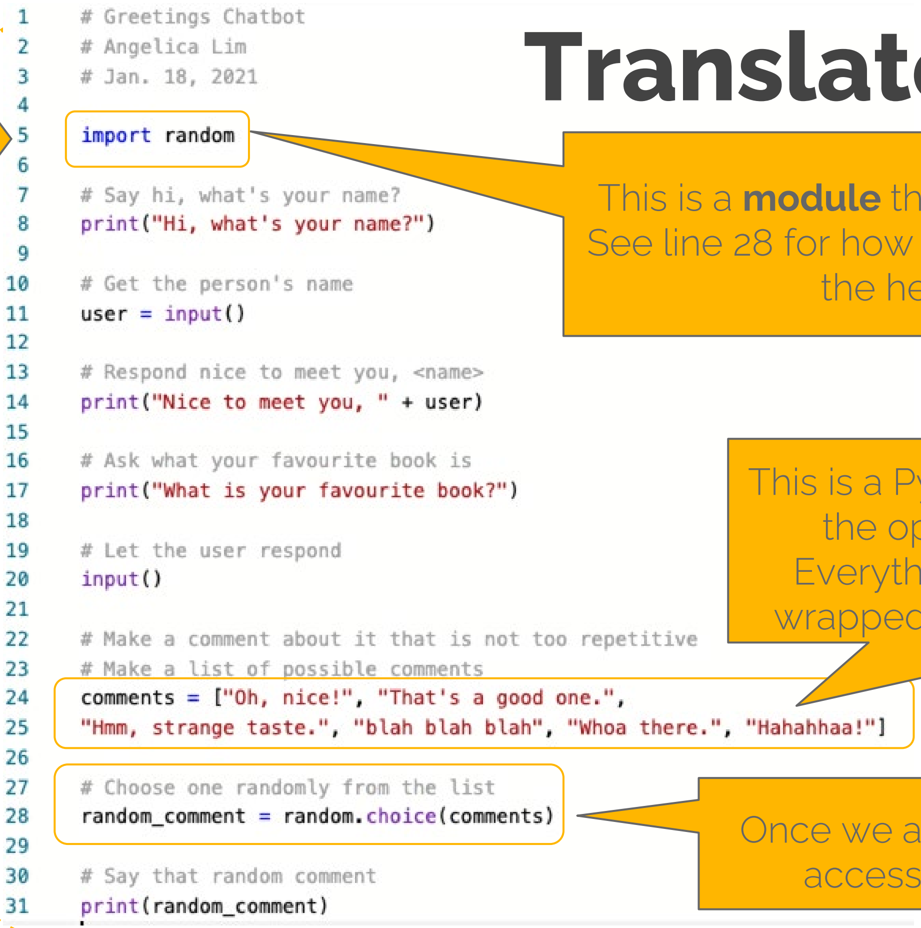
>> The comment **should not be too obviously repetitive.**

Update your algorithm



```
# Greetings Chatbot
# Author: Angelica Lim
# Date: Nov. 29, 2017

# Say hi, what's your name?
print("Hi, what's your name?")

# Get the person's name
user = input()

# Respond nice to meet you, <name>
print("Nice to meet you, " + user)

# Ask what your favourite book is
print("What is your favourite book?")

# Let the user respond
input()

# Make a comment about it that is not too repetitive

# Have a list of possible comments
# Choose one randomly from the list
# Say that random comment
```

Translate to Python 3



```
 1  # Greetings Chatbot
 2  # Angelica Lim
 3  # Jan. 18, 2021
 4
 5  import random
 6
 7  # Say hi, what's your name?
 8  print("Hi, what's your name?")
 9
10  # Get the person's name
11  user = input()
12
13  # Respond nice to meet you, <name>
14  print("Nice to meet you, " + user)
15
16  # Ask what your favourite book is
17  print("What is your favourite book?")
18
19  # Let the user respond
20  input()
21
22  # Make a comment about it that is not too repetitive
23  # Make a list of possible comments
24  comments = ["Oh, nice!", "That's a good one.",
25  "Hmm, strange taste.", "blah blah blah", "Whoa there.", "Hahahhaa!"]
26
27  # Choose one randomly from the list
28  random_comment = random.choice(comments)
29
30  # Say that random comment
31  print(random_comment)
```

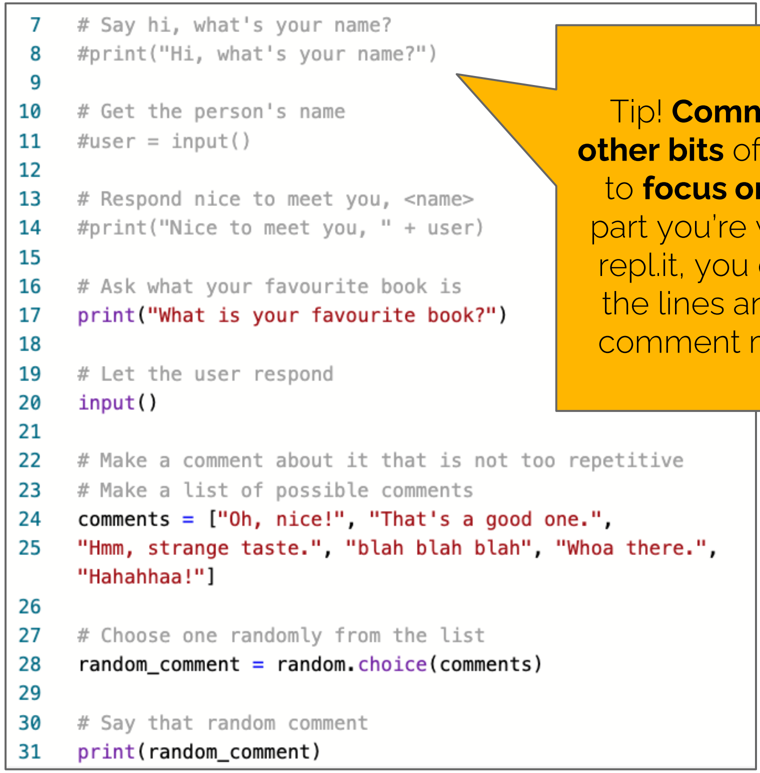
This is a **module** that we can **import** to have more functionality. See line 28 for how to use it. All **import** statements should follow the header at the **top** of the program.

This is a Python **list**, indicated with brackets. All the options are separated by commas. Everything should be on one line (here it's wrapped due to space, but it's all on line 22.)

Once we add line 5 to import the module, we can access that module's functions with a **dot**.

http://interactivepython.org/runestone/static/thinkcspy/PythonModules/modules.html

Test **how it works**



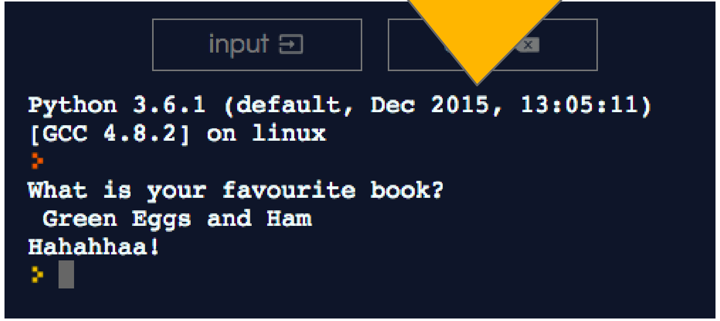
```
# Say hi, what's your name?
#print("Hi, what's your name?")

# Get the person's name
#user = input()

# Respond nice to meet you, <name>
#print("Nice to meet you, " + user)

# Ask what your favourite book is
print("What is your favourite book?")

# Let the user respond
input()

# Make a comment about it that is not too repetitive
# Make a list of possible comments
comments = ["Oh, nice!", "That's a good one.",
"Hmm, strange taste.", "blah blah blah", "Whoa there.",
"Hahahhaa!"]

# Choose one randomly from the list
random_comment = random.choice(comments)

# Say that random comment
print(random_comment)
```

Tip! **Comment** out **the other bits** of your program to **focus on testing** the part you're working on. In repl.it, you can select all the lines and hit alt-/ to comment multiple lines.

Remember the feature we wanted to make:

The comment **should not be too obviously repetitive.**

What happens when you run it multiple times? Is it still repetitive?



```
Python 3.6.1 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
›
What is your favourite book?
 Green Eggs and Ham
Hahahhaa!
›
```

Question 1

Which symbol is used to designate **comments** in Python?

Question 2

What would you type in the comments at the top of your Python file? (Hint: There should be at least 3 elements.)

Readings Check!

Question 3

What does the = in this line of code refer to?

```
name = "poppy"
```

Question 4

What does the following code produce?

```
print("LO"*3+"L")
```

Let's review some concepts

How do we make a list in Python?

What module do we need to import to randomly choose something from a list?

How can we test smaller pieces of our Python code?

What is concatenation?

What does a dot after a module name do?

What characters can we have in a variable name?