Control Flow Revisited
Chris Piech and Mehran Sahami
Stanford University
Housekeeping

- Assignment #2 goes out today
  - It is due in one week
  - Worth the focus
- Code while you watch:
  - https://us.edstem.org/courses/2695/lessons/5749
- It is Friday! Which is just a beautiful thing.
Review
num_students = 700

Piech + Sahami, CS106A, Stanford University
My computer has space for about 10 billion suitcases
# Create a variable, of type int
# called age with the value 30.
age = 31

# Use the value in age (output it)
print("age is: " + str(age))

# Modify age to be one greater.
age = age + 1
Create, Modify, Use

# Create a variable, of type int
# called age with the value 30.
age = 31

# Use the value in age (output it)
print("age is: ", age)

# Modify age to be one greater.
age = age + 1
Binary Operators

+  Addition
−  Subtraction
*  Multiplication
/  Division
Cool Example: Carbon Dating

C\textsubscript{14} = 1.2 \text{ dpm}

C\textsubscript{14} = 13.6 \text{ dpm}

\[ \text{age} = K \cdot \log \left( \frac{c_{14}}{100} \right) \]
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14 in Sample:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

* It calculates the age of a C14 sample
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14 in Sample:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

* It calculates the age of a C14 sample

Piech + Sahami, CS106A, Stanford University
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14 in Sample:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

* It calculates the age of a C14 sample
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

terminal
> python carbondate.py
% of natural c14: 50
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14:"))
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

terminal
> python carbondate.py
% of natural c14: 50

float

50.0

pct_left
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

cool example: carbon dating

terminal

> python carbondate.py
% of natural c14: 50

50.0

float

pct_left

5730.0
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

terminal
> python carbondate.py
% of natural c14: 50

50.0
pct_left

5730.0
age

float

float
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

terminal

> python carbondate.py
% of natural c14: 50
Sample is 5730.0 years old
Today’s Goal

1. Be able to use For / While / If in Python
Today’s Route

Core
Python

You are here

The River of Control Flow

For Loops

Booleans

Conditions

Review

Piech + Sahami, CS106A, Stanford University
Today’s Route

You are here

Core Python

For Loops

Booleans

Conditions

Review

The River of Control Flow

Piech + Sahami, CS106A, Stanford University
While Loop in Karel

while front_is_clear() :
    body

if beepers_present() :
    body
While Loop Redux

```python
while condition :
    body

if condition :
    body
```

The condition should be a “boolean” which is either True or False.
def main():
    calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14 in Sample:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

* It calculates the age of a C14 sample
def main():
    while True:
        calculate_age_single_sample()

def calculate_age_single_sample():
    # ask the user to enter the percent c14 left in their sample
    pct_left = float(input("% of natural c14 in Sample:"))
    # calc the age: https://en.wikipedia.org/wiki/Radiocarbon_dating
    age = math.log(pct_left / 100) * HALF_LIFE_CONSTANT
    # print the result
    print("Sample is " + str(age) + " years old.")

* It calculates the age of a C14 sample
Today’s Route

You are here

Core Python

For Loops

Booleans

Conditions

Review

The River of Control Flow
Today’s Route

The River of Control Flow

Review → Conditions → Booleans → For Loops → Core Python

You are here
1 < 2
Booleans

\[ 1 < 2 \]

True
## Comparison Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Example</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>equals</td>
<td>1 + 1 == 2</td>
<td>True</td>
</tr>
<tr>
<td>!=</td>
<td>does not equal</td>
<td>3.2 != 2.5</td>
<td>True</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than</td>
<td>10 &lt; 5</td>
<td>False</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than</td>
<td>10 &gt; 5</td>
<td>True</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal to</td>
<td>126 &lt;= 100</td>
<td>False</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal to</td>
<td>5.0 &gt;= 5.0</td>
<td>True</td>
</tr>
</tbody>
</table>

* All have equal precedence
## Comparison Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Example</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>equals</td>
<td><code>1 + 1 == 2</code></td>
<td>True</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>does not equal</td>
<td><code>3.2 != 2.5</code></td>
<td>True</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>less than</td>
<td><code>10 &lt; 5</code></td>
<td>False</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>greater than</td>
<td><code>10 &gt; 5</code></td>
<td>True</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>less than or equal to</td>
<td><code>126 &lt;= 100</code></td>
<td>False</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>greater than or equal to</td>
<td><code>5.0 &gt;= 5.0</code></td>
<td>True</td>
</tr>
</tbody>
</table>

* All have equal precedence
Comparison Operators

```python
if 1 < 2 :
    print("1 is less than 2")

num = int(input("Enter a number: "))
if num == 0:
    print("That number is 0")
else :
    print("That number is not 0.")
```
num = int(input("Enter a number: 
if num == 0:
    print("Your number is 0 ")
else:
    if num > 0:
        print("Your number is positive")
    else:
        print("Your number is negative")
If Else Revisited

```python
num = int(input("Enter a number: "))
if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
```

Piech + Sahami, CS106A, Stanford University
num = int(input("Enter a number: "))

if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
num = int(input("Enter a number: "))

if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
num = \textbf{int}(\textit{input("Enter a number: ")})

\textbf{if} num == 0:
    \textbf{print("Your number is 0 ")}

\textbf{elif} num > 0:
    \textbf{print("Your number is positive")}

\textbf{else}:
    \textbf{print("Your number is negative")}

Enter a number: 5
```python
num = int(input("Enter a number: 

if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
```
num = \texttt{int}(\texttt{input}("Enter a number: "))

\texttt{if} num == 0:
    \texttt{print}("Your number is 0 ")
\texttt{elif} num > 0:
    \texttt{print}("Your number is positive")
\texttt{else}:
    \texttt{print}("Your number is negative")
```python
num = int(input("Enter a number: "))
if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
```

```
Enter a number: 5
```

```
5
num
```
num = int(input("Enter a number: "))
if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
If Else Revisited

```python
num = int(input("Enter a number: "))
if num == 0:
    print("Your number is 0 ")
elif num > 0:
    print("Your number is positive")
else:
    print("Your number is negative")
```

Enter a number: 5
Your number is positive
Use `while` and `if` statements in Python.

They are the same as in Karel, except that the `test` can be any expression that evaluates to `True` or `False`. 
Amazing
Guess My Number

I am thinking of a number between 0 and 99...
Enter a guess: 50
Your guess is too high

Enter a new number: 25
Your guess is too low

Enter a new number: 40
Your guess is too low

Enter a new number: 45
Your guess is too low

Enter a new number: 48
Congrats! The number was: 48
Guess My Number

```
import random

secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))

while guess != secret_number:
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print(""")
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: "+str(secret_number))
```

```
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")
print("") # an empty line
guess = int(input("Enter a new guess: "))
print("Congrats! The number was: " + str(secret_number))
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))

# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
**Guess My Number**

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print(""")  # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```

50  int

92  int
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
**Guess My Number**

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")
print(""") # an empty line
guess = int(input("Enter a new guess: "))
print("Congrats! The number was: " + str(secret_number))
```
```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line

# an empty line

guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")
print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```

95 | int
---|---
guess

92 | int
---|---
secret_number
```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("")  # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```

![Diagram](image)
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("")  # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```

- `secret_number`: 92
- `guess`: 95
- `int`: To convert a string to an integer.

Note: The code includes logic to prompt the user for guesses, compare them to a secret number generated randomly, and provide feedback on whether the guess is too high or too low. Once the correct number is guessed, the program displays the number. The diagram shows the data flow with the `guess` and `secret_number`. The `int` function is used to convert user input to an integer.
### Guess My Number

```python
import random

secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))

# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

    print("")  # an empty line
    guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```

---

```plaintext
<table>
<thead>
<tr>
<th>guess</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>95</td>
<td></td>
</tr>
</tbody>
</table>
```

---

```plaintext
<table>
<thead>
<tr>
<th>secret_number</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>92</td>
<td></td>
</tr>
</tbody>
</table>
```
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line

print("Congrats! The number was: ") + str(secret_number)
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))
print("Congrats! The number was: " + str(secret_number))
```
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("") # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
Behind the Scenes
Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")

print("Congrats! The number was: " + str(secret_number))
```
def guess_my_number():
    secret_number = random.randint(1, 99)
    print("I am thinking of a number between 1 and 99...")

    while True:
        print("Enter a guess:")
        guess = int(input())

        if guess == secret_number:
            print("Congrats! The number was: " + str(secret_number))
            break

    print("You guessed it!"")

guess_my_number()
```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")

while ???:
    # Get a new guess

    # Report high/low

print("Congrats! The number was: " + str(secret_number))
```
PIECH + SAHAMI, CS106A, STANFORD UNIVERSITY

# Guess My Number

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")

while ???:
    # Get a new guess
    guess = int(input("Enter a new guess: "))

    # Report high/low

    print("Congrats! The number was: " + str(secret_number))
```

**Guess My Number**

```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")

while guess != secret_number:
    # Get a new guess
    guess = int(input("Enter a new guess: "))

    # Report high/low

print("Congrats! The number was: " + str(secret_number))
```
Guess My Number

secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a new guess: "))

while guess != secret_number:
    # Report high/low
    # Get a new guess
    guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```python
secret_number = random.randint(1, 99)
print("I am thinking of a number between 1 and 99...")
guess = int(input("Enter a guess: "))
# True if guess is not equal to secret number
while guess != secret_number:
    # True if guess is less than secret number
    if guess < secret_number:
        print("Your guess is too low")
    else:
        print("Your guess is too high")

print("")  # an empty line
guess = int(input("Enter a new guess: "))

print("Congrats! The number was: " + str(secret_number))
```
Time to shine
• **sentinel**: A value that signals the end of user input.
  
  – **sentinel loop**: Repeats until a sentinel value is seen.

• Example: Write a program that prompts the user for numbers until the user types -1, then output the total of the numbers.
  
  – In this case, -1 is the sentinel value.

  
  Type a number: **10**
  Type a number: **20**
  Type a number: **30**
  Type a number: **-1**
  total is **60**
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0

while num != -1:
    num = int(input("Enter a number: "))
    total += num

print("total is " + total)
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

```python
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
```
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + str(total))
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))

while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))

while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: 

print("total is " + total)
Example: Sentinel Loops

```python
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
```
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
```
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
```
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

```python
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is "+total)
```
Example: Sentinel Loops

```python
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + str(total))
```

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
num = int(input("Enter a number: "))
while num != -1:
    total += num
    num = int(input("Enter a number: "))

print("total is " + total)
Example: Sentinel Loops

```
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0
while True:
    num = int(input("Enter a number: "))
    if num == -1:
        break  # immediately exits loop
    total += num
print("total is " + total)
```
# fencepost problem!
# ask for number - post
# add number to total - fence

total = 0

while True:
    num = int(input("Enter a number: "))
    if num == SENTINEL:
        break  # immediately exits loop
    total += num

print("total is " + total)
Logical Operators

In order of precedence:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Example</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>not</td>
<td>not (2 == 3)</td>
<td>True</td>
</tr>
<tr>
<td>and</td>
<td>(2 == 3) and (-1 &lt; 5)</td>
<td>False</td>
</tr>
<tr>
<td>or</td>
<td>(2 == 3) or (-1 &lt; 5)</td>
<td>True</td>
</tr>
</tbody>
</table>

Can "chain" tests as in algebra

```python
# assume x is 15
2 <= x <= 10
```

```python
# identical version
2 <= x and x <= 10
```
Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
5 * 7 >= 3 + 5 * 6 and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence Madness

Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False

35 >= 3 + 5 * 6 and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
35 >= 3 + 5 * 6 and not False
35 >= 3 + 30 and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
35 >= 3 + 5 * 6 and not False
35 >= 3 + 30 and not False
35 >= 33 and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence Madness

Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
35 >= 3 + 5 * 6 and not False
35 >= 3 + 30 and not False
35 >= 33 and not False
True and not False

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence:

- arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
35 >= 3 + 5 * 6 and not False
35 >= 3 + 30 and not False
35 >= 33 and not False
True and not False
True and True

https://docs.python.org/3/reference/expressions.html#operator-totalmary
Precedence Madness

Precedence:

arithmetic > comparison > not > and/or

5 * 7 >= 3 + 5 * (7 - 1) and not False
35 >= 3 + 5 * 6 and not False
35 >= 3 + 30 and not False
35 >= 33 and not False
True and not False
True and True
True

https://docs.python.org/3/reference/expressions.html#operator-totalmary
George Boole

English Mathematician teaching in Ireland 1815 – 1864
Boole died of being too cool

Piech + Sahami, CS106A, Stanford University
# Store expressions that evaluate to True/False

x = 1 < 2  # True
y = 5.0 == 4.0  # False
# Store expressions that evaluate to True/False
x = 1 < 2  # True
y = 5.0 == 4.0  # False

# Directly set to True/False
is_sheltering = True
is_raining = False
# Store expressions that evaluate to True/False
x = 1 < 2  # True
y = 5.0 == 4.0  # False

# Directly set to True/False
is_sheltering = True
is_raining = False

play_again = input('Play again? "y" or "n"') == 'y'
if play_again:
    ...

Boolean Variables
Please...

NO FOOD OR DRINKS

is_allowed = not food or drinks

*know your logical precedence
is_allowed = not food or drinks

False

*know your logical precedence
Please...

NO FOOD OR DRINKS

is_allowed = not food or drinks

False        True

*know your logical precedence
Today’s Route

You are here

Core
Python

For Loops

Booleans

Conditions

Review

The River of Control Flow

Piech + Sahami, CS106A, Stanford University
Today’s Route

The River of Control Flow

Core Python

For Loops

Booleans

Conditions

Review

You are here
How would you print “Python rocks socks” 100 times
public void run() :
   for i in range(100):
       print("Python rocks socks!")
for i in range(100):
    print("Python rocks socks!")

i = 0
while i < 100:
    print("Python rocks socks!")
    i += 1
Create a counting variable `i`

```python
for i in range(100):
    print("Python rocks socks!")
```

Which takes on the values 0 to 99 one at a time
For Loop Redux

```
for i in range(3):
    print("Python rocks socks!")
```

`range(3) -> [0, 1, 2]`
For Loop Redux

```python
for i in range(3):
    print("Python rocks socks!")
```

`range(3) -> [0, 1, 2]`
For Loop Redux

\[
\begin{array}{c|c}
  i & 0 \\
\end{array}
\]

\[
\begin{array}{ccc}
  \text{range}(3) \rightarrow [0, 1, 2] \\
\end{array}
\]

for \(i\) in range(3):

\[
\text{print("Python rocks socks!")}
\]
For Loop Redux

```
for i in range(3):
    print("Python rocks socks!")
```

```
Python rocks socks
```
For Loop Redux

For Loop Redux

```
for i in range(3):
    print("Python rocks socks!")
```

Python rocks socks
For Loop Redux

```python
i = 1

for i in range(3):
    print(“Python rocks socks!”)
```

Python rocks socks
For Loop Redux

for i in range(3):
    print("Python rocks socks!")

range(3) -> [0, 1, 2]
for i in range(3):
    print("Python rocks socks!")

Python rocks socks
Python rocks socks
i 2

```python
for i in range(3):
    print("Python rocks socks!")
```

`range(3) → [0, 1, 2]`
For Loop Redux

```
for i in range(3):
    print("Python rocks socks!")
```

```
Python rocks socks
Python rocks socks
Python rocks socks
```
```python
for i in range(3):
    print("Python rocks socks!")
```

```
Python rocks socks
Python rocks socks
Python rocks socks
```
For Loop Redux

$\textbf{for } i \textbf{ in } \text{range}(3):$
\begin{align*}
\text{print("Python rocks socks!")}
\end{align*}
You can use the for loop variable
How would you print the first 100 even numbers?
Printing Even Numbers

```
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
```
for i in range(3):
    print(i * 2)
Printing Even Numbers

```python
for i in range(3):
    print(i * 2)
```
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

```
for | 0
```

Piech + Sahami, CS106A, Stanford University
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

<table>
<thead>
<tr>
<th>i</th>
<th>0</th>
</tr>
</thead>
</table>

Piech + Sahami, CS106A, Stanford University
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

```
0
```
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

```
For Loop Redux

0
```
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

i   1

0
2
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

<table>
<thead>
<tr>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
</tbody>
</table>

Output:

```
0
2
```
Printing Even Numbers

```
for i in range(3):
    print(i * 2)
```

```
0
2
4
```
Printing Even Numbers

for i in range(3):
    print(i * 2)
Printing Even Numbers

\[
\begin{array}{c|c}
  i & 3 \\
\end{array}
\]

```python
for i in range(3):
    print(i * 2)
```

```
0
2
4
```
for i in range(3):
    print(i * 2)
Printing Even Numbers

```python
# our solution 0, 1, 2
for i in range(3):
    print(i * 2)

# equivalently
for i in range(0, 6, 2):
    print(i)
```

- Start at 0
- Stop before 6
- Skip by 2 each time
Welcome to the CS106A game show!
Choose a door and win a prize
Door: 2
You chose door 2
You win $100

* To be delivered via amazon gift cards

Piech + Sahami, CS106A, Stanford University
Choose a Door

door = int(input("Door: "))
# while the input is invalid
while door < 1 or door > 3:
    # tell the user the input was invalid
    print("Invalid door!")
    # ask for a new input
    door = int(input("Door: "))
The Door Logic

prize = 4

if door == 1:
    prize = 2 + 9 // 10 * 100

elif door == 2:
    locked = prize % 2 != 0
    if not locked:
        prize += 6

elif door == 3:
    for i in range(door):
        prize += i
def main() :
    for i in range(999999):
        print("You rock!")
        print("See you on Monday")
Today’s Route

You are here

Core Python

The River of Control Flow

Review

Conditions

Booleans

For Loops

Piech + Sahami, CS106A, Stanford University
Today’s Goal

1. Be able to use For / While / If in Python