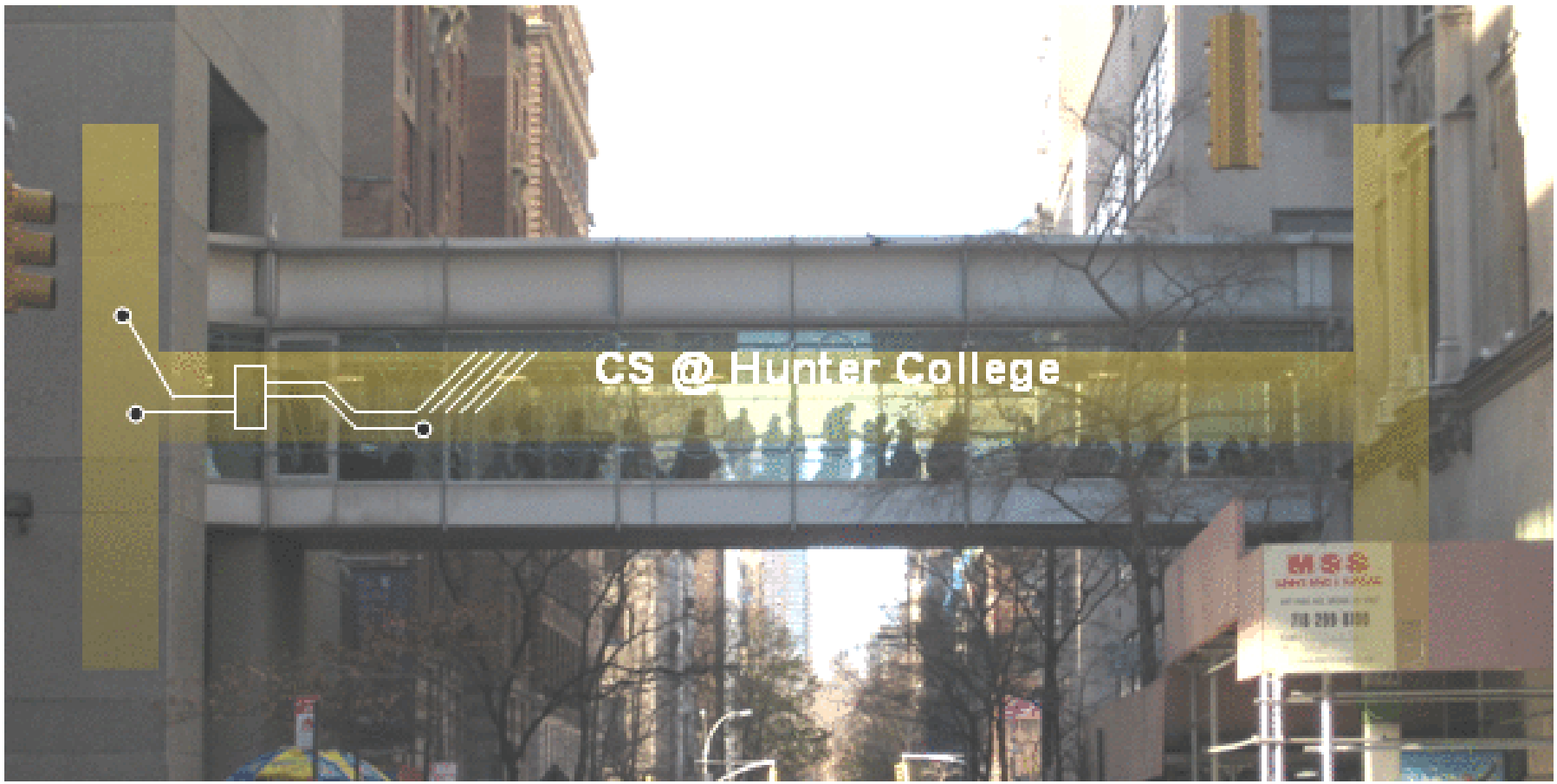
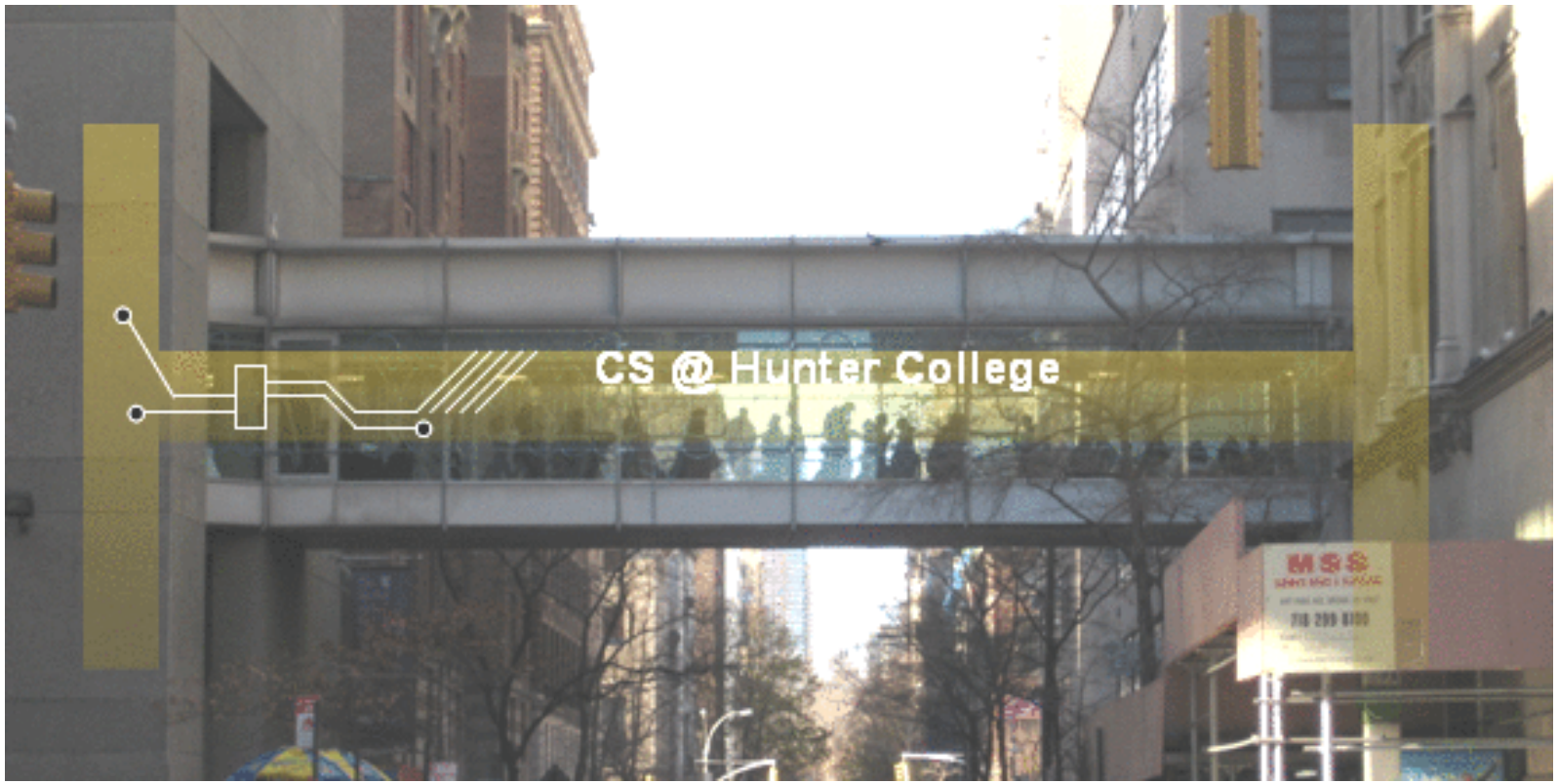
# CSci 127: Introduction to Computer Science




hunter.cuny.edu/csci

# Welcome




- Prof. Sakas, Department Chair and Co-Instructor

# Introductions

Katherine Howitt
Tutor Coordinator

Genady Maryash
Recitation Coordinator

Dr. Katherine St. John
Course Coordinator

Dr. William Sakas
Co-Instructor

# Introductions

Katherine Howitt
Tutor Coordinator

Genady Maryash
Recitation Coordinator

Dr. Katherine St. John
Course Coordinator

Dr. William Sakas
Co-Instructor

Not pictured: recitation instructors, undergraduate teaching assistants, faculty planning committee, technical & administrative staff.

# Syllabus

**CSci 127: Introduction to Computer Science**

*Catalog Description: 3 hours, 3 credits: This course presents an overview of computer science (CS) with an emphasis on problem-solving and computational thinking through 'coding': computer programming for beginners. Other topics include: organization of hardware, software, and how information is structured on contemporary computing devices. This course is pre-requisite to several introductory core courses in the CS Major. The course is also required for the CS minor. MATH 12500 or higher is strongly recommended as a co-req for intended Majors.*

# Syllabus

**CSci 127: Introduction to Computer Science**

*Catalog Description: 3 hours, 3 credits: This course presents an overview of computer science (CS) with an emphasis on problem-solving and computational thinking through 'coding': computer programming for beginners. Other topics include: organization of hardware, software, and how information is structured on contemporary computing devices. This course is pre-requisite to several introductory core courses in the CS Major. The course is also required for the CS minor. MATH 12500 or higher is strongly recommended as a co-req for intended Majors.*

(Show syllabus webpage)

# Introductions: Your Turn

- Introduce yourself to two classmates (that you have not met before).
- Write down names & interesting fact on lecture slip.

# Course Topics

- Overview of Computer Science
- Problem Solving & Computational Thinking
- Hardware & software of contemporary computing devices
- Programming in Python & C++
- Logical Circuits
- Overview of Machine Language
- Introduction to Unix (& command line interface)

# Today's Topics

- Introduction to Python
- Definite Loops (`for`-loops)
- Turtle Graphics
- Algorithms

# First Program: Hello, World!

```
#Name:  Thomas Hunter
#Date:  September 1, 2017
#This program prints:  Hello, World!

print("Hello, World!")
```

# First Program: Hello, World!

```
#Name:  Thomas Hunter                          ← These lines are comments
#Date:  September 1, 2017                      ← (for us, not computer to read)
#This program prints:  Hello, World!           ← (this one also)

print("Hello, World!")                         ← Prints the string "Hello, World!" to the screen
```

# First Program: `Hello, World!`

```
#Name:  Thomas Hunter                          ← These lines are comments
#Date:  September 1, 2017                      ← (for us, not computer to read)
#This program prints:  Hello, World!           ← (this one also)

print("Hello, World!")                         ← Prints the string "Hello, World!" to the screen
```

- Output to the screen is: `Hello, World!`
- Can replace `Hello, World!` with another string to be printed.

# Variations on Hello, World!

```
#Name:  L-M Miranda
#Date:  Hunter College HS ‘98
#This program prints intro lyrics

print(’Get your education,’)
```

# Variations on Hello, World!

```
#Name:  L-M Miranda
#Date:  Hunter College HS ‘98
#This program prints intro lyrics

print(’Get your education,’)
print("don’t forget from whence you came, and")
print("The world’s gonna know your name.")
```

# Variations on Hello, World!

```
#Name:  L-M Miranda
#Date:  Hunter College HS '98
#This program prints intro lyrics

print('Get your education,')
print("don't forget from whence you came, and")
print("The world's gonna know your name.")
```

- Each print statement writes its output on a new line.

# Variations on Hello, World!

```
#Name:  L-M Miranda
#Date:  Hunter College HS '98
#This program prints intro lyrics

print('Get your education,')
print("don't forget from whence you came, and")
print("The world's gonna know your name.")
```

- Each print statement writes its output on a new line.
- Resulting in three lines of output.

# Variations on Hello, World!

```
#Name:  L-M Miranda
#Date:  Hunter College HS ‘98
#This program prints intro lyrics

print(’Get your education,’)
print("don’t forget from whence you came, and")
print("The world’s gonna know your name.")
```

- Each print statement writes its output on a new line.
- Resulting in three lines of output.
- Can use single or double quotes, just need to match.

# Turtles Introduction

- A simple, whimsical graphics package for Python

# Turtles Introduction

- A simple, whimsical graphics package for Python
- Dates back to Logos Turtles in the 1960s

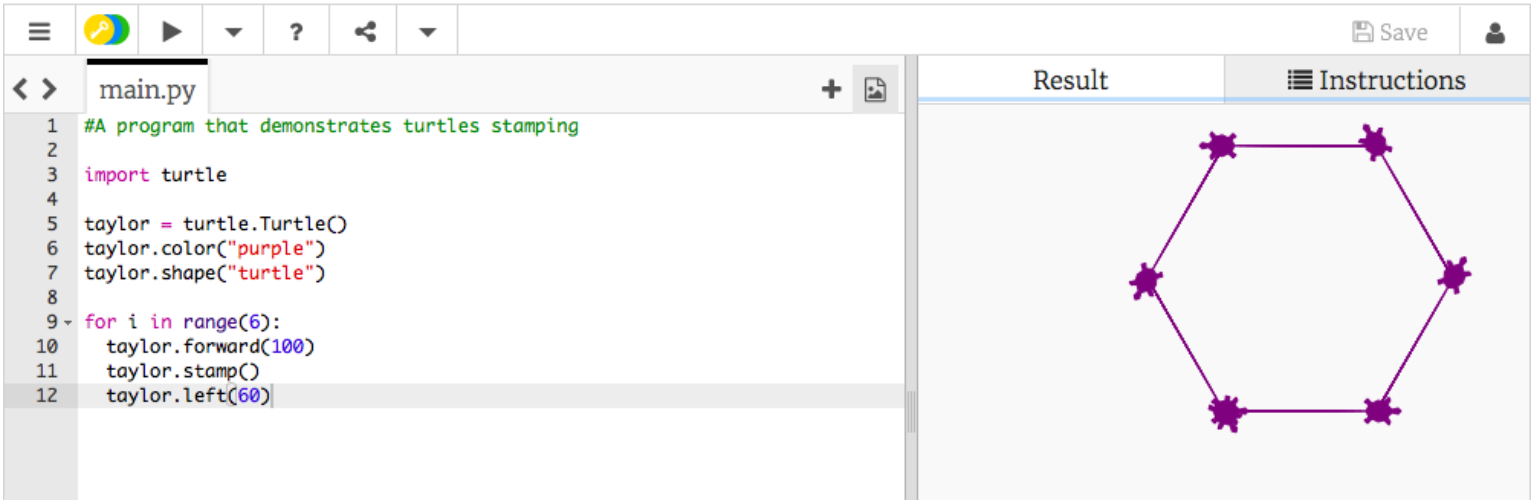
# Turtles Introduction

- A simple, whimsical graphics package for Python
- Dates back to Logos Turtles in the 1960s
- (Demo from webpage)

# Turtles Introduction

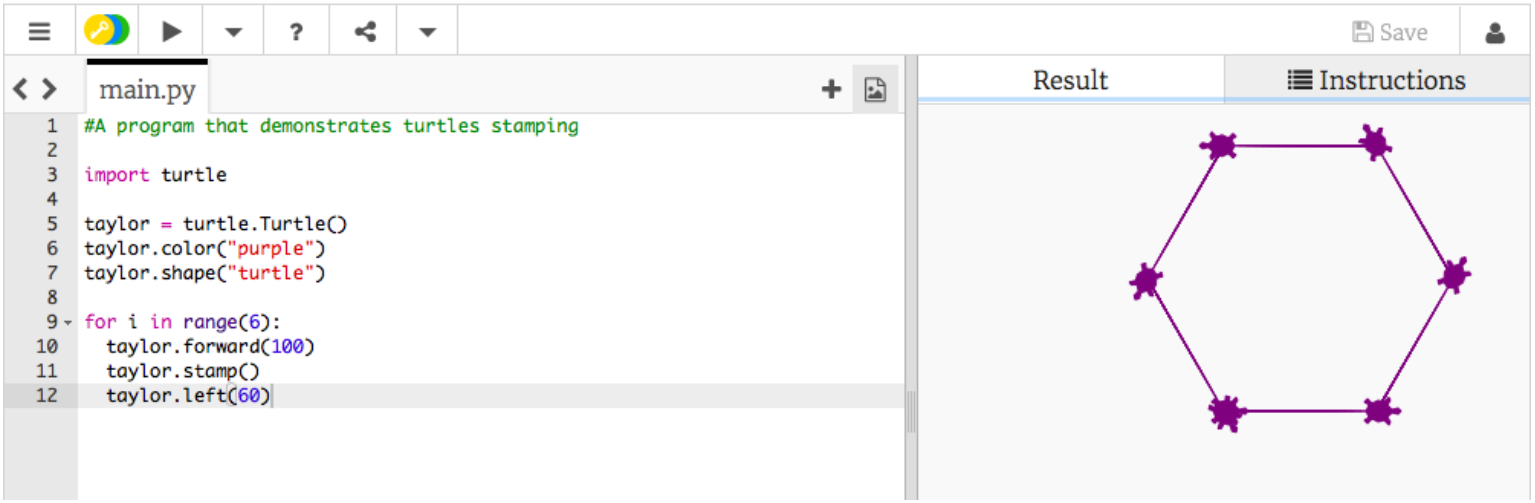
- A simple, whimsical graphics package for Python
- Dates back to Logos Turtles in the 1960s
- (Demo from webpage)
- (Fancier turtle demo)

# Turtles Introduction



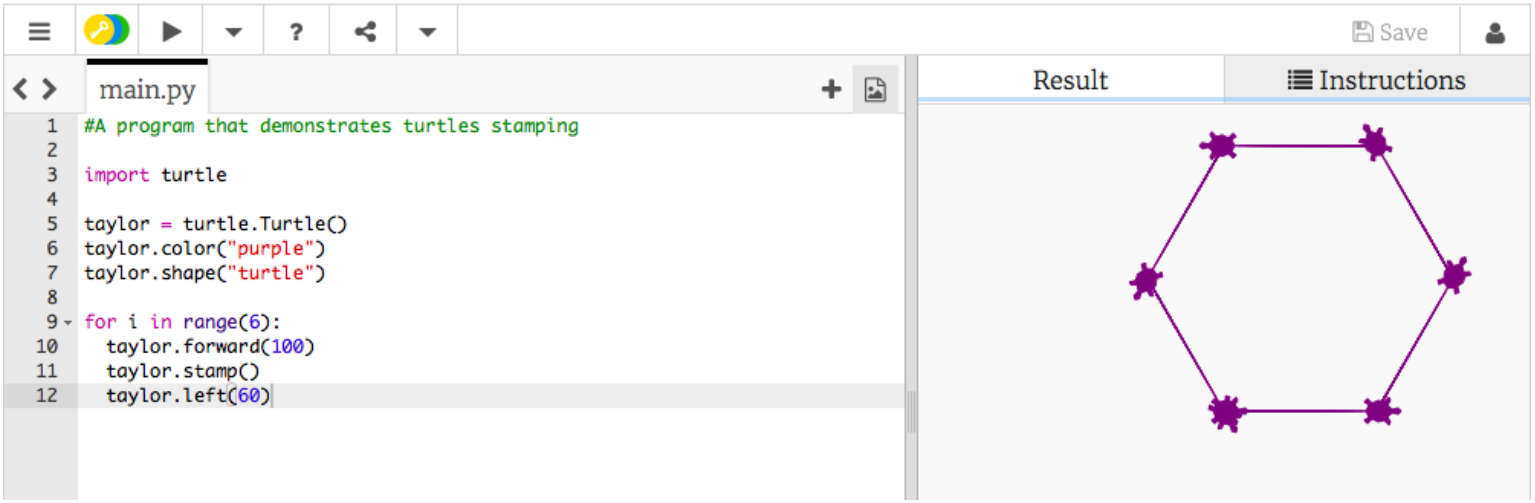

- Creates a turtle, called `taylor`

# Turtles Introduction



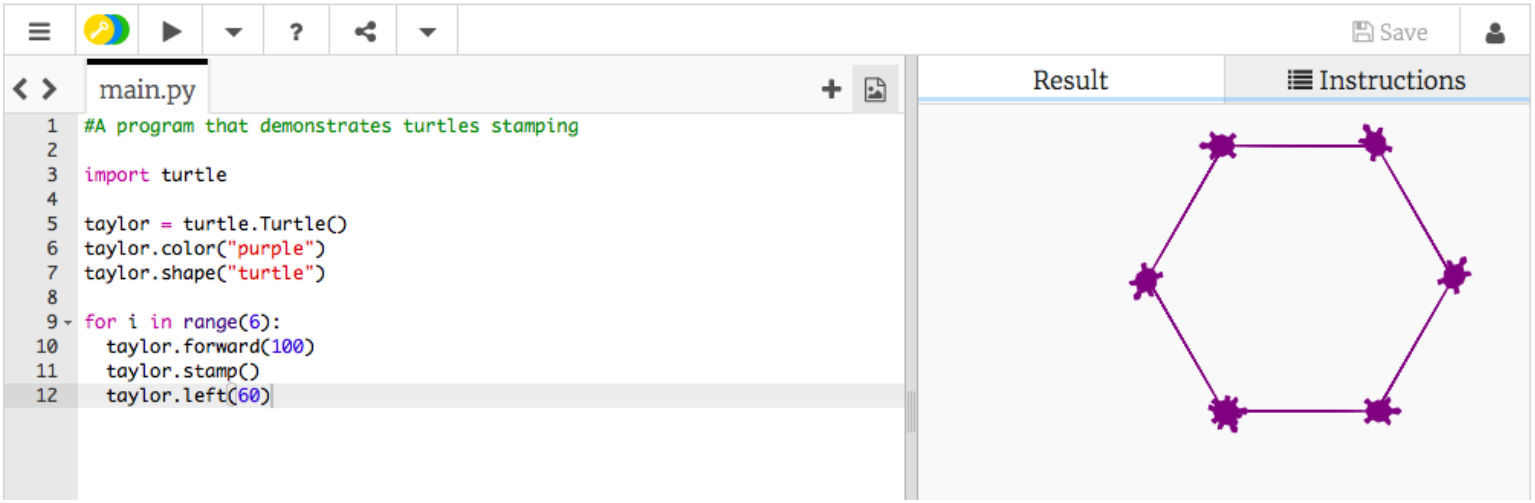

- Creates a turtle, called `taylor`
- Changes the color (to purple) and shape (to turtle-shaped)

# Turtles Introduction




- Creates a turtle, called `taylor`
- Changes the color (to purple) and shape (to turtle-shaped)
- Repeats 6 times:

# Turtles Introduction




- Creates a turtle, called `taylor`
- Changes the color (to purple) and shape (to turtle-shaped)
- Repeats 6 times:
- Move forward; stamp; and turn left 60 degrees

# Group Work

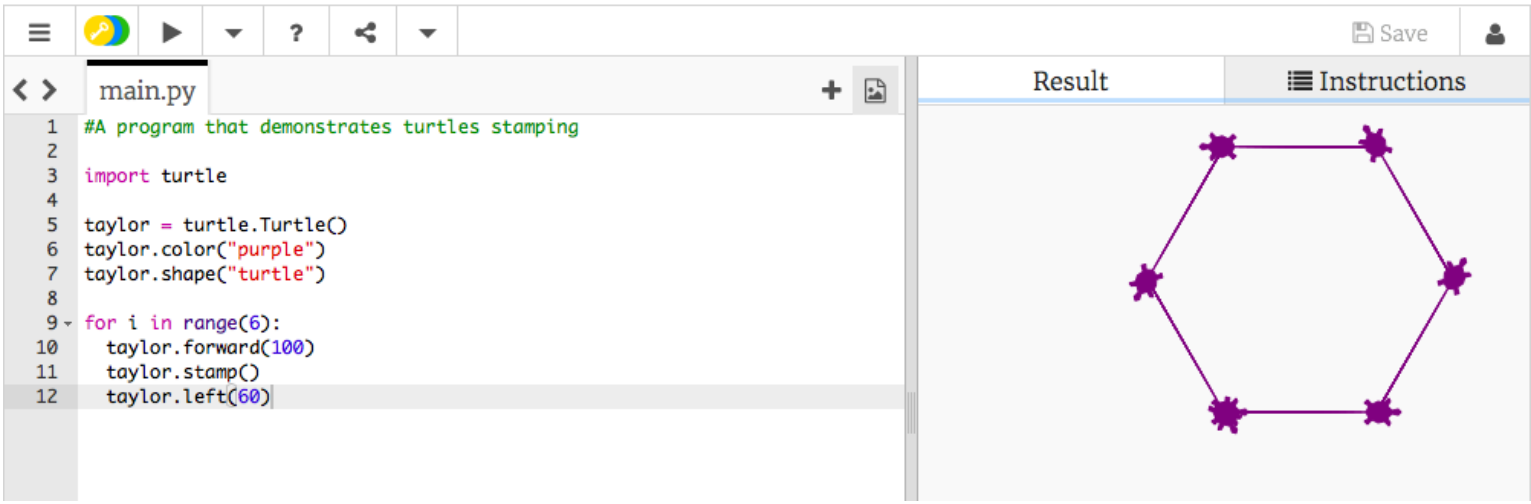
Working in pairs or triples:

1. Write a program that will draw a 10-sided polygon.
2. Write a program that will repeat the line:

   ```
   I'm lookin' for a mind at work!
   ```

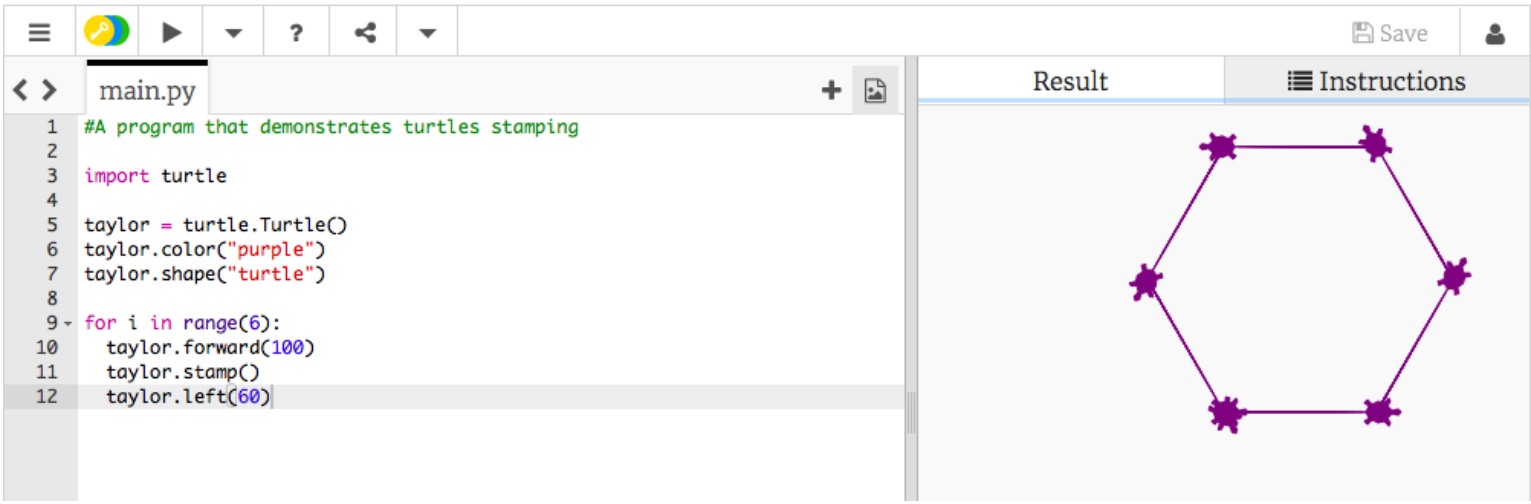
   three times.

# Decagon Program



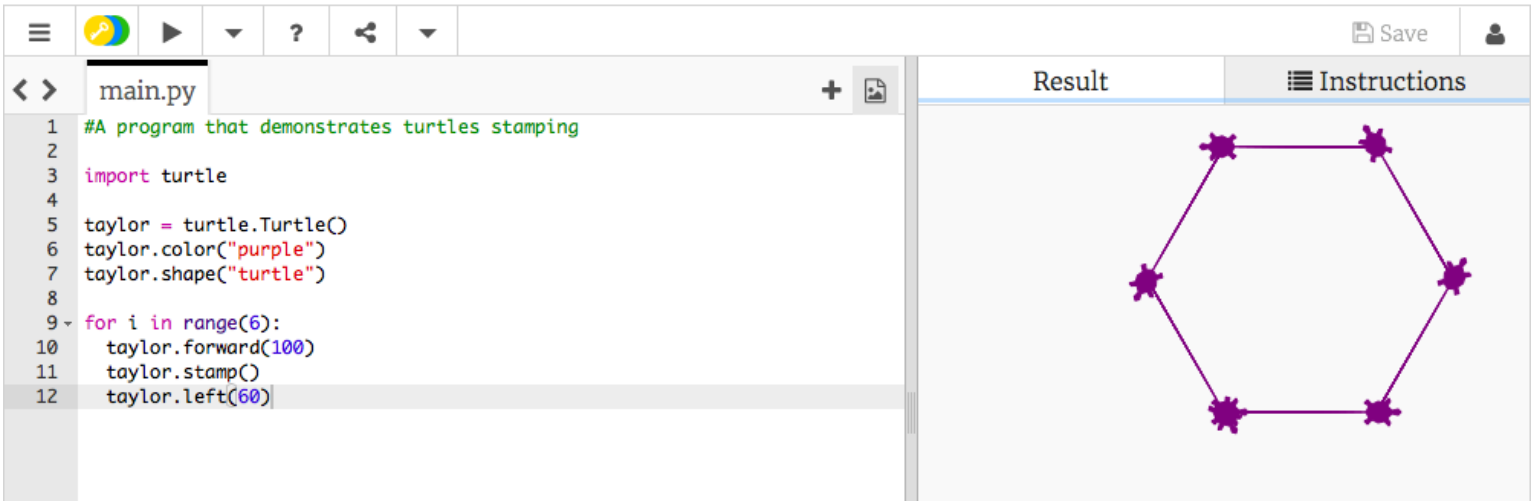

- Start with the hexagon program.

# Decagon Program




- Start with the hexagon program.
- Has 10 sides (instead of 6), so change the `range(6)` to `range(10)`.

# Decagon Program




- Start with the hexagon program.
- Has 10 sides (instead of 6), so change the `range(6)` to `range(10)`.
- Makes 10 turns (instead of 6),
  so change the `taylor.left(60)` to `taylor.left(360/10)`.

# Work Program

3. Write a program that will repeat the line:

   ```
   I'm lookin' for a mind at work!
   ```

   three times.

# Work Program

3. Write a program that will repeat the line:

   ```
   I'm lookin' for a mind at work!
   ```

   three times.

- Repeats three times, so, use `range(3)`:

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

# Work Program

3. Write a program that will repeat the line:

   ```
   I'm lookin' for a mind at work!
   ```

   three times.

- Repeats three times, so, use `range(3)`:

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

- Instead of turtle commands, repeating a print statement.

# Work Program

3. Write a program that will repeat the line:

   `I'm lookin' for a mind at work!`

   three times.

- Repeats three times, so, use `range(3)`:

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

- Instead of turtle commands, repeating a print statement.
- Completed program:

```
# Your name here!
for i in range(3):
    print("I'm lookin' for a mind at work!")
```

# What is an Algorithm?

From our textbook:

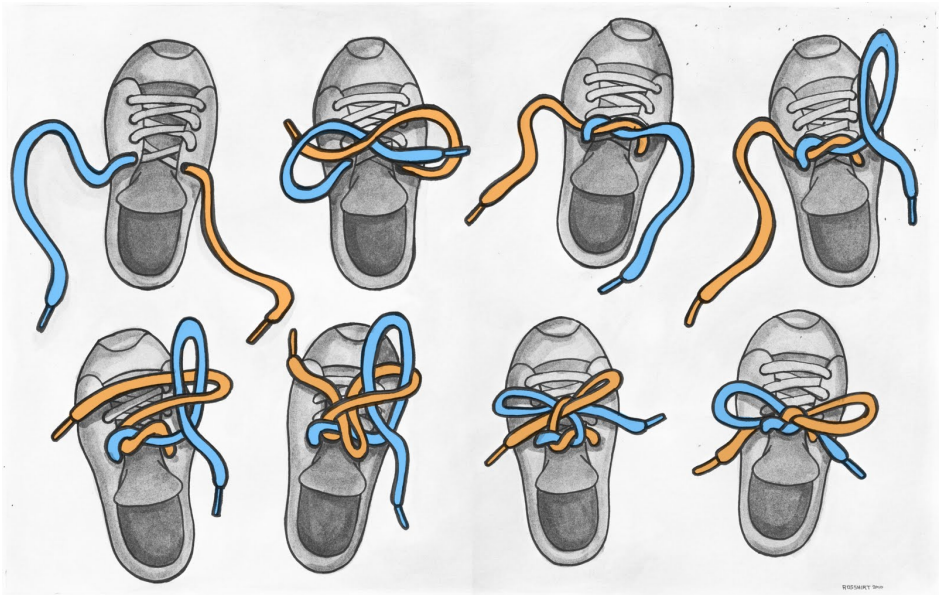
- An algorithm is a process or set of rules to be followed to solve a problem.

# What is an Algorithm?

From our textbook:

- An algorithm is a process or set of rules to be followed to solve a problem.
- Programming is a skill that allows a computer scientist to take an algorithm and represent it in a notation (a program) that can be followed by a computer.

# Group Work



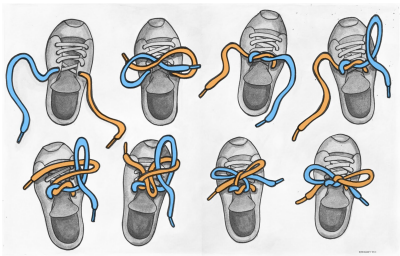
Rose Reible, prezi.com

Working in pairs or triples:

1. Write an algorithm to tie your shoes.

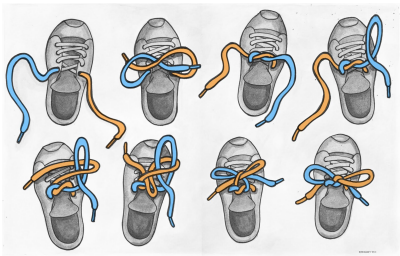
# Recap

- Writing precise algorithms is difficult.

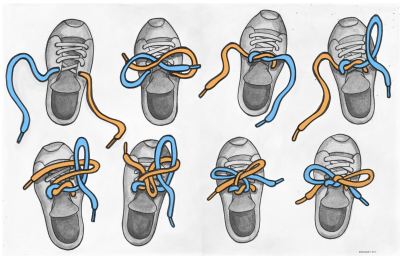


# Recap

- Writing precise algorithms is difficult.
- In Python, we introduced:

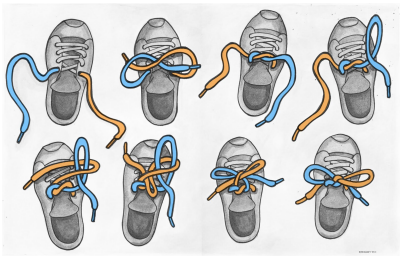


# Recap



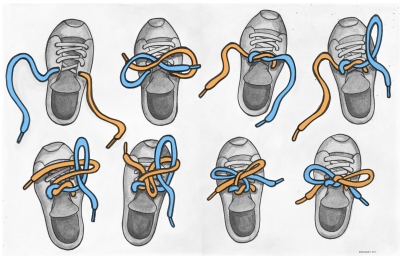
- Writing precise algorithms is difficult.
- In Python, we introduced:
  - strings, or sequences of characters,

# Recap



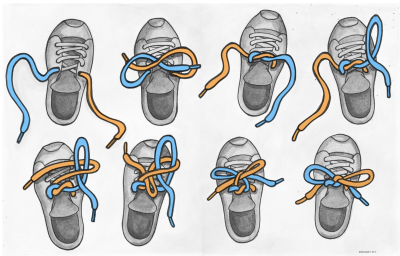
- Writing precise algorithms is difficult.
- In Python, we introduced:
  - strings, or sequences of characters,
  - `print()` statements,

# Recap



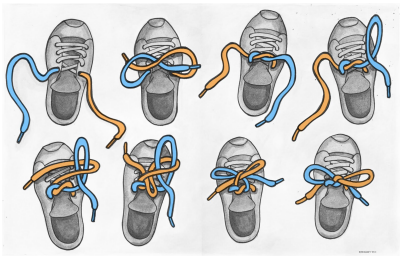
- Writing precise algorithms is difficult.
- In Python, we introduced:
  - strings, or sequences of characters,
  - `print()` statements,
  - `for`-loops with `range()` statements, &

# Recap



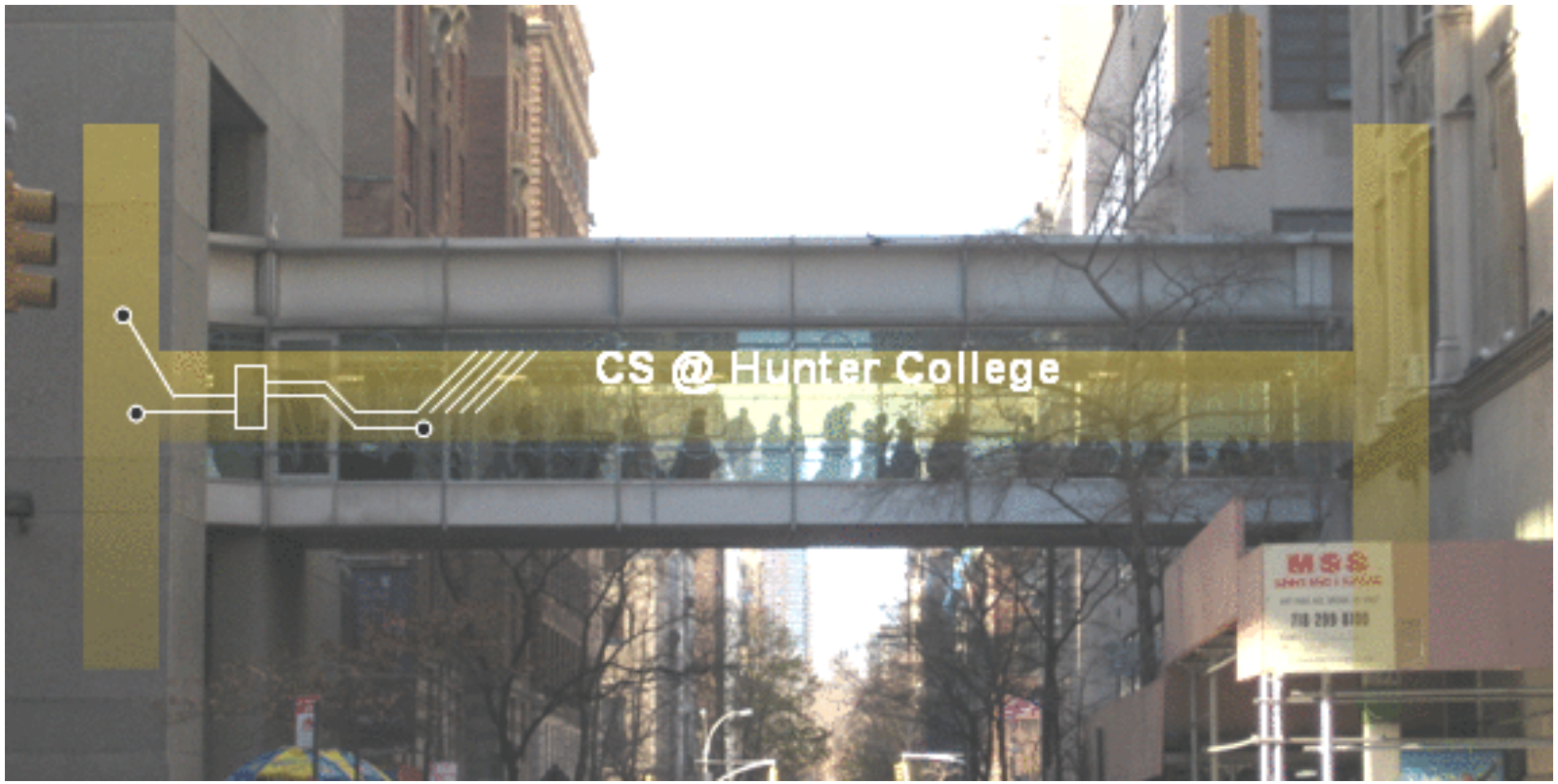
- Writing precise algorithms is difficult.
- In Python, we introduced:
  - strings, or sequences of characters,
  - `print()` statements,
  - `for`-loops with `range()` statements, &
  - variables containing turtles.

# Recap



- Writing precise algorithms is difficult.
- In Python, we introduced:
  - strings, or sequences of characters,
  - `print()` statements,
  - `for`-loops with `range()` statements, &
  - variables containing turtles.
- On lecture slip, write down a topic you wish we had spent more time.

# Lecture Slips & Writing Boards




- Turn in lecture slips & writing boards as you leave...