Data science for everyone

Prof. Jones-Rooy & Prof. Policastro

Feb. 10, 2020

3.1: Introduction to Python

ANNOUNCEMENTS

1. Lab 0 due Wed., Feb. 12, 8p
2. Homework 1 due Tue., Feb. 18, 8p
3. It's DS major/minor and DS-CS joint major declaration kickoff day!
   1. Declare a major/minor in DS here!
   2. Declare a joint major in DS-CS here!

Questions:
cds-undergraduate@nyu.edu

Outline

0. Wrap up causality from Lecture 2.2

1. Programming background
2. The art & science of programming
3. Python building blocks

SEE LECTURE 2.2 SLIDES

One more example

Combining selecting on the DV with necessary and sufficient conditions

Selecting on the DV: Example from the *NYT* Nonfiction Bestseller List

Book	DV: Best-seller?	Memoir?	Female author?	Political figure?	Social issue?	Colorful cover? (non-white background)
Becoming	Y	Y	Y	Y	Y	Y
Educated	Y	Y	Y	N	Y	Y
Maid	Y	Y	Y	N	Y	N

Being written by a political figure is not a **necessary** condition for being a bestselling nonfiction book

Having a colorful cover is not a **necessary** condition for being a bestselling nonfiction book

Potential necessary conditions: Memoir, female author, social issue; **Can't say sufficiency w/ just bestsellers**

Book	DV: Best-seller?	IV: Memoir?	IV: Female author?	IV: Political figure?	IV: Social issue?	IV: Colorful cover? (non-white background)
Becoming	Y	Y	Y	Y	Y	Y
Educated	Y	Y	Y	N	Y	Y
Maid	Y	Y	Y	N	Y	N
Women Rowing North	N	N	Y	N	Y	Y
Inheritance	N	Y	Y	N	Y	Y
Factfulness	N	N	N	N	Y	N

Being about a social issue may be a necessary but not sufficient condition

In this sample of 6 books only, being female also may be necessary but not sufficient

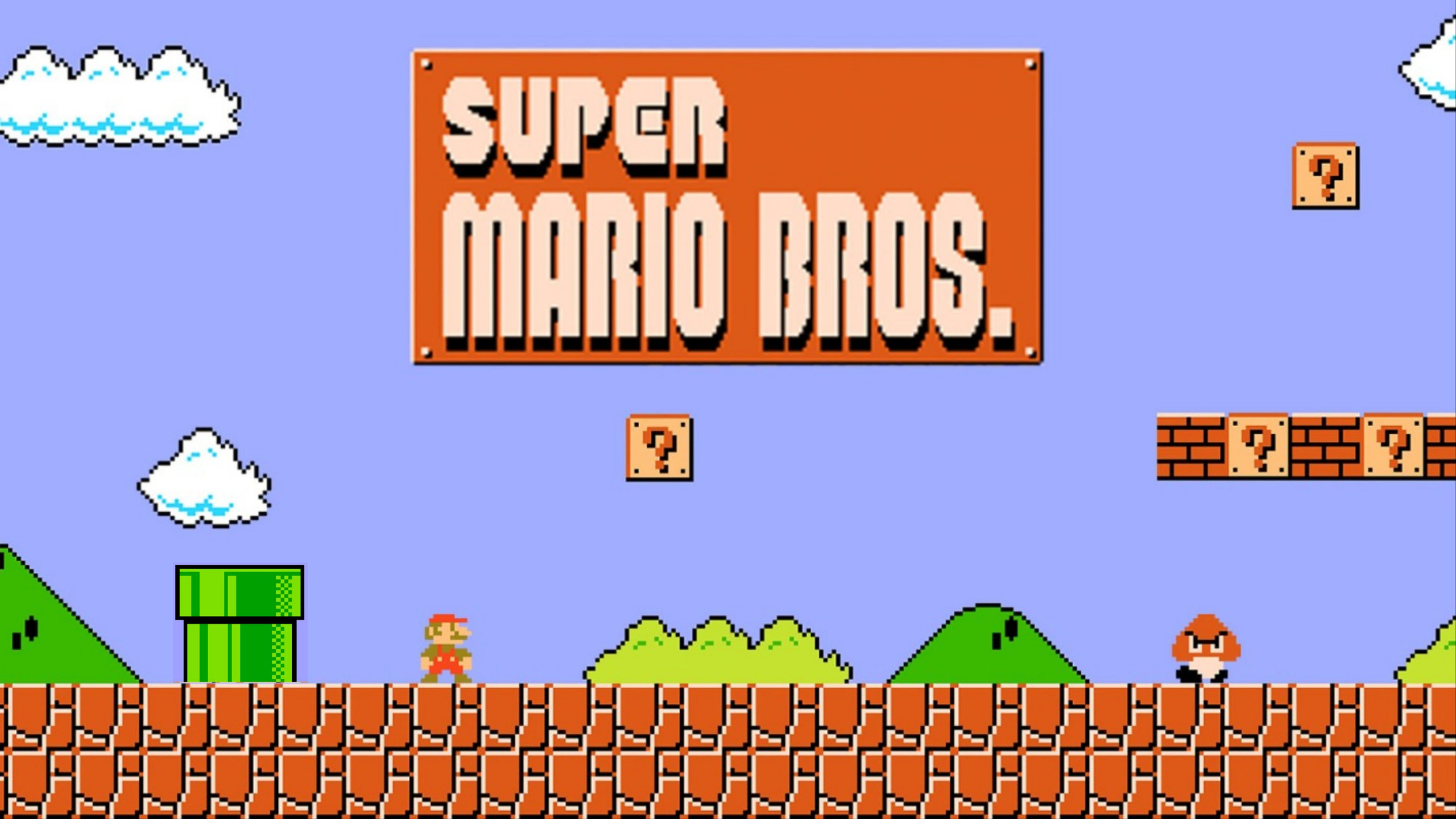
Outline

0. Wrap up causality from Lecture 2.2

1. Programming background
2. The art & science of programming
3. Python building blocks

COMPUTER PROGRAM

- Set of instructions to tell a computer what to do
- It's a way of telling a computer how to solve a specific problem
  - Usually requires telling it to solve a set of smaller problems first
- Example problem: Making a video game
  - Sub-problems:
    - How far or fast to move the character when you indicate →
    - What happens when the character runs into an object
    - What the background looks like, how it changes


SUPER
MARIO BROS.

PROGRAMMING

- Lots of approaches to programming
  - Write entire programs from scratch
  - Buy other people's programs completely (e.g., an app)
  - Fold existing programs into your own (e.g., import a module or package)
  - Use an existing program and modify it to do what you want

COMPONENTS OF A PROGRAM

- Source code:
  - Human-readable code that you're writing to tell the computer what to do
- Translation to computer:
  - Compiler: translates source code into object code or machine code
  - Interpreter: executes source code line by line
- Interpreters: Easier to build, slower to run
- Writing source code: Programming, coding, software developing
  - Individually, in teams, in massive groups

Outline

0. Wrap up causality from Lecture 2.2

1. Programming background
2. The art & science of programming
3. Python building blocks

PROGRAMMING

- One often hard part is figuring out what the sub-problems are
  - It's a high-level logic and architecture challenge more than knowing the vocabulary or a long list of commands (tools)
  - But, more tools → more solutions you can think of
- It can be both conceptually challenging and often just flat-out time-consuming
  - To build all the steps
  - Debugging!!! (BTW: Scientific method works here, too!)
- Creativity is an asset
  - Analogies: Designing experiments; filmmaking

Outline

0. Wrap up causality from Lecture 2.2

1. Programming background
2. The art & science of programming
3. Python building blocks

PYTHON BUILDING BLOCKS

Arithmetic

"Expressions" in textbook

Naming objects

"Names" using an assignment statement in textbook

Built-in functions

Call expressions in textbook

Modules & packages

More on this in lecture 3.2

Ch. 3 of *Inferential Thinking* (and all over the Internet!)

ARITHMETIC

- Mathematical expressions in Python
- Basically, using Python as a calculator

```
2+3
```

```
4*8
```

```
12+8*4
```

NAMING OBJECTS

- Assign a name to something you've created
- Allows you to use it again and again without re-creating
- Generally, we try to assign names that are useful, clear, informative, easy to re-type (for time and so we are less likely to make errors)!

```
price = 2+3
```

```
myformula = 12+8*4
```

BUILT-IN FUNCTIONS

- Python has built-in functions that perform specific operations
- We can call them using specific syntax (generally relatively intuitive!)
- If we want to do something more than what's built in, we either need to import a module or package (lecture 3.2) or create our own (in coming weeks)

```
abs(-6)
```

```
max(4, 8, 9)
```

PACKAGES & MODULES

- Beyond relatively basic/general built-in functions, we will want to conduct more complicated processes with our code
- We can use more specialized and sophisticated built-in functions by importing entire packages or specific modules within that package
- These come with their own built-in functions, which we can use directly, or fold into further, more complicated, functions we write ourselves
- Every (good) package has its own documentation and examples (and informal support on forums, etc.)

```
import math
```

```
import numpy as np
```

Outline

0. Wrap up causality from Lecture 2.2

1. Programming background
2. The art & science of programming
3. Python building blocks