Rotman

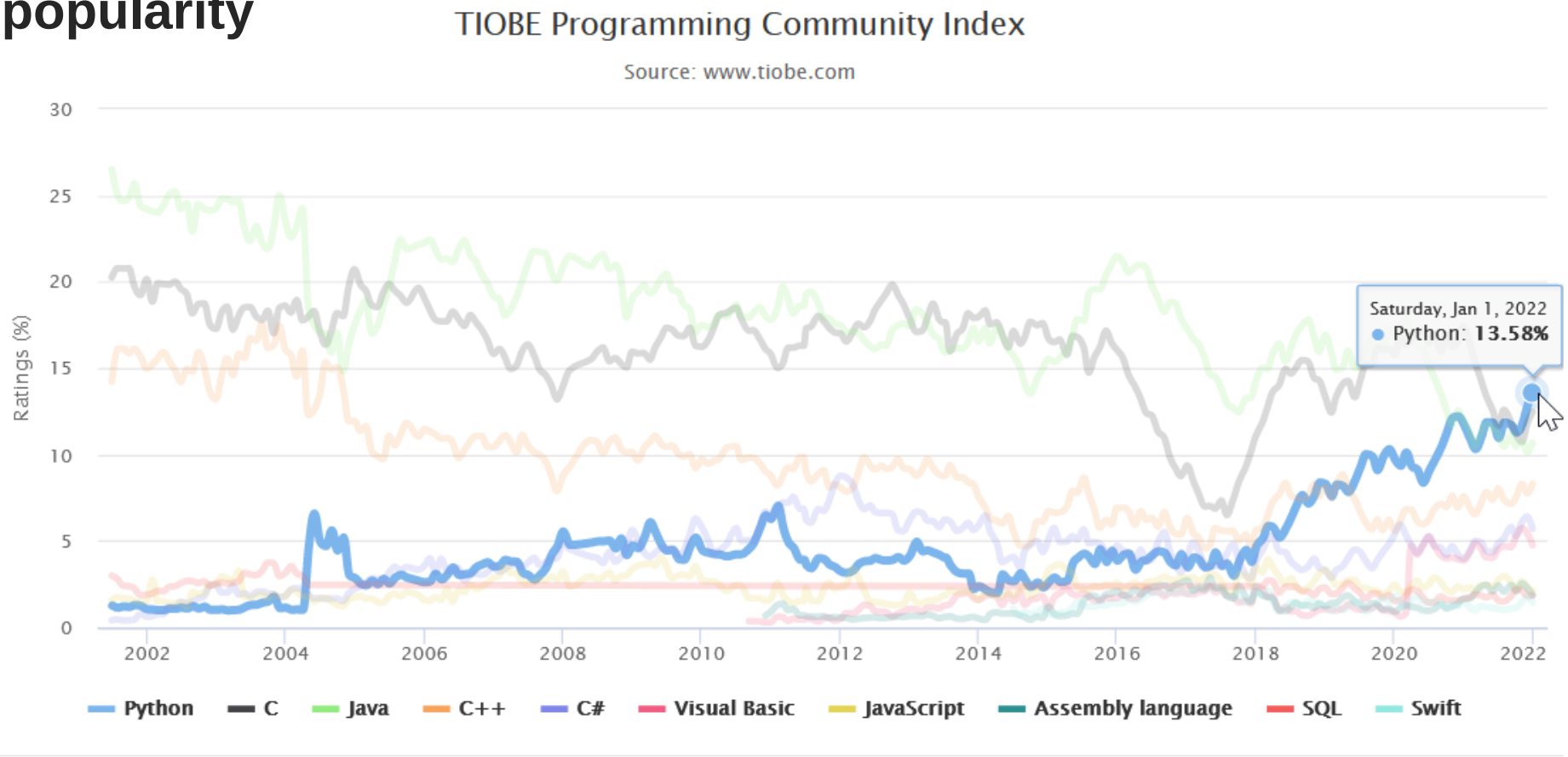
BASIC PYTHON INTRODUCTION

February 8, 2022 Prepared by Niti
TDMDAL & FinHUB

Rotman School of Management
UNIVERSITY OF TORONTO

Python's Popularity

Python gained the highest increase in one year in TIOBE index of programming language popularity



Source: https://www.tiobe.com/tiobe-index/

Rotman

Python’s Popularity

1. Statistical analysis
2. Scientific computing
3. Machine learning
4. Data visualization
5. Artificial intelligence
6. Others:
    i. Scripting & automation
    ii. Web development
    iii. Systems testing & prototyping
    iv. Desktop & mobile applications
    v. Education!

Getting Python

- **Anaconda**
  - Anaconda installation is the **recommended** method for getting **Python**.
  - Anaconda is a package manager that allows installing many applications at once.
  - **Installation Guide - Video :** https://youtu.be/Z1Yd7upQsXY?t=4m19s *timestamped to start minutes 19 seconds watch until 5:59*
  - **Installation Guide - Text** https://bit.ly/2FRyakD

Writing Python Codes

- **Jupyter Notebook**
  - Among other applications, Anaconda also installs **Jupyter notebook**,
  - Jupyter Notebook is an application where you can easily write and execute Python codes.

- **Google Colab**
  - Google's colaboratory, which is a free Jupyter notebook environment that requires no setup and runs entirely on Google's cloud.

Rotman

1. VARIABLES & BASIC DATA TYPES

Variables

Variables Definition

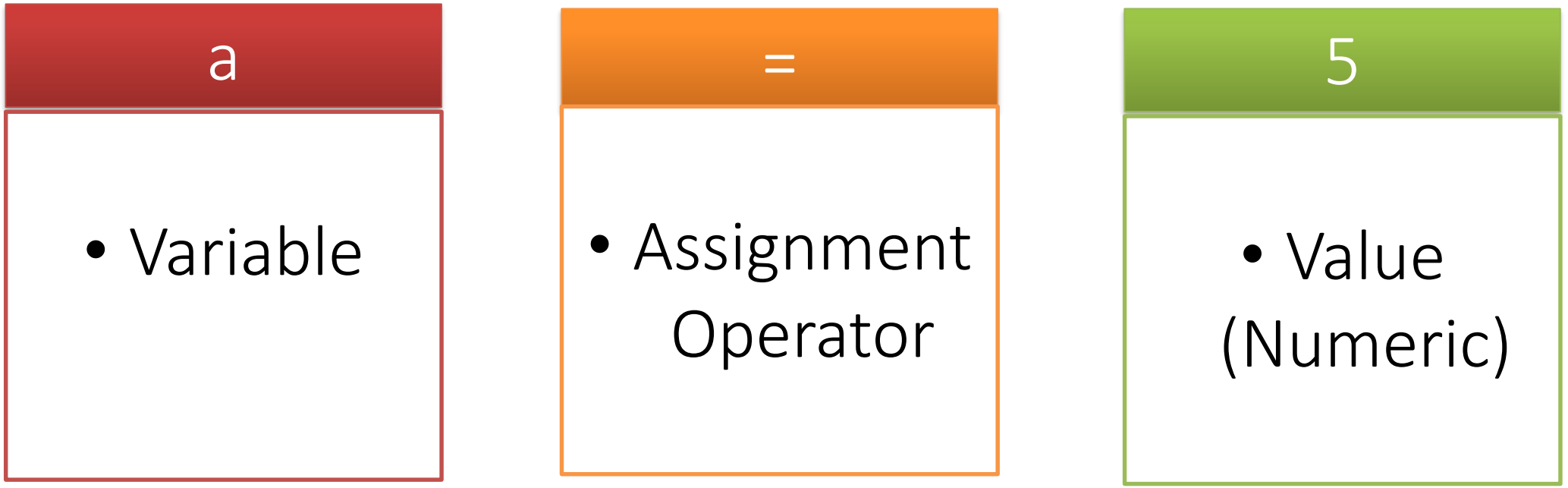
- A program works with values.
- Variable is a name that refers to a value.
- A powerful feature of programming language is the ability to manipulate variables

Creating a Variable

- Python provides assignment operators to assign values to a variable.
- "=” is the most common assignment operator.
- Once a variable is created, its value can be called by referring to it by its name.
- Variables can be used as the assigned value, until the value is changes or the variable is deleted.
- Be careful not to overwrite a variable name unless you really mean to do so!

Assigning Value to a Variable

- This is an example of an assignment statement.



Rotman

Variable Naming Convention

- Python admits that the naming convention can be a bit of a mess.
- As you start to program and create variables, this will become more evident.
- Nevertheless, programmers should aim to follow recommended naming conventions.
- Some variable names are not accepted altogether.

Variable Names and Keywords

- Python has some keywords that defines its rule and structure.
- These keywords cannot be used as a variable name.
- Example of Python keywords are:

and	as	assert	break	class	continue
def	del	elif	else	except	exec
finally	for	from	global	if	import
in	is	lambda	not	or	pass
print	raise	return	try	while	with
yield					

Rotman

Operators

Operators

- Operators are special symbols or keywords used to perform some designated computation.

- The values that an operator acts on are called operands. If variable name is used in the place of operands, it is replaced with its value before the operation is performed.

- Operators are mapped to various functions such as addition, subtraction, assigning value, comparing values, combining two or more operations, etc.

Rotman

Types of Operators

- There are many types of operators in Python. The most common of them are:

Operator	Function
Assignment	Assign values to variables
Arithmetic	perform arithmetic operation on numeric values (works differently on strings)
Comparison	Compare two values
Logical	test if multiple conditions are satisfied at the same time
Membership	test if a sequence with a specified value is present in another value.

Examples of Operators

Assignment	Arithmetic	Comparison	Logical	Membership
X = 5	+ (Addition)	== (Equal to)	and (Use: X < 5 and X < 10)	in (Use: x in y)
X += 5 (same as X = X+5)	- (Subtraction)	!= (Not equal to)	or (Use: X < 5 or X = 10)	not in (Use: x not in y)
X *= 5 (Same as X = X*5)	* (Multiplication)	> (is greater than)	not (Use: not(x < 5 and x = 10)	
	** (Exponent)	< (is less than)		

Rotman

Inplace Operators

- Some operators are inplace operators because they directly change the content of a given linear algebra, or vector or matrices without making a copy.

- For example, for `x=2`, the line of code, `x +=3`, will change the value of `x` to `5`.

- Here, `+=` is an inplace operator because it changes the value of x “in place”.

Data Types

Data Types

- Variables store values that belong to certain data types. Numbers are usually of data type integer, float or Boolean. Texts are of data type string. A hierarchy of Python's standard data types can be found here.

- A data type is simply an attribute of data which tells the interpreter how the programmer intends to use the data. Data types are essential because they define what we can do with values and how they can be stored.

- Normally, the data type of a variable is set when you assign a certain value to it. However, Python also provides built-in functions to set a specific data type if you want.

- The **type** function can be used to get the data type of a variable.

Rotman

Examples of Basic Data Types

Data Type	Examples	Function
Integer	5, 7, 19,	int()
Float	5.0, 25.81, 0.9997	float()
Boolean	True, False	bool()
String	“2021-1-15”, “aapl”, “CAD”	str()

Numeric Data Types

Integers

- Integers are positive or negative whole numbers with no decimal points.
- Integers are not only useful for mathematical calculation but also for indexing.

Floats

- Floating-point numbers, known as floats, are numbers with a decimal point on it.
- Floats are particularly useful when precision in mathematical calculation is important.

Booleans

- Boolean data type have two values – True or False.
- Booleans are extremely helpful when we want to check if a condition has been met.

String Data Type

- String data types are special data types.
- Most arithmetic operations are illegal on strings.
- However, two arithmetic operators can be used with strings:
  - The + operator, which concatenates strings
    `'a' + 'a'` results in `'aa'`
  - The * operator, which performs repetitions
    'a'*4 results in `'aaaa'`
- We learn more about strings in the next module.

None Data Type

- None in Python is used to define a null value, an undefined value or no value at all.
- None is a keyword and belongs to NoneType data type.
- None is useful when comparing whether a value is missing or null as any value other than None will always return False.
    - None is not a 0.
    - None is not the same as False.
    - None is not an empty string.

Data Type Conversion

- The process of converting the value of one data type to another is called type conversion.

Implicit Data Conversion	• Data type conversion without the user's involvement. • For example, Python promotes the conversion of integer (lower data type) to float (higher data type) to avoid data loss.
Explicit Data Conversion	• Data type conversion to a desired data type by user. • Also called typecasting. • Python has built-in functions such as int(), float() and str() to support data type changes by users.

Rotman

String Data Types

What are Strings?

- Strings are values enclosed in either single or double quotes.
- Strings are the most complex of the four basic data types.
- Formally, strings are arrays of bytes representing unicode characters, which means characters, symbols, numbers, whitespaces are all strings.
- Python's support for unicode standard is what makes it possible to store (in Python 3) characters written in any language in the world.
- Strings in Python are **immutable**, meaning once a string variable is created it cannot be changed.

String is a Sequence

- If you recall from Python's standard data type hierarchy, strings fall under the **sequence** data type group.
- Sequences in Python are data types that can store one or more element. Thus, strings are sequence of characters.
- A one-element sequence is simply a sequence of one character.
- Some examples of string are:
  `'a', 'aapl', 'Apple Inc.', 'Apple Inc. - aapl'`

Rotman

Strings can be Indexed

- Sequences in Python can be indexed by its position.
- Positions of each element in a sequence can be indicated by an integer value called **index**.
- Indexing ability of string makes it possible to extract part(s) of string for further manipulation.
- Python indexing starts from 0 not 1.

```
In [1]: astring = 'Apple Inc.'
        astring[0]

Out[1]: 'A'
```


Rotman

String has Length

- Sequences have length, which is equal to the number of elements in that sequence.
- A string '`Apple Inc.`' has 10 characters and therefore, has a length of 10.
- Python provides a built-in function `len()` which can be used to easily get the length of sequences.

```
In [2]: astring = 'Apple Inc.'
        len(astring)

Out[2]: 10
```


Rotman

Looping through a String

- It is also possible to loop through each element of a sequence in Python.
- So, we can also loop through a string.
- We will learn more about looping in subsequent modules. For now, here is an example of looping through each character of a string and printing it on the console:

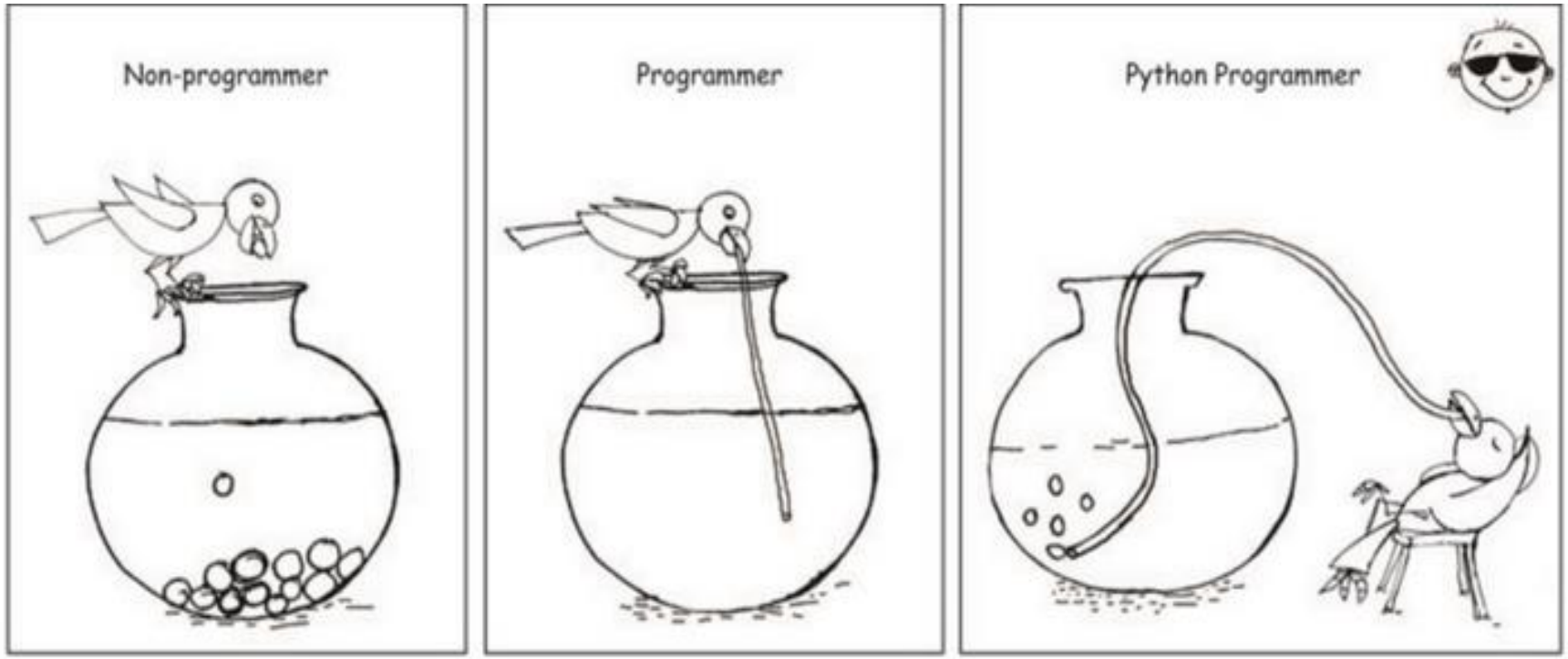
```
In [3]:  for i in 'aapl':
             print(i)

a
a
p
l
```


Rotman

String Methods

- Methods can be used to manipulate strings and get new values.
- A **dot notation** (".") is used to access these methods. Simply type the string variable followed by a dot and then the method.
- Each method serves its own function. For example,
  - .capitalize() capitalizes the first character in a string
  - .count() counts the frequency of a specified substring within the string, and
  - .format() is useful to format specified values in a string.

Questions?



Who wants to become a Python Programmer?

Thank you