Course Introduction

BBM 101 - Introduction to Programming I

Hacettepe University
Fall 2015

Fuat Akal, Aykut Erdem, Erkut Erdem, Vahid Garousi

Slides based on material prepared by Ruth Anderson, Michael Ernst and Bill Howe in the course CSE 140
University of Washington

Welcome to BBM 101

- This course teaches core programming concepts with an emphasis on data manipulation tasks from science, engineering, and business
- Goal by the end of the semester: Given a data source and a problem description, you can independently write a complete, useful program to solve the problem

Course Staff

- Lecturers:
  - Asst. Prof. Dr. Fuat Akal
  - Asst. Prof. Dr. Aykut Erdem
  - Asst. Prof. Dr. Erkut Erdem
  - Assoc. Prof. Dr. Vahid Garousi

Course Staff

- TAs (teaching assistants):
  - Burçak Asal
  - Feyza Nur Çubukçuoğlu
  - Levent Karacan
  - Selim Yılmaz

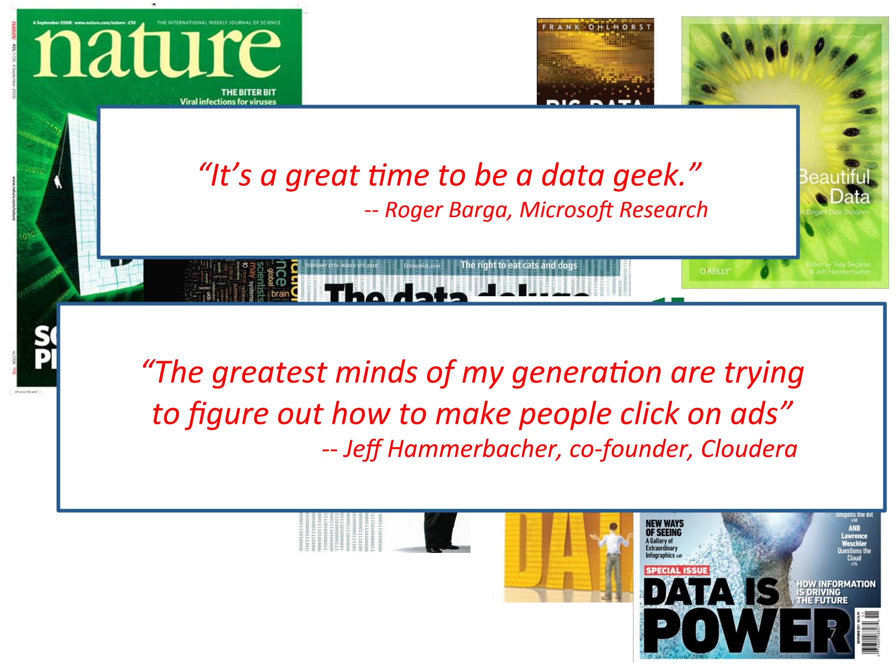
Do not hesitate to ask TAs for help!

Learning Objectives

- Computational problem-solving
  - Writing a program will become your "go-to" solution for data analysis tasks.
- Basic Python proficiency
  - Including experience with relevant libraries for data manipulation, scientific computing, and visualization.
- Experience working with real datasets
  - astronomy, biology, linguistics, oceanography, open government, social networks, and more.
  - You will see that these are easy to process with a program, and that doing so yields insight.

What This Course is not

- A "skills course" in Python
  - ...though you will become proficient in the basics of the Python programming language
  - ...and you will gain experience with some important Python libraries
- A data analysis / "data science" / data visualization course
  - There will be very little statistics knowledge assumed or taught
- A "project" course
  - the assignments are "real," but are intended to teach specific programming concepts
- A "software engineering" course
  - Programming is the starting point of computer science and software engineering



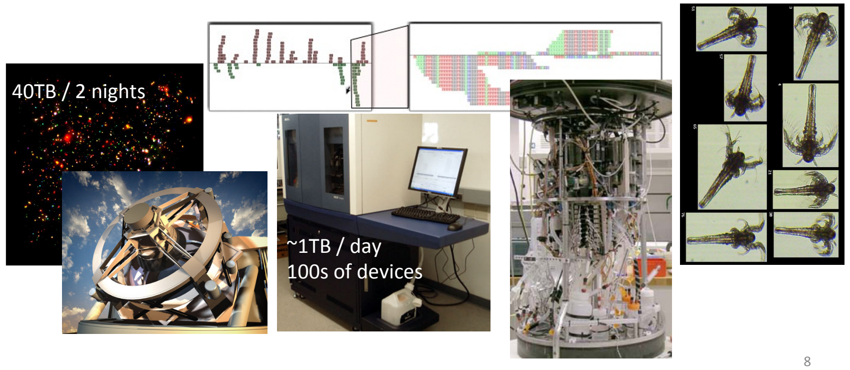
"It's a great time to be a data geek."
-- Roger Barga, Microsoft Research

"The greatest minds of my generation are trying to figure out how to make people click on ads"
-- Jeff Hammerbacher, co-founder, Cloudera

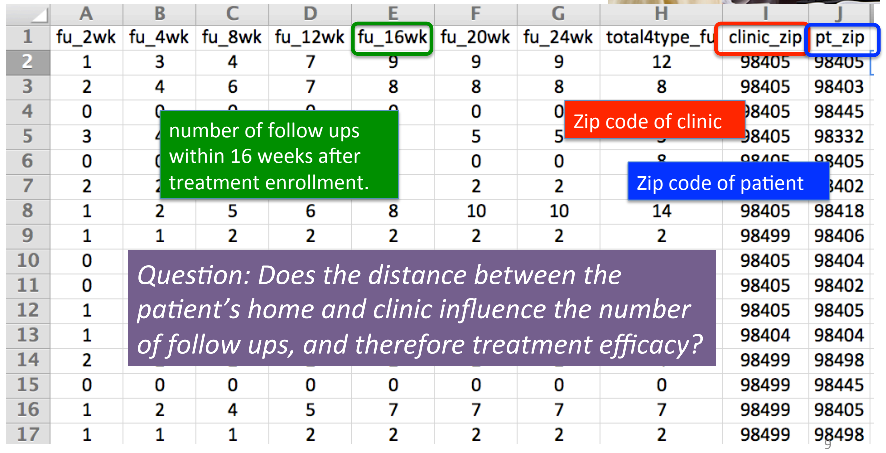
All of Science is Reducing to Computational Data Manipulation

Old model: "Query the world" (Data acquisition coupled to a specific hypothesis)
New model: "Download the world" (Data acquisition supports many hypotheses)

- **Astronomy: High-resolution, high-frequency sky surveys (SDSS, LSST, PanSTARRS)**
- **Biology: lab automation, high-throughput sequencing,**
- **Oceanography: high-resolution models, cheap sensors, satellites**



Example: Assessing Treatment Efficacy



	A	B	C	D	E	F	G	H	I	J
1	fu_2wk	fu_4wk	fu_8wk	fu_12wk	fu_16wk	fu_20wk	fu_24wk	total4type_fu	clinic_zip	pt_zip
2	1	3	4	7	9	9	9	12	98405	98405
3	2	4	6	7	8	8	8	8	98405	98403
4	0	0	0	0	0	0	0	[illegible]	98405	98445
5	3	4	[illegible]	[illegible]	[illegible]	5	5	[illegible]	98405	98332
6	0	0	[illegible]	[illegible]	[illegible]	0	0	8	98405	98405
7	2	2	[illegible]	[illegible]	[illegible]	2	2	[illegible]	[illegible]	98402
8	1	2	5	6	8	10	10	14	98405	98418
9	1	1	2	2	2	2	2	2	98499	98406
10	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98405	98404
11	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98405	98402
12	1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98405	98405
13	1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98404	98404
14	2	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	98499	98498
15	0	0	0	0	0	0	0	0	98499	98445
16	1	2	4	5	7	7	7	7	98499	98405
17	1	1	1	2	2	2	2	2	98499	98498

number of follow ups within 16 weeks after treatment enrollment.

Zip code of clinic

Zip code of patient

Question: Does the distance between the patient's home and clinic influence the number of follow ups, and therefore treatment efficacy?

Python Program to Assess Treatment Efficacy

```
# This program reads an Excel spreadsheet whose penultimate
# and antepenultimate columns are zip codes.
# It adds a new last column for the distance between those zip
# codes, and outputs in CSV (comma-separated values) format.
# Call the program with two numeric values:  the first and last
# row to include.
# The output contains the column headers and those rows.

# Libraries to use
import random
import sys
import xlrd        # library for working with Excel spreadsheets
import time
from gdapi import GoogleDirections

# No key needed if few queries
gd = GoogleDirections('dummy-Google-key')

wb = xlrd.open_workbook('mhip_zip_eScience_121611a.xls')
sheet = wb.sheet_by_index(0)

# User input:  first row to process, first row not to process
first_row = max(int(sys.argv[1]), 2)
row_limit = min(int(sys.argv[2]+1), sheet.nrows)

def comma_separated(lst):
  return ",".join([str(s) for s in lst])

headers = sheet.row_values(0) + ["distance"]
print comma_separated(headers)

for rownum in range(first_row,row_limit):
  row = sheet.row_values(rownum)
  (zip1, zip2) = row[-3:-1]
  if zip1 and zip2:
    # Clean the data
    zip1 = str(int(zip1))
    zip2 = str(int(zip2))
    row[-3:-1] = [zip1, zip2]
    # Compute the distance via Google Maps
    try:
      distance = gd.query(zip1,zip2).distance
    except:
      print >> sys.stderr, "Error computing distance:", zip1, zip2
      distance = ""
   # Print the row with the distance
   print comma_separated(row + [distance])
   # Avoid too many Google queries in rapid succession
   time.sleep(random.random()+0.5)
```

23 lines of executable code!

Course Logistics

- Website: http://web.cs.hacettepe.edu.tr/~bbm101/
- See the website for all administrative details
- Read the handouts and required texts, *before* the lecture
- Take notes!
- Follow the course in Piazza https://piazza.com/hacettepe.edu.tr/fall2015/bbm101

Academic Integrity

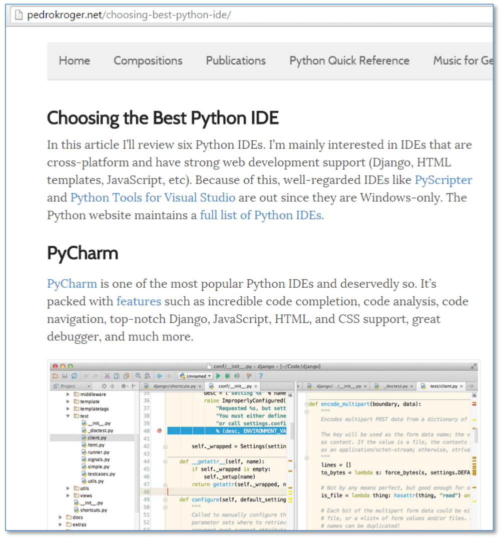
- Honest work is required of a scientist or engineer.
- Collaboration policy on the course web. **Read it!**
  - Discussion is permitted.
  - **Carrying materials from discussion is not permitted.**
  - Everything you turn in must be your own work.
    - Cite your sources, explain any unconventional action.
  - **You may not view others' work.**
  - If you have a question, ask.
- We trust you completely.
- But we have no sympathy for trust violations – nor should you!

How to Succeed

- No prerequisites
- Non-predictors for success:
  - Past programming experience
  - Enthusiasm for games or computers
- Programming and data analysis are challenging
- Every one of you can succeed
  - There is no such thing as a “born programmer”
  - Work hard
  - Follow directions
  - Be methodical
  - *Think* before you act
  - Try on your own, then ask for help
  - Start early

Integrated Development Environment (IDE)

- There are many!


pedrokroger.net/choosing-best-python-ide/

Home | Compositions | Publications | Python Quick Reference | Music for Ge

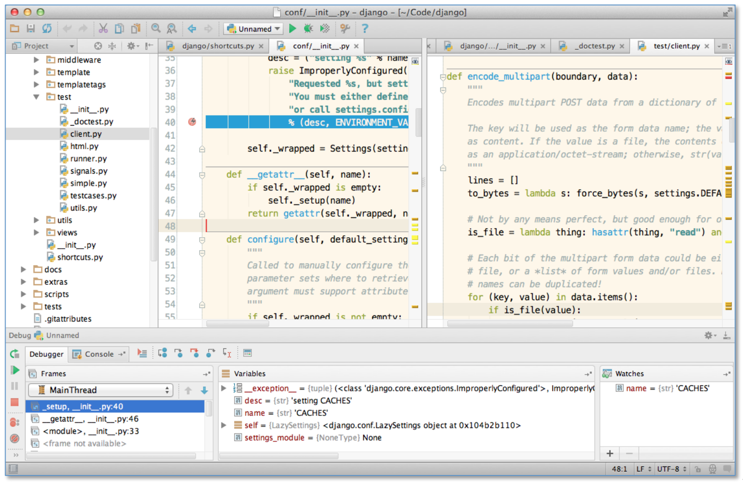
Choosing the Best Python IDE

In this article I'll review six Python IDEs. I'm mainly interested in IDEs that are cross-platform and have strong web development support (Django, HTML templates, JavaScript, etc). Because of this, well-regarded IDEs like PyScripter and Python Tools for Visual Studio are out since they are Windows-only. The Python website maintains a full list of Python IDEs.

PyCharm

PyCharm is one of the most popular Python IDEs and deservedly so. It's packed with features such as incredible code completion, code analysis, code navigation, top-notch Django, JavaScript, HTML, and CSS support, great debugger, and much more.

Our Recommendation: PyCharm


conf/__init__.py – django – [~/Code/django]

```
        desc = ("setting %s" % name
        raise ImproperlyConfigured(
            "Requested %s, but sett
            "You must either define
            "or call settings.confi
            % (desc, ENVIRONMENT_VA

    self._wrapped = Settings(settin

def __getattr__(self, name):
    if self._wrapped is empty:
        self._setup(name)
    return getattr(self._wrapped, n

def configure(self, default_setting
    """
    Called to manually configure th
    parameter sets where to retriev
    argument must support attribute
    """
    if self._wrapped is not empty:
```

```
def encode_multipart(boundary, data):
    """
    Encodes multipart POST data from a dictionary of

    The key will be used as the form data name; the v
    as content. If the value is a file, the contents
    as an application/octet-stream; otherwise, str(va
    """
    lines = []
    to_bytes = lambda s: force_bytes(s, settings.DEFA

    # Not by any means perfect, but good enough for o
    is_file = lambda thing: hasattr(thing, "read") an

    # Each bit of the multipart form data could be ei
    # file, or a *list* of form values and/or files.
    # names can be duplicated!
    for (key, value) in data.items():
        if is_file(value):
```

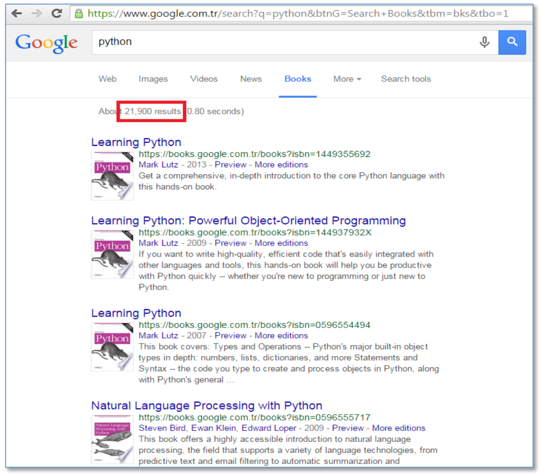
```
Frames
MainThread
_setup, __init__.py:40
__getattr__, __init__.py:46
<module>, __init__.py:33
<frame not available>

Variables
__exception__ = {tuple} (<class 'django.core.exceptions.ImproperlyConfigured'>, Improperly
desc = {str} 'setting CACHES'
name = {str} 'CACHES'
self = {LazySettings} <django.conf.LazySettings object at 0x104b2b110>
settings_module = {NoneType} None

Watches
name = {str} 'CACHES'
```

Python Version

- Whatever IDE you choose to work with, always stick to **Python version 2.7.10**
- **Always** use this version to code your assignments.

Books

- There are many!



https://www.google.com.tr/search?q=python&btnG=Search+Books&tbm=bks&tbo=1

About 21,900 results (0.80 seconds)

Learning Python
https://books.google.com.tr/books?isbn=1449355692
Mark Lutz - 2013 - Preview - More editions
Get a comprehensive, in-depth introduction to the core Python language with this hands-on book.

Learning Python: Powerful Object-Oriented Programming
https://books.google.com.tr/books?isbn=144937932X
Mark Lutz - 2009 - Preview - More editions
If you want to write high-quality, efficient code that's easily integrated with other languages and tools, this hands-on book will help you be productive with Python quickly -- whether you're new to programming or just new to Python.

Learning Python
https://books.google.com.tr/books?isbn=0596554494
Mark Lutz - 2007 - Preview - More editions
This book covers: Types and Operations -- Python's major built-in object types in depth: numbers, lists, dictionaries, and more Statements and Syntax -- the code you type to create and process objects in Python, along with Python's general ...

Natural Language Processing with Python
https://books.google.com.tr/books?isbn=0596555717
Steven Bird, Ewan Klein, Edward Loper - 2009 - Preview - More editions
This book offers a highly accessible introduction to natural language processing, the field that supports a variety of language technologies, from predictive text and email filtering to automatic summarization and

Our Recommendation for Books

- The Python Tutorial, available from the Python website.
  - This is good for explaining the nuts and bolts of how Python works.
- Think Python, 2nd edition
  - Freely available online in HTML and PDF.
  - Also available for purchase as a printed book, but don't buy the first edition.
  - This book introduces more conceptual material, motivating computational thinking.
- There is an interactive version of "How to Think Like a Computer Scientist" (the first edition of "Think Python"), which lets you type and run Python code directly while reading the book.

Introduction to Python and Programming

BBM 101 - Introduction to Programming I

Hacettepe University
Fall 2015

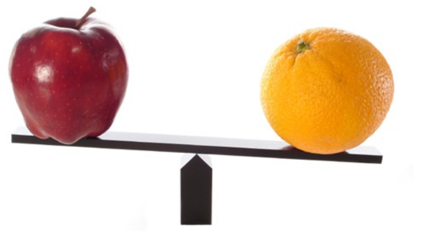
Fuat Akal, Aykut Erdem, Erkut Erdem, Vahid Garousi

Slides based on material prepared by Ruth Anderson, Michael Ernst and Bill Howe in the course CSE 140
University of Washington

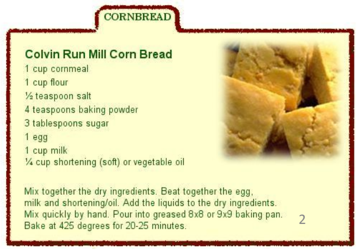
1. Python is a calculator

2. A variable is a container

3. Different types cannot be compared



4. A program is a recipe



CORNBREAD

Colvin Run Mill Corn Bread
1 cup cornmeal
1 cup flour
½ teaspoon salt
4 teaspoons baking powder
3 tablespoons sugar
1 egg
1 cup milk
¼ cup shortening (soft) or vegetable oil

Mix together the dry ingredients. Beat together the egg, milk and shortening/oil. Add the liquids to the dry ingredients. Mix quickly by hand. Pour into greased 8x8 or 9x9 baking pan. Bake at 425 degrees for 20-25 minutes.

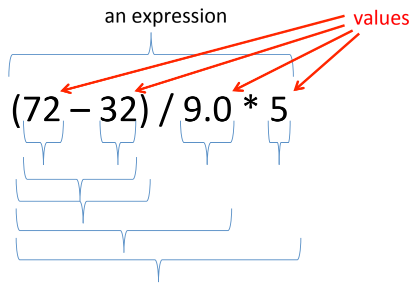
1. Python is Like a Calculator

You Type *Expressions*. Python Computes Their *Values*.

- 5
- 3+4
- 44/2
- 2**3
- 3*4+5*6
  – If precedence is unclear, use parentheses
- (72 – 32) / 9 * 5

An Expression is Evaluated From the Inside Out

- How many expressions are in this Python code?



(72 – 32) / 9.0 * 5
(40) / 9.0 * 5
40 / 9.0 * 5
4.44 * 5
22.2

Another Evaluation Example

(72 – 32) / (9.0 * 5)
(40) / (9.0 * 5)
40 / (9.0 * 5)
40 / (45.0)
40 / 45.0
.888

2. A Variable is a Container

Variables Hold Values

- Recall variables from algebra:
  - Let x = 2 …
  - Let y = x …

- To assign a variable, use "*varname* = *expression*"

```
pi = 3.14
pi
var = 6*10**23
22 = x                # Error!
```

No output from an assignment statement

- Not all variable names are permitted

Changing Existing Variables (“re-binding” or “re-assigning”)

```
x = 2
x
y = x
y
x = 5
x
Y
```

- “=” in an assignment is not a promise of eternal equality
  - This is different than the mathematical meaning of “=”
- Evaluating an expression gives a new (copy of a) number, rather than changing an existing one

How an Assignment is Executed

1. Evaluate the right-hand side to a value
2. Store that value in the variable

```
x = 2
print(x)
y = x
print(y)
z = x + 1
print(z)
x = 5
print(x)
print(y)
print(z)
```

State of the computer:

x: ~~2~~ 5

y: 2

z: 3

Printed output:

```
2
2
3
5
2
3
```

To visualize a program’s execution: http://pythontutor.com

More Expressions: Conditionals (value is `True` or `False`)

```
22 > 4           # condition, or conditional
22 < 4           # condition, or conditional
22 == 4          …
x == 100         # Assignment, not conditional!
22 = 4           # Error!
x >= 5
x >= 100
x >= 200
not True
not (x >= 200)
3<4 and 5<6
4<3 or 5<6
temp = 72
water_is_liquid = (temp > 32 and temp < 212)
```

Numeric operators: `+`, `*`, `**`
Boolean operators: `not`, `and`, `or`
Mixed operators: `<`, `>=`, `==`

More Expressions: strings

A string represents text

```
'Python'
myString = "BBM 101-Introduction to Programming"
""
```

Empty string is not the same as an unbound variable

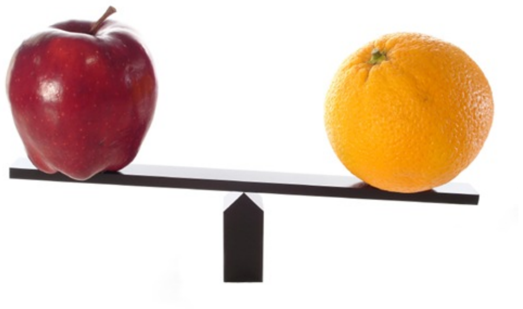
- **"“ and ‘’ are the same**

Operations:

- **Length:**
  `len(myString)`
- **Concatenation:**
  `“Hacettepe" + ‘ ‘ + ' University'`
- **Containment/searching:**
  `‘a' in myString`
  `“a" in myString`

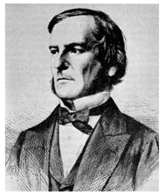
3. Different Types cannot be Compared

```
anInt = 2
aString = "Hacettepe"
anInt == aString                # Error
```



Types of Values

- Integers (**`int`**): **`-22, 0, 44`**
  - Arithmetic is exact
  - Some funny representations: **`12345678901L`**
- Real numbers (**`float`**, for "floating point"): **`2.718, 3.1415`**
  - Arithmetic is approximate, e.g., **`6.022*10**23`**
  - Some funny representations: **`6.022e+23`**
- Strings (**`str`**): **`"I love Python", ""`**
- Truth values (**`bool`**, for "Boolean"): **`True, False`**


George Boole

Operations Behave differently on Different Types

```
3.0 + 4.0
3 + 4
3 + 4.0
"3" + "4"              # Concatenation
3 + "4"                # Error
3 + True               # Error
```

Moral: Python only *sometimes* tells you when you do something that does not make sense.

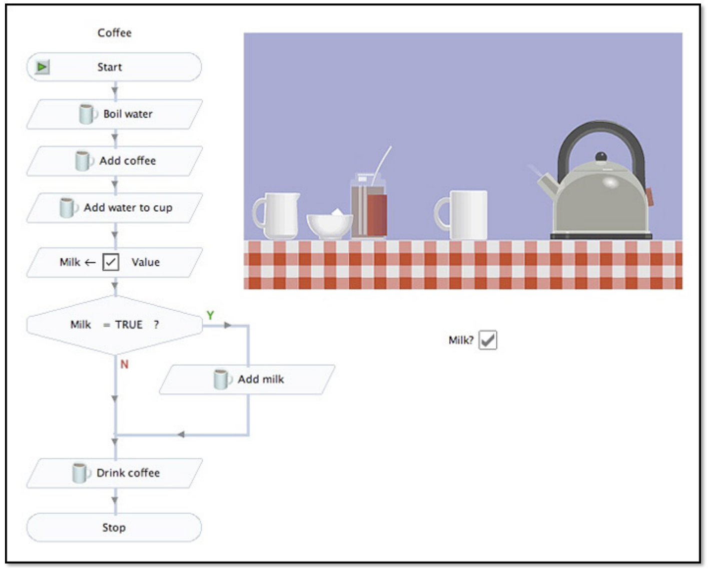
Operations on Different Types

```
15.0 / 4.0
15 / 4
15.0 / 4
15 / 4.0
```

Type conversion:

```
float(15)
int(15.0)
int(15.5)
int("15")
str(15.5)
float(15) / 4
```

A Program is a Recipe



Design the Algorithm Before Coding

- We should think (design the algorithm) before coding
- Algorithmic thinking is the logic. Also, called problem solving
- Coding is the syntax
- Make this a habit
- Some students do not follow this practice and they get challenged in all their courses and careers!

What is a Program?

- A program is a sequence of instructions
- The computer executes one after the other, as if they had been typed to the interpreter
- Saving your work as a program is better than re-typing from scratch

```
x = 1
y = 2
x + y
print(x + y)
print("The sum of", x, "and", y, "is", x+y)
```

The `print()` Statement

- The **`print`** statement always prints one line
  - The next print statement prints below that one
- Write 0 or more expressions after **`print`**, separated by commas
  - In the output, the values are separated by spaces
- Examples:

```
x=1
y=2
print(3.1415)
print(2.718, 1.618)
Print()
print(20 + 2, 7 * 3, 4 * 5)
print("The sum of", x, "and", y, "is", x+y)
```

Exercise: Convert Temperatures

- Make a temperature conversion chart as the following
- Fahrenheit to Centigrade, for Fahrenheit values of: -40, 0, 32, 68, 98.6, 212
- C = (F - 32) × 5/9
- Output:

```
Fahrenheit Centigrade
-40 -40.0
0 -17.7778
32 0.0
68 20.0
98.6 37.0
212 100.0
```

- You have created a Python program!
- (It doesn't have to be this tedious, and it won't be.)

Expressions, Statements, and Programs

- An expression evaluates to a value

```
3 + 4
pi * r**2
```

- A statement causes an effect

```
pi = 3.14159
print(pi)
```

- Expressions appear within other expressions and within statements

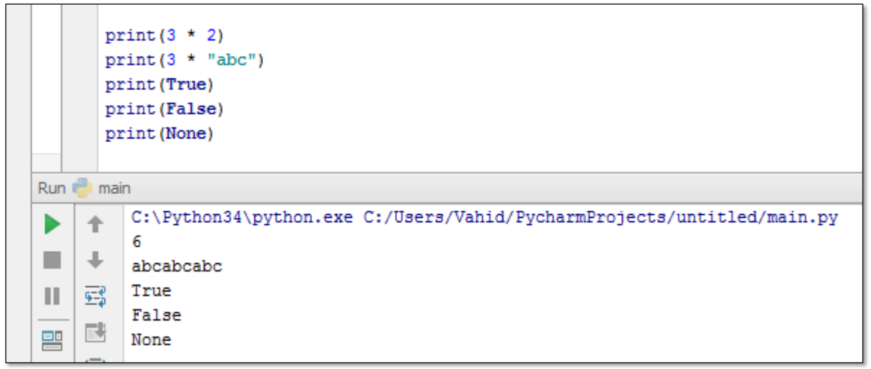
```
(fahr – 32) * (5.0 / 9)
print(pi * r**2)
```

- A statement may *not* appear within an expression

```
3 + print(pi)          # Error!
```

- A program is made up of statements
  - A program should do something or communicate information

print() Function



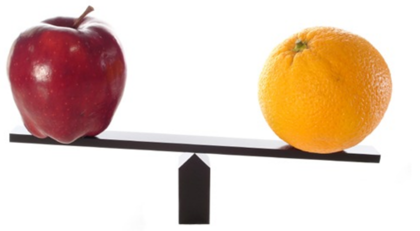
```
print(3 * 2)
print(3 * "abc")
print(True)
print(False)
print(None)
```

```
C:\Python34\python.exe C:/Users/Vahid/PycharmProjects/untitled/main.py
6
abcabcabc
True
False
None
```

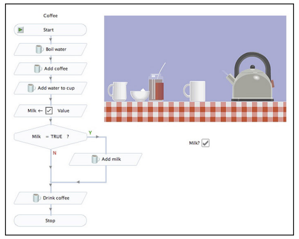
1. Python is a calculator

2. A variable is a container

3. Different types cannot be compared



4. A program is a recipe



Programming Languages

- A programming language is a “language” to write programs in, such as Python, C, C++, Java
- The concept of programming languages are quite similar
- Python:

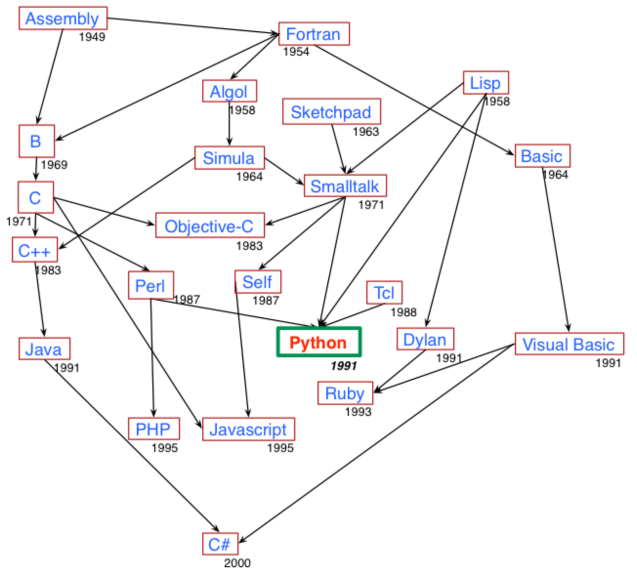
```
print("Hello, World!")
```

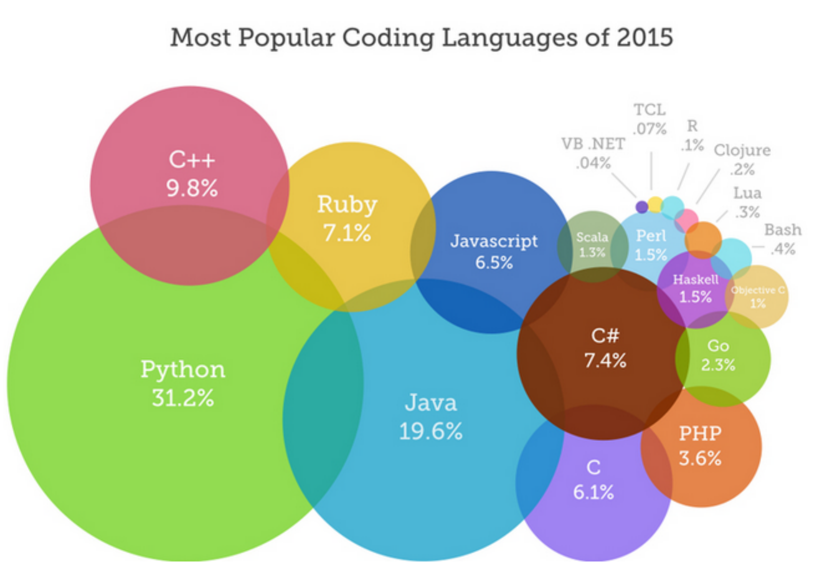
- Java:

```
public static void main(String[] args) {
    System.out.println("Hello, World");
}
```

- Python is simpler! That’s why we are learning it first ☺

Evolution of Programming Languages





- http://blog.codeeval.com/codeevalblog/2015