Artificial Intelligence Tools & Techniques

Dr. Tassadaq Hussain

Assistant Professor Riphah International University

Collaborations:

Microsoft Research and Barcelona Supercomputing Center Barcelona, Spain

European Network on High Performance and Embedded Architecture and Compilation

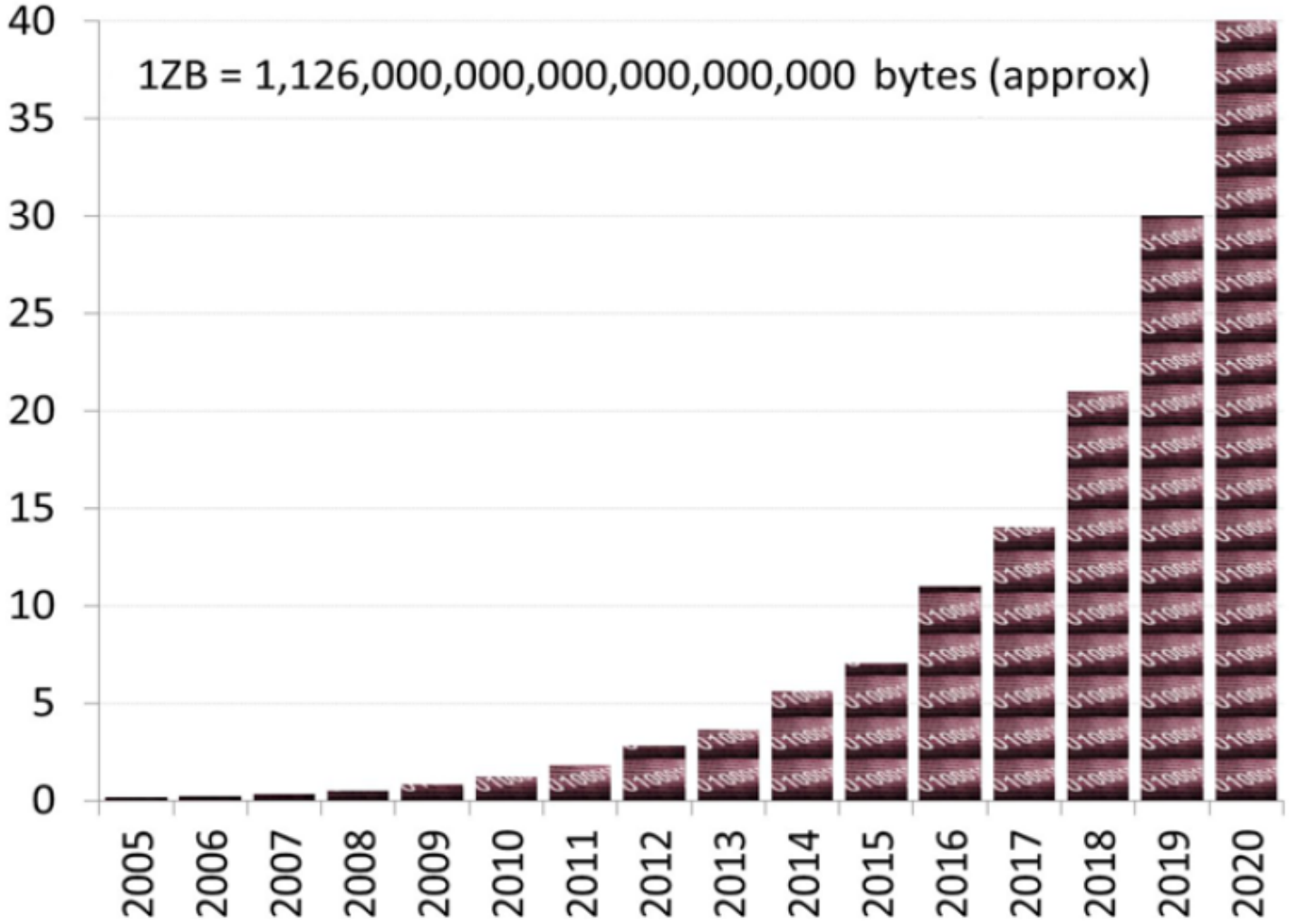
UCERD Pvt Ltd Islamabad

UCERD
Gathering
Intellectuals
www.ucerd.com

Agenda

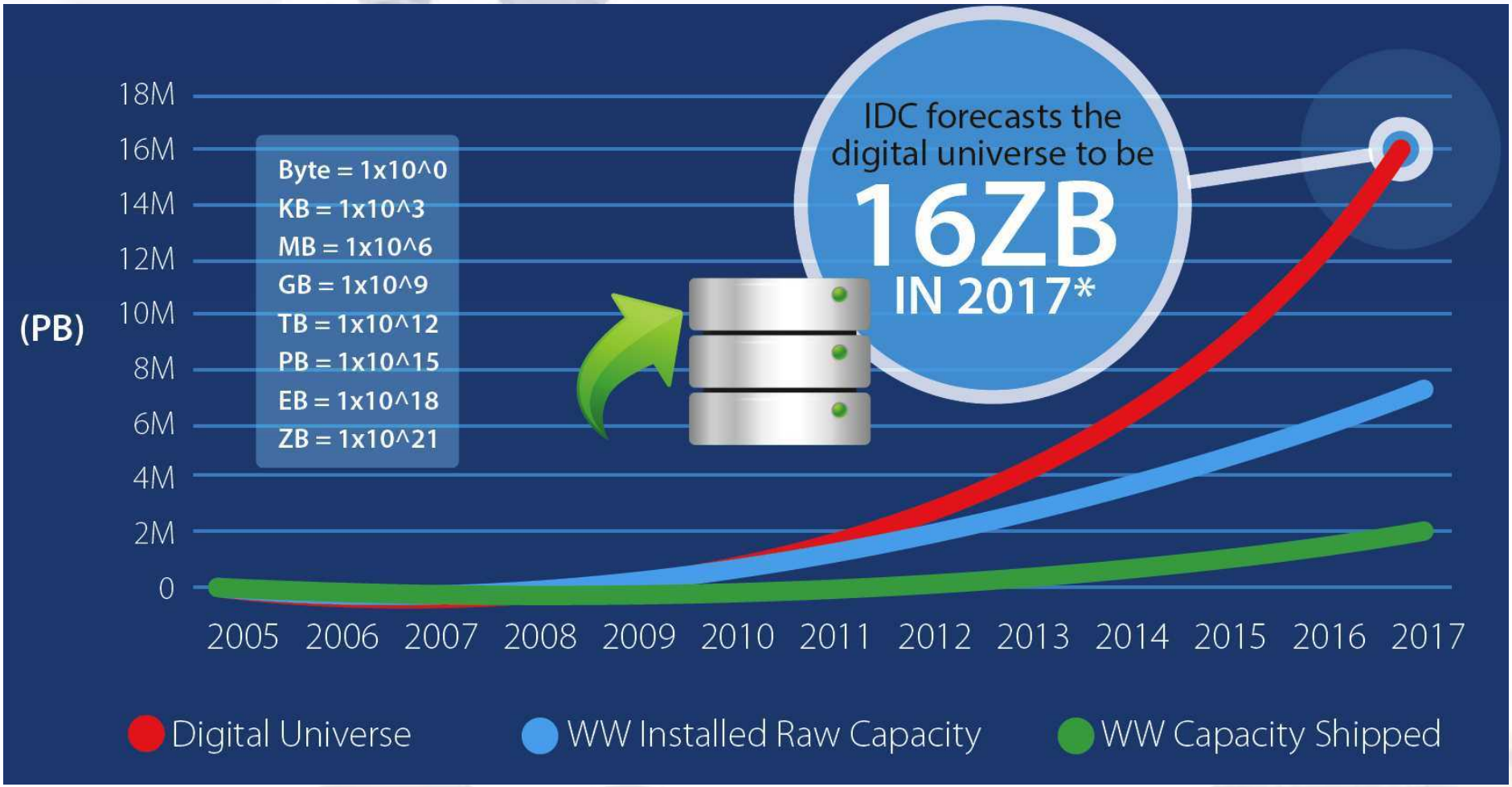
- **Importance**
- Coding Languages
- Conventional Programming
- Python Language
- Python for ML

All Global Data in Zettabytes



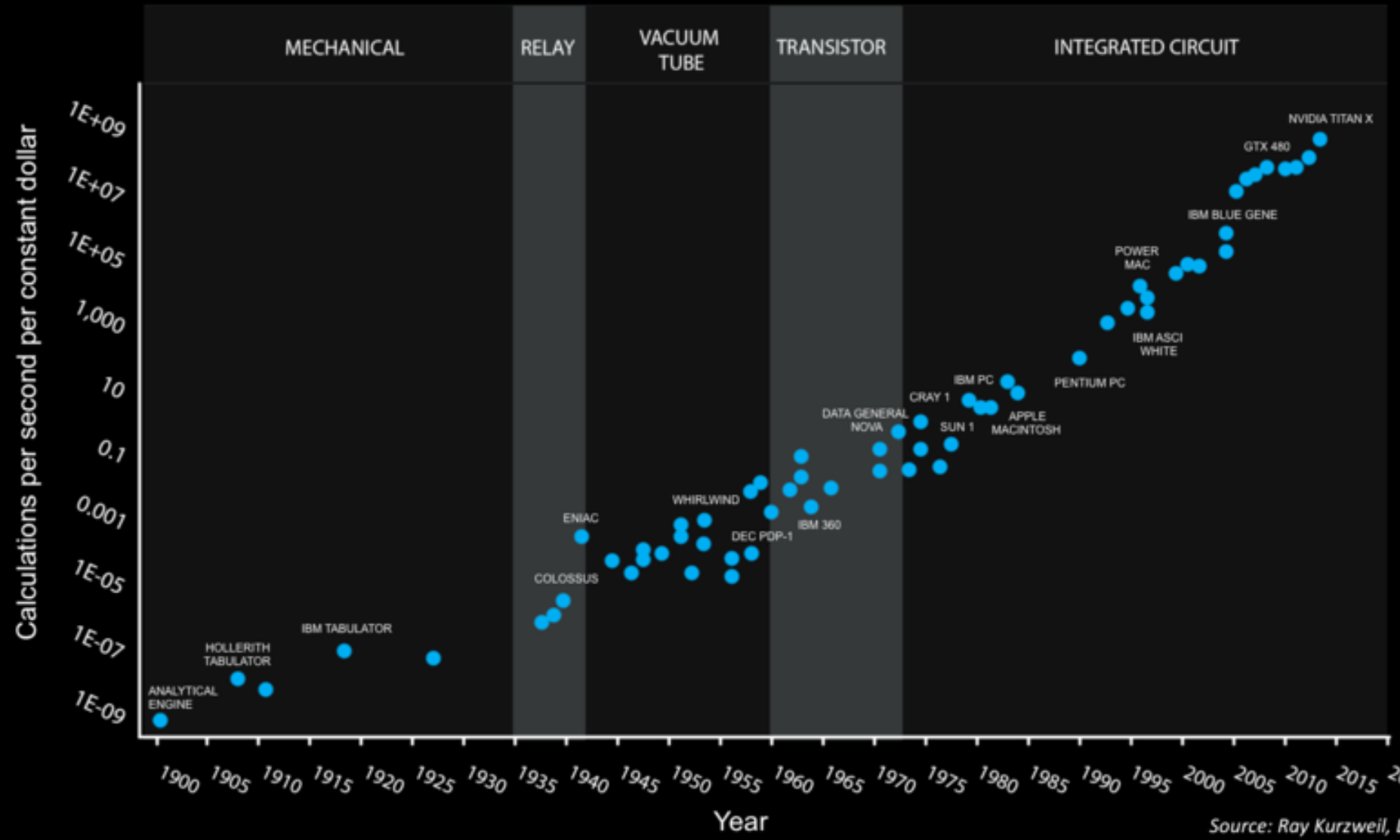
Source: http://www1.unece.org/stat/platform/display/msis/Big+Data

Information Future Trend



Technology Research Gartner Inc. states that information data volume doubles after every 18 months.

120 Years of Moore's Law

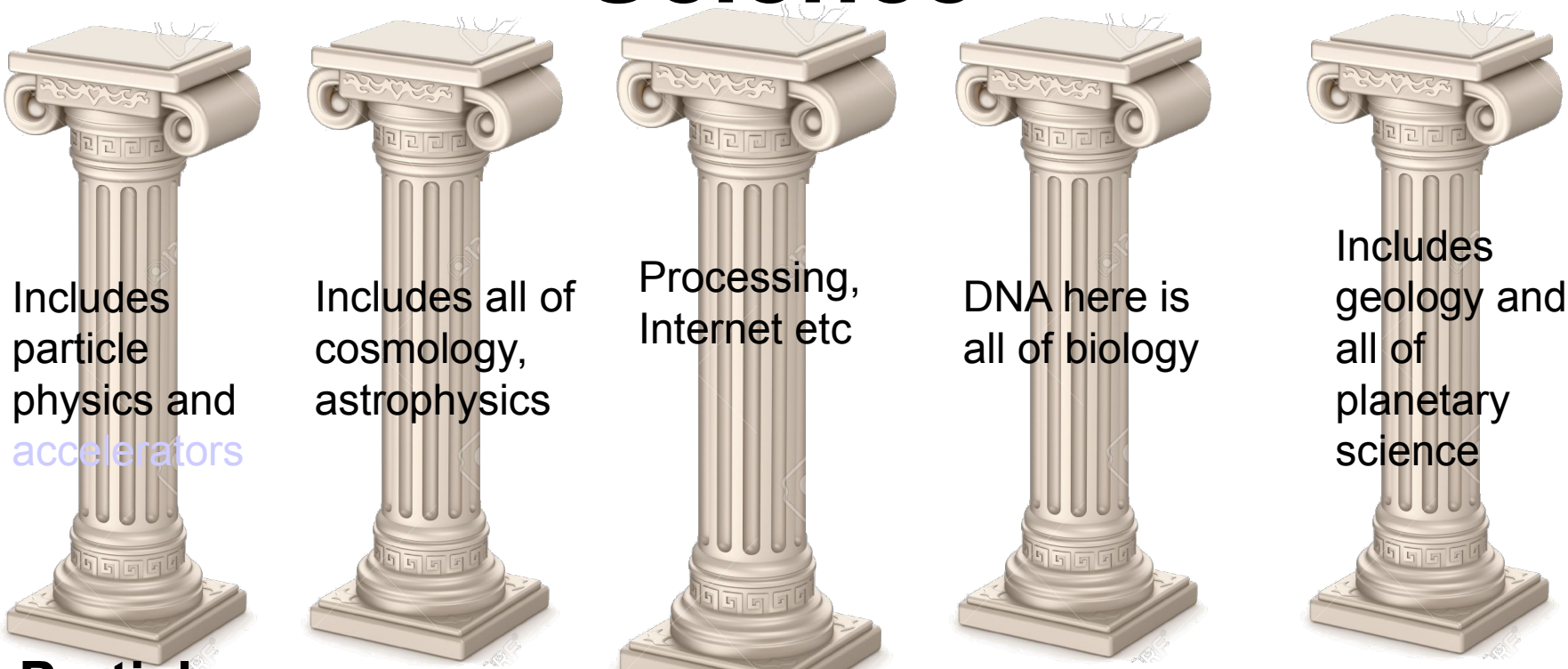


Source: Ray Kurzweil, DFJ

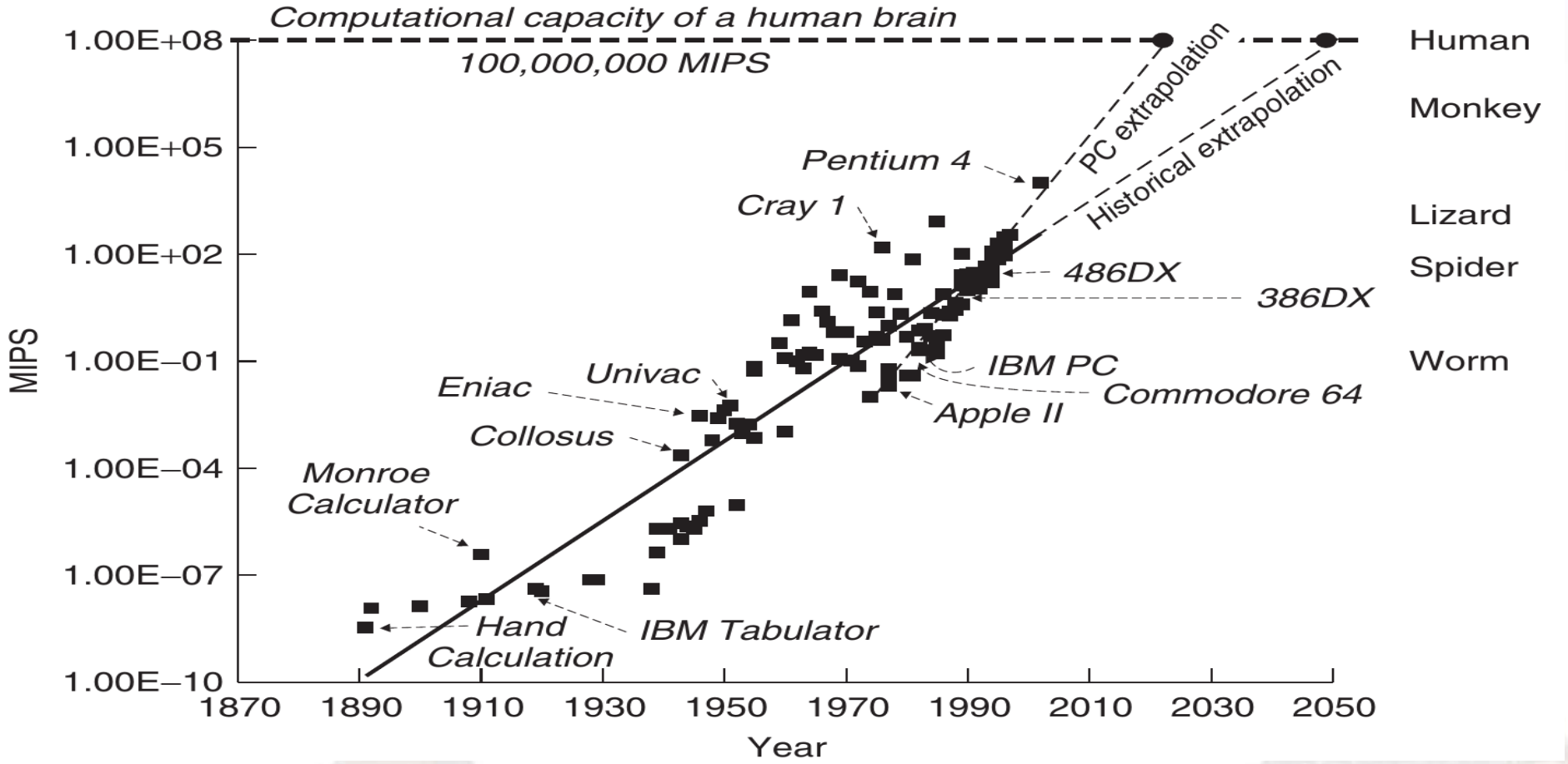
Pillars of Science

Fermi National Accelerator Laboratory

Science



Computational Capability



It is estimated that sometime between the years **2025 and 2050**, a **personal computers** will exceed the calculation power of a human brain.

Agenda

- Importance
- **Coding Languages**
- Conventional Programming
- Python Language
- Python for ML

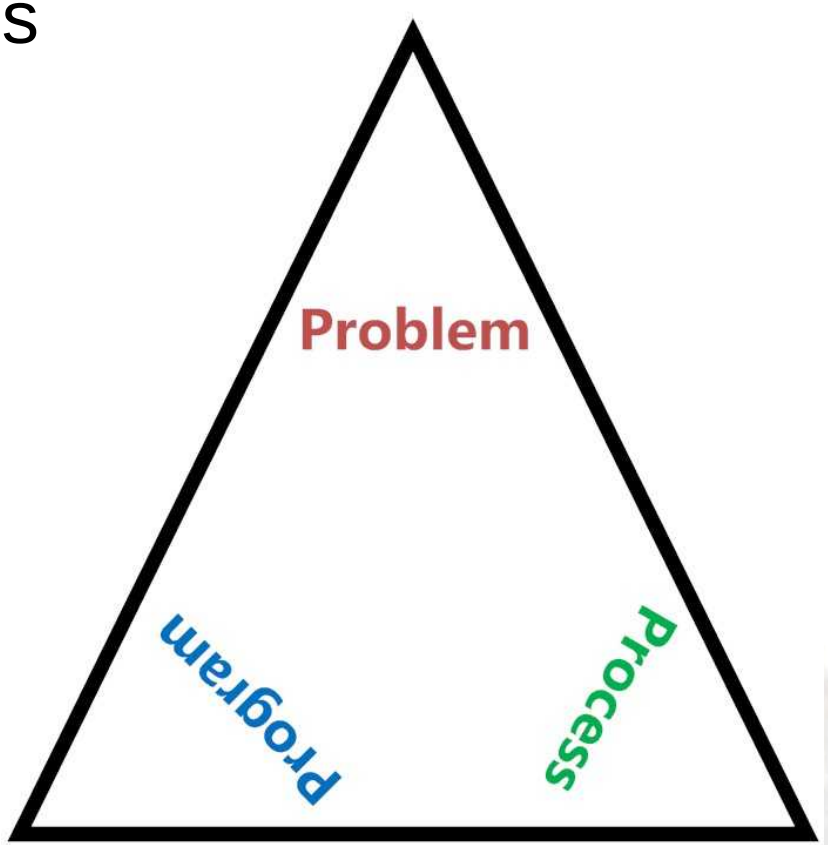
Coding Languages

- A coding language is a language that has its own set of symbols, words, tokens, and operators having their own significance and meaning. Thus syntax and semantics combine to form a formal language in itself.

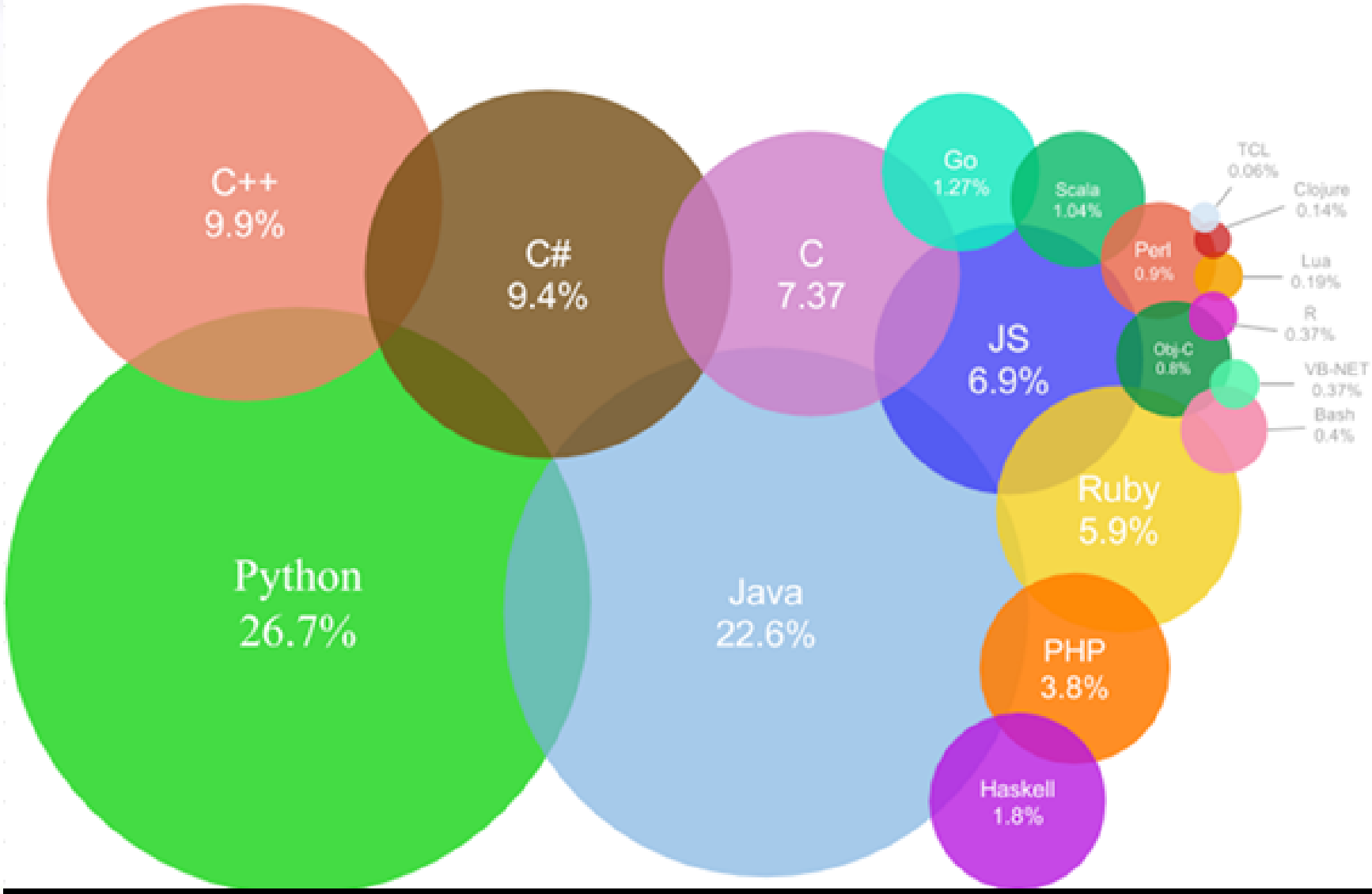
Data Processing

Coding Languages

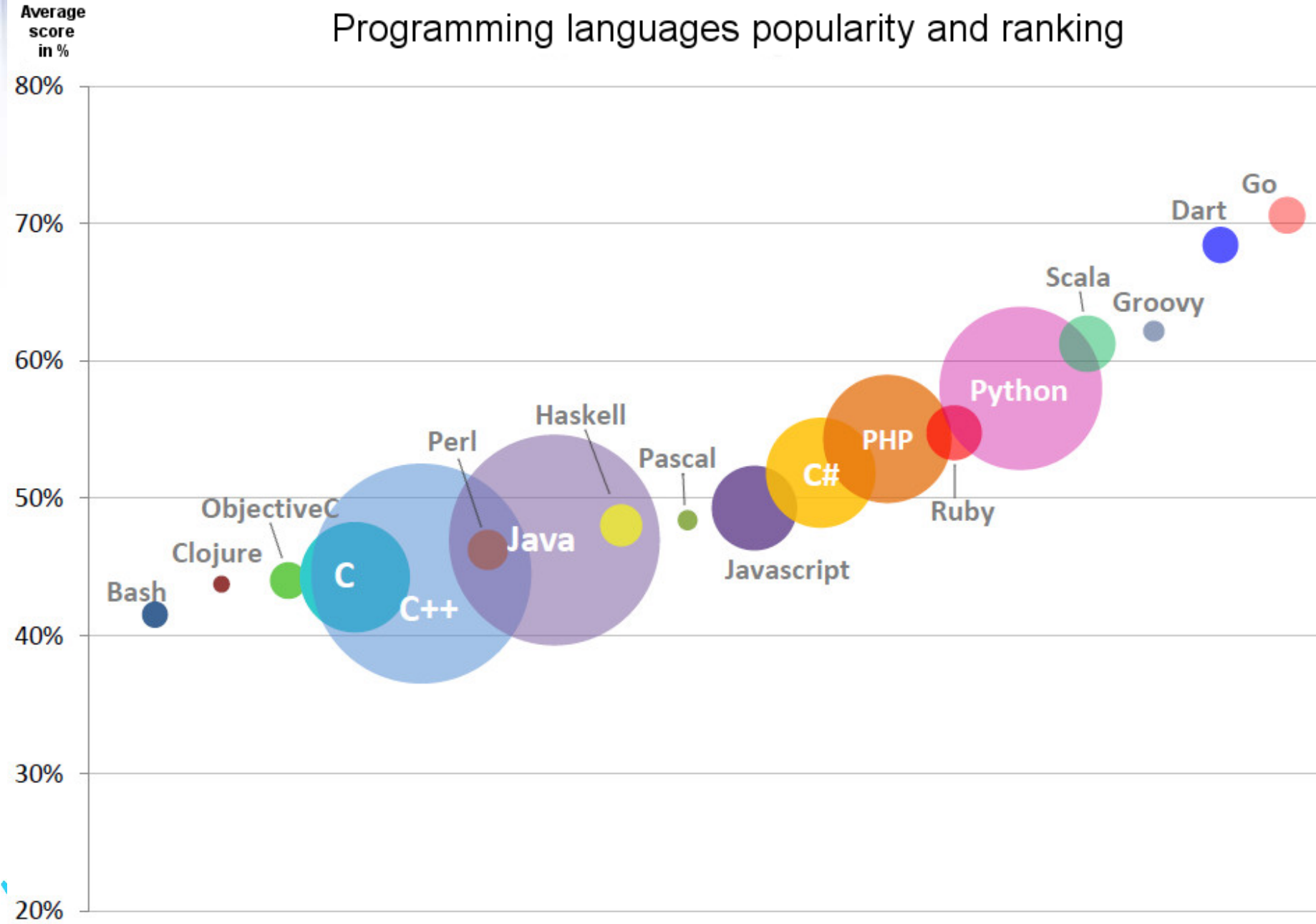
- Programming Languages
- Scripting Languages



Most Popular Coding Languages of 2016

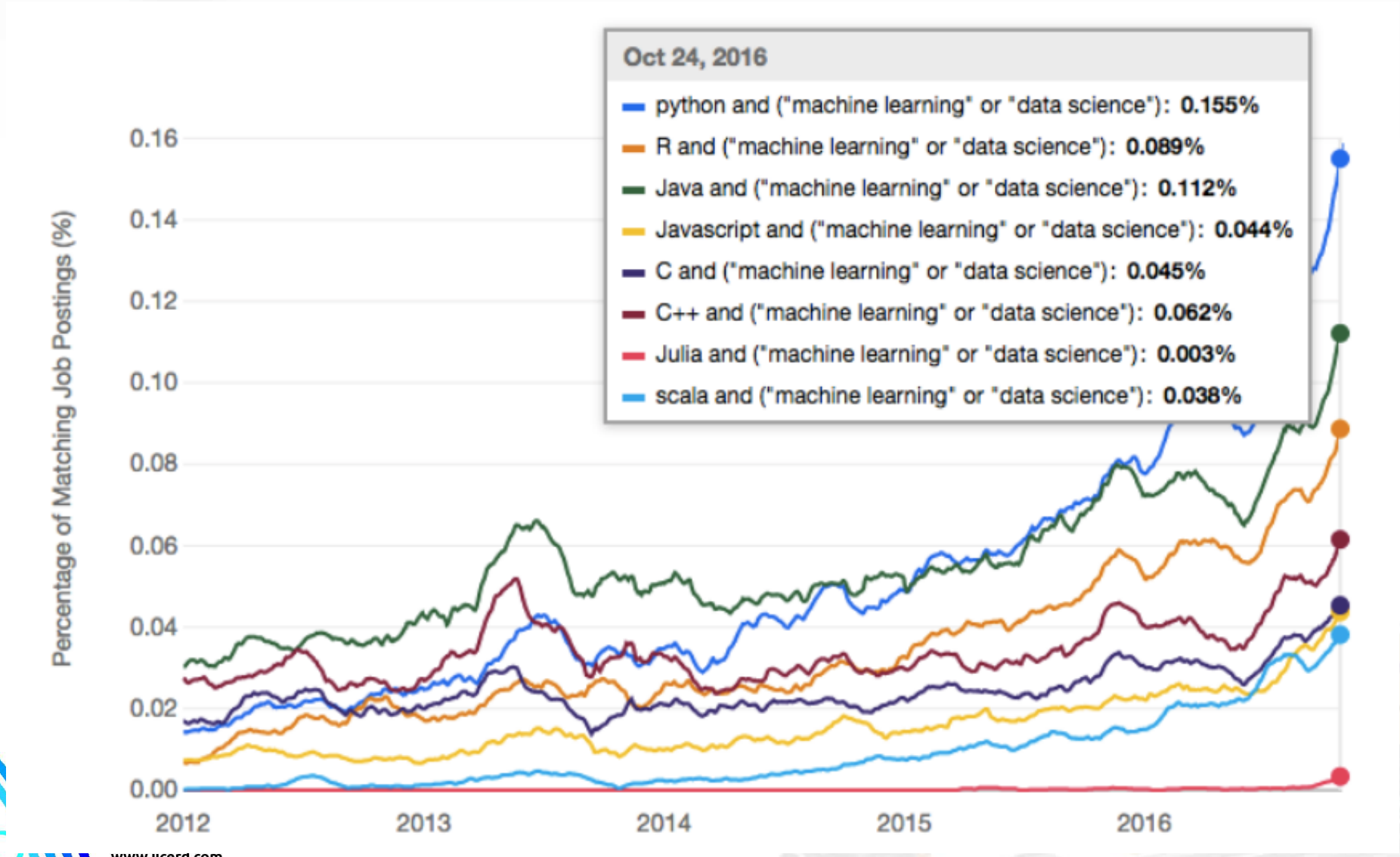


Programming languages popularity and ranking



Based on the most used programming languages in the coding contests of the first half of 2014

CodinGame


Oct 24, 2016
python and ("machine learning" or "data science"): 0.155%
R and ("machine learning" or "data science"): 0.089%
Java and ("machine learning" or "data science"): 0.112%
Javascript and ("machine learning" or "data science"): 0.044%
C and ("machine learning" or "data science"): 0.045%
C++ and ("machine learning" or "data science"): 0.062%
Julia and ("machine learning" or "data science"): 0.003%
scala and ("machine learning" or "data science"): 0.038%
Percentage of Matching Job Postings (%)
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
2012
2013
2014
2015
2016

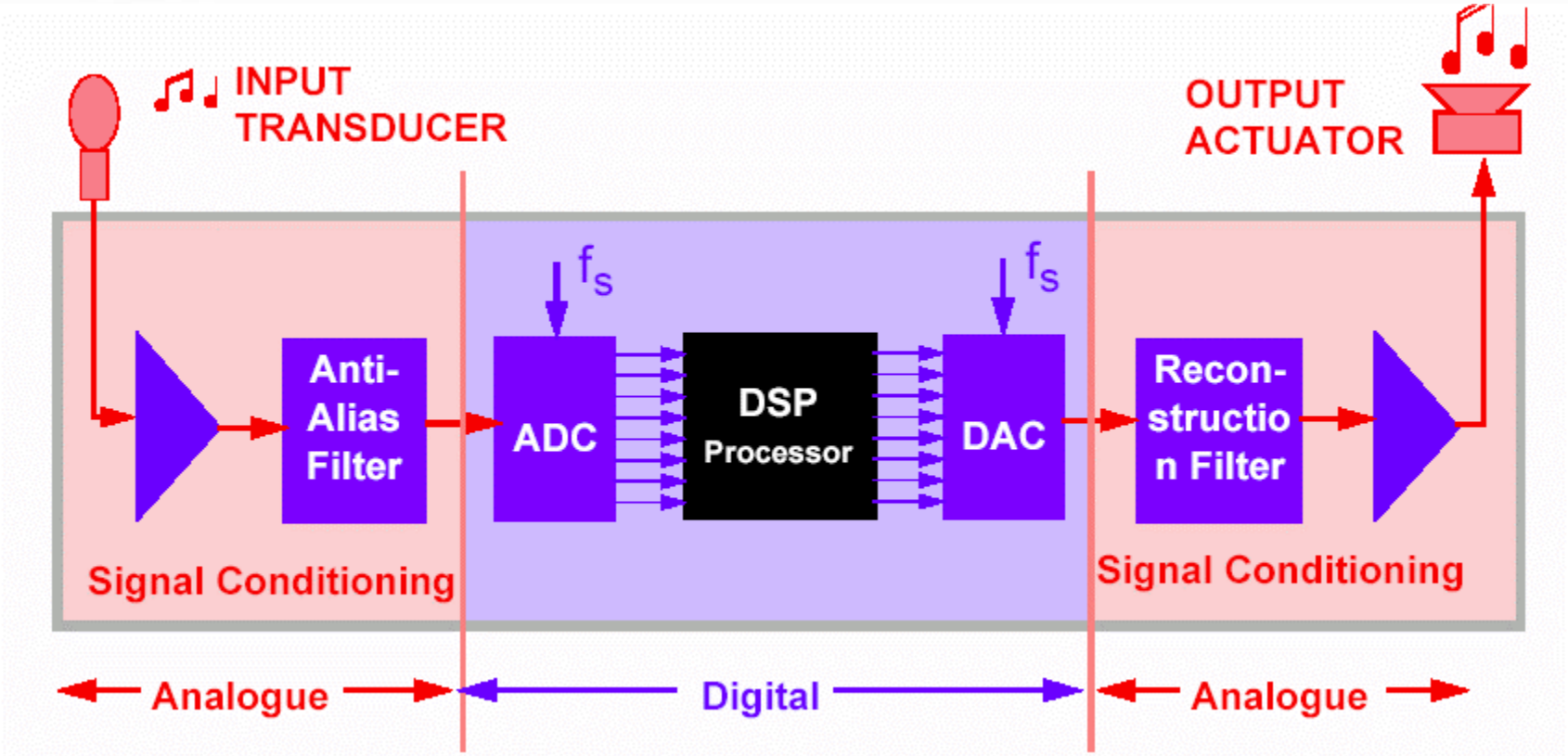
Agenda

- Importance
- Coding Languages
- **Conventional Programming**
- Python Language
- Python for ML

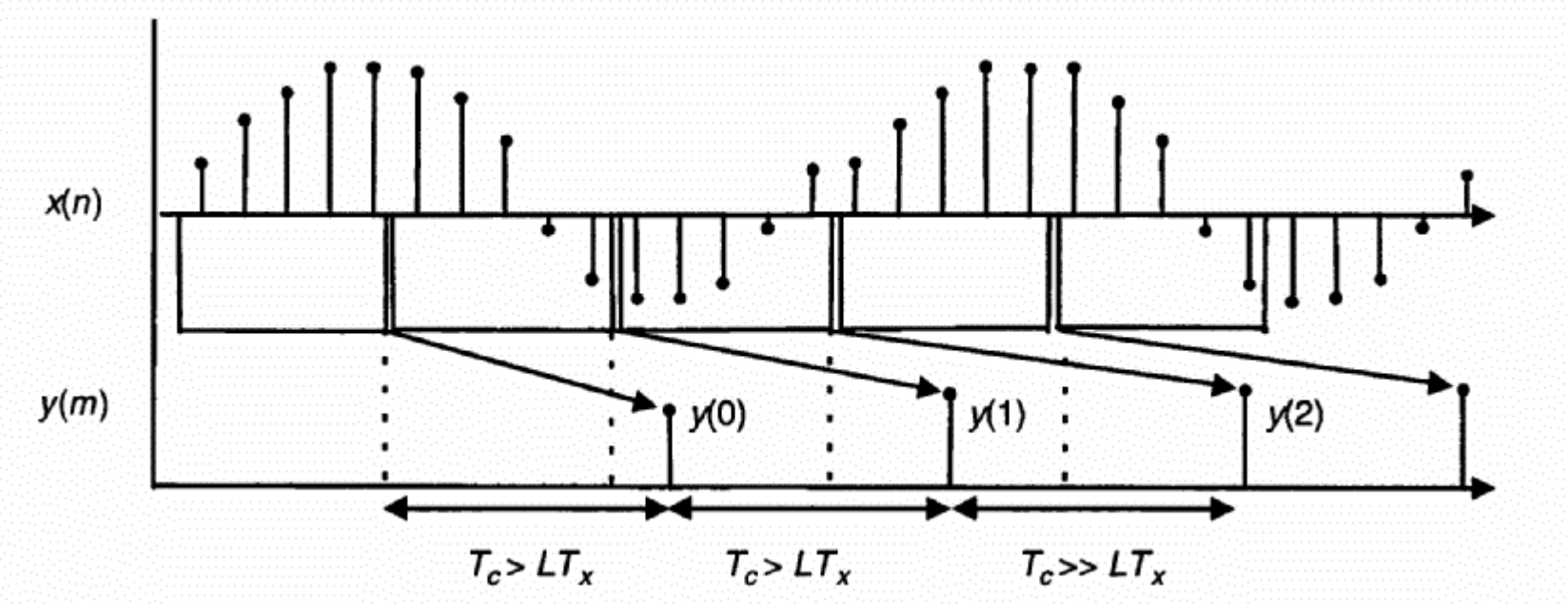
Signal Processing Applications

- Sound applications
  - Compression, enhancement, special effects, synthesis, recognition, echo cancellation,…
  - Cell Phones, MP3 Players, Movies, Dictation, Text-to-speech,…
- Communication
  - Modulation, coding, detection, equalization, echo cancellation,…
  - Cell Phones, dial-up modem, DSL modem, Satellite Receiver,…
- Automotive
  - ABS, GPS, Active Noise Cancellation, Cruise Control, Parking,…
- Medical
  - Magnetic Resonance, Tomography, Electrocardiogram,…
- Military
  - Radar, Sonar, Space photographs, remote sensing,…
- Image and Video Applications
  - DVD, JPEG, Movie special effects, video conferencing,…
- Mechanical
  - Motor control, process control, oil and mineral prospecting,…

Signal Processing System



Signals



Levels of processing

Stream processing

- All computations with one input sample are completed before the next input sample arrives

Block processing

- Each input sample x(n) is stored in memory before any processing occurs upon it. After *L* input samples have arrived, the entire collection of samples is processed at once.

Vector processing

- Systems with several input and/or output signals being computed at once: can work with streams or blocks

Executing a Program on HPC System

1. If you are starting with an existing serial program, debug the serial code completely
2. Identify the parts of the program that can be executed concurrently:
   - Requires a thorough understanding of the algorithm
   - Exploit any inherent parallelism which may exist.
   - May require restructuring of the program and/or algorithm. May require an entirely new algorithm.
3. Decompose the program:
   - Functional Parallelism
   - Data Parallelism
   - Combination of both
4. Code development
   - Code may be influenced/determined by machine architecture
   - Choose a programming paradigm
   - Determine communication
   - Add code to accomplish task control and communications
5. Compile, Test, Debug
6. Optimization
   - Measure Performance
   - Locate Problem Areas
   - Improve them

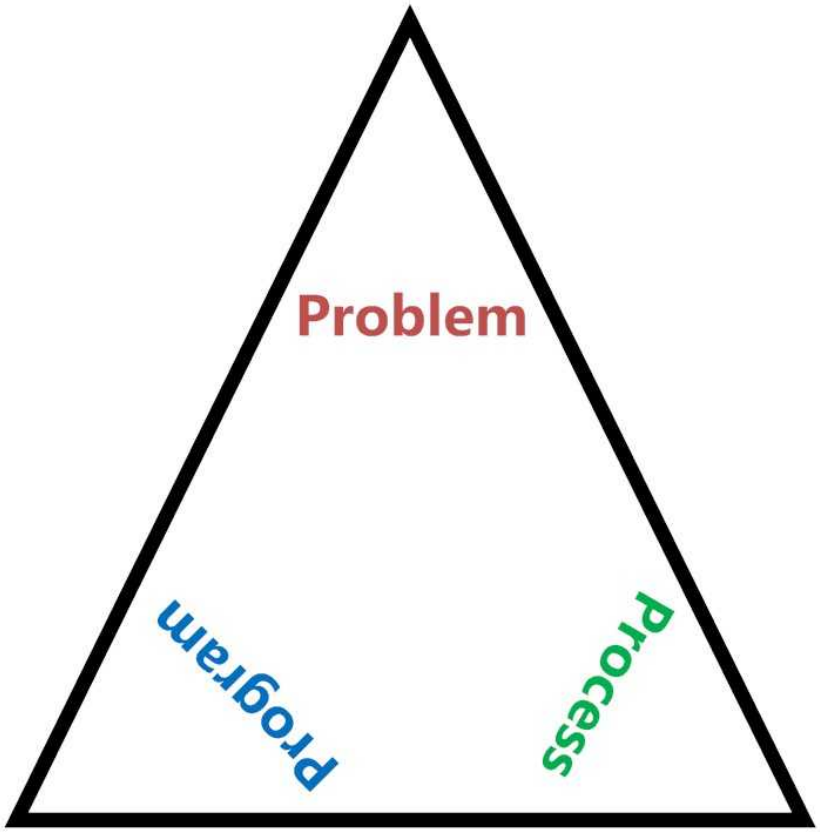
Types of Parallelism

Instruction Level Parallelism
Task Level Parallelism
Data Level Parallelism

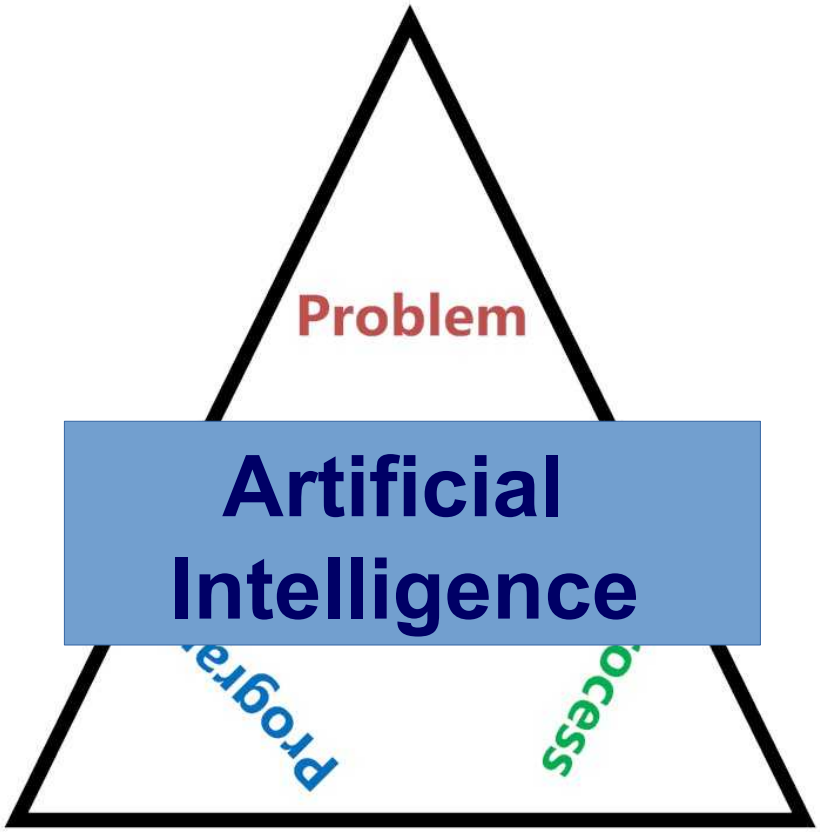
Parallel Programmings

- C++
  - Pthread Libraries
- Parallel Programming Models
  - OpenMP
  - MPI
  - OpenACC
  - OpenCL

Problem Program and Process



Problem Program and Process



Agenda

- Importance
- Coding Languages
- Conventional Programming
- **Python Language**
- Python for ML

Python Language

Everything in Python is an object.

The objects can be either mutable or immutable.

A mutable object can be changed after it is created, and an immutable object can't.

Python

Python code is fast to develop: As the code is no required to be compiled and built, Python code can be much readily changed and executed. This makes for a fast development cycle.

Python code is not as fast in execution: Since the code is not directly compiled and executed and an additional layer of the Python virtual machine is responsible for execution, Python code runs a little slow as compared to conventional languages like C, C++, etc.

Strengths

- Easy to learn:
- Supports multiple programming paradigms
- Extensible
- Active open source community
- Large and Active Community Support
- Powerful Set of Packages
- Easy and Rapid Prototyping
- Easy to Collaborate

Strength of Python

Easy to learn: Python is a relatively easy-to-learn language. Its syntax is simple for a beginner to learn and understand. When compared with languages like C or Java, there is minimal boilerplate code required in executing a Python program.

Supports multiple programming paradigms: Python is a multi-paradigm, multi-purpose programming language. It supports object oriented programming, structured programming, functional programming, and even aspect oriented programming. This versatility allows it to be used by a multitude of programmers.

Extensible: Extensibility of Python is one of its most important characteristics. Python has a huge number of modules easily available which can be readily installed and used. These modules cover every aspect of programming from data access to implementation of popular algorithms. This easy-to-extend feature ensures that a Python developer is more productive as a large array of problems can be solved by available libraries.

Active open source community: Python is open source and supported by a large developer community. This makes it robust and adaptive. The bugs encountered are easily fixed by the Python community. Being open source, developers can tinker with the Python source code if their requirements call for it.

Pitfall

- Although Python is a very popular programming language, it comes with its own share of pitfalls. One of the most important limitations it suffers is in terms of execution speed. Being an interpreted language, it is slow when compared to compiled languages. This limitation can be a bit restrictive in scenarios where extremely high performance code is required. This is a major area of improvement for future implementations of Python and every subsequent Python version addresses it. Although we have to admit it can never be as fast as a compiled language, we are convinced that it makes up for this deficiency by being super-efficient and effective in other departments.

Why Python

According to a 2017 survey by StackOverflow,

- Python is world's 5th most used language. It is one of the top three languages used by data scientists and one of the most "wanted" language among StackOverflow users.
- In a recent poll by KDnuggets in 2017, Python got the maximum number of votes for being the leading platform for Analytics, Data Science, and Machine Learning based on the choice of users
- Python has a lot of advantages that makes it a language of choice when it comes to the practices of Data Science. We will now try to illustrate these advantages and argue our case for "Why Python is a language of choice for Data scientists?"

Setting Up a Python Environment

- Set Up Anaconda Python Environment
- Installing Libraries
  - pip install required_package
-

- Scripting languages focus flexibility, rapid development and dynamic checking.
- Their type systems embrace very high level concepts such as tables, patterns, lists and files.
- There a number of distinct groups that fall under the scripting language family.
- Languages such as Perl and Python are known as ``glue'' languages because they were designed to glue existing programs together.
- There are other extensible types of scripting languages used in the WWW also.

Development Environments

1. PyDev with Eclipse
2. Komodo
3. Emacs
4. Vim
5. TextMate
6. Gedit
7. Idle
8. PIDA (Linux)(VIM Based)
9. NotePad++ (Windows)
10. BlueFish (Linux)

Python Keywords

FALSE	Class	Finally	Is	return
None	Continue	For	Lambda	try
TRUE	Def	From	nonlocal	while
And	Del	Global	Not	with
As	Elif	If	Or	yield
Assert	Else	Import	Pass	
Break	Except	In	Raise	

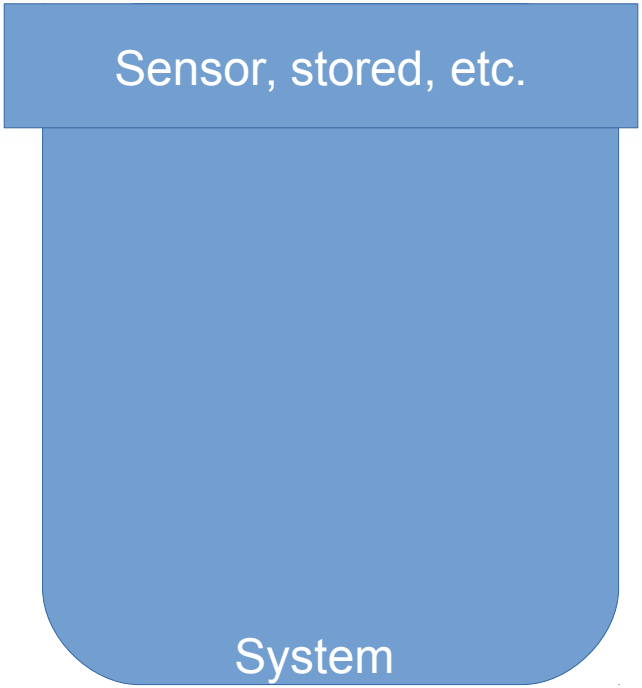
Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries

System
Method
Algorithm
.
.

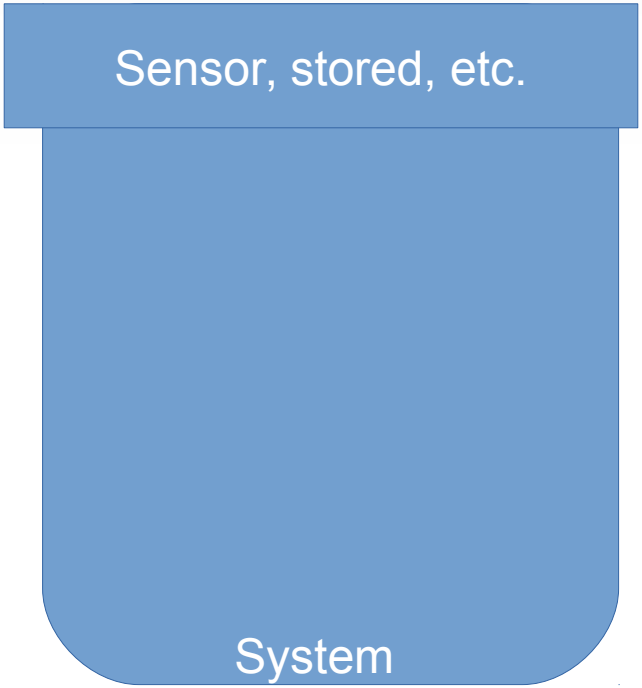
Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries



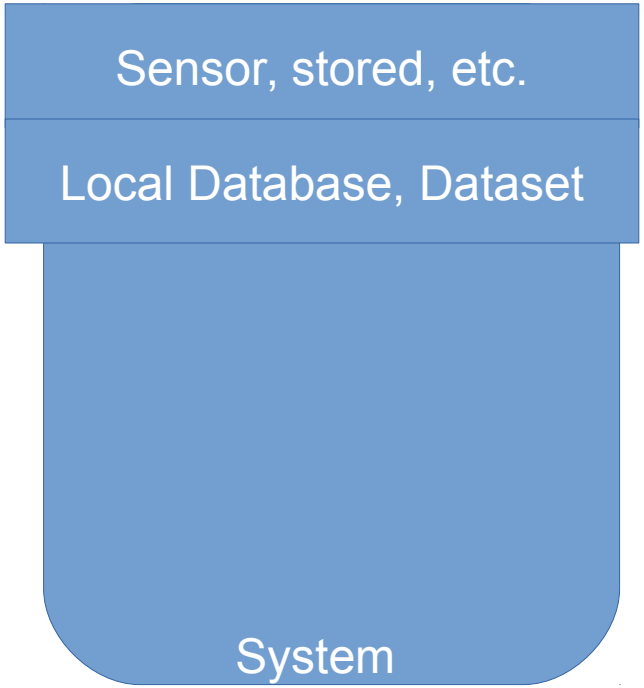
Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries



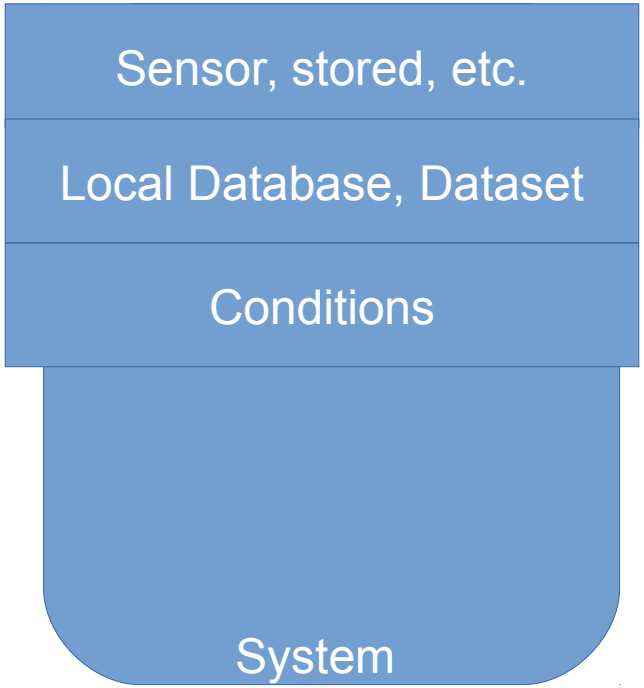
Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries



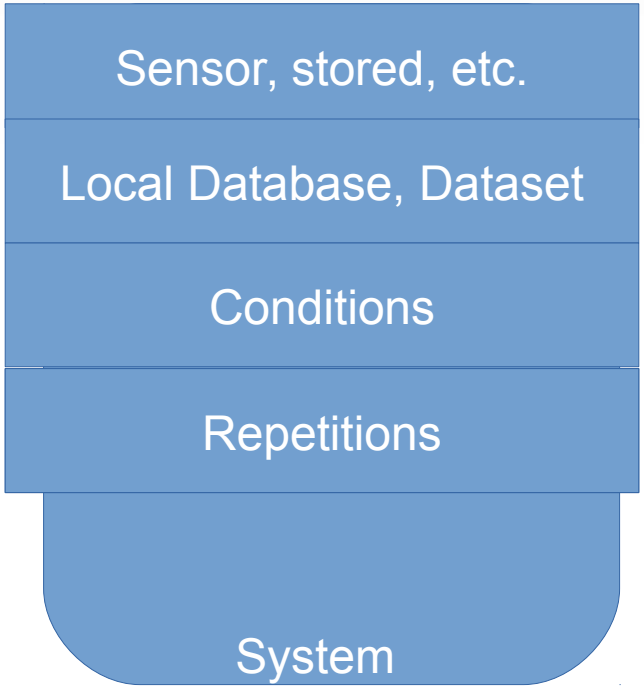
Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries



Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries



Python Scripting Language

- Data Input Output
- Data Types
- Conditional Statements
- Repetition Statements
- Functions and Libraries

Python Environment

Libraries

Read data

Operations: Filtering, Processing, Classification, Control etc.

Visualizing

Write, Operate etc

Modules and Functions

import math as mt

mt.functions…

import math

math.cos

from math import cos, pi

cos

from math import *

UCERD
Gathering
Intellectuals
www.ucerd.com

Example 1

Start ipython

Import some libraries

Create vectors or arrays

display

Reading and Writing Data

- Understand the source and type of data

Reading Files

```
 f = open(“names.txt")
>>> f.readline()
```

Results

Uses libraries to deal with complex databases and datastructures.

Data Types/Structure

Lists

Tuples

Set

Dictionary

List, Tuple, Set and Dictionary

- List: Use when user need an ordered sequence of homogenous collections, whose values can be changed later in the program.
- Tuple: User when you need an ordered sequence of heterogeneous collections whose values need not be changed later in the program.
- Set: It is ideal for user when user don't have to store duplicates and is not concerned about the order or the items. User just want to know whether a particular value already exists or not.
- Dictionary: It is ideal for use when user need to relate values with keys, in order to look them up efficiently using a key.

Condition

```
if continuation :
                    print value


if gpa > 2 :
    print gpa
```

Repetition

```
for x in range(1, 6, +1): # range(start, stop, step)
    print x
```

Agenda

- Importance
- Coding Languages
- Conventional Programming
- Python Language
- **Python for ML**

Numpy

- Numpy is the backbone of Machine Learning in Python. It is one of the most important libraries in Python for numerical computations. It adds support to core Python for multi-dimensional arrays (and matrices) and fast vectorized operations on these arrays.

Module: Numpy

NumPy is the fundamental package for scientific computing with Python.

It contains among other things:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

NumPy is a Python C extension library for array-oriented computing
- Efficient
- In-memory
- Contiguous (or Strided)
- Homogeneous (but types can be algebraic)

Source: http://www.numpy.org/

NumPy UFunctions

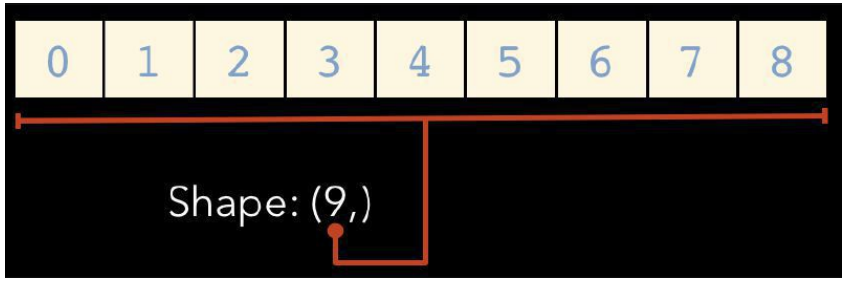
- comparison: `<, <=, ==, !=, >=, >`
- arithmetic: `+, -, *, /, reciprocal, square`
- exponential: `exp, expm1, exp2, log, log10, log1p, log2, power, sqrt`
- trigonometric: `sin, cos, tan, acsin, arccos, atctan`
- hyperbolic: `sinh, cosh, tanh, acsinh, arccosh, atctanh`
- bitwise operations: `&, |, ~, ^, left_shift, right_shift`
- logical operations: `and, logical_xor, not, or`
- predicates: `isfinite, isinf, isnan, signbit`
- other: `abs, ceil, floor, mod, modf, round, sinc, sign, trunc`

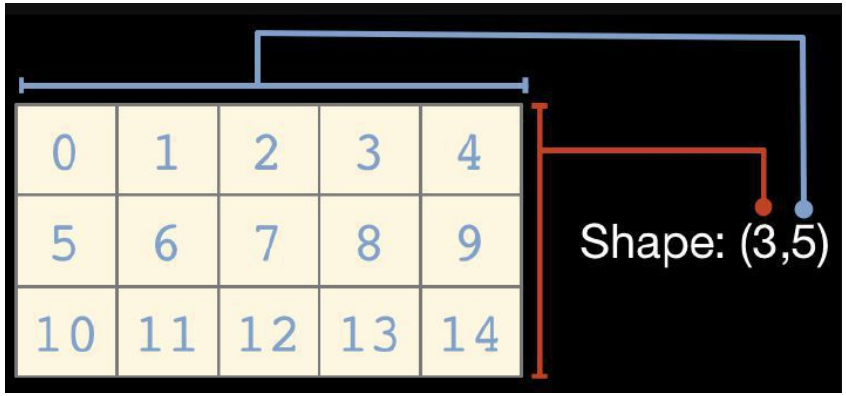
Understand Dataset

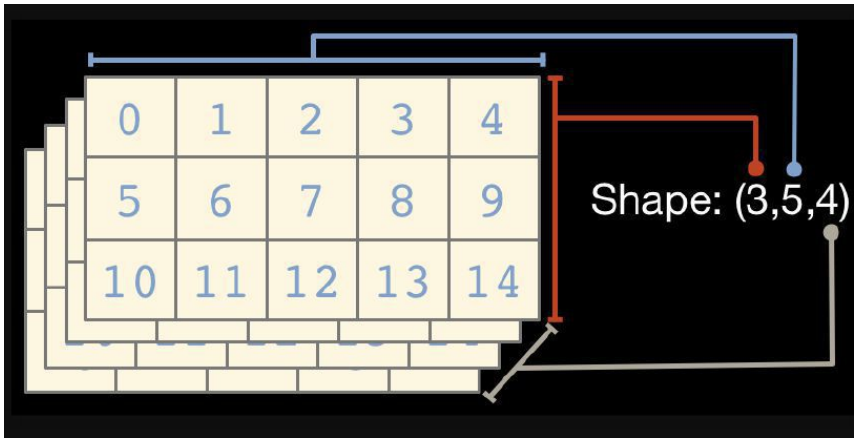
data.dtype

np.size(data)

np.shape(data)







```
data2 = genfromtxt('./signals/ecg.csv',
   delimiter=',')
np.shape(data2)

Signal = data2[:,1]
```

```
In [4]: import numpy as np
   ...: arr = np.array([1,3,4,5,6])
   ...: arr

Out[4]: array([1, 3, 4, 5, 6])

In [5]: arr.shape

Out[5]: (5,)

In [6]: arr.dtype

Out[6]: dtype('int32')


In [16]: arr = np.array([1,'st','er',3])
    ...: arr.dtype
Out[16]: dtype('<U11')

In [17]: np.sum(arr)
```

```
In [19]: arr = np.array([[1,2,3],[2,4,6],[8,8,8]])
    ...: arr.shape

Out[19]: (3, 3)

 In [20]: arr

Out[20]:
array([[1, 2, 3],
       [2, 4, 6],
       [8, 8, 8]])


  In [21]: arr = np.zeros((2,4))
      ...: arr
  Out[21]:array([[ 0.,  0.,  0.,  0.],
                 [ 0.,  0.,  0.,  0.]])
```

Accessing Array Element

```
In [50]: arr[0]
Out[50]:
array([[0, 1, 2],
       [3, 4, 5]])

In [57]: arr = np.arange(10)
    ...: arr[5:]
Out[57]: array([5, 6, 7, 8, 9])

In [58]: arr[5:8]
Out[58]: array([5, 6, 7])

In [60]: arr[:-5]
Out[60]: array([0, 1, 2, 3, 4])
```

Linear Algebra Using numpy

```
In [39]: A = np.array([[1,2,3],[4,5,6],[7,8,9]])
    ...: B = np.array([[9,8,7],[6,5,4],[1,2,3]])
```

```
In [40]: A.dot(B)
Out[40]:
array([[ 24,  24,  24],
       [ 72,  69,  66],
       [120, 114, 108]])
```

```
In [48]: np.linalg.svd(A)
```

Pandas

Pandas is an important Python library for data manipulation, wrangling, and analysis.

Pandas allows you to work with both cross-sectional data and time series based data. So let's get started exploring pandas!

All the data representation in pandas is done using two primary data structures:

- Series
- Dataframes

Series

Series in pandas is a one-dimensional ndarray with an axis label. It means that in functionality, it is almost similar to a simple array. The values in a series will have an index that needs to be hashable. This requirement is needed when we perform manipulation and summarization on data contained in a series data structure. Series objects can be used to represent time series data also. In this case, the index is a datetime object.

Dataframe

Dataframe is the most important and useful data structure, which is used for almost all kind of data representation and manipulation in pandas. Unlike numpy arrays (in general) a dataframe can contain heterogeneous data. Typically tabular data is represented using dataframes, which is analogous to an Excel sheet or a SQL table. This is extremely useful in representing raw datasets as well as processed feature sets in Machine Learning and Data Science. All the operations can be performed along the axes, rows, and columns, in a dataframe. This will be the primary data structure which we will leverage, in most of the use cases in our later chapters.

Datatypes

List of dictionaries

CSV files

Databases

Scikit-learn

Scikit-learn is one of the most important and indispensable Python frameworks for Data Science and Machine Learning in Python. It implements a wide range of Machine Learning algorithms covering major areas of Machine Learning like classification, clustering, regression, and so on. All the mainstream Machine Learning algorithms like support vector machines, logistic regression, random forests, K-means clustering, hierarchical clustering, and many many more, are implemented efficiently in this library. Perhaps this library forms the foundation of applied and practical Machine Learning. Besides this, its easy-to-use API and code design patterns have been widely adopted across other frameworks too!

```
from sklearn import datasets
iris = datasets.load_iris()
digits = datasets.load_digits()
```

Image Processing: OpenCV

OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. It has C++, C, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV was designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform.

Source: https://opencv.org/

Read and Display

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

#input handler
img = cv2.imread('./images/L1.jpg')
plt.imshow(img)
plt.show()
```