Introduction to Computer and Programming

Lecture 2

Yue Zhang
Westlake University

August 1, 2023

WestlakeNLP

Chapter 2.

Numbers and Expressions

WestlakeNLP

Overview

WestlakeNLP

Overview

WestlakeNLP

Binary Numbers

Figure: The movie, *The Matrix*

WestlakeNLP

Why binary numbers?

- The most robust form for information storage and communicational processing.
- Natural materials binary state.
  - Voltage
  - Solid and liquid

WestlakeNLP

Let us start with number systems.

Computers "think" in Binary Way

Base 10	Base 2
2	10
3	11
4	100
5	101
15	1111

Figure: Base-2 and Base-10

- We think in **base-10** systems.
  - 100, 256, -55, 0
- Computers think in **base-2** systems.
  - 010, 10110, 1101, 0

WestlakeNLP

What does the base do?

- How large is 256?
  - $2 \times 100 + 5 \times 10 + 6$

What does the base do?

- How large is 256?
  - $2 \times 100 + 5 \times 10 + 6$
  - $2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$

What does the base do?

- How large is 256?
  - $2 \times 100 + 5 \times 10 + 6$
  - $2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0$
- The base decides the order of magnitude for each digit!

WestlakeNLP

The General Form

$d = d_n d_{n-1} d_{n-2} \cdots d_2 d_1$

e.g., $18235 = 1\ 8\ 2\ 3\ 5$

base $= R$

e.g., $R = 10$

value $= d_n \times R^{n-1} + d_{n-1} \times R^{n-2} + \cdots + d_2 \times R^1 + d_1 \times R^0$

e.g., $18235 = 1 \times 10^4 + 8 \times 10^3 + 2 \times 10^2 + 3 \times 10^1 + 5 \times 10^0$

Other Bases

- What is 256 in base 8?

Other Bases

- What is 256 in base 8?

$$
\begin{aligned}
&2 \times 8^2 + 5 \times 8^1 + 6 \times 8^0 \\
&= 2 \times 64 + 5 \times 8 + 6 \\
&= 174
\end{aligned}
$$

WestlakeNLP

Other Bases

- What is 256 in base 16?

WestlakeNLP

Other Bases

- What is 256 in base 16?

$$
\begin{aligned}
& 2 \times 16^2 + 5 \times 16^1 + 6 \times 16^0 \\
= & \ 2 \times 256 + 5 \times 16 + 6 \\
= & \ 588
\end{aligned}
$$

WestlakeNLP

The larger the base, the larger the value.

$$d = 256 \qquad R = 10 \qquad \text{value} = 256$$
$$R = 8 \qquad \text{value} = 174$$
$$R = 16 \qquad \text{value} = 588$$

$$R = 2?$$

WestlakeNLP

The larger the base, the larger the value.

$$d = 256 \qquad R = 10 \qquad \text{value} = 256$$
$$R = 8 \qquad \text{value} = 174$$
$$R = 16 \qquad \text{value} = 588$$

$$R = 2?$$

Cannot hold 2, 5, 6!
The digit must be smaller than the base.

WestlakeNLP

Digit Symbols and Base

- Decimal numbers (base 10) have 10 digit symbols.
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Octal numbers (base 8) have 8 digit symbols.
  - 0, 1, 2, 3, 4, 5, 6, 7
- Binary numbers (base 2) have 2 digit symbols.
  - 0, 1

WestlakeNLP

Digit Symbols and Base

Can 257 be octal?
binary?

Can 259 be octal?
hexadecimal (base 16)?

WestlakeNLP

Digit Symbols and Base

How are digits in bases higher than 10 represented?

- with distinct symbols for 10 and above.
- hexadecimal numbers (base 16) have 16 digits.
  0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- What is the value of **1B** in hexadecimal?

Converting Binary to Decimal

What is the **decimal** equivalent of the binary number **1101110**?

$$
\begin{aligned}
& 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \\
= & 1 \times 64 + 1 \times 32 + 0 \times 16 + 1 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 1 \\
= & 110
\end{aligned}
$$

WestlakeNLP

Converting Decimal to Binary

Binary for 19?

$$19 = 16 + 2 + 1 = 2^4 + 2^1 + 2^0 = 10011$$

Converting Decimal to Binary

Binary for 19?

$$19 = 16 + 2 + 1 = 2^4 + 2^1 + 2^0 = 10011$$

$$19 \div 2 = 9 \cdots\cdots 1$$
$$9 \div 2 = 4 \cdots\cdots 1$$
$$4 \div 2 = 2 \cdots\cdots 0$$
$$2 \div 2 = 1 \cdots\cdots 0$$
$$1 \div 2 = 0 \cdots\cdots 1$$

↑ read

Can you prove this?

WestlakeNLP

Arithmatic in Binary

$$
\text{Addition}\quad \begin{aligned} 0+0&=0\\ 0+1&=1\\ 1+1&=0 \text{ with a carry}\end{aligned}
$$

WestlakeNLP

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
+ \quad 1\,0\,0\,1\,0\,1\,1 \\
\hline
\end{array}
$$

WestlakeNLP

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
+ \quad 1\,0\,0\,1\,0\,1\,1 \\
\hline
0
\end{array}
$$

Carry Value

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{rccccccc}
 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
+ & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hline
 & & & & & & 1 & 0
\end{array}
$$

Carry Value

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

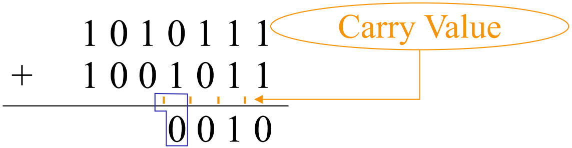
$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
+ \quad 1\,0\,0\,1\,0\,1\,1 \\
\hline
0\,1\,0
\end{array}
$$

Carry Value

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

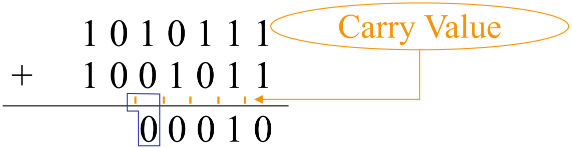


Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry



WestlakeNLP

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{rl}
 & 1\,0\,1\,0\,1\,1\,1 \\
+ & 1\,0\,0\,1\,0\,1\,1 \\
\hline
 & 1\,0\,0\,0\,1\,0
\end{array}
$$

Carry Value

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
+ \quad 1\,0\,0\,1\,0\,1\,1 \\
\hline
0\,1\,0\,0\,0\,1\,0
\end{array}
$$

Carry Value

Arithmatic in Binary

Addition $0 + 0 = 0$

$0 + 1 = 1$

$1 + 1 = 0$ with a carry

$$
\begin{array}{rcccccccc}
 & & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\
+ & & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hline
 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0
\end{array}
$$

Carry Value

Arithmatic in Binary

Subtraction

$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
-\quad 1\,1\,1\,0\,1\,1 \\
\hline
\end{array}
$$

Arithmatic in Binary

Subtraction

$$
\begin{array}{r}
1\,0\,1\,0\,1\,1\,1 \\
-\quad 1\,1\,1\,0\,1\,1 \\
\hline
\boxed{0}
\end{array}
$$

WestlakeNLP

Subtraction

	1	0	1	0	1	1	1
-		1	1	1	0	1	1
						0	0

Arithmatic in Binary

Subtraction

```
    1 0 1 0 1 1 1
-     1 1 1 0 1 1
-----------------
            1 0 0
```

Arithmatic in Binary

Subtraction

borrowing

-1
+2

```
  1 0 1 0 1 1 1
-   1 1 1 0 1 1
---------------
        1 1 0 0
```

Subtraction

borrowing

-1 -1

+2 +2

```
  1 0 1 0 1 1 1
-     1 1 1 0 1 1
-----------------
        1 1 1 0 0
```

Subtraction

borrowing

```
  -1 -1 -1
     +2 +2 +2
      1 0 1 0 1 1 1
-       1 1 1 0 1 1
---------------------
        0 1 1 1 0 0
```

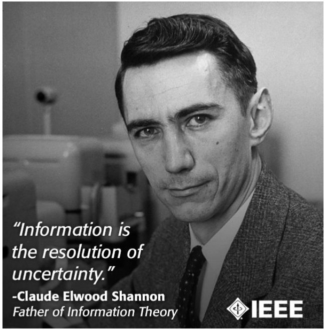
Subtraction

borrowing

```
        -1  -1  -1
            +2  +2  +2
        1 0 1 0 1 1 1
-         1 1 1 0 1 1
---------------------
        0 0 1 1 1 0 0
```

WestlakeNLP

- Binary numbers are connected with information theory.



Claude Elwood Shannon
(April 30, 1916 – February 24, 2001)

Information theory, a pillar of modern computing and communication, originated from the seminal work of Claude Shannon in the mid-20th century.

Information Theory

- What is information?
  - Amount of ambiguities resolved from a message.
  - Measured in bits.
  - Encoded in binary numbers.

WestlakeNLP

Information Theory – Example 1

[exam / no exam] 2 choices

knowing no exam $\rightarrow$ 2 choices to 1

information: $log_2 2 - log_2 1 = 1$ bit

encoding: 1 - exam

0 - no exam

WestlakeNLP

Information Theory – Example 2

[a deck of cards] 4 suits, 52 cards

$\spadesuit \heartsuit \clubsuit \diamondsuit$

knowing the suit $\rightarrow$ 4 choices to 1

information: $log_2 52 - log_2 13 = 2$ bits

encoding: 00 - $\spadesuit$ 01 - $\heartsuit$

10 - $\clubsuit$ 11 - $\diamondsuit$

WestlakeNLP

[a deck of cards] 4 suits, 52 cards

knowing the card ♠3 $\rightarrow$ 52 choices to 1

information: $log_2 52 < 6$ bits

encoding: 00 - ♠ 0011 - 3

$\therefore$ 000011 - ♠3

WestlakeNLP

Information Theory – Example 3

[a deck of cards] 4 suits, 52 cards

knowing the card ♠3 $\rightarrow$ 52 choices to 1

information: $log_2 52 < 6$ bits

encoding: 00 - ♠ 0011 - 3

$\therefore$ 000011 - ♠3

similarly, 01 - ♡ 1010 - 10

$\therefore$ 011010 - ♡10

Information Theory – Example 3

[a deck of cards] 4 suits, 52 cards

knowing the card ♠3 $\rightarrow$ 52 choices to 1

information: $log_2 52 < 6$ bits

encoding: 00 - ♠ 0011 - 3

$\therefore$ 000011 - ♠3

similarly, 01 - ♡ 1010 - 10

$\therefore$ 011010 - ♡10

♢K? 11 - ♢ 1101 - 13 $= K$

WestlakeNLP

Information Theory – Example 4

[a deck of cards] 4 suits, 52 cards

knowing the number 5 $\rightarrow$ 52 choices to 1

information: $log_2 52 - log_2 4 = log_2 13 < 4$ bits

encoding: A - 0001, 2 - 0010, 3 - 0011, 4 - 0100,
5 - 0101, 6 - 0110, 7 - 0111, 8 - 1000,
9 - 1001, 10 - 1010, 11 - 1011, 12 - 1100,
13 - 1101

WestlakeNLP

Overview

Start Learning Python

What does a computer do?

- calculate mathematical functions

What does a computer do?

- calculate mathematical functions
- step by step

Start Learning Python

What does a computer do?

- calculate mathematical functions
- step by step
- in underlying binary format

WestlakeNLP

Start Learning Python

What does a computer do?

- calculate mathematical functions
- step by step
- in underlying binary format

Thus a simple way of learning Programming is to start with math calculator.

WestlakeNLP

Using Python as a Calculator – Integer Functions (IDLE)

```
>>> 3+5
8
>>> 8-7
1
>>> 6*9
54
>>> 24/6
4.0
```

WestlakeNLP

Using Python as a Calculator – Integer Functions (IDLE)

```
>>> 3+5
8
>>> 8-7
1
>>> 6*9
54
>>> 24/6
4.0
```

Literals

3, 5, 8, 7, 6, 9, 24, 40

Operators

$+, -, *(\times), /(\div)$

Expressions

$3+5,\ 8-7,\ 6*9,\ 24/6$

WestlakeNLP

Using Python as a Calculator – Integer Functions (IDLE)

```
>>> 3+5
8
>>> 8-7
1
>>> 6*9
54
>>> 24/6
4.0
```

Literals

$3, 5, 8, 7, 6, 9, 24, 40$

Operators

$+, -, *(\times), /(\div)$

Expressions

$3+5,\ 8-7,\ 6*9,\ 24/6$

- Expressions have **values** wherever you type one expression in IDLE, the value is shown.
- **Evaluation** of expressions: find the **values** according to **operands** and **operators**.

Using Python as a Calculator – Composite Expressions

```
>>> 3+2-5+1
1
```

- left-to-right evaluation

WestlakeNLP

Using Python as a Calculator – Composite Expressions

```
>>> 3+2-5+1
1
```

```
>>> 3+2*5-4
9
```

- Left-to-Right Evaluation
- **Operator Precedence**
  (1) $*$, $/$ higher
  (2) $+$, $-$ lower

WestlakeNLP

Using Python as a Calculator – Composite Expressions

```
>>> 3+2-5+1
1
```

```
>>> 3+2*5-4
9
```

```
>>> (3+2)*(5-4)
5
```

- Left-to-Right Evaluation
- **Operator Precedence**
  (1) $*$, $/$ higher
  (2) $+$, $-$ lower
- **Brackets** specify order of evaluation.

WestlakeNLP

Using Python as a Calculator – More Operators

```
>>> 5**2
25
```

```
>>> 5%2
1
```

```
>>> -(5*1)
-5
```

- Power – **Binary Operator**
  takes two operands
- Modulus – **Binary Operator**
- Negation – **Unary Operator**
  takes one operand

There are unary, binary and ternary operators in python.

We will learn ternary operator later on.

WestlakeNLP

Using Python as a Calculator – More Literals

```
>>> 0b1101011
107
>>> 0b11111111
255
```

- Binary Numbers

Using Python as a Calculator – More Literals

```
>>> 0b1101011
107
>>> 0b11111111
255
```

```
>>> 5e2
500.0
>>> 6e-1
0.6
>>> 3E5
300000.0
```

- Binary Numbers
- Scientific Notation

WestlakeNLP

Using Python as a Calculator – Floating Point Numbers

```
>>> type(1)
<class 'int'>
>>> type(4.0)
<class 'float'>
>>> type(0b11111)
<class 'int'>
>>> type(3E2)
<class 'float'>
```

- Literals have **types**
  int - integer
  float - floating point numbers

- Similarly, expressions (their values) have types.

```
>>> 24/6
4.0
>>> type(24/6)
<class 'float'>
```

WestlakeNLP

Using Python as a Calculator – Floating Point Numbers

- Floating Point Expressions

```
>>> 7.0/2
3.5
>>> 26/2.0
13.0
>>> 25**0.5
5.0
>>> 3.1+2.4
5.5
```

WestlakeNLP

Using Python as a Calculator – Floating Point Numbers

- Floating Point Expressions

```
>>> 7.0/2
3.5
>>> 26/2.0
13.0
>>> 25**0.5
5.0
>>> 3.1+2.4
5.5
```

- Integer Division

```
>>> 26//4
6
>>> 26%4
2
```

WestlakeNLP

Using Python as a Calculator – Functions

```
>>> type(5.0)
<class 'float'>
```

- What is type(5.0)?
  - type is a **function**
  - A function takes a set of **arguments** and yields a **return value**.
  - type(5.0) is a **function call**.
  - A function call is an expression.

WestlakeNLP

More Function Calls

```
>>> int(3.0)
3
>>> int(3.1)
3
>>> int(3.9)
3
>>> round(3.1)
3
>>> round(3.5)
4
```

- float to int

WestlakeNLP

More Function Calls

```
>>> int(3.0)
3
>>> int(3.1)
3
>>> int(3.9)
3
>>> round(3.1)
3
>>> round(3.5)
4
```

- float to int

```
>>> float(3)
3.0
```

- int to float

More Function Calls

```
>>> round(3.333,1)
3.3
>>> round(3.333,2)
3.33
```

- functions with more than one arguments

WestlakeNLP

Using Python as a Calculator – Functions

```
>>> round(10**0.5, 2)
3.16
```

- The order for evaluation composite expressions with functions.

arguments $\rightarrow$ function call

WestlakeNLP

What is the type of functions?

```
>>> type(int)
<class 'type'>
```

```
>>> type(round)
<class 'builtin_function_or_method'>
```

- function objects

We will learn objects.

WestlakeNLP

Using Python as a Calculator – Variables

```
>>> a=1000
>>> w=a**0.5
>>> round(w,2)
31.62
```

- **identifiers** (v.s. literals)

  a, w (v.s. 1000, 0.5, 0.2)
- **Variables** can change their values, and are represented by identifiers.
- **Constants** do not change their values, and are represented by literals.

WestlakeNLP

Using Python as a Calculator – Variables

```
>>> a=1000
>>> w=a**0.5
>>> round(w,2)
31.62
```

- **assignment**
  a=1000
  w=a**0.5

- Assignment of values to variables.
- Assignments are **statements**, which are the basic commands of a programming language.
- A statement does not have a value.

WestlakeNLP

Using Python as a Calculator – One Problem-solving Example

- If a ball is thrown upwards with an initial velocity $v0 = 5m/s$ from an initial altitude of $0m$, what is its altitude after $0.1s$? $(h = v_0t + \frac{1}{2}gt^2)$

```
>>> v0=5
>>> g=9.81
>>> t1=0.1
>>> h1=v0*t1-0.5*g*t1**2
>>> round(h1,2)
0.45
```

WestlakeNLP

Using Python as a Calculator – One Problem-solving Example

- If a ball is thrown upwards with an initial velocity $v0 = 5m/s$ from an initial altitude of $0m$, what is its altitude after $0.1s$? ($h = v_0 t + \frac{1}{2} g t^2$)

```
>>> v0=5
>>> g=9.81
>>> t1=0.1
>>> h1=v0*t1-0.5*g*t1**2
>>> round(h1,2)
0.45
```

- what is its altitude after $1s$?

```
>>> t2=1
>>> h2=v0*t2-0.5*g*t2**2
>>> round(h2,2)
0.09
```

WestlakeNLP

Using Python as a Calculator – Valid Identifiers

- start with letter or underscore
- contain letters, numbers and underscore

valid	invalid
a, a0, _a, area	0a, area!

- cannot be keywords

WestlakeNLP

Using Python as a Calculator – Keywords

and	as	assert	async	await	break	class	continue	def
del	elif	else	except	False	finally	for	from	global
if	import	in	is	lambda	None	nonlocal	not	or
pass	raise	return	True	try	while	with	yield	__peg_parser__

Using Python as a Calculator – Variable Reassignments

```
>>> x=1
>>> x
1
>>> x=2
>>> x
2
```

- initial assignments
- reassignment
  variable value changes

WestlakeNLP

Using Python as a Calculator – Variable Reassignments

```
>>> x=1
>>> x
1
>>> x=2
>>> x
2
```

```
>>> x=x+1
>>> x
3
```

- initial assignments
- reassignment

  variable value changes

WestlakeNLP

Using Python as a Calculator – Variable Reassignments

```
>>> x=1
>>> x
1
>>> x=2
>>> x
2
```

```
>>> x=x+1
>>> x
3
```

- initial assignments
- reassignment

  variable value changes
- How can x=x+1?
  - right hand side evaluated first
  - value assigned to x

WestlakeNLP

- Shortcut

```
>>> x=1
>>> x+=1
>>> x
2
```

x+=1	x=x+1
x-=2	x=x-2
x*=3	x=x*3
x/=4	x=x/4

WestlakeNLP

Using Python as a Calculator – One More Case

- Width and Area of a Square

```
>>> w=2
>>> a=w*w
>>> a
4
>>> w=3
>>> a=w*w
>>> a
9
```

WestlakeNLP

Using Python as a Calculator – One More Case

- Width and Area of a Square

```
>>> w=2
>>> a=w*w
>>> a
4
>>> w=3
>>> a=w*w
>>> a
9
```

- Can this a=w*w be omitted?

WestlakeNLP

Using Python as a Calculator – Math Module

The math module has more math utilities.

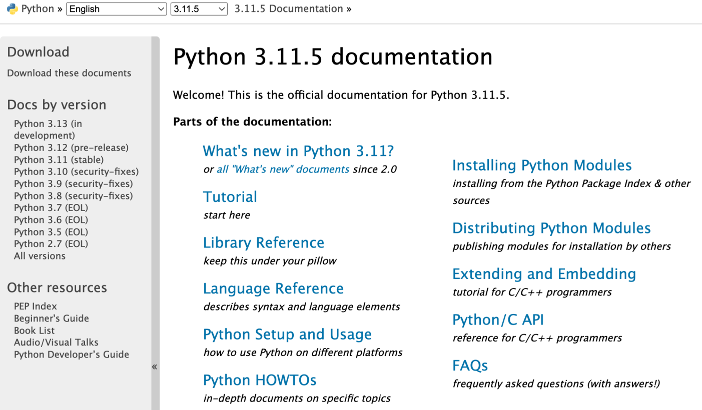
```
>>> import math
>>> math.pi
3.141592653589793
>>> math.e
2.718281828459045
>>> math.factorial(10)
3628800
>>> math.log(100)
4.605170185988092
>>> math.sin(3)
0.1411200080598672
>>> math.cos(7)
0.7539022543433046
```

- keyword import is a second statement
- identifier math is a module name
- math is an object
- **Content**
  - math.pi
  - math.log
  - ...

Checkout math, cmath and random modules from **Python Documentation**!

Using Python as a Calculator – Math Module

- Python Documentation (https://docs.python.org/3/)



Python » English 3.11.5 3.11.5 Documentation »

Download

Download these documents

Docs by version

- Python 3.13 (in development)
- Python 3.12 (pre-release)
- Python 3.11 (stable)
- Python 3.10 (security-fixes)
- Python 3.9 (security-fixes)
- Python 3.8 (security-fixes)
- Python 3.7 (EOL)
- Python 3.6 (EOL)
- Python 3.5 (EOL)
- Python 2.7 (EOL)
- All versions

Other resources

- PEP Index
- Beginner's Guide
- Book List
- Audio/Visual Talks
- Python Developer's Guide

Python 3.11.5 documentation

Welcome! This is the official documentation for Python 3.11.5.

Parts of the documentation:

What's new in Python 3.11?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

Distributing Python Modules
publishing modules for installation by others

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

FAQs
frequently asked questions (with answers!)

Using Python as a Calculator – Math Module

```
>>> type(math)
<class 'module'>
```

- module object

- Types we learned: int, float, type, function, module

This week check-off:
Solving Mathematical Problems

WestlakeNLP