

Python

- What exactly is Python?
 - A tool for the practical data scientist
 - High-level programming language that has become widely used in a variety of settings
 - Provides libraries to do several things
 - A good general-purpose, high-level language
 - Makes very readable
 - Diverse range of open source libraries
 - Anaconda
 - Includes over 195 most popular python packages



Using Python

- Python must be installed and configured prior to use
 - One of the items installed is the Python interpreter
- Python interpreter can be used in two modes:
 - Interactive mode: enter statements on keyboard
 - Script mode: save statements in Python script



Script Mode

- Statements entered in interactive mode are not saved as a program
- To have a program use script mode
 - Save a set of Python statements in a file
 - The filename should have the .py extension
 - To run the file, or script, type
`python filename`
at the operating system command line

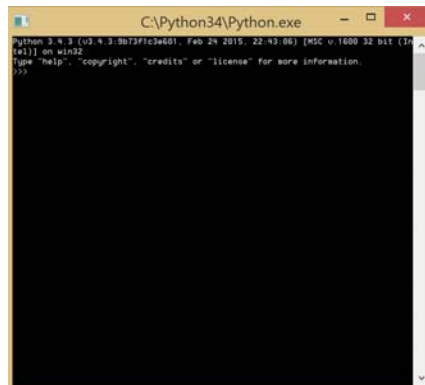
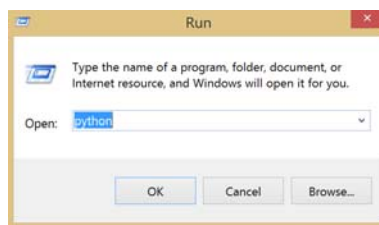


Interactive Mode

- When you start Python in interactive mode, you will see a prompt
 - Indicates the interpreter is waiting for a Python statement to be typed
 - Prompt reappears after previous statement is executed
 - Error message displayed If you incorrectly type a statement
- Good way to learn new parts of Python

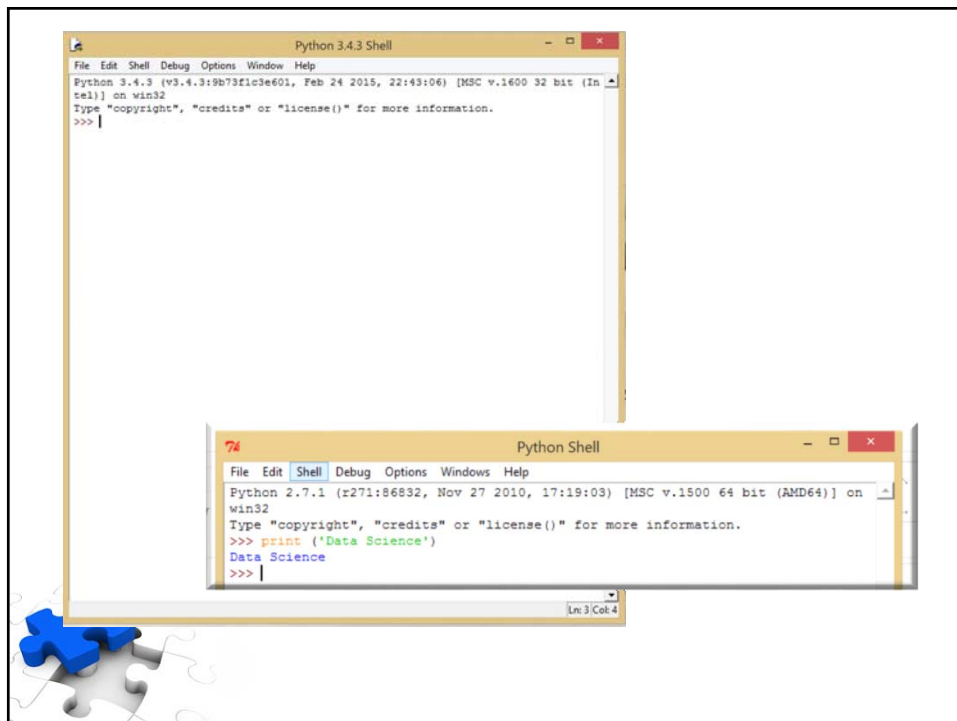


A Console Window

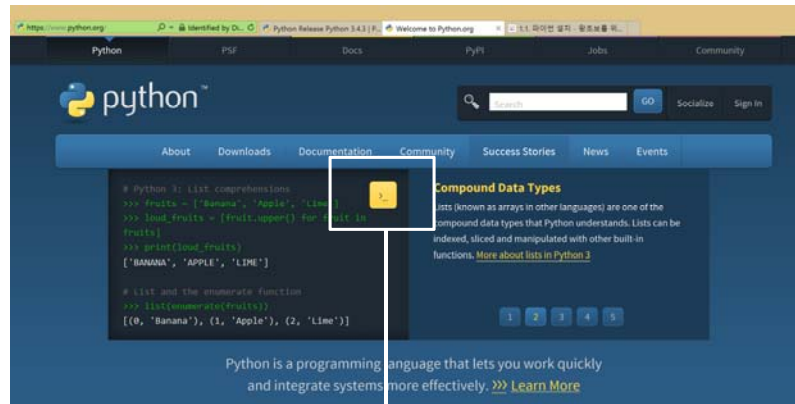


The IDLE Programming Environment

- IDLE (Integrated Development Program): single program that provides tools to write, execute and test a program
 - Automatically installed when Python language is installed
 - Runs in interactive mode
 - Has built-in text editor with features designed to help write Python programs



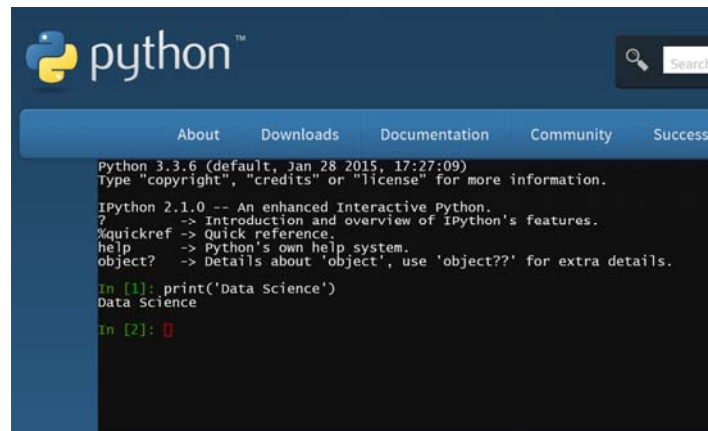
Python on Web Browser



Launch Interactive Shell



Python on Web Browser




Install Python



Install Python






Anaconda




Log In [or](#) [Support](#) [Contact](#)

ANACONDA COMMUNITY CONSULTING TRAINING ABOUT RESOURCES

DOWNLOAD ANACONDA NOW

Download for   




GET SUPERPOWERS WITH ANACONDA

Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages

Which version should I download and install?

With Anaconda you can run multiple versions of Python in isolated environments, so choose the download with the Python version that you use more often, as that will be your default Python version.



Anaconda

[Download for Windows](#) [Download for OSX](#) [Download for Linux](#)

Anaconda 4.1.1

For Windows

Anaconda is BSD licensed which gives you permission to use Anaconda commercially and for redistribution.

[Changelog](#)

1. Download the installer
2. Optional: Verify data integrity with [MD5](#) or [SHA-256](#)
3. Double-click the .exe file to install Anaconda and follow the instructions on the screen

Behind a Firewall? Use these [zipped Windows installers](#)

Python 3.5 version

64-BIT INSTALLER (351M)

32-BIT INSTALLER (292M)


Python 2.7 version

64-BIT INSTALLER (340M)

32-BIT INSTALLER (285M)


For older versions of Anaconda installers, see the [Anaconda installer archive](#)

For long-term support of the packages found in the Anaconda archives, please [contact us](#).



Anaconda

Program → anaconda → Ipython (Py 3.4) QtConsole



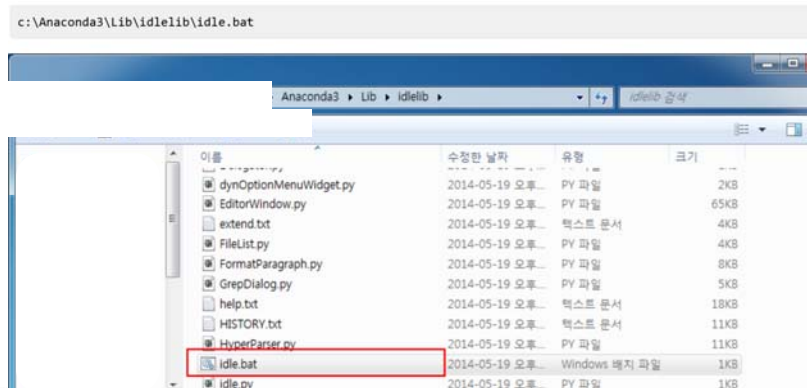
```
Python
File Edit View Kernel Magic Window Help
Python 3.4.1 [Anaconda 2.1.0 (32-bit)] (default, Sep 24 2014, 18:34:57) [MSC
v.1600 32 bit (Intel)]
Type "copyright", "credits" or "license()" for more information.

IPython 2.2.0 -- An enhanced Interactive Python.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://binstar.org
? -> Introduction and overview of IPython's features.
Quickref -> Quick reference.
help -> Python's own help system.
object? -> Details about 'object', use 'object??' for extra details.
Agiuref -> A brief reference about the graphical user interface.

In [1]: print("hello")
hello

In [2]:
```

IDLE in Anaconda



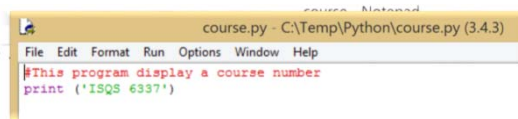
Displaying Output

- *print* function
 - *function*
 - piece of prewritten code that performs an operation
 - displays output on the screen
 - *Argument*.
 - data given to a function
 - Example: data that is printed to screen
 - Statements in a program execute in the order that they appear
 - From top to bottom



String and String Literals & Comments

```
>>> print('Jaeki Song')
Jaeki Song
>>> print("jaeki song")
jaeki song
>>> print("""Jaeki "K" Song""")
Jaeki "K" Song
```

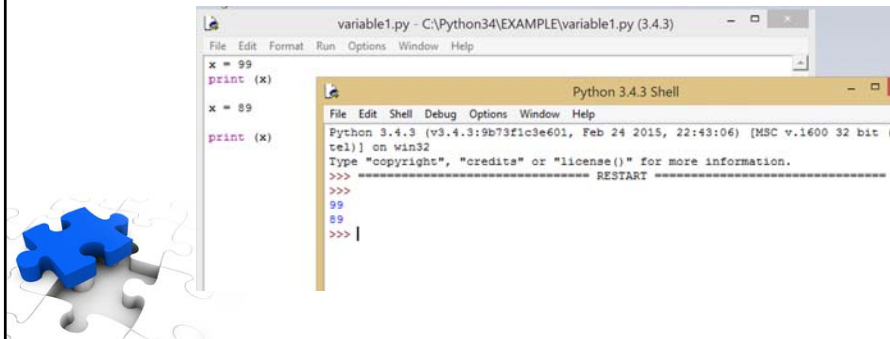


```
tel] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>> ISQS 6337
>>> |
```



Variables

- Variable: name that represents a value stored in the computer memory
- Assignment statement: used to create a variable and make it reference data



Reading Input


- Programs need to read input typed by the user on the keyboard
- Built-in `input` function reads input from keyboard
 - Returns the data as a string
 - Format: `variable = input(prompt)`
 - `prompt` is typically a string instructing user to enter a value

The image shows a snippet of Python code using the `input` function. The code prompts the user to enter their name and then prints the entered name. The output shows the user entering 'Jaeki Song' and the program printing 'Jaeki Song'.

```
>>> name=input('Enter Your Name:')
Enter Your Name:Jaeki Song
>>> print (name)
Jaeki Song
>>> |
```

Reading Input

- `input` function **always returns** a string
- Built-in functions convert between data types
 - `int(item)` converts *item* to an `int`
 - `float(item)` converts *item* to a `float`
 - Nested function call: general format:
`function1(function2(argument))`



```
input1.py - C:/Python34/EXAMPLE
File Edit Format Run Options Window Help
#Get the user's info
name = input('Enter your name:')
age = int(input('What is your age?'))
income = float(input('What is your income?'))

#Display the data
print('Here is the data you entered: ')
print('Name: ', name)
print('Age: ', age)
print('Income: ', income)
```

Enter your name: Jaeki
What is your age? 25
What is your income? 1000000.50
Here is the data you entered:
Name: Jaeki
Age: 25
Income: 1000000.5

Example

- A retail business is planning to have a storewide sale where the prices of all items will be 20 percent off. Write a program to calculate the sale price of an item after the discount is subtracted.

```
Enter the item's original price: 100
The sale price is 80.0
```



Performing Calculation

- Math expression: performs calculation and gives a value
- Two types of division:
 - / operator performs floating point division
 - `>>> 5/2 → 2.5`
 - // operator performs integer division
 - Positive results truncated
 - `>>> 5//2 → 2`
 - negative rounded away from zero
 - `>>> -5 // 2 → -3`



Operator Precedence

- Python operator precedence:
 1. Operations enclosed in parentheses
 - Forces operations to be performed before others
 2. Exponentiation (`**`)
 3. Multiplication (`*`), division (`/` and `//`), and remainder (`%`)
 4. Addition (`+`) and subtraction (`-`)
- Higher precedence performed first
 - Same precedence operators execute from left to right



To do..

```
>>>
Enter a number of seconds: 11730
Here is the time in hours minutes, and seconds:
Hours: 3.0
Minutes: 15.0
Seconds: 30.0
>>> |
```



Breaking Long Statements into Multiple Lines

- Long statements cannot be viewed on screen without scrolling and cannot be printed without cutting off
- Multiline continuation character (\)
 - Allows to break a statement into multiple lines
 - Example:

```
print('my first name is',\
first_name)
```



Data

- Integer
 - Range: -2,147,483,648 to 2,147,483,647
- Float
 - Floating-point numbers have decimal points

```
>>> int(True)
1
>>> int(False)
0
>>> float(True)
1.0
>>> float(False)
0.0
```



To do...

- Suppose you want to deposit a certain amount of money into a savings account and then leave it alone to draw interest for the next 10 years. At the end of 10 years you would like to have \$10,000 in the account. How much do you need to deposit today to make that happen?

$$P = \frac{F}{(1 + r)^n}$$

, where P is the present value, F is the future value,
 r is the interest rate, n is the number of years

```
Enter the desired future value: 10000
Enter the annual interest rate: 0.05
Enter the number of years the money will grow: 10
You will need to deposit this amount: 6139.132535407592
```



String

- Enclose characters in either single or double quotes

```
>>> name = 'Jaeki
SyntaxError: EOL while scanning string literal
>>> name = "jaeki
SyntaxError: EOL while scanning string literal
>>> name = '''jaeki
song'''
>>> print (name)
jaeki
song
>>> |
```



String

- Escape with \
- Combine with +
- Duplicate with *

```
>>> address = '703 Flint Avenue\nLubbock, TX, 79409'
>>> print(address)
703 Flint Avenue
Lubbock, TX, 79409
```

```
>>> add1 = '703 Flint Avenue'
>>> add2 = 'Lubbock, TX, 79409'
>>> add1+add2
'703 Flint AvenueLubbock, TX, 79409'
```

```
>>> start = 'Na ' * 4 + '\n'
>>> middle = 'Hey ' * 3 + '\n'
>>> end = 'Goodbye.'
>>> print (start + start + middle + end)
Na Na Na Na
Na Na Na Na
Hey Hey Hey
Goodbye.
```



String

- Extract a character with []

```
>>> letters = 'abcdefghijklmnopqrstuvwxyz'
>>> letters[0]
'a'
>>> letters[1]
'b'
>>> letters[-1]
'z'
>>> letters[-2]
'y'
>>> letters[25]
'z'
```

replace

```
>>> name = 'Jaeki Tong'
>>> name.replace('T', 'S')
'Jaeki Song'
```



String

- Indexing
 - “python is very powerful”

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
p	y	t	h	o	n		i	s		v	e	r	y		p

...



String

- Slice with [start:end:step]
 - [:] extracts the entire sequence from start to end
 - [start :] specifies from the *start* offset to the end
 - [: end] specifies from the beginning to the *end* offset minus 1
 - [start : end] indicates from the *start* offset to the *end* offset minus 1
 - [start: end: step] extract from the *start* offset to the *end* offset minus 1, skipping characters by *step*



String

```
>>> letters = 'abcdefghijklmnopqrstuvwxyz'
>>> letters [:]
'abcdefghijklmnopqrstuvwxyz'
>>> letters [20:]
'uvwxyz'
>>> letters [12:15]
'mno'
>>> letters [-6:-2]
'uvwx'
>>> letters [4:20:3]
'ehknqt'
```



String

- len()

```
>>> len('letters')
26
>>> empty=""
>>> len(empty)
0
```

- split()

```
>>> tools = 'get gloves, get mask, get cat vitamins, call ambulance'
>>> tools.split(',')
['get gloves', ' get mask', ' get cat vitamins', ' call ambulance']
```



Data Output

- Formatting numbers

- Two arguments:
 - Numeric value to be formatted
 - Format specifier
- Returns string containing formatted number
- Format specifier typically includes precision and data type



Data Output

- Floating-point number
 - Format(1234.56789, '.2f')
 - .2 → specifies the precision
 - **f** → specifies that the data type of the number formatting is a floating-point number
- Integer
 - Use d as the type designator
 - Cannot specify precision
 - format (123456, 'd') → 123456
 - format (123456, ',d') → 123,456



InClass 5-1

Write a program that will ask the user to enter the amount of a purchase. The program should compute the state and sales tax. Assume that state sales tax is 5 percent and the county sales tax is 2.5 percent. The program should display the amount of the purchase, the state sales tax, the county sales tax, the total sales tax, and the total of the sale (which is the sum of the amount of purchase plus the total sales tax).

```
Enter the amount of the purchase: 1000
Purchase Amount: 1000.00
State Tax: 50.00
County Tax: 25.00
Total Tax: 75.00
Sale Total: 1075.00
```

