TUG TEC 20i8

"TEC it to the limit"

TEC 25th edition -- Wednesday May 30 & Thursday May 31, 2018

FRESCHE SOLUTIONS

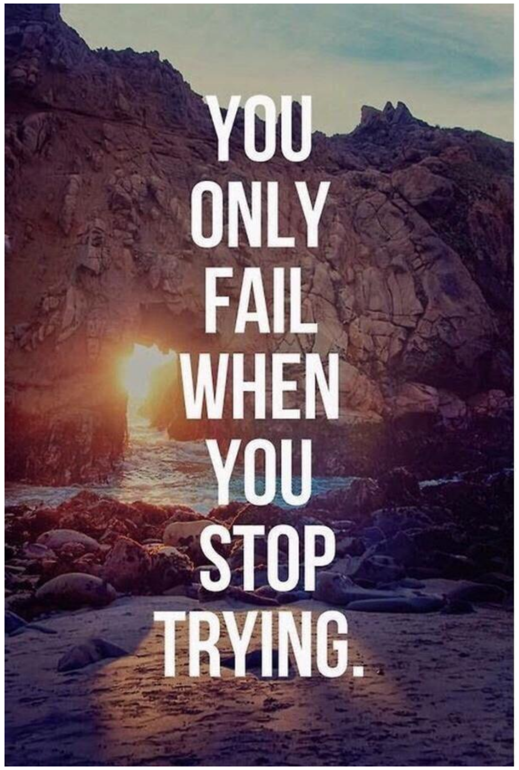
Introduction to Python for IBM i

Mike Pavlak – IT Strategist

mike.pavlak@freschesolutions.com

Agenda

- A little about Python
- Why use Python
- How to install/determine if installed
  - IDE
- Syntax 101
  - Variables
  - Strings
  - Functions



Acknowledgements

- Kevin Adler
- Tony Cairns
- Jesse Gorzinski
- Google
- Memegenerator
- Corn chips and salsa
- Parrots
- And, of course, spam

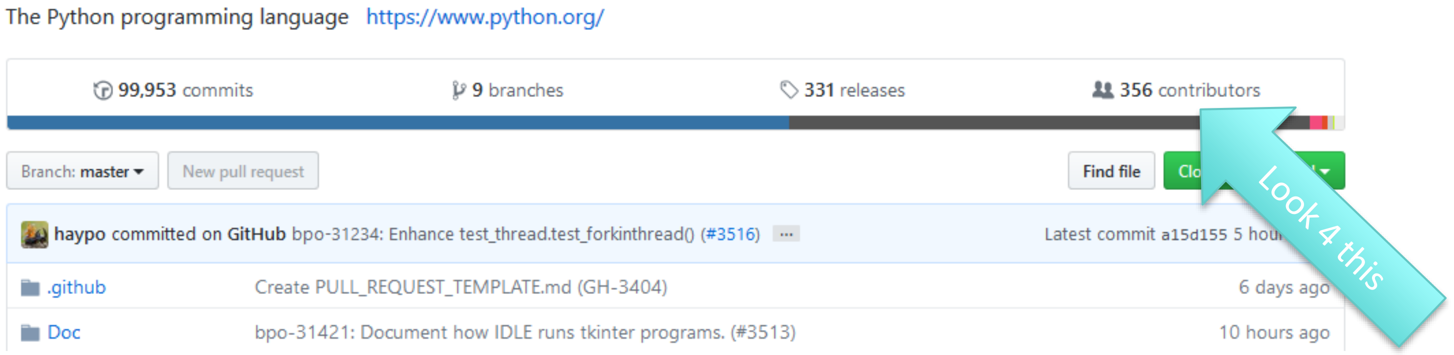
A little about Python

What is it, really?

- General purpose language
- Easy to get started
- Simple syntax
- Great for integrations (glue between systems)
- Access to C and other APIs
- Infrastructure first, but applications, too

Historically…

- Python was conceptualized by Guido Van Rossum in the late 1980's
- Rossum published the first version of Python code (0.9.0) in February of 1991 at the CWI(Centrum Wiskunde & Informatica) in the Netherlands, Amsterdam
- Python is derived from the ABC programming language, which is a general purpose language that was also developed at CWI.
- Rossum chose the name "Python" since he was a fan of Monty Python's Flying Circus.
- Python is now maintained by a core development team at the institute, although Rossum still holds a vital role in directing its progress and as leading "commitor".


The Python programming language https://www.python.org/

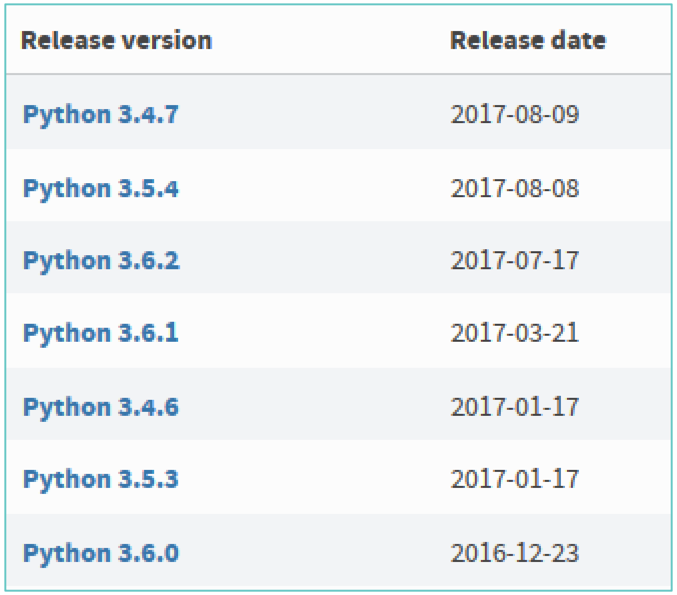
99,953 commits | 9 branches | 331 releases | 356 contributors

haypo committed on GitHub bpo-31234: Enhance test_thread.test_forkinthread() (#3516) … Latest commit a15d155 5 hou

.github	Create PULL_REQUEST_TEMPLATE.md (GH-3404)	6 days ago
Doc	bpo-31421: Document how IDLE runs tkinter programs. (#3513)	10 hours ago

Python lineage

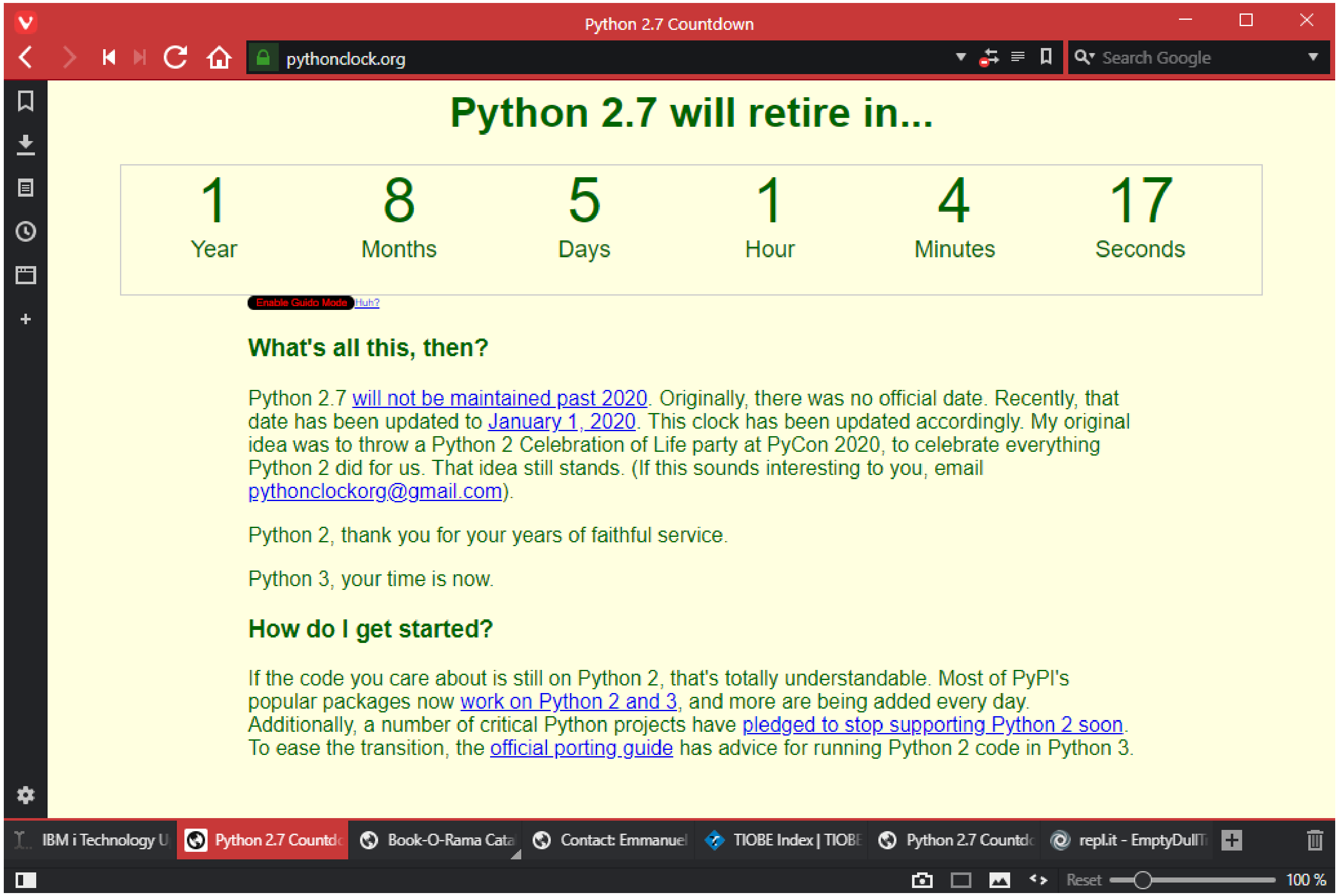
- Python 1 – 1994
- Python 2 – 2000 (Not dead yet…)
  - 2.7 – 2010
- Python 3 – 2008
  - 3.5 – 2015
  - 3.6.2 – July 2017
  - 3.7 → ETA July 2018



Release version	Release date
Python 3.4.7	2017-08-09
Python 3.5.4	2017-08-08
Python 3.6.2	2017-07-17
Python 3.6.1	2017-03-21
Python 3.4.6	2017-01-17
Python 3.5.3	2017-01-17
Python 3.6.0	2016-12-23

FRESCHE

Python 2 or 3?



Python 2.7 will retire in...

1	8	5	1	4	17
Year	Months	Days	Hour	Minutes	Seconds

Enable Guido Mode Huh?

What's all this, then?

Python 2.7 will not be maintained past 2020. Originally, there was no official date. Recently, that date has been updated to January 1, 2020. This clock has been updated accordingly. My original idea was to throw a Python 2 Celebration of Life party at PyCon 2020, to celebrate everything Python 2 did for us. That idea still stands. (If this sounds interesting to you, email pythonclockorg@gmail.com).

Python 2, thank you for your years of faithful service.

Python 3, your time is now.

How do I get started?

If the code you care about is still on Python 2, that's totally understandable. Most of PyPI's popular packages now work on Python 2 and 3, and more are being added every day. Additionally, a number of critical Python projects have pledged to stop supporting Python 2 soon. To ease the transition, the official porting guide has advice for running Python 2 code in Python 3.

What’s the diff?

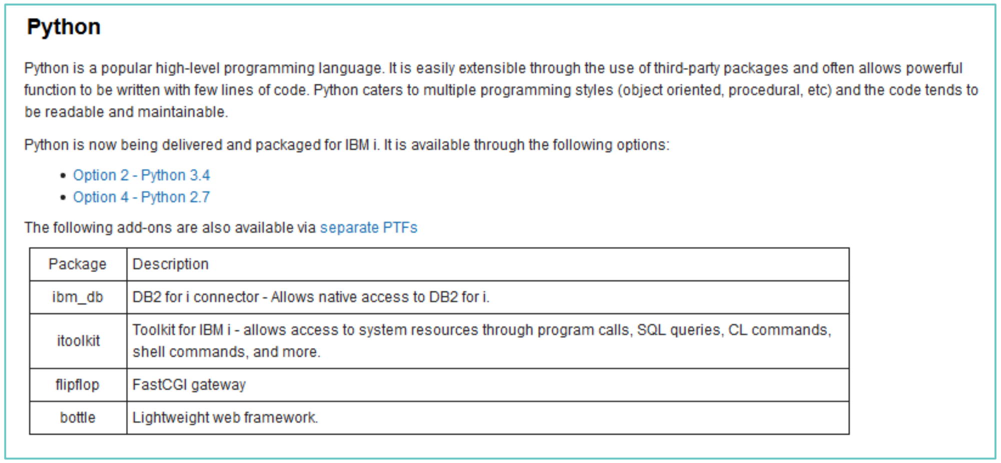
- Example:
  - Python 2 print statement replaced by function
    - Python 2 – print “Hello World!”
    - Python 3 – print(“Hello World!”)
- *Many more differences, tho…*
- *Which one?*
  - Correct answer: It depends…
    - Many existing libraries are Python 2
    - But 90%+ are also Python 3 compliant, or on their way

Got Python?

Details at Developerworks

Updated March 2018, Thanks Jesse!

- https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/IBM%20i%20Technology%20Updates/page/Open%20Source%20Technologies



Python

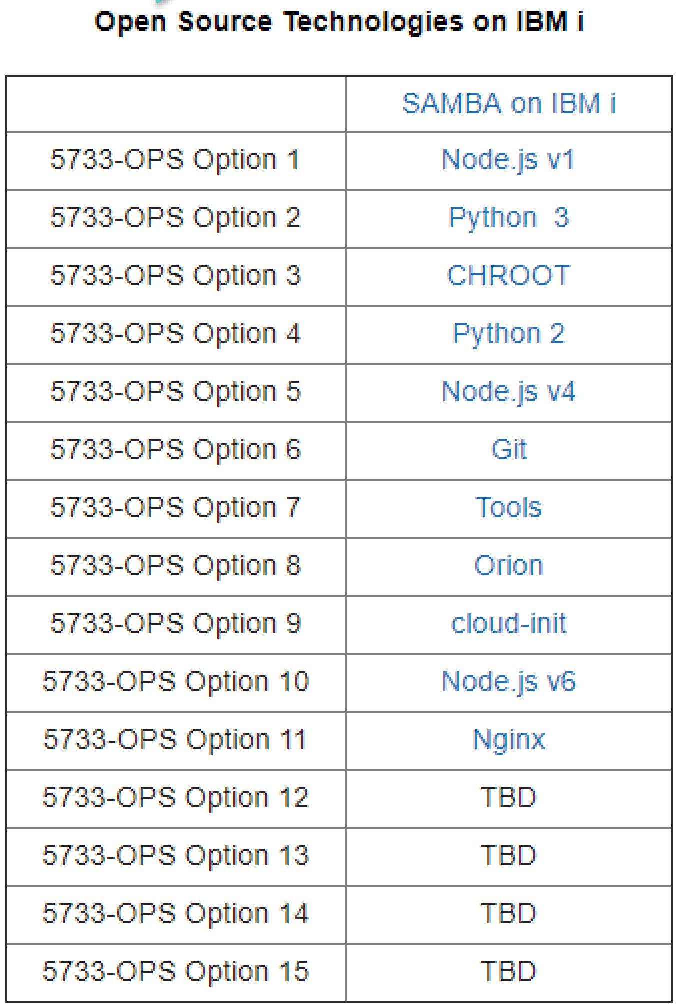
Python is a popular high-level programming language. It is easily extensible through the use of third-party packages and often allows powerful function to be written with few lines of code. Python caters to multiple programming styles (object oriented, procedural, etc) and the code tends to be readable and maintainable.

Python is now being delivered and packaged for IBM i. It is available through the following options:

- Option 2 - Python 3.4
- Option 4 - Python 2.7

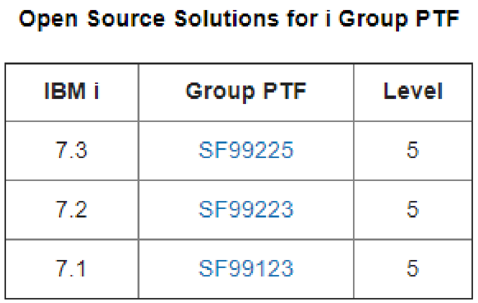
The following add-ons are also available via separate PTFs

Package	Description
ibm_db	DB2 for i connector - Allows native access to DB2 for i.
itoolkit	Toolkit for IBM i - allows access to system resources through program calls, SQL queries, CL commands, shell commands, and more.
flipflop	FastCGI gateway
bottle	Lightweight web framework.



Open Source Technologies on IBM i

	SAMBA on IBM i
5733-OPS Option 1	Node.js v1
5733-OPS Option 2	Python 3
5733-OPS Option 3	CHROOT
5733-OPS Option 4	Python 2
5733-OPS Option 5	Node.js v4
5733-OPS Option 6	Git
5733-OPS Option 7	Tools
5733-OPS Option 8	Orion
5733-OPS Option 9	cloud-init
5733-OPS Option 10	Node.js v6
5733-OPS Option 11	Nginx
5733-OPS Option 12	TBD
5733-OPS Option 13	TBD
5733-OPS Option 14	TBD
5733-OPS Option 15	TBD

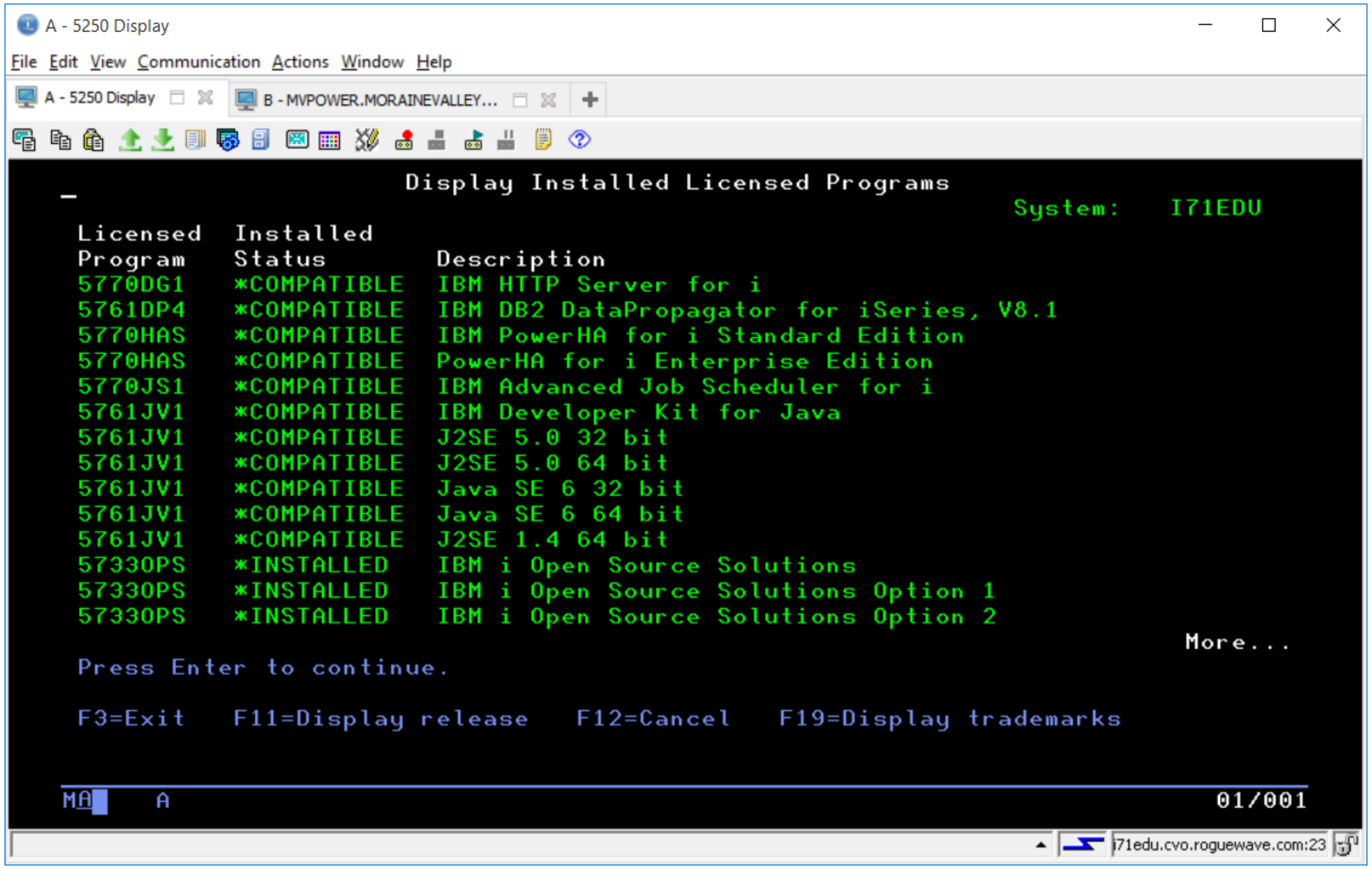


Open Source Solutions for i Group PTF

IBM i	Group PTF	Level
7.3	SF99225	5
7.2	SF99223	5
7.1	SF99123	5

Need licensed program

- 5733OPS Base and option 2 or 4

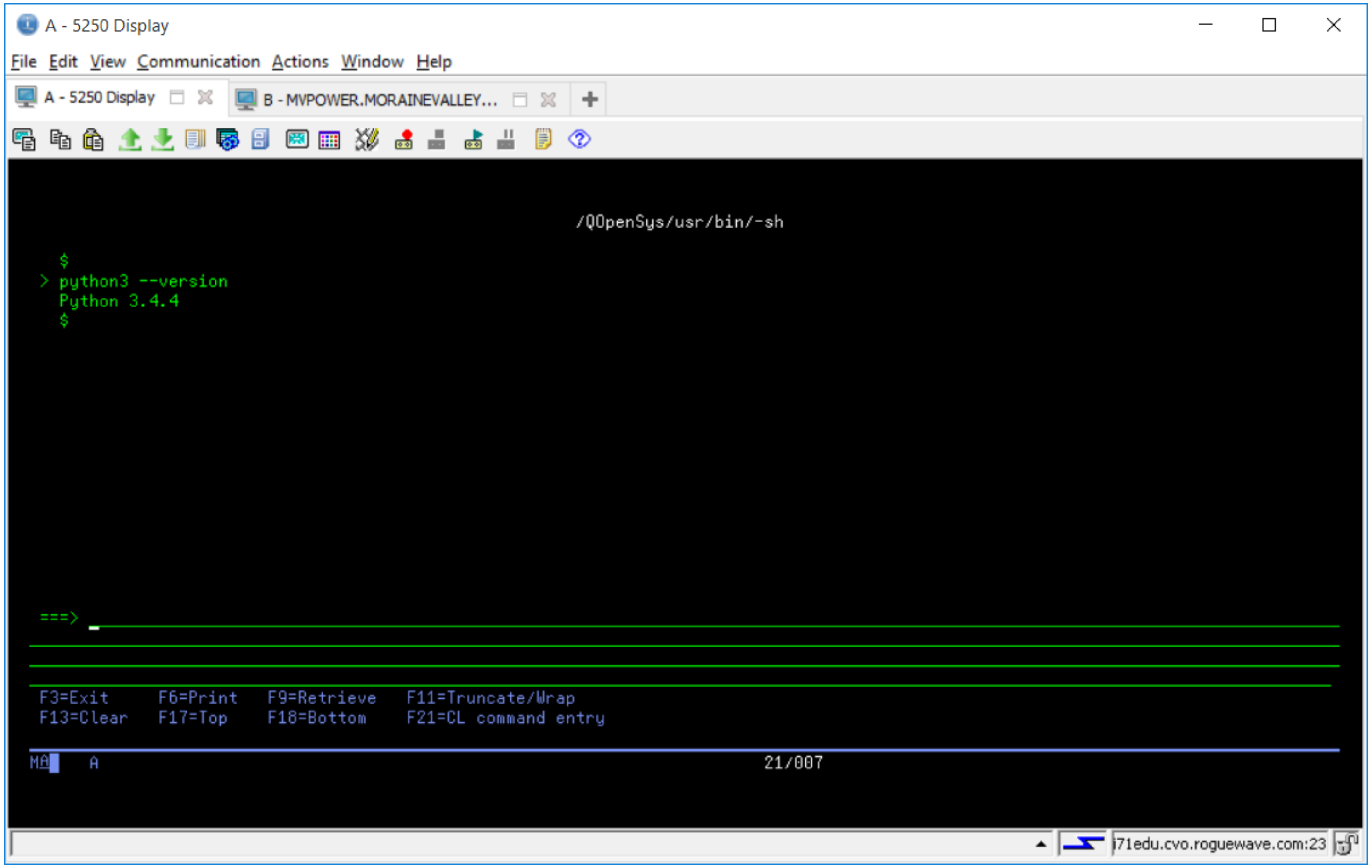

A - 5250 Display
File Edit View Communication Actions Window Help
A - 5250 Display
B - MVPOWER.MORAINEVALLEY...

```
                          Display Installed Licensed Programs
                                                                System:   I71EDU
  Licensed  Installed
  Program   Status       Description
  5770DG1   *COMPATIBLE  IBM HTTP Server for i
  5761DP4   *COMPATIBLE  IBM DB2 DataPropagator for iSeries, V8.1
  5770HAS   *COMPATIBLE  IBM PowerHA for i Standard Edition
  5770HAS   *COMPATIBLE  PowerHA for i Enterprise Edition
  5770JS1   *COMPATIBLE  IBM Advanced Job Scheduler for i
  5761JV1   *COMPATIBLE  IBM Developer Kit for Java
  5761JV1   *COMPATIBLE  J2SE 5.0 32 bit
  5761JV1   *COMPATIBLE  J2SE 5.0 64 bit
  5761JV1   *COMPATIBLE  Java SE 6 32 bit
  5761JV1   *COMPATIBLE  Java SE 6 64 bit
  5761JV1   *COMPATIBLE  J2SE 1.4 64 bit
  5733OPS   *INSTALLED   IBM i Open Source Solutions
  5733OPS   *INSTALLED   IBM i Open Source Solutions Option 1
  5733OPS   *INSTALLED   IBM i Open Source Solutions Option 2
                                                                       More...
  Press Enter to continue.

  F3=Exit   F11=Display release   F12=Cancel   F19=Display trademarks

MA     A                                                                01/001
```

i71edu.cvo.roguewave.com:23

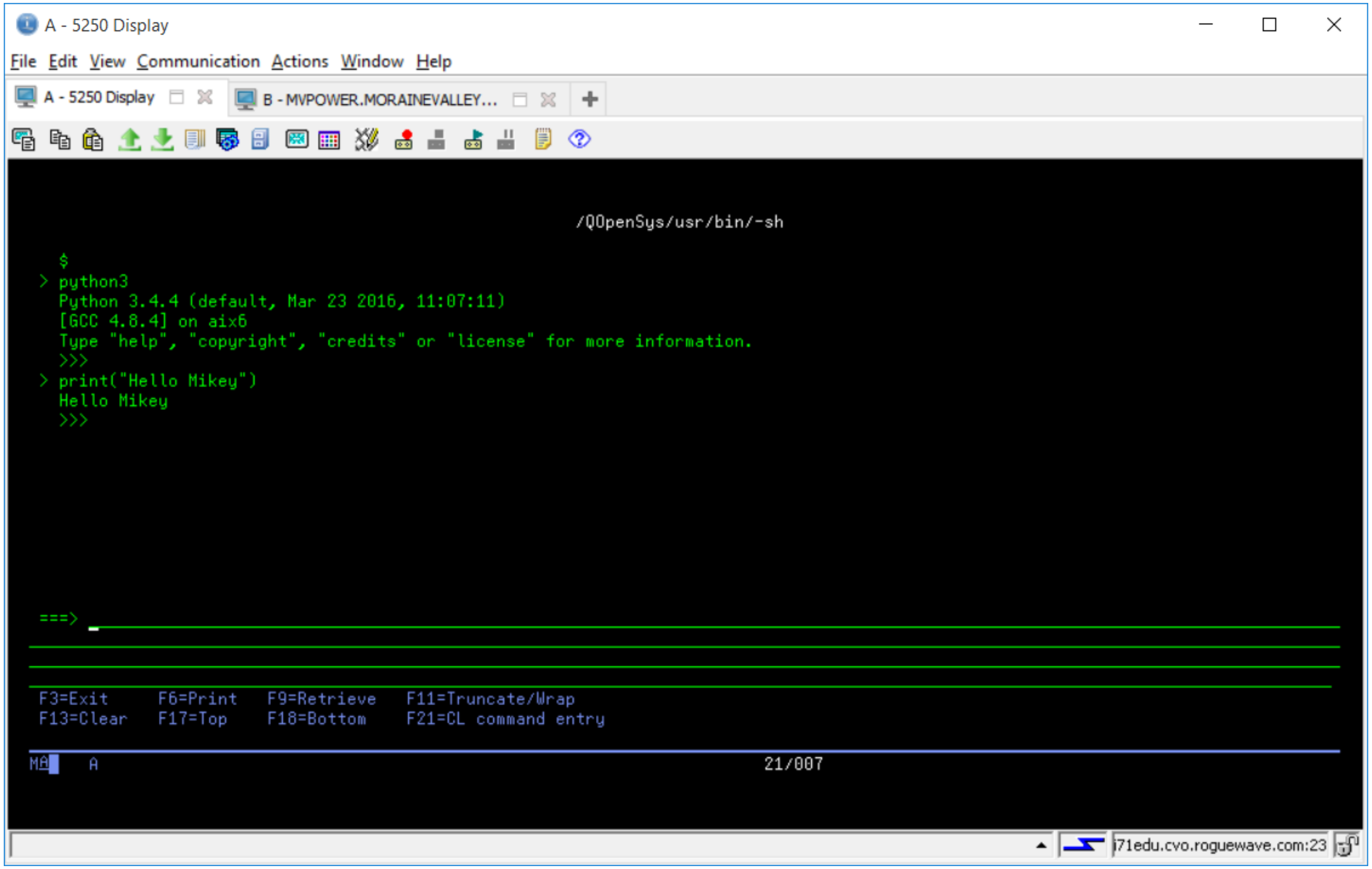
Python in action

- Command line via green screen


A - 5250 Display
File Edit View Communication Actions Window Help
A - 5250 Display
B - MVPOWER.MORAINEVALLEY...
i71edu.cvo.roguewave.com:23

```
                              /QOpenSys/usr/bin/-sh

   $
 > python3 --version
   Python 3.4.4
   $

===> _

F3=Exit     F6=Print    F9=Retrieve    F11=Truncate/Wrap
F13=Clear   F17=Top     F18=Bottom     F21=CL command entry

MA    A                                       21/007
```

Hello World

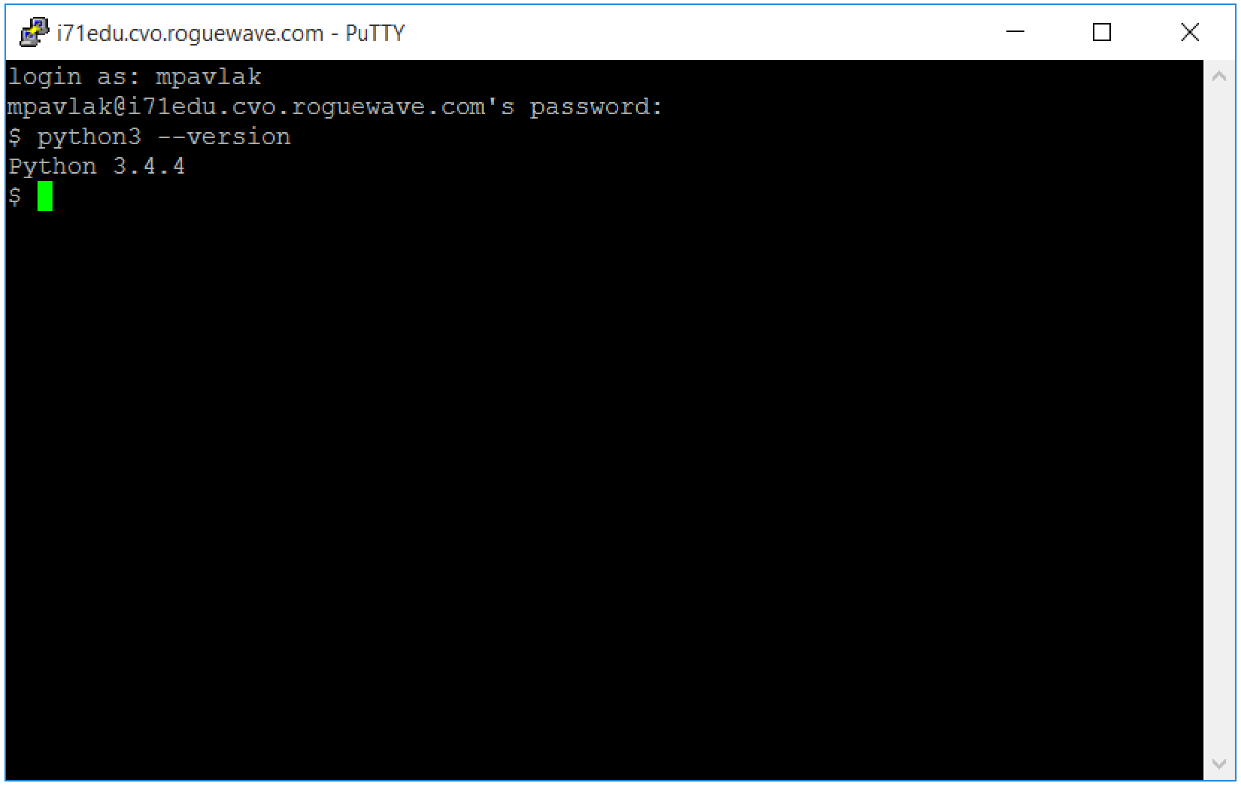

A - 5250 Display
File Edit View Communication Actions Window Help
A - 5250 Display
B - MVPOWER.MORAINEVALLEY...
i71edu.cvo.roguewave.com:23

```
                                 /QOpenSys/usr/bin/-sh

   $
 > python3
   Python 3.4.4 (default, Mar 23 2016, 11:07:11)
   [GCC 4.8.4] on aix6
   Type "help", "copyright", "credits" or "license" for more information.
   >>>
 > print("Hello Mikey")
   Hello Mikey
   >>>




 ===> _


 F3=Exit      F6=Print     F9=Retrieve     F11=Truncate/Wrap
 F13=Clear    F17=Top      F18=Bottom      F21=CL command entry

MA     A                                          21/007
```

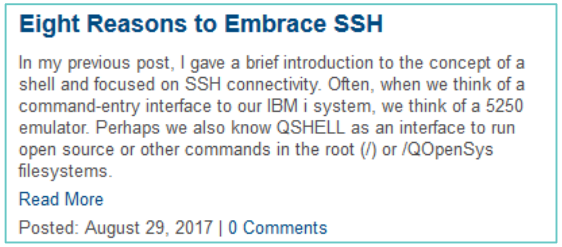
Most prefer SSH

- Command line via SSH terminal
  - Recommended strongly by Jesse!

http://ibmsystemsmag.com/blogs/open-your-i/



```
login as: mpavlak
mpavlak@i71edu.cvo.roguewave.com's password:
$ python3 --version
Python 3.4.4
$
```



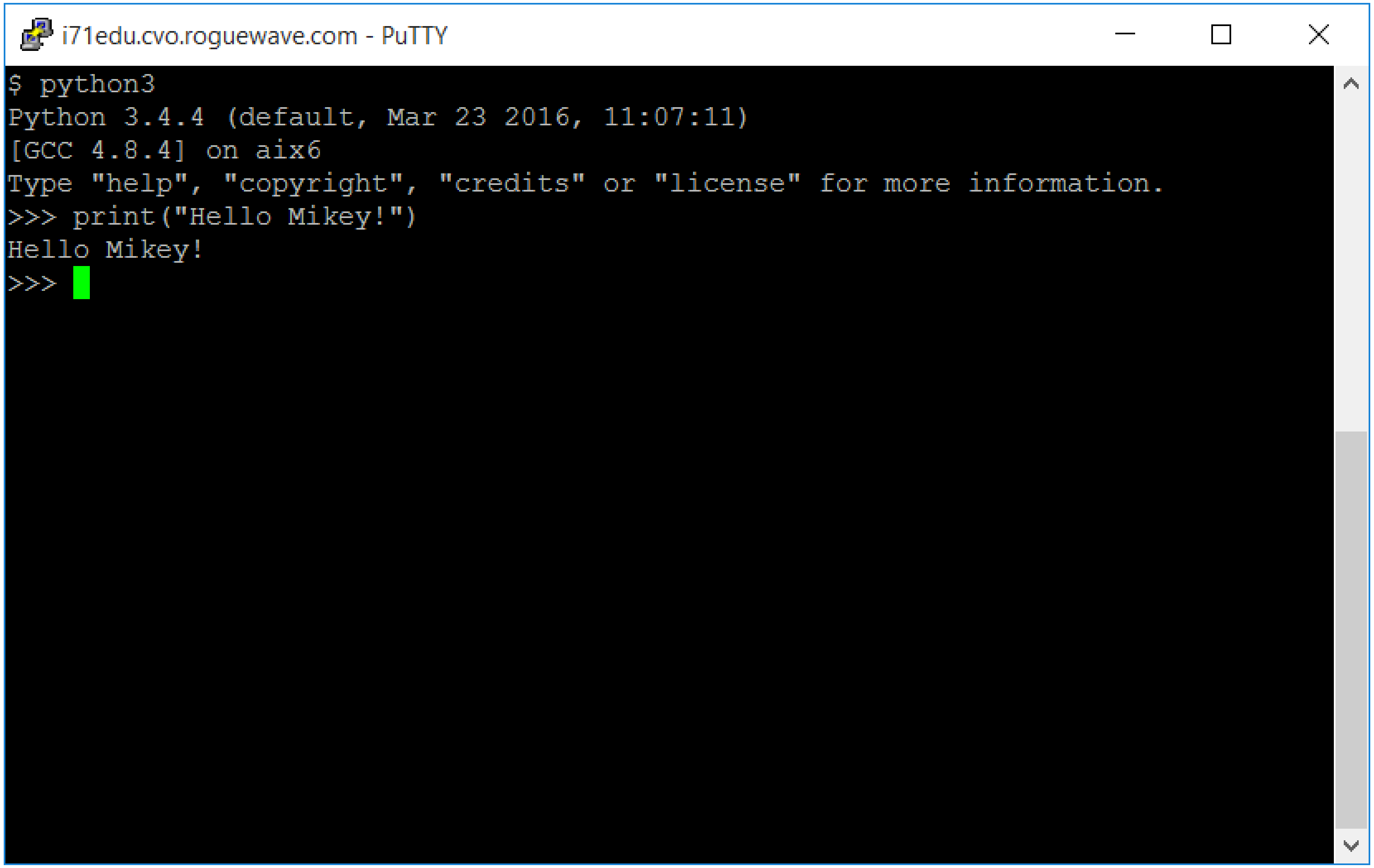
Eight Reasons to Embrace SSH

In my previous post, I gave a brief introduction to the concept of a shell and focused on SSH connectivity. Often, when we think of a command-entry interface to our IBM i system, we think of a 5250 emulator. Perhaps we also know QSHELL as an interface to run open source or other commands in the root (/) or /QOpenSys filesystems.

Read More

Posted: August 29, 2017 | 0 Comments

Hello World, again…



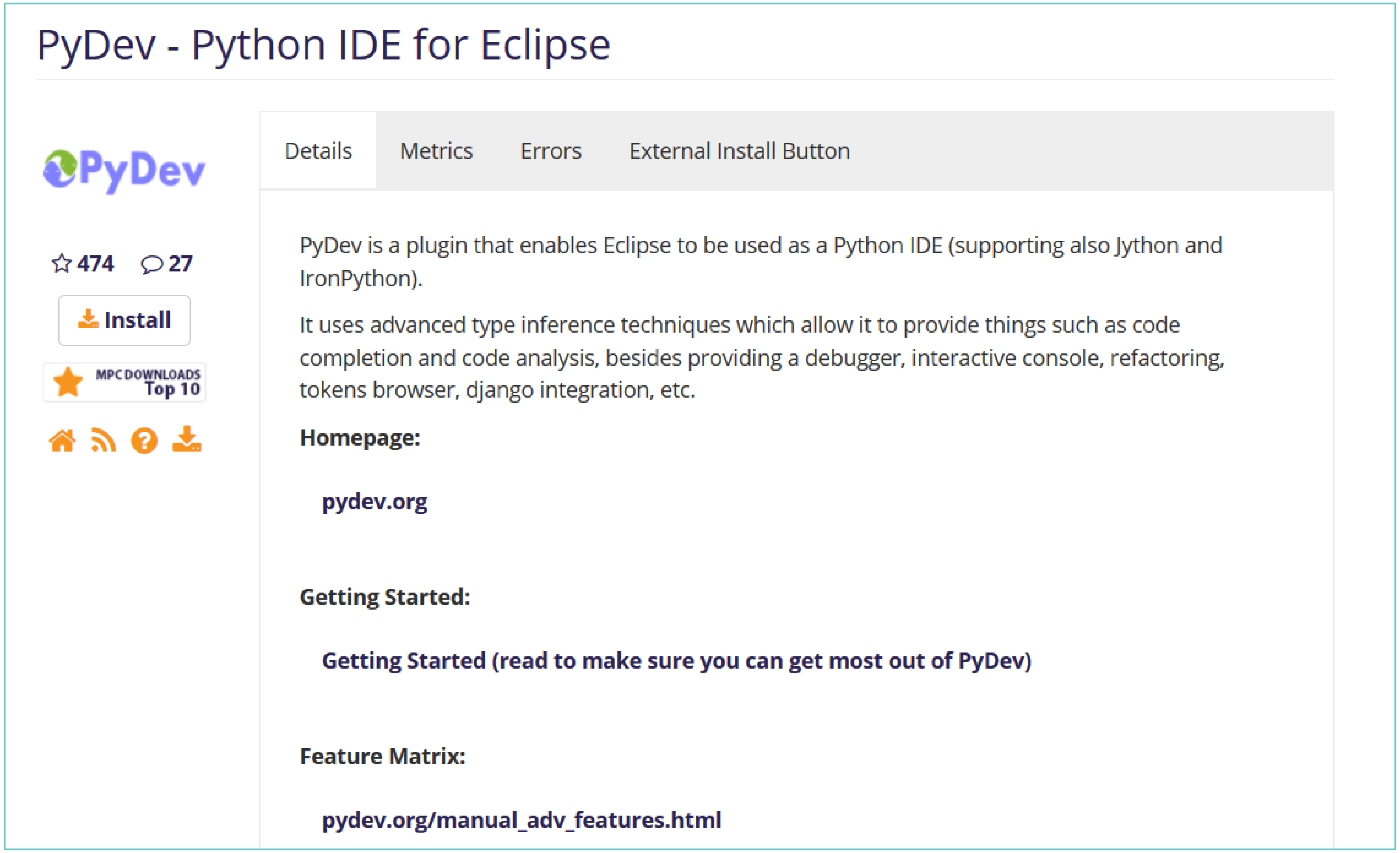
```
$ python3
Python 3.4.4 (default, Mar 23 2016, 11:07:11)
[GCC 4.8.4] on aix6
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello Mikey!")
Hello Mikey!
>>>
```

IDE

Zend Studio

- No, you don't need to buy Zend Studio
- Use Orion, etc.
- But if you have Studio or RDi…
  - Consider something from Eclipse.org
  - I grabbed PyDev

Eclipse



PyDev - Python IDE for Eclipse

Details | Metrics | Errors | External Install Button

PyDev is a plugin that enables Eclipse to be used as a Python IDE (supporting also Jython and IronPython).

It uses advanced type inference techniques which allow it to provide things such as code completion and code analysis, besides providing a debugger, interactive console, refactoring, tokens browser, django integration, etc.

Homepage:

pydev.org

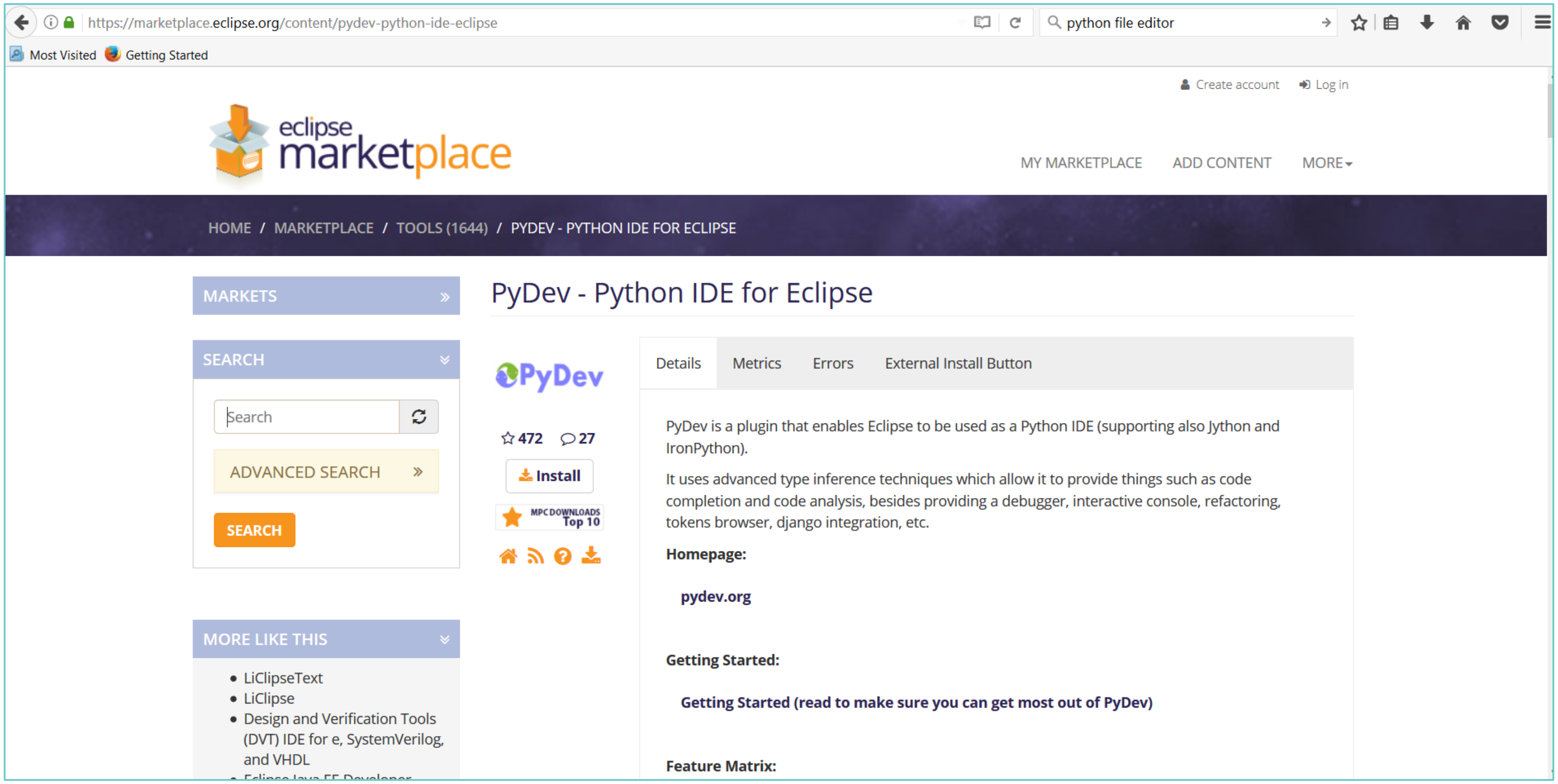
Getting Started:

Getting Started (read to make sure you can get most out of PyDev)

Feature Matrix:

pydev.org/manual_adv_features.html

Download PyDev from Eclipse


https://marketplace.eclipse.org/content/pydev-python-ide-eclipse

python file editor

Most Visited   Getting Started

Create account   Log in

eclipse marketplace

MY MARKETPLACE   ADD CONTENT   MORE

HOME / MARKETPLACE / TOOLS (1644) / PYDEV - PYTHON IDE FOR ECLIPSE

MARKETS

SEARCH

Search

ADVANCED SEARCH

SEARCH

MORE LIKE THIS

- LiClipseText
- LiClipse
- Design and Verification Tools (DVT) IDE for e, SystemVerilog, and VHDL

PyDev - Python IDE for Eclipse

PyDev

472   27

Install

MPC DOWNLOADS Top 10

Details   Metrics   Errors   External Install Button

PyDev is a plugin that enables Eclipse to be used as a Python IDE (supporting also Jython and IronPython).

It uses advanced type inference techniques which allow it to provide things such as code completion and code analysis, besides providing a debugger, interactive console, refactoring, tokens browser, django integration, etc.

Homepage:

pydev.org

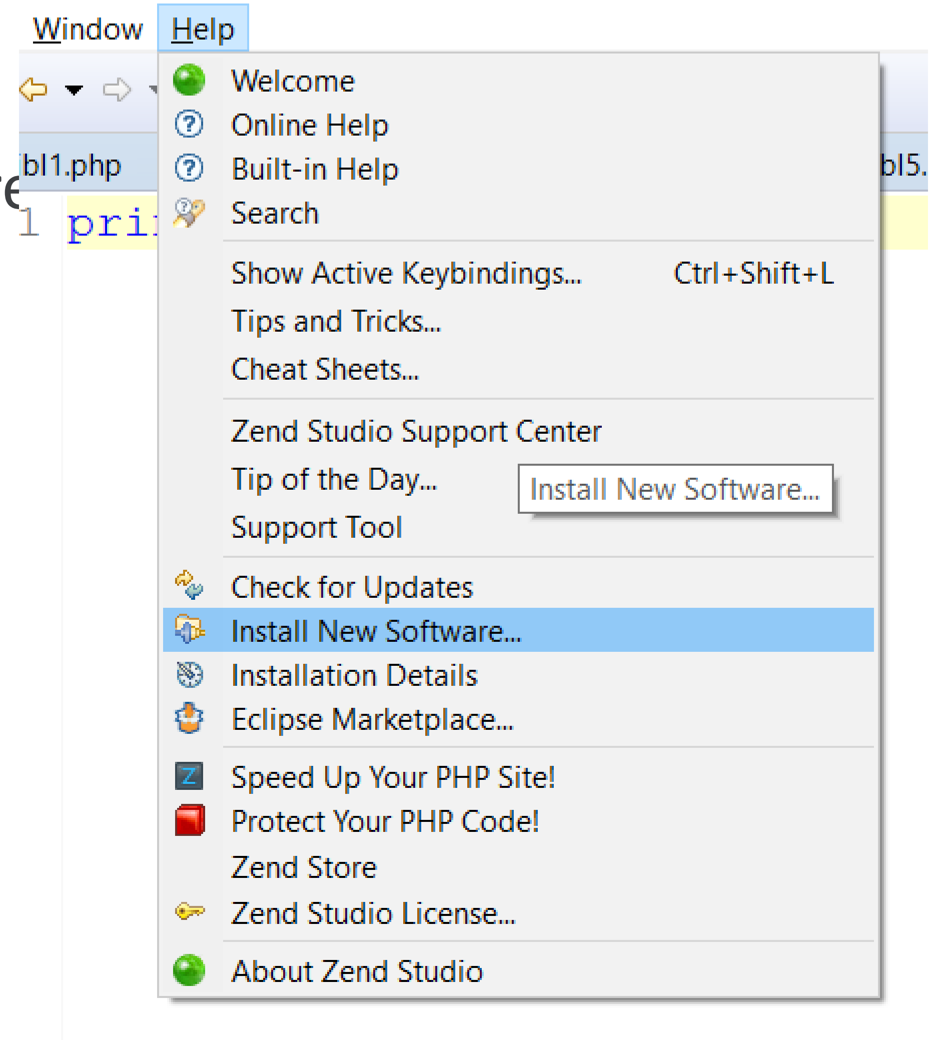
Getting Started:

Getting Started (read to make sure you can get most out of PyDev)

Feature Matrix:

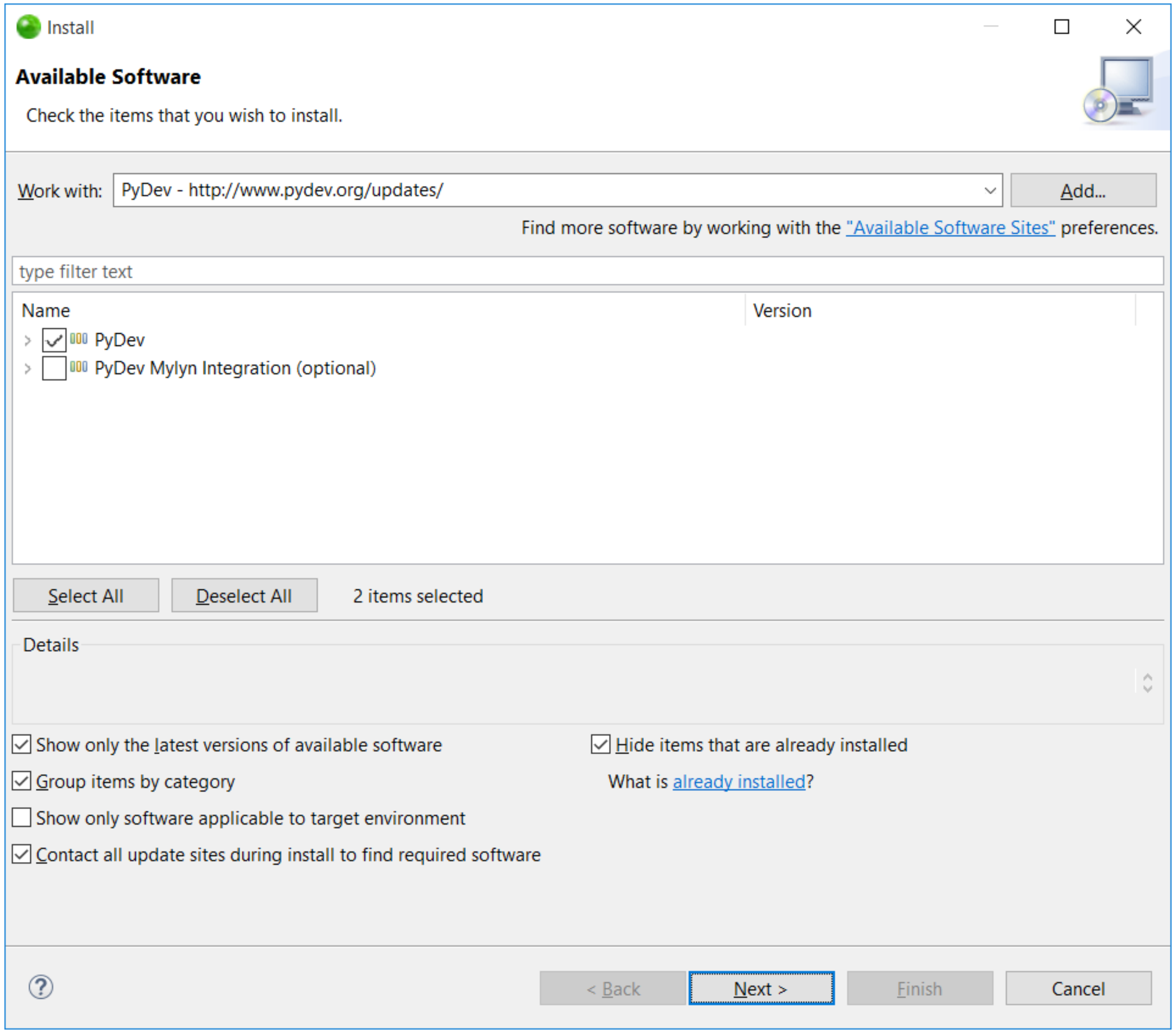
Capture URL

- Help➔
  - Install New Software
  - Follow prompts



Editor for Eclipse

- Select what you like
  - Click next



Install

Available Software

Check the items that you wish to install.

Work with: PyDev - http://www.pydev.org/updates/ [Add...]

Find more software by working with the "Available Software Sites" preferences.

type filter text

Name	Version
✓ PyDev	
☐ PyDev Mylyn Integration (optional)	

[Select All] [Deselect All] 2 items selected

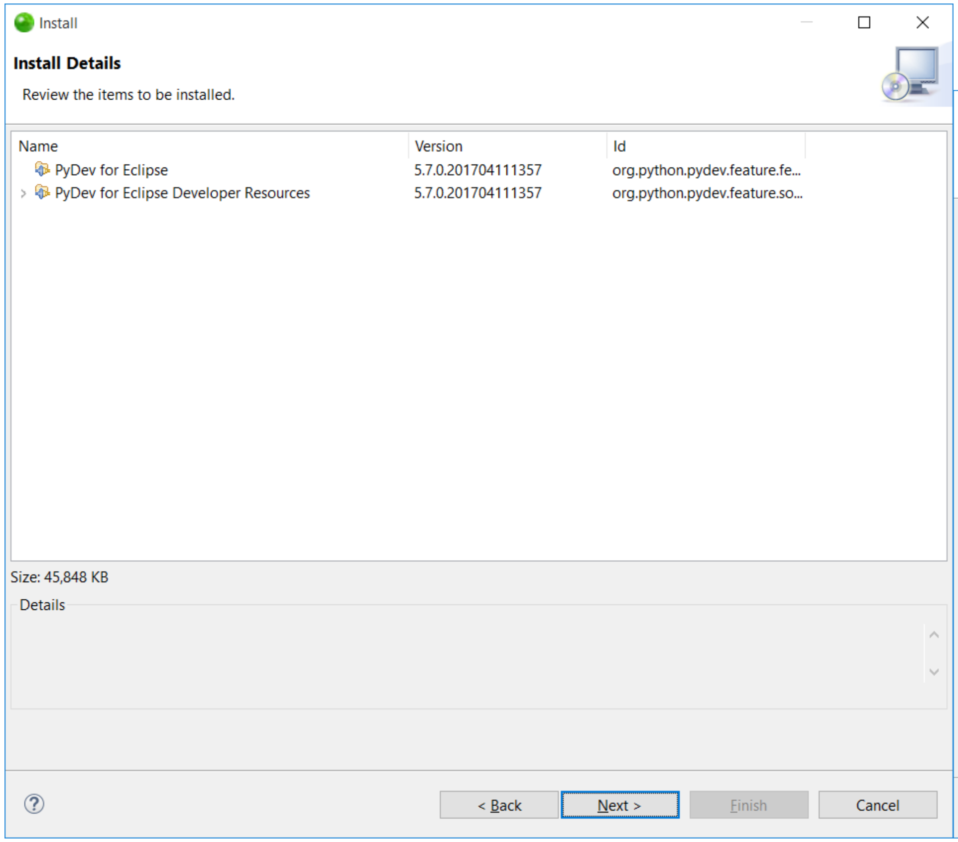
Details

- ✓ Show only the latest versions of available software
- ✓ Hide items that are already installed
- ✓ Group items by category
- What is already installed?
- ☐ Show only software applicable to target environment
- ✓ Contact all update sites during install to find required software

[< Back] [Next >] [Finish] [Cancel]

Confirm versions

- Click next again
  - Then accept EULA


Install

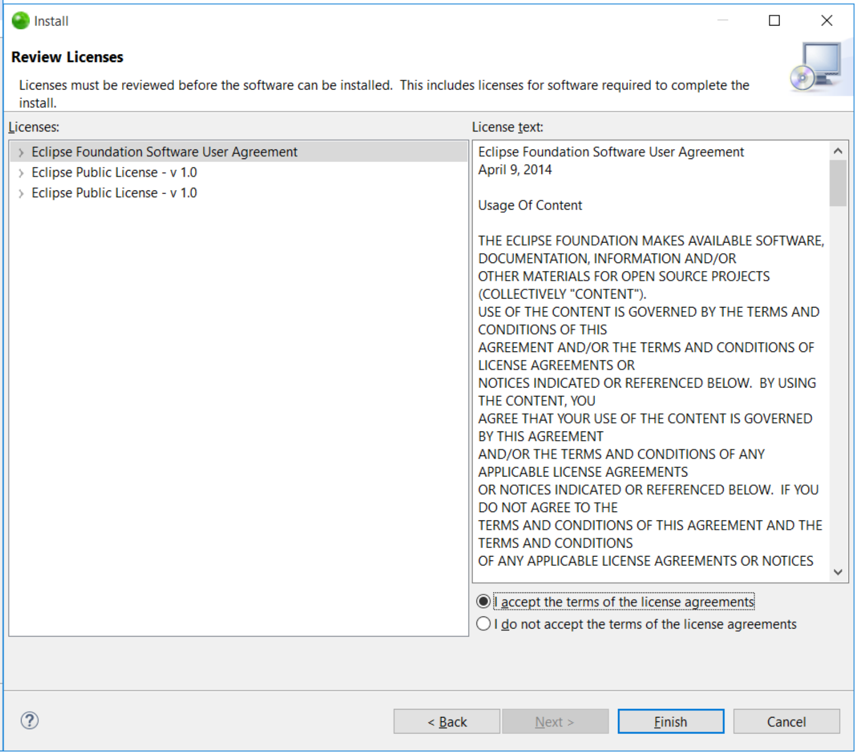
Install Details

Review the items to be installed.

Name	Version	Id
PyDev for Eclipse	5.7.0.201704111357	org.python.pydev.feature.fe...
PyDev for Eclipse Developer Resources	5.7.0.201704111357	org.python.pydev.feature.so...

Size: 45,848 KB

Details


Install

Review Licenses

Licenses must be reviewed before the software can be installed. This includes licenses for software required to complete the install.

Licenses:
- Eclipse Foundation Software User Agreement
- Eclipse Public License - v 1.0
- Eclipse Public License - v 1.0

License text:

Eclipse Foundation Software User Agreement
April 9, 2014

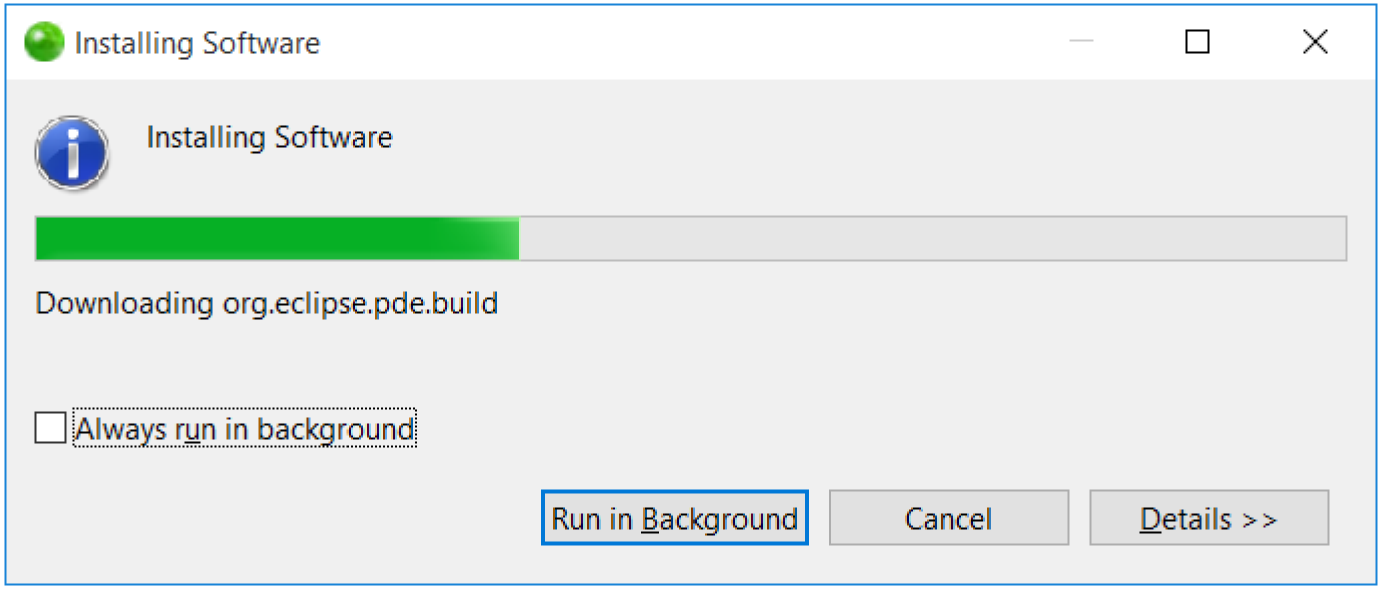
Usage Of Content

THE ECLIPSE FOUNDATION MAKES AVAILABLE SOFTWARE, DOCUMENTATION, INFORMATION AND/OR OTHER MATERIALS FOR OPEN SOURCE PROJECTS (COLLECTIVELY "CONTENT"). USE OF THE CONTENT IS GOVERNED BY THE TERMS AND CONDITIONS OF THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. BY USING THE CONTENT, YOU AGREE THAT YOUR USE OF THE CONTENT IS GOVERNED BY THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT AND THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES

[✓ I accept the terms of the license agreements]
I do not accept the terms of the license agreements

FRESCHE

Watch the pretty status bar


Installing Software

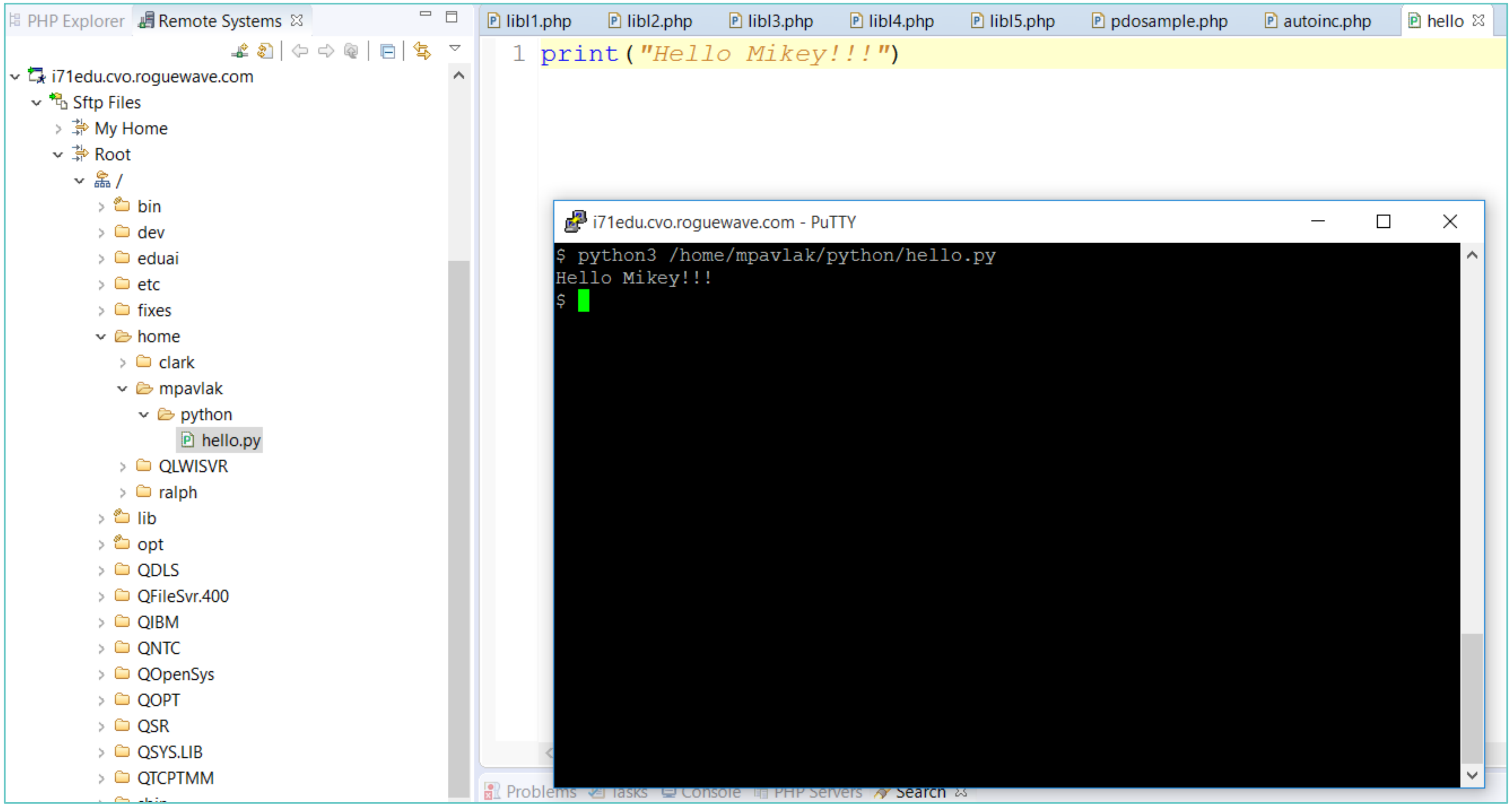
Installing Software

Downloading org.eclipse.pde.build

skwib.com

Python in Eclipse (i.e. Zend Studio)

- I bet RDi works, too!

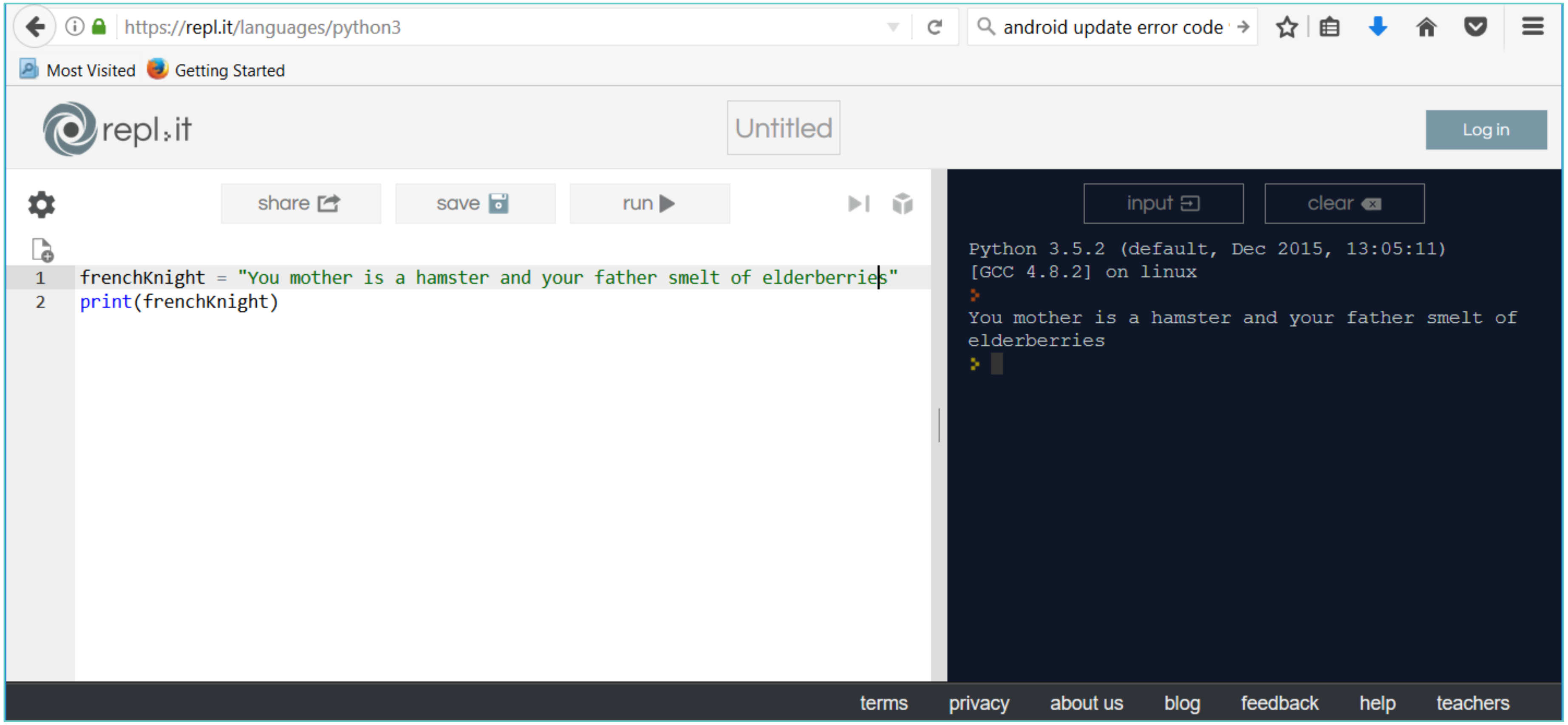


```
print("Hello Mikey!!!")
```

```
$ python3 /home/mpavlak/python/hello.py
Hello Mikey!!!
$
```

Alternatives to IBM i when learning

- What's that? The boss won't let you install Python on the IBM i?

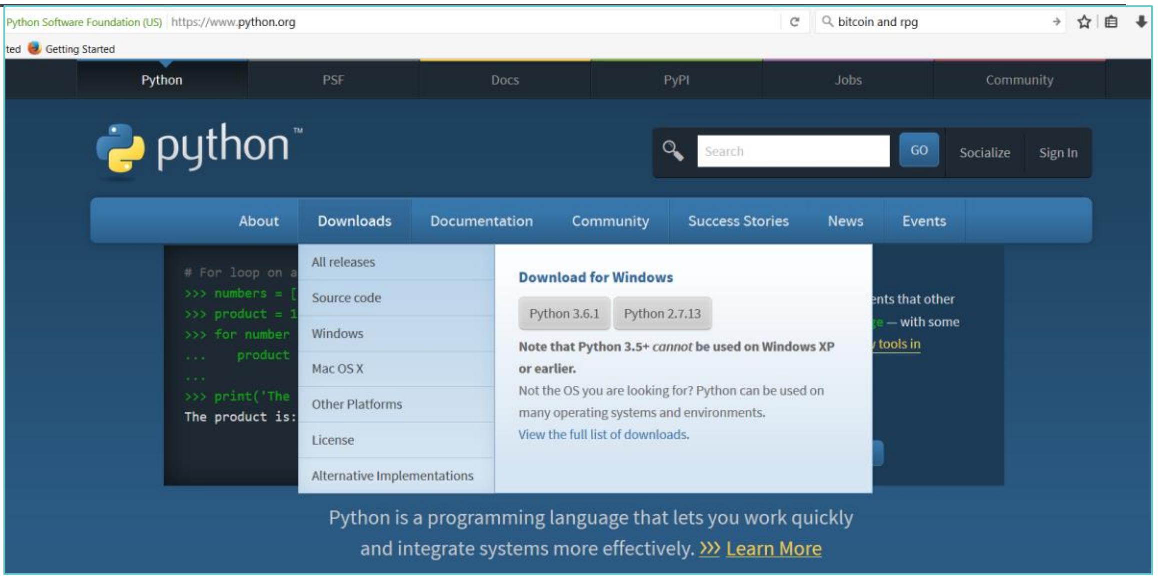


```
frenchKnight = "You mother is a hamster and your father smelt of elderberries"
print(frenchKnight)
```

```
Python 3.5.2 (default, Dec 2015, 13:05:11)
[GCC 4.8.2] on linux
›
You mother is a hamster and your father smelt of
elderberries
›
```

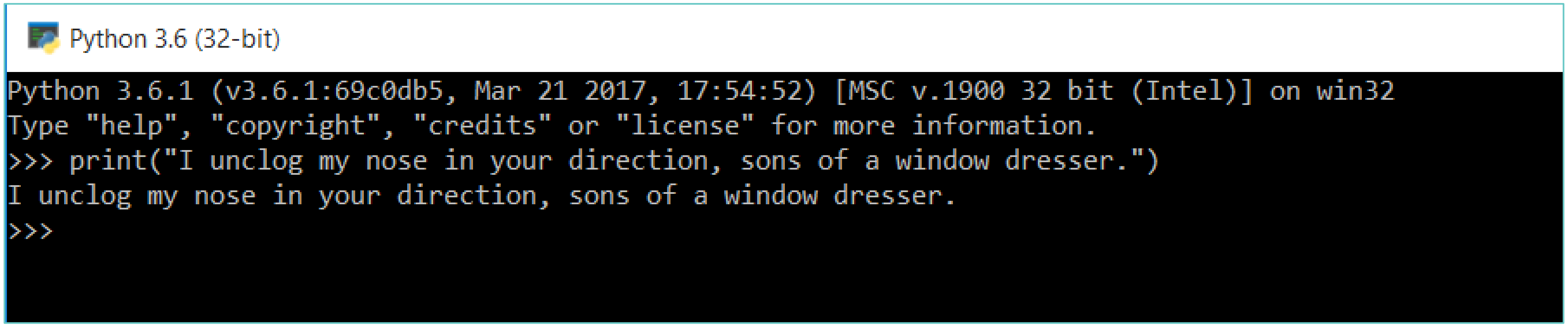
Desktop education at it's finest

- How about your PC?
- Head to Python.org site:
  - Download
  - Install
  - Viola!



Note that Python 3.5+ *cannot* be used on Windows XP or earlier.
Not the OS you are looking for? Python can be used on many operating systems and environments.
View the full list of downloads.

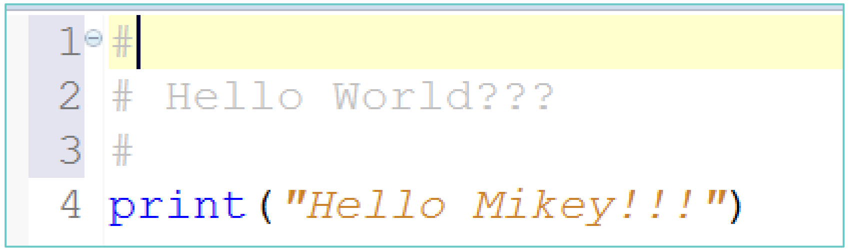
Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More


Python 3.6 (32-bit)

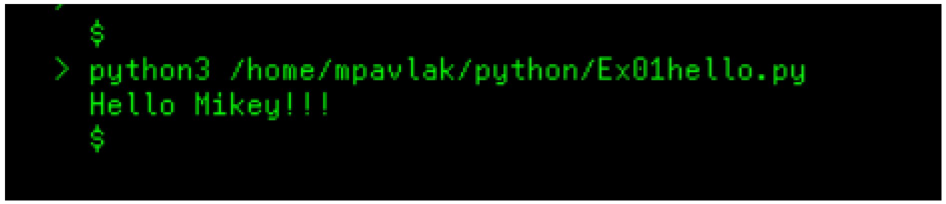
```
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("I unclog my nose in your direction, sons of a window dresser.")
I unclog my nose in your direction, sons of a window dresser.
>>>
```

Python Script in IFS

- Create a file like Ex01hello.py
- Open the file
- Key up some code and click save
- Rinse, repeat…



```
#
# Hello World???
#
print("Hello Mikey!!!")
```



```
$
> python3 /home/mpavlak/python/Ex01hello.py
Hello Mikey!!!
$
```

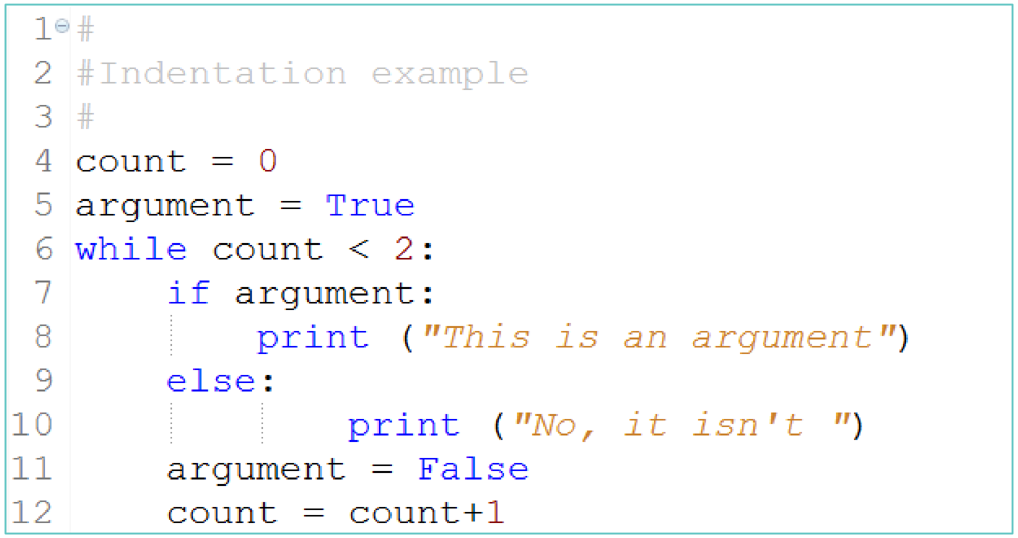
Syntax !== sin-tax

(eh, Cook county?)

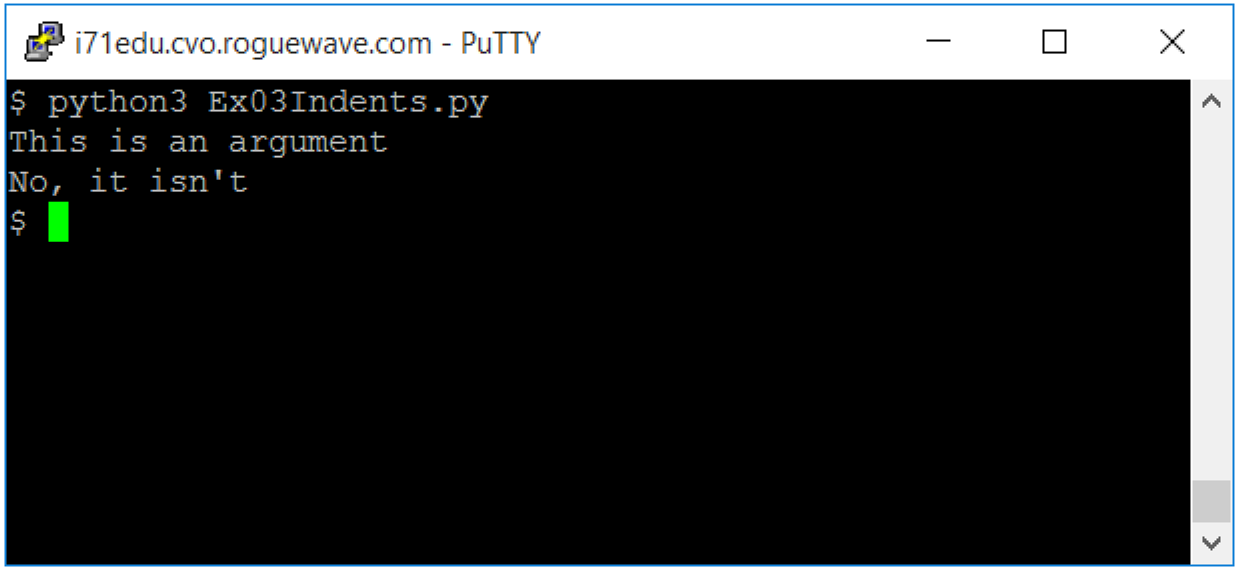
How it is written

- Indentation means EVERYTHING
    - Don’t use tab
    - 4 spaces – No more, no less
    - Mismatched indents can cause failures. Good luck finding…
- No scope terminators like other languages
- Colon introduces start block, then indent
- Much more readable than other languages
- Get a good editor!!!

Indentation



```
#
#Indentation example
#
count = 0
argument = True
while count < 2:
    if argument:
        print ("This is an argument")
    else:
            print ("No, it isn't ")
    argument = False
    count = count+1
```



```
$ python3 Ex03Indents.py
This is an argument
No, it isn't
$ 
```

Operators – Similar to other C derivatives

- Comparison
  - Assignment =
  - Comparison ==
  - Inequality !=
  - Less than <
  - Greater than >
  - Less than or equal to <=
  - Greater than or equal to >=

- Mathematical
  - Addition +
  - Multiplication *
  - Division /
  - Floor division //
  - Modulus %
  - Exponentiation **
- Booleans
  - And
  - Or
  - Not

Syntax

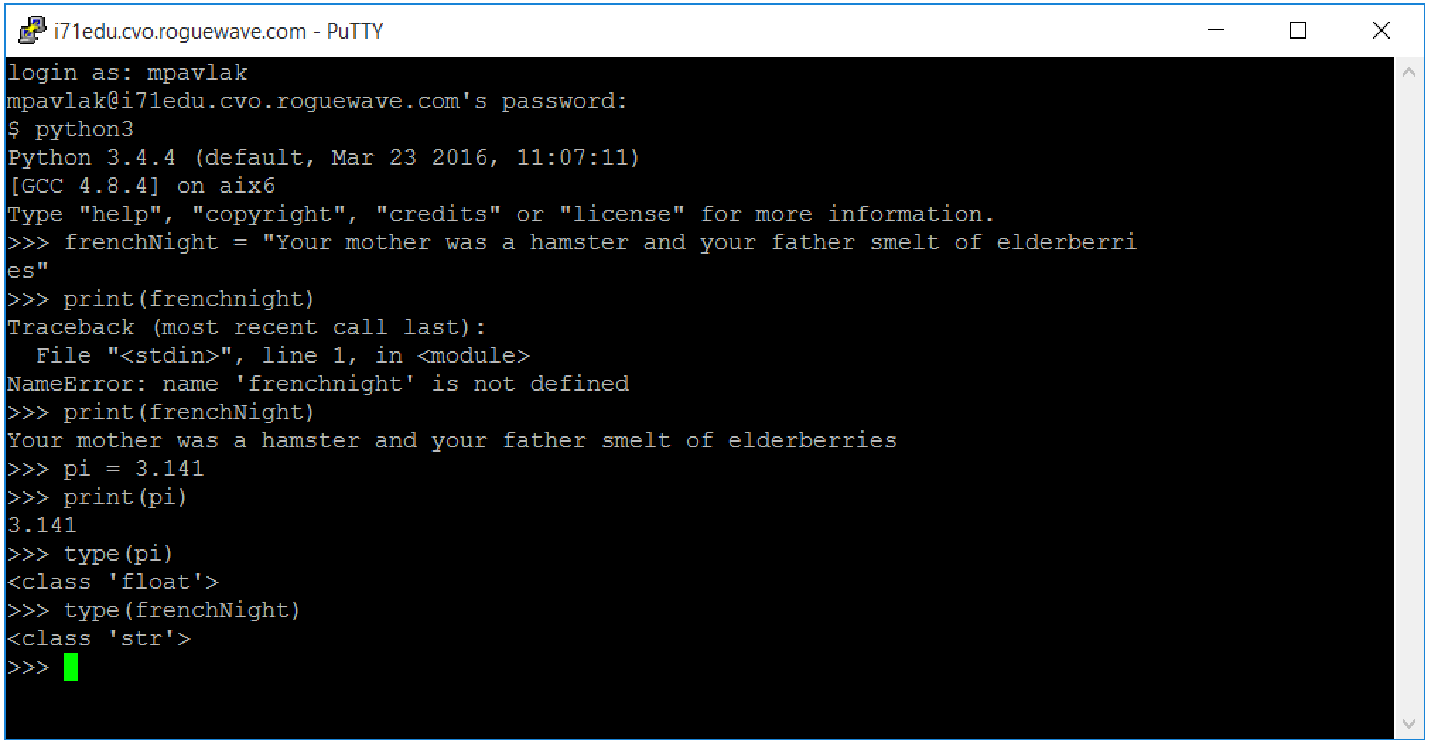
Variables

Data types – yeah…about that…

- Int
  - Integer of unlimited size
- Float
  - System defined precision
- Complex
  - Complex with real and imaginary parts
- Bool
  - TRUE & FALSE

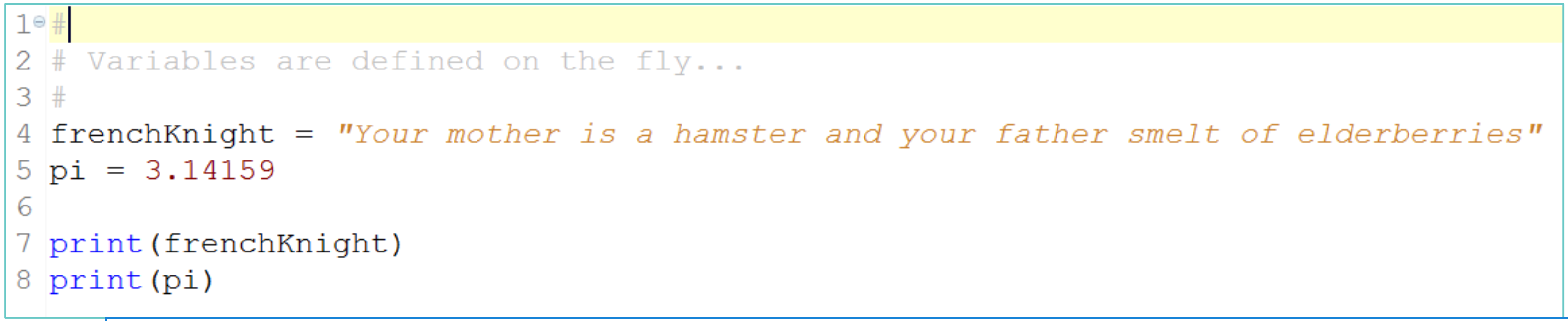
Variables on the fly

- Case sensitive
- camelCase
- Who are you? type()

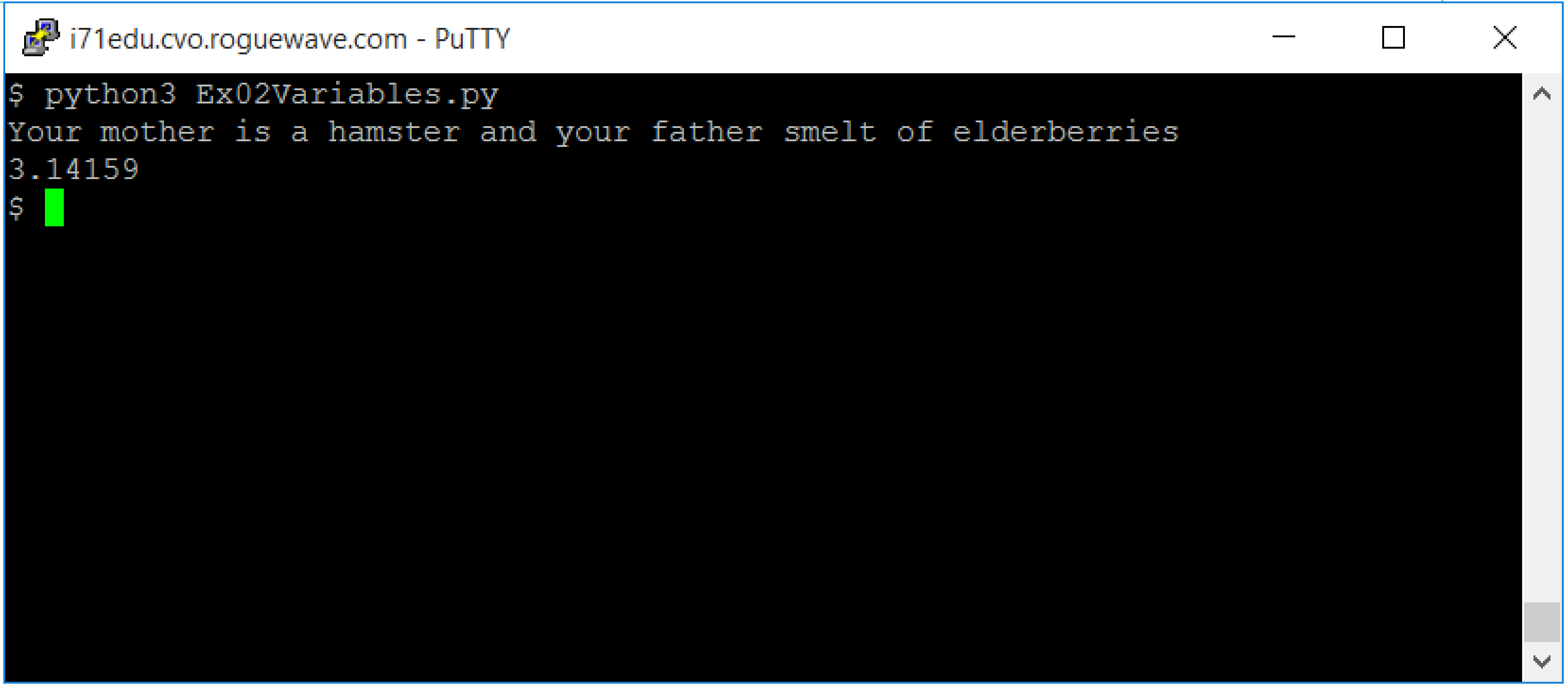


```
login as: mpavlak
mpavlak@i71edu.cvo.roguewave.com's password:
$ python3
Python 3.4.4 (default, Mar 23 2016, 11:07:11)
[GCC 4.8.4] on aix6
Type "help", "copyright", "credits" or "license" for more information.
>>> frenchNight = "Your mother was a hamster and your father smelt of elderberri
es"
>>> print(frenchnight)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'frenchnight' is not defined
>>> print(frenchNight)
Your mother was a hamster and your father smelt of elderberries
>>> pi = 3.141
>>> print(pi)
3.141
>>> type(pi)
<class 'float'>
>>> type(frenchNight)
<class 'str'>
>>>
```

Variables in a file

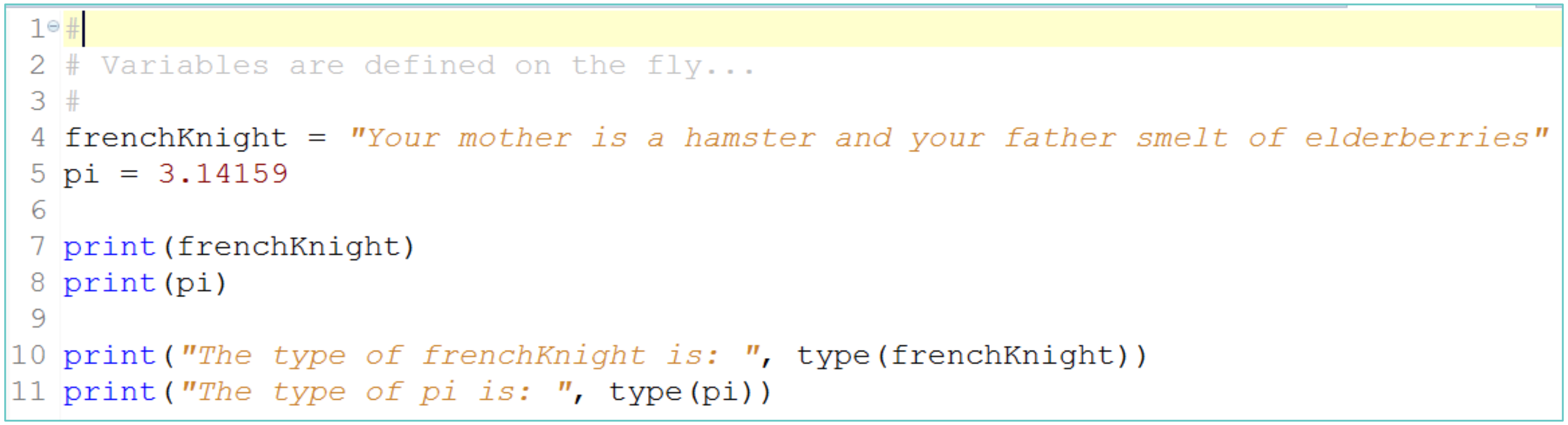


```python
#
# Variables are defined on the fly...
#
frenchKnight = "Your mother is a hamster and your father smelt of elderberries"
pi = 3.14159

print(frenchKnight)
print(pi)
```

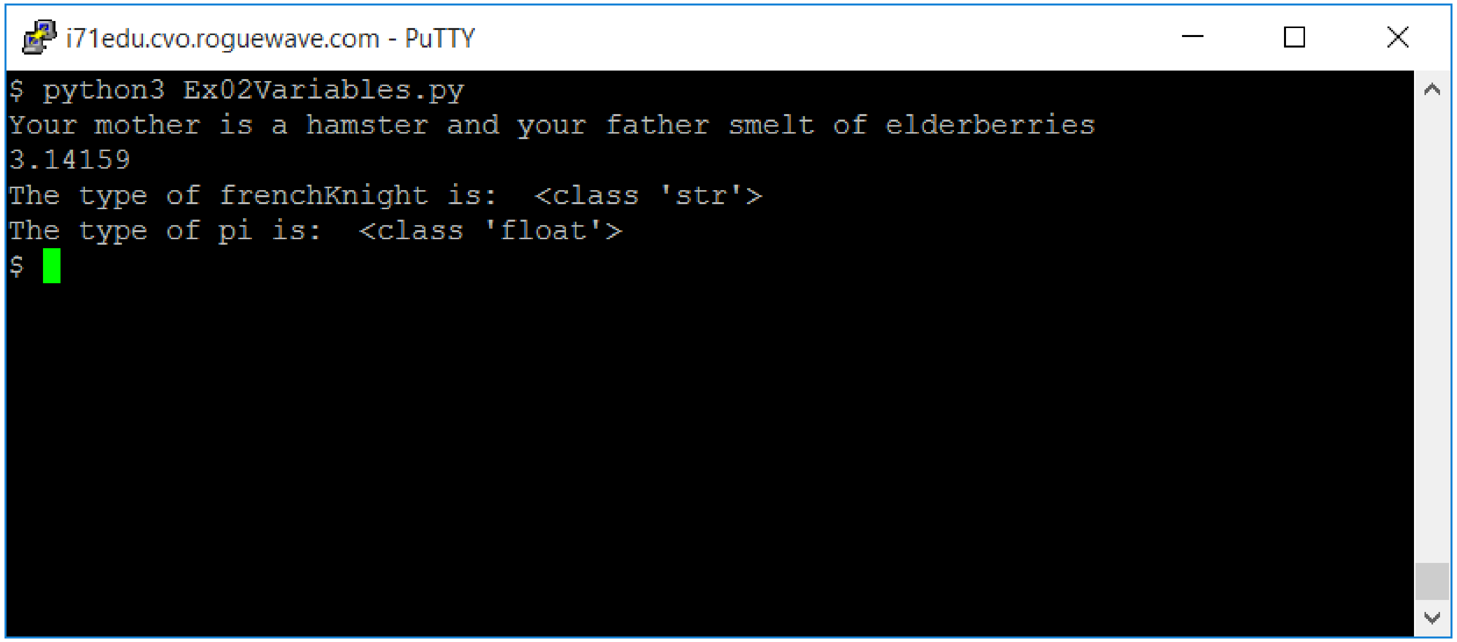


```
$ python3 Ex02Variables.py
Your mother is a hamster and your father smelt of elderberries
3.14159
$
```

Data type?



```
#
# Variables are defined on the fly...
#
frenchKnight = "Your mother is a hamster and your father smelt of elderberries"
pi = 3.14159

print(frenchKnight)
print(pi)

print("The type of frenchKnight is: ", type(frenchKnight))
print("The type of pi is: ", type(pi))
```

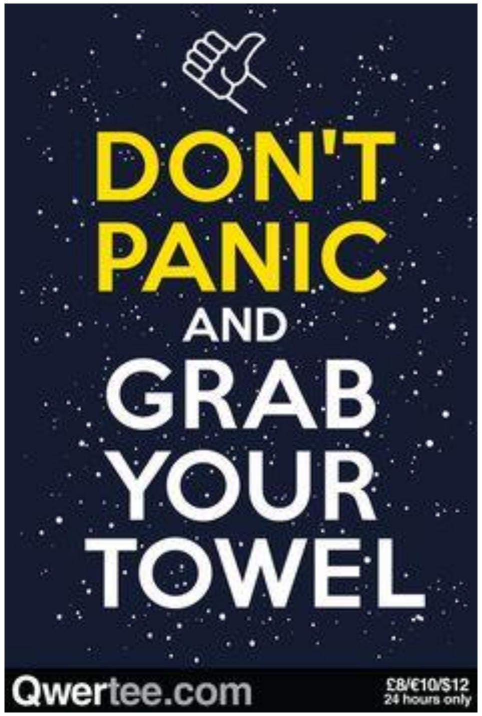


```
$ python3 Ex02Variables.py
Your mother is a hamster and your father smelt of elderberries
3.14159
The type of frenchKnight is:  <class 'str'>
The type of pi is:  <class 'float'>
$
```

Every variable is implemented as a class

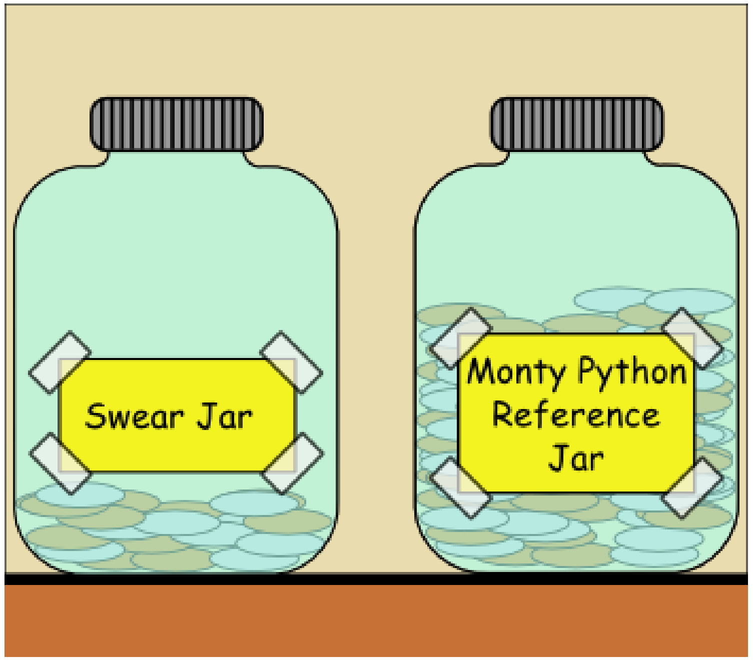
FRESCHE

And now for something completely different



It’s OK...

- Monty Python references are not only acceptable...
  - They are encouraged!
- Documentation is littered with references
- Examples are well covered

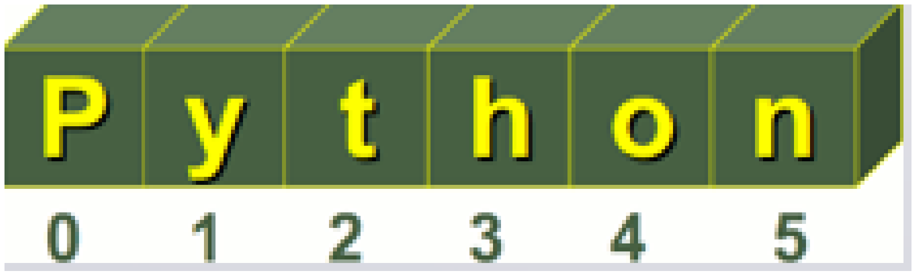


Back to variables

- Numbers – 3 Data types
  - Integer 1,2,42
  - Float 3.14159
  - Complex: <real> + <imaginary> (not used much…)

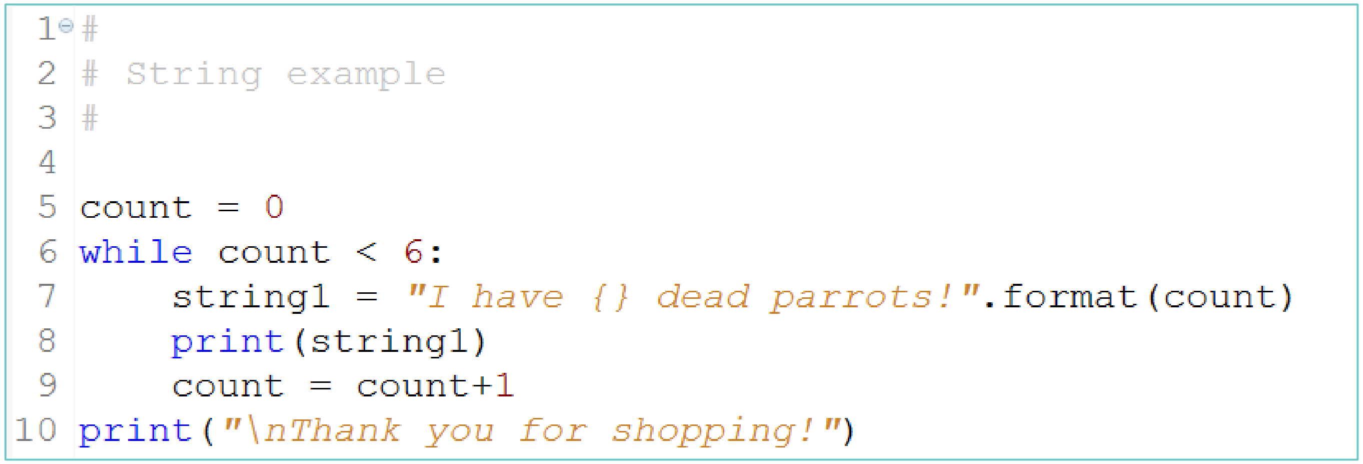
Strings

- Immutable objects, cannot change value
- Can reassign. (dynamic typing)
- Single or Double quotes, OK (even triple…)
- Index starts at 0 (of course…)

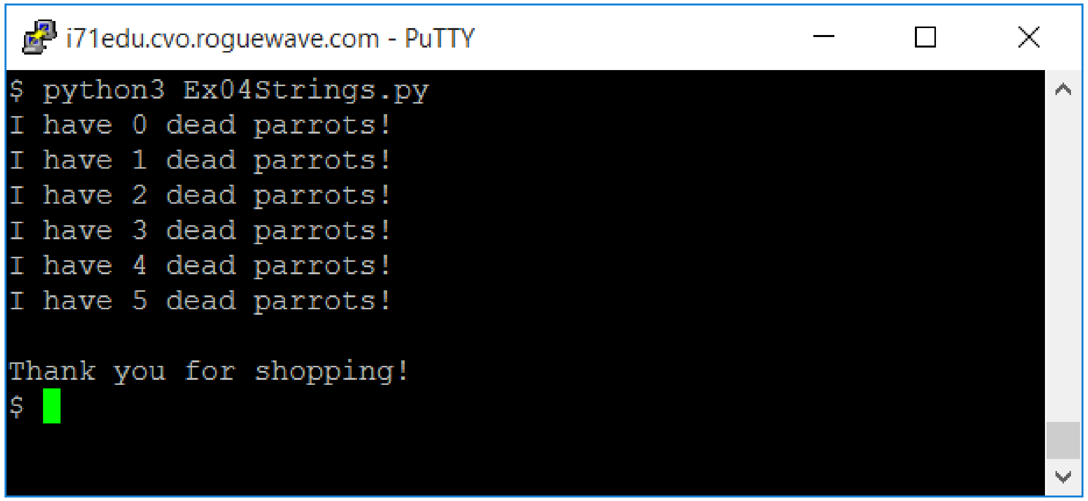


String formatting

- Interpolation, of sorts



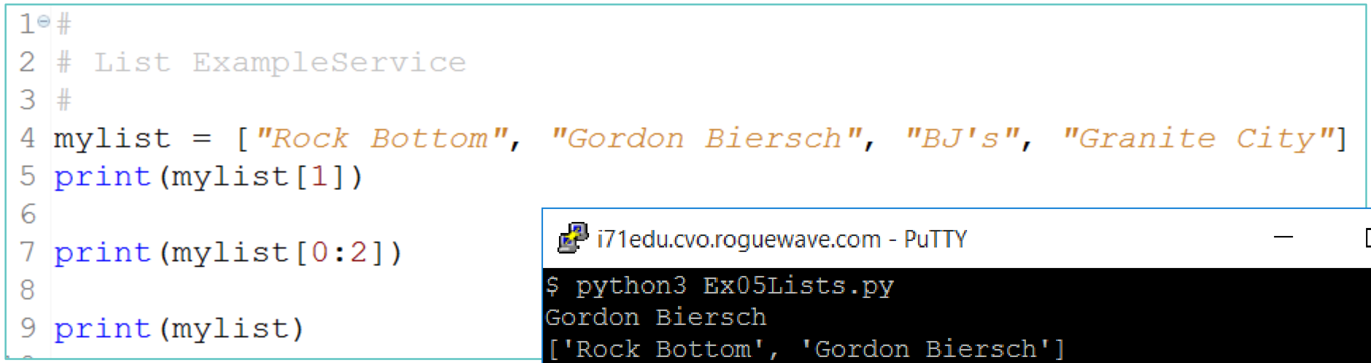
```python
#
# String example
#

count = 0
while count < 6:
    string1 = "I have {} dead parrots!".format(count)
    print(string1)
    count = count+1
print("\nThank you for shopping!")
```



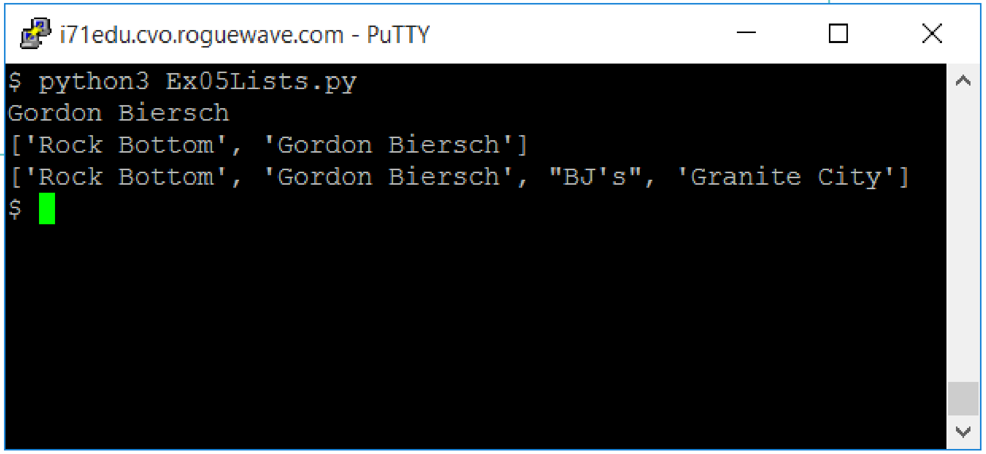
```
$ python3 Ex04Strings.py
I have 0 dead parrots!
I have 1 dead parrots!
I have 2 dead parrots!
I have 3 dead parrots!
I have 4 dead parrots!
I have 5 dead parrots!

Thank you for shopping!
$
```

Lists

- Ordered group, similar to array
- Different data types, ok
- Multi-dimensional (sub lists)
- Mutable (changeable)



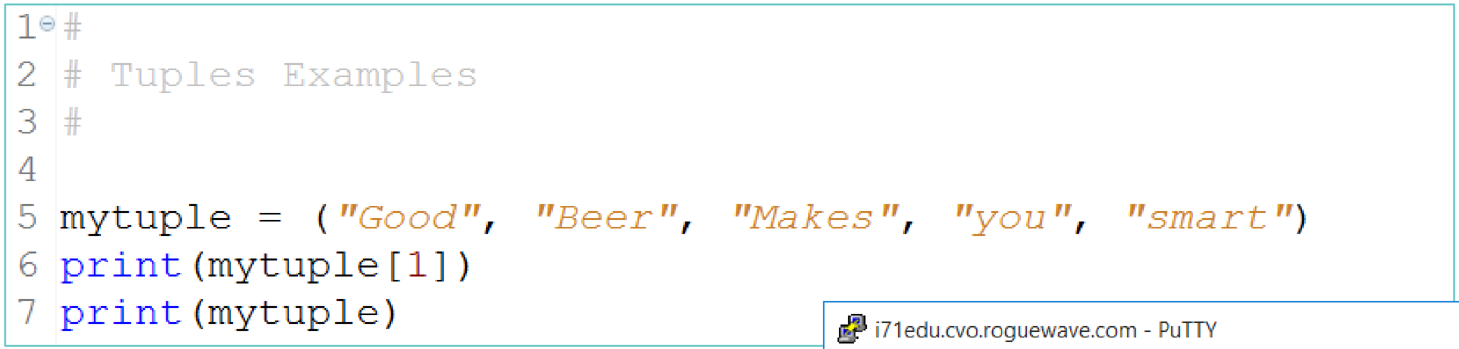
```
#
# List ExampleService
#
mylist = ["Rock Bottom", "Gordon Biersch", "BJ's", "Granite City"]
print(mylist[1])

print(mylist[0:2])

print(mylist)
```



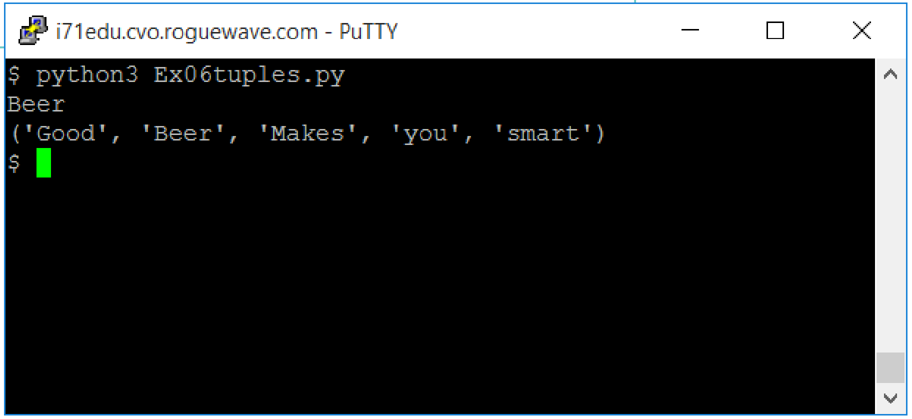
```
$ python3 Ex05Lists.py
Gordon Biersch
['Rock Bottom', 'Gordon Biersch']
['Rock Bottom', 'Gordon Biersch', "BJ's", 'Granite City']
$
```

Tuples

- Similar to lists
- Immutable (don’t change once created)
- Use parenthesis instead of brackets



```
#
# Tuples Examples
#

mytuple = ("Good", "Beer", "Makes", "you", "smart")
print(mytuple[1])
print(mytuple)
```

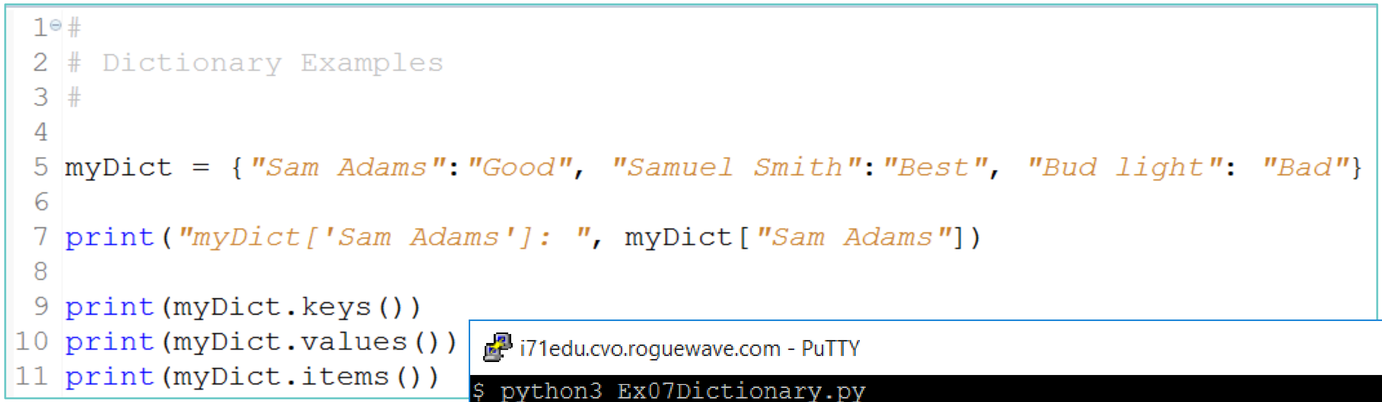


i71edu.cvo.roguewave.com - PuTTY

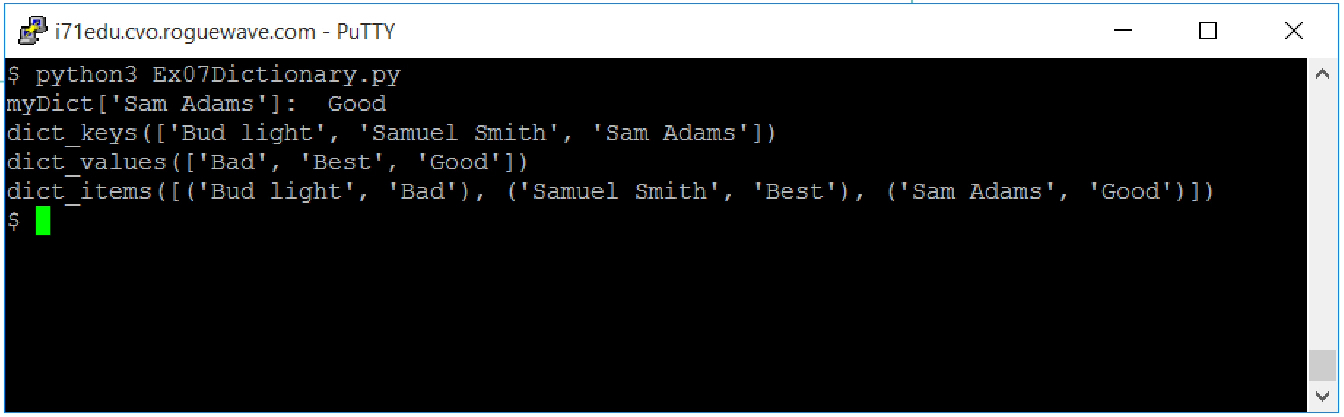
```
$ python3 Ex06tuples.py
Beer
('Good', 'Beer', 'Makes', 'you', 'smart')
$
```

Dictionary

- Again, like lists but more like hash or PHP Assoc. Array
- Mutable
- Key value pairs



```
#
# Dictionary Examples
#

myDict = {"Sam Adams":"Good", "Samuel Smith":"Best", "Bud light": "Bad"}

print("myDict['Sam Adams']: ", myDict["Sam Adams"])

print(myDict.keys())
print(myDict.values())
print(myDict.items())
```



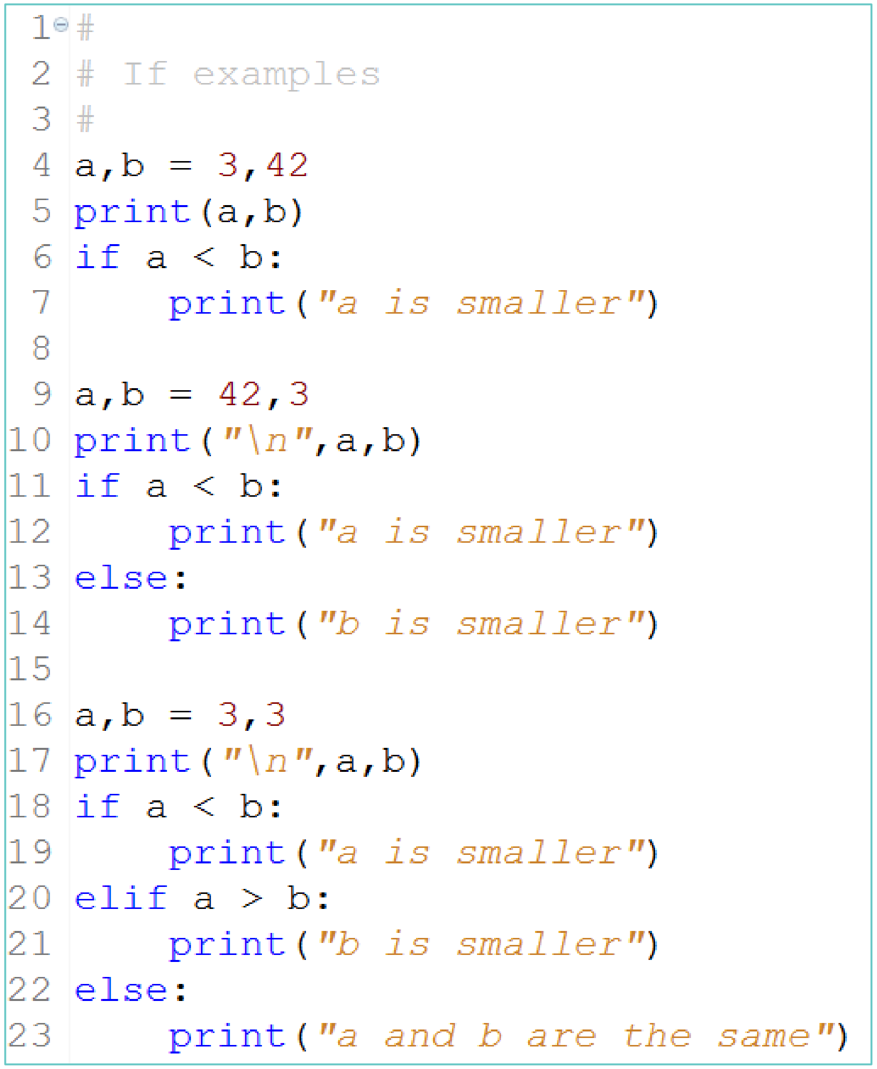
i71edu.cvo.roguewave.com - PuTTY

```
$ python3 Ex07Dictionary.py
myDict['Sam Adams']:  Good
dict_keys(['Bud light', 'Samuel Smith', 'Sam Adams'])
dict_values(['Bad', 'Best', 'Good'])
dict_items([('Bud light', 'Bad'), ('Samuel Smith', 'Best'), ('Sam Adams', 'Good')])
$
```

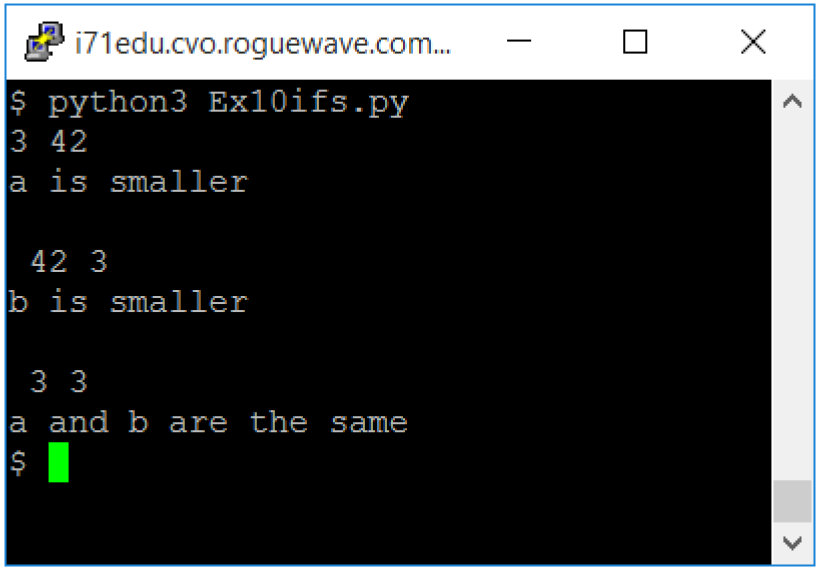
Syntax

Control Structures

ifs



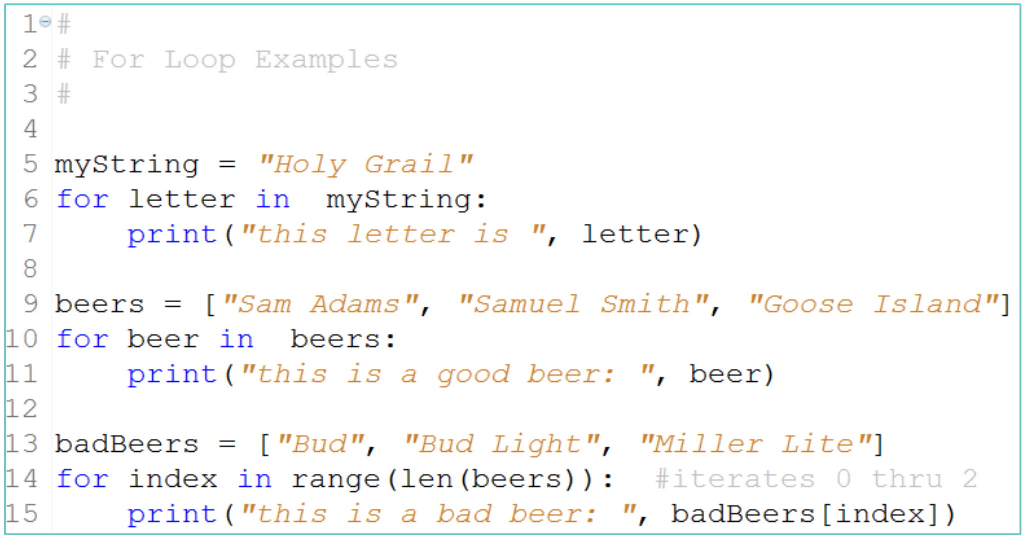
```
#
# If examples
#
a,b = 3,42
print(a,b)
if a < b:
    print("a is smaller")

a,b = 42,3
print("\n",a,b)
if a < b:
    print("a is smaller")
else:
    print("b is smaller")

a,b = 3,3
print("\n",a,b)
if a < b:
    print("a is smaller")
elif a > b:
    print("b is smaller")
else:
    print("a and b are the same")
```

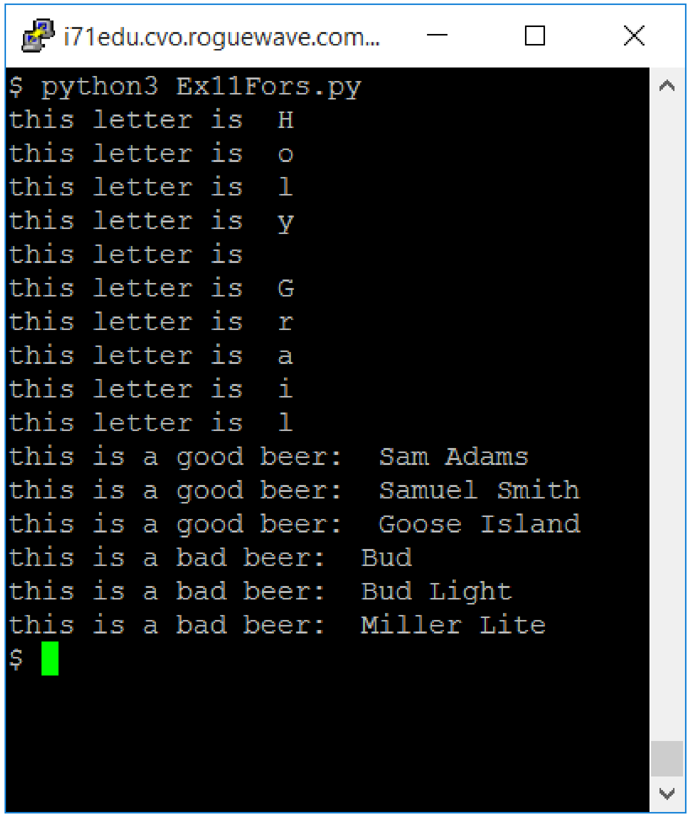


```
$ python3 Ex10ifs.py
3 42
a is smaller

 42 3
b is smaller

 3 3
a and b are the same
$ 
```

FRESCHE

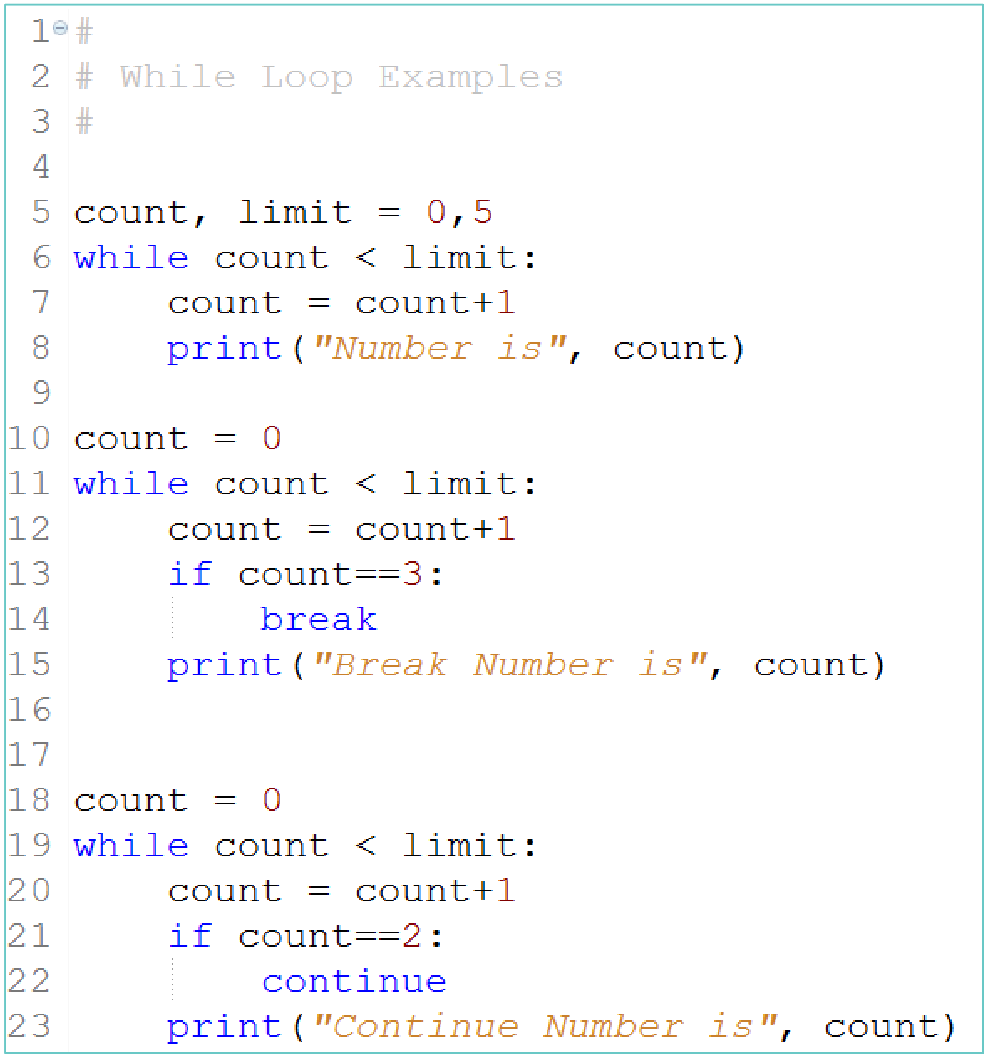
for loop


```
#
# For Loop Examples
#

myString = "Holy Grail"
for letter in  myString:
    print("this letter is ", letter)

beers = ["Sam Adams", "Samuel Smith", "Goose Island"]
for beer in  beers:
    print("this is a good beer: ", beer)

badBeers = ["Bud", "Bud Light", "Miller Lite"]
for index in range(len(beers)):  #iterates 0 thru 2
    print("this is a bad beer: ", badBeers[index])
```

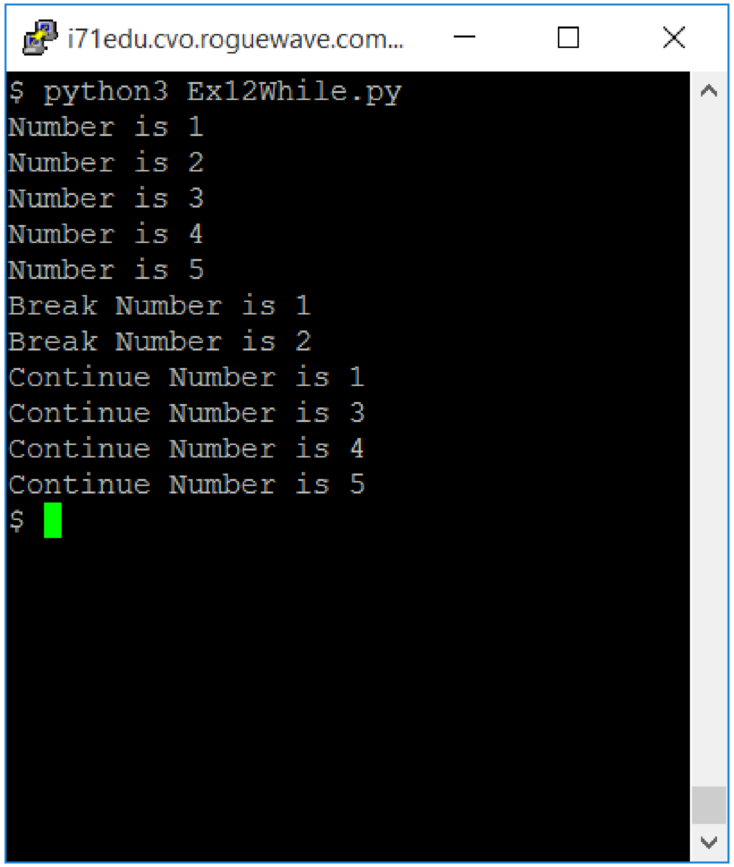

i71edu.cvo.roguewave.com...

```
$ python3 Ex11Fors.py
this letter is  H
this letter is  o
this letter is  l
this letter is  y
this letter is
this letter is  G
this letter is  r
this letter is  a
this letter is  i
this letter is  l
this is a good beer:  Sam Adams
this is a good beer:  Samuel Smith
this is a good beer:  Goose Island
this is a bad beer:  Bud
this is a bad beer:  Bud Light
this is a bad beer:  Miller Lite
$
```

while loop



```python
#
# While Loop Examples
#

count, limit = 0,5
while count < limit:
    count = count+1
    print("Number is", count)

count = 0
while count < limit:
    count = count+1
    if count==3:
        break
    print("Break Number is", count)


count = 0
while count < limit:
    count = count+1
    if count==2:
        continue
    print("Continue Number is", count)
```



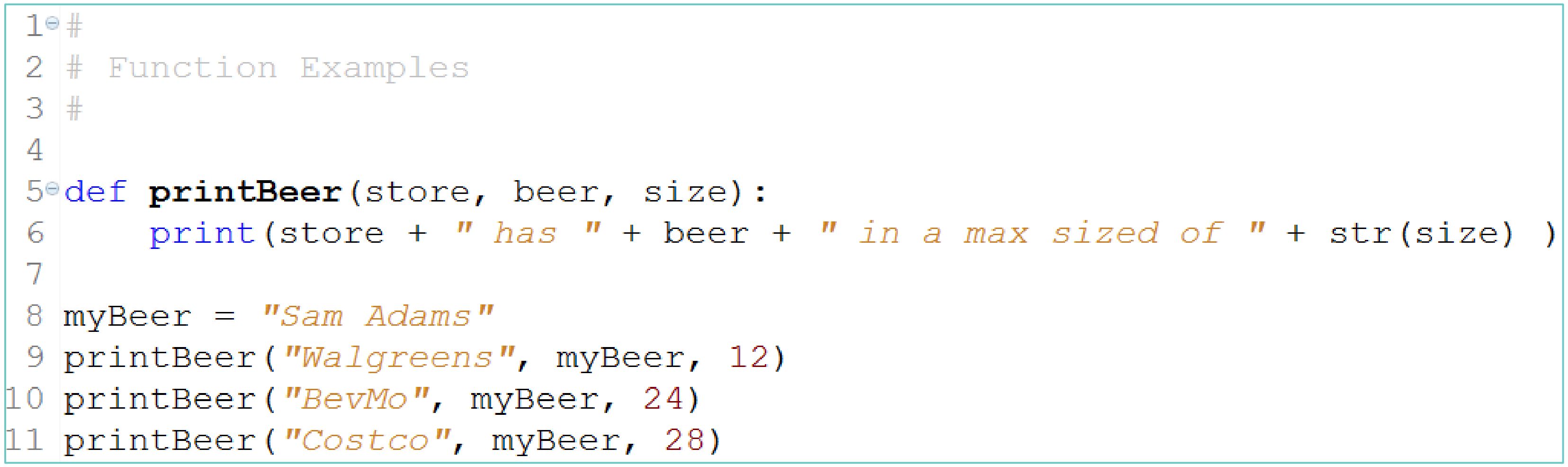
i71edu.cvo.roguewave.com...

```
$ python3 Ex12While.py
Number is 1
Number is 2
Number is 3
Number is 4
Number is 5
Break Number is 1
Break Number is 2
Continue Number is 1
Continue Number is 3
Continue Number is 4
Continue Number is 5
$
```

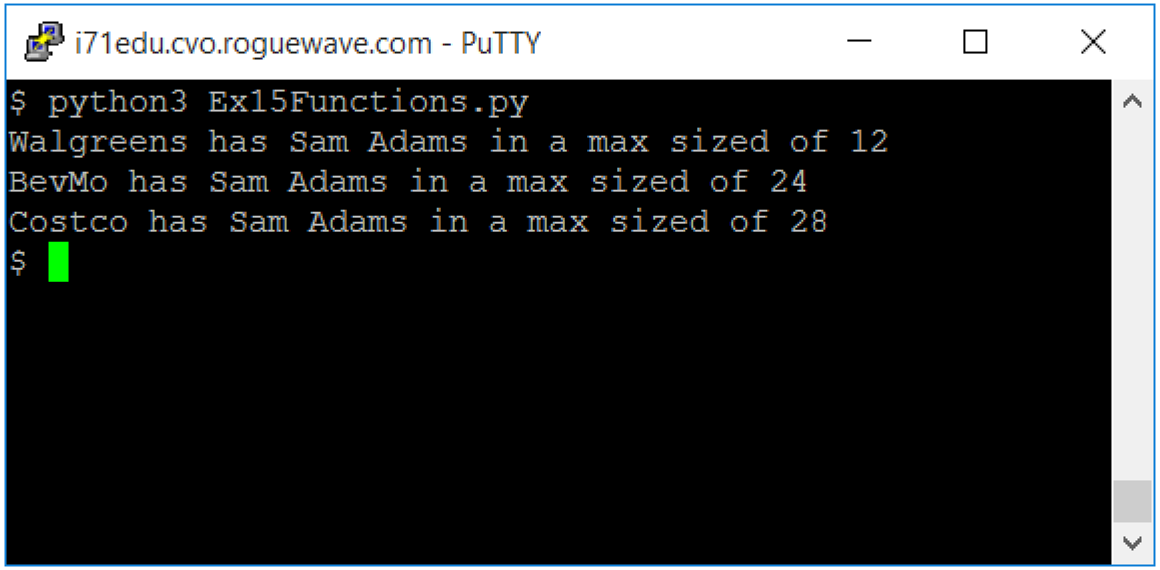
Syntax

Functions

Basic functions

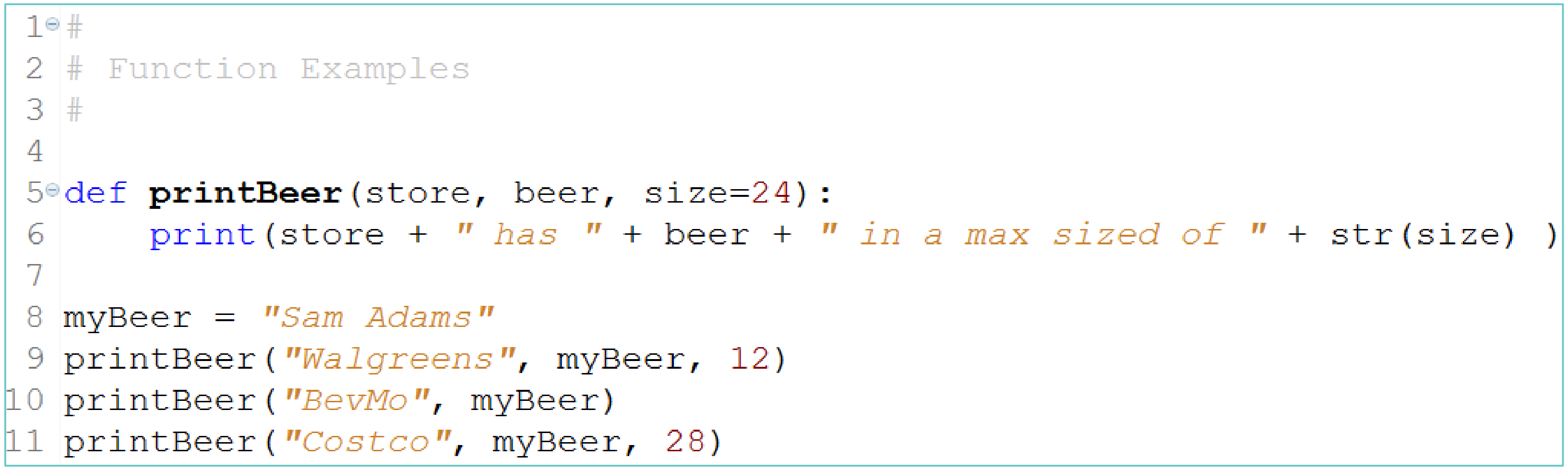


```
#
# Function Examples
#

def printBeer(store, beer, size):
    print(store + " has " + beer + " in a max sized of " + str(size) )

myBeer = "Sam Adams"
printBeer("Walgreens", myBeer, 12)
printBeer("BevMo", myBeer, 24)
printBeer("Costco", myBeer, 28)
```

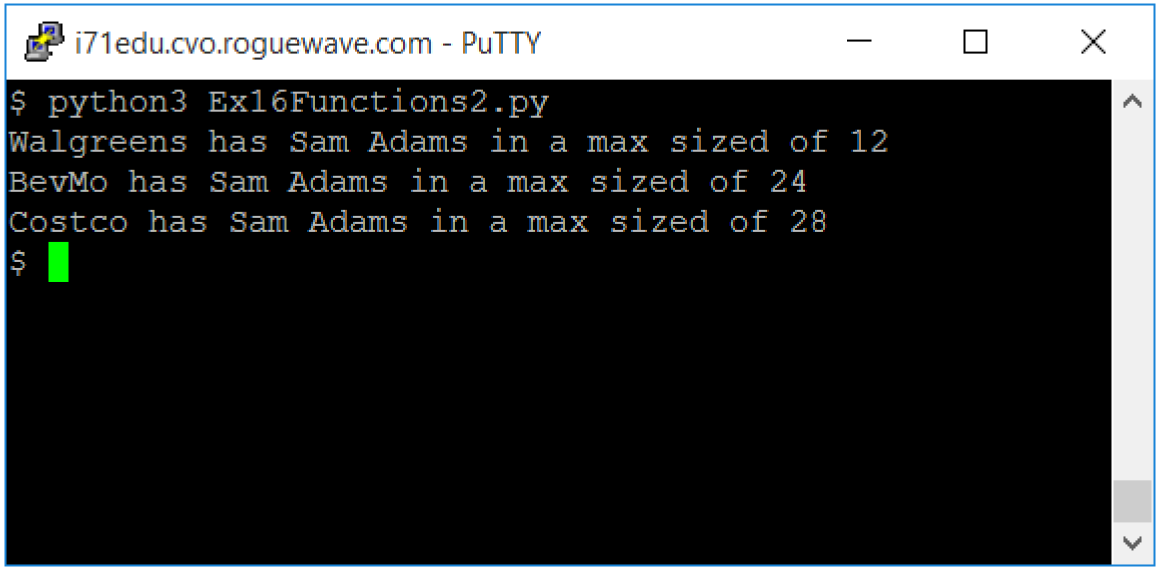


```
$ python3 Ex15Functions.py
Walgreens has Sam Adams in a max sized of 12
BevMo has Sam Adams in a max sized of 24
Costco has Sam Adams in a max sized of 28
$
```

Functions with defaults



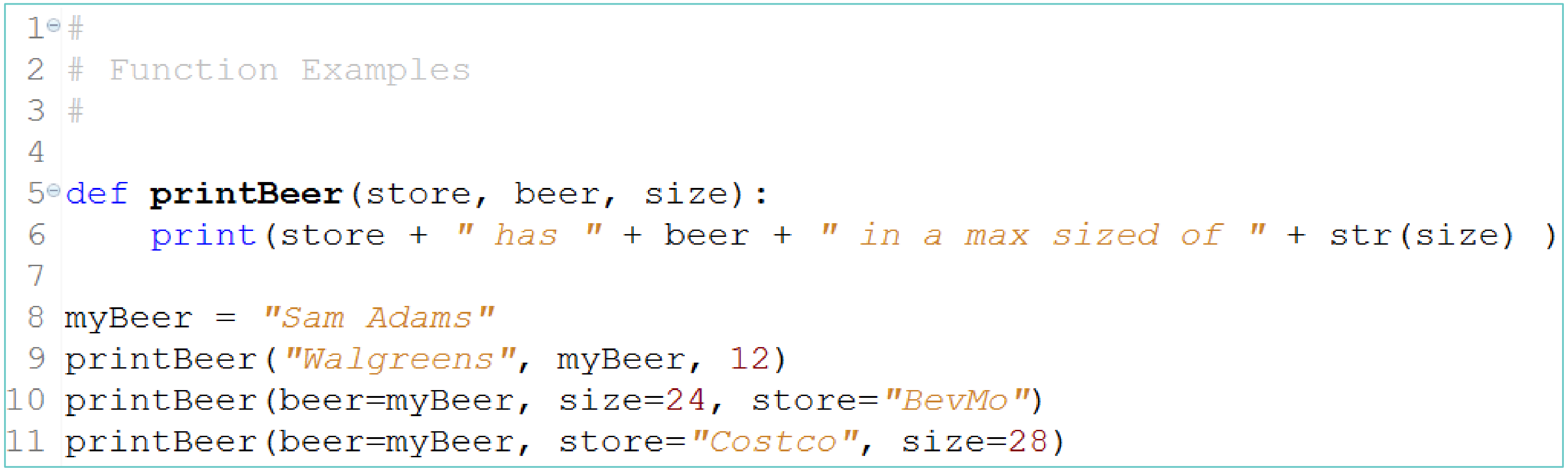
```python
#
# Function Examples
#

def printBeer(store, beer, size=24):
    print(store + " has " + beer + " in a max sized of " + str(size) )

myBeer = "Sam Adams"
printBeer("Walgreens", myBeer, 12)
printBeer("BevMo", myBeer)
printBeer("Costco", myBeer, 28)
```



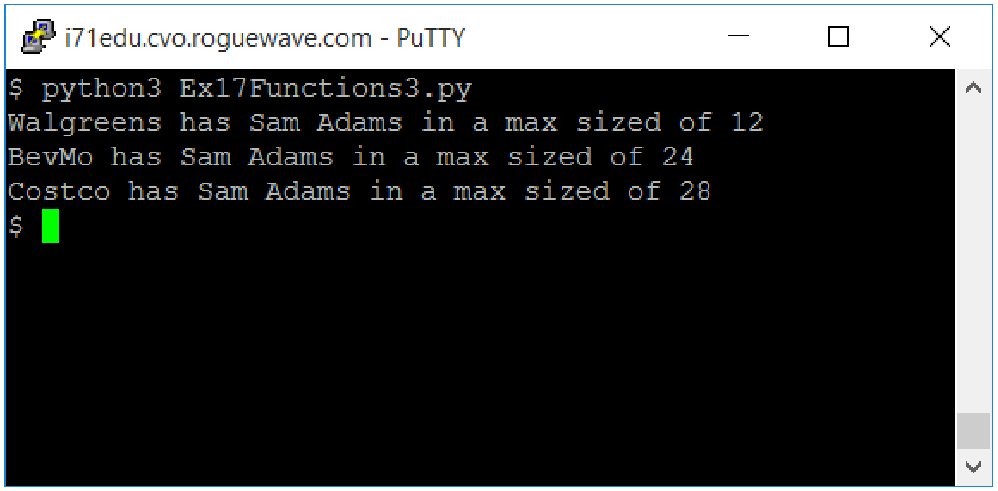
i71edu.cvo.roguewave.com - PuTTY

```
$ python3 Ex16Functions2.py
Walgreens has Sam Adams in a max sized of 12
BevMo has Sam Adams in a max sized of 24
Costco has Sam Adams in a max sized of 28
$
```

Functions with Keyword arguments



```
#
# Function Examples
#

def printBeer(store, beer, size):
    print(store + " has " + beer + " in a max sized of " + str(size) )

myBeer = "Sam Adams"
printBeer("Walgreens", myBeer, 12)
printBeer(beer=myBeer, size=24, store="BevMo")
printBeer(beer=myBeer, store="Costco", size=28)
```



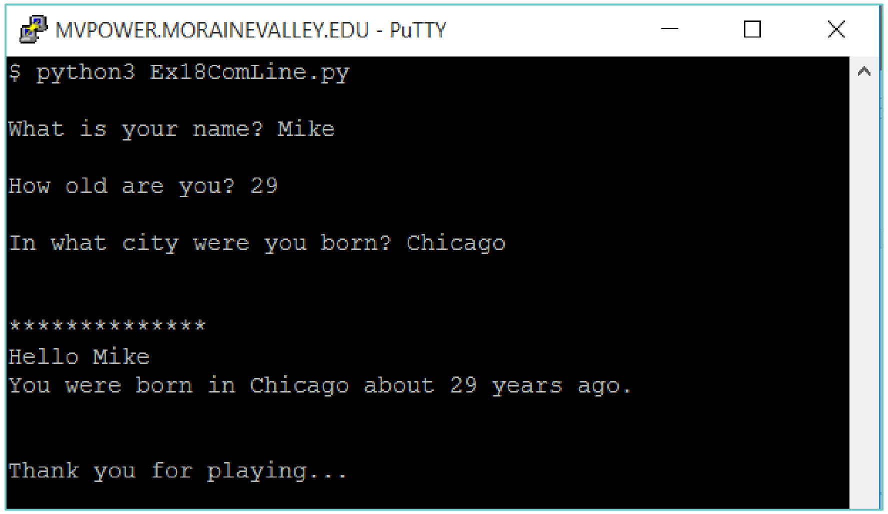
```
$ python3 Ex17Functions3.py
Walgreens has Sam Adams in a max sized of 12
BevMo has Sam Adams in a max sized of 24
Costco has Sam Adams in a max sized of 28
$ 
```

Command Line

Input from command line

- “Talk” to the script…

```python
# Get input from user and then embed in string
from pip._vendor.distlib.compat import raw_input

name = raw_input("\nWhat is your name? ")
age = raw_input("\nHow old are you? ")
city = raw_input("\nIn what city were you born? ")
print("\n\n**************")
print("Hello %s" % (name))
print("You were born in %s about %s years ago." % (city, str(age)))
print("\n\nThank you for playing...\n\n")
```

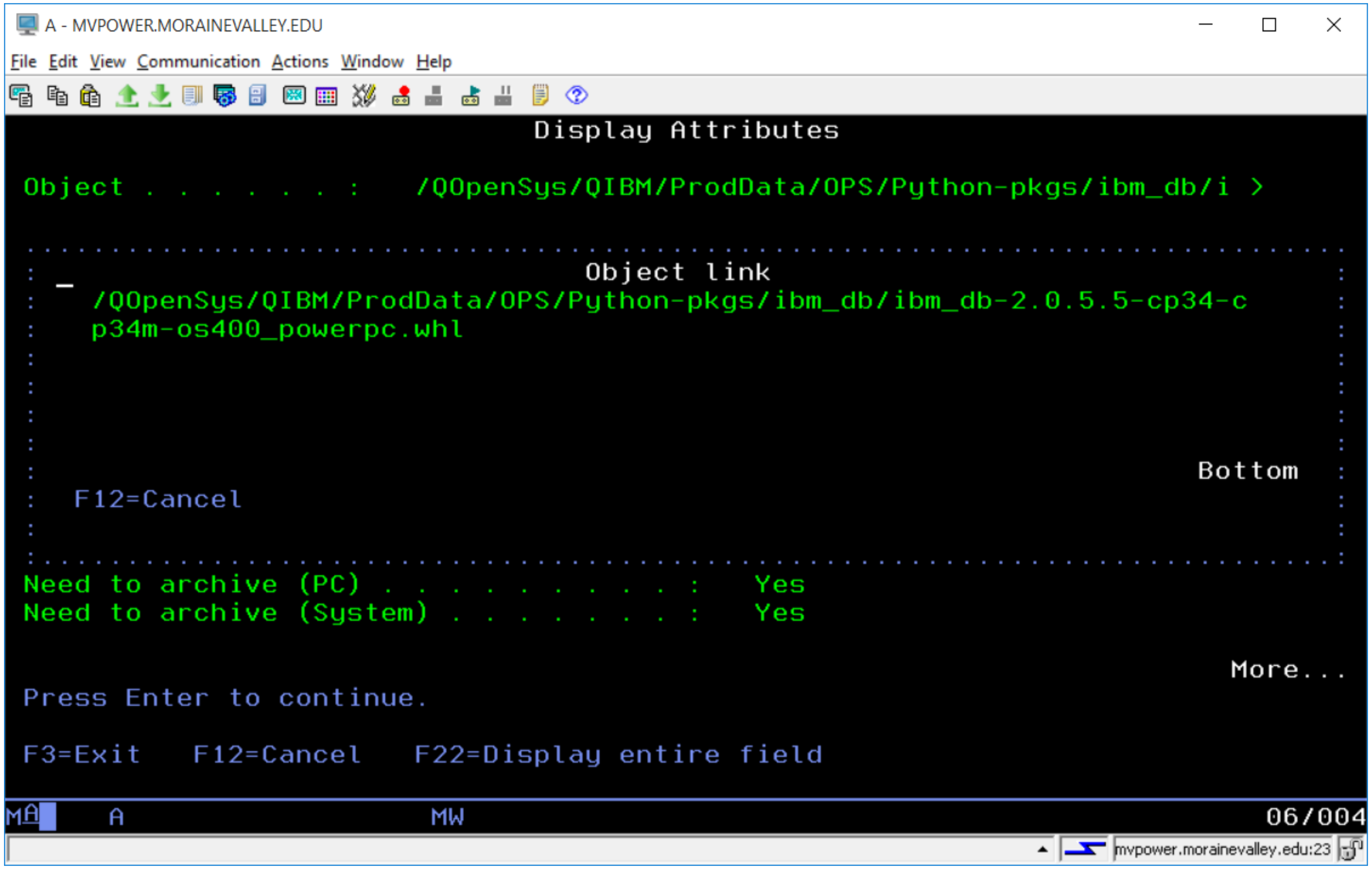


MVPOWER.MORAINEVALLEY.EDU - PuTTY

```
$ python3 Ex18ComLine.py

What is your name? Mike

How old are you? 29

In what city were you born? Chicago


**************
Hello Mike
You were born in Chicago about 29 years ago.


Thank you for playing...
```

Database

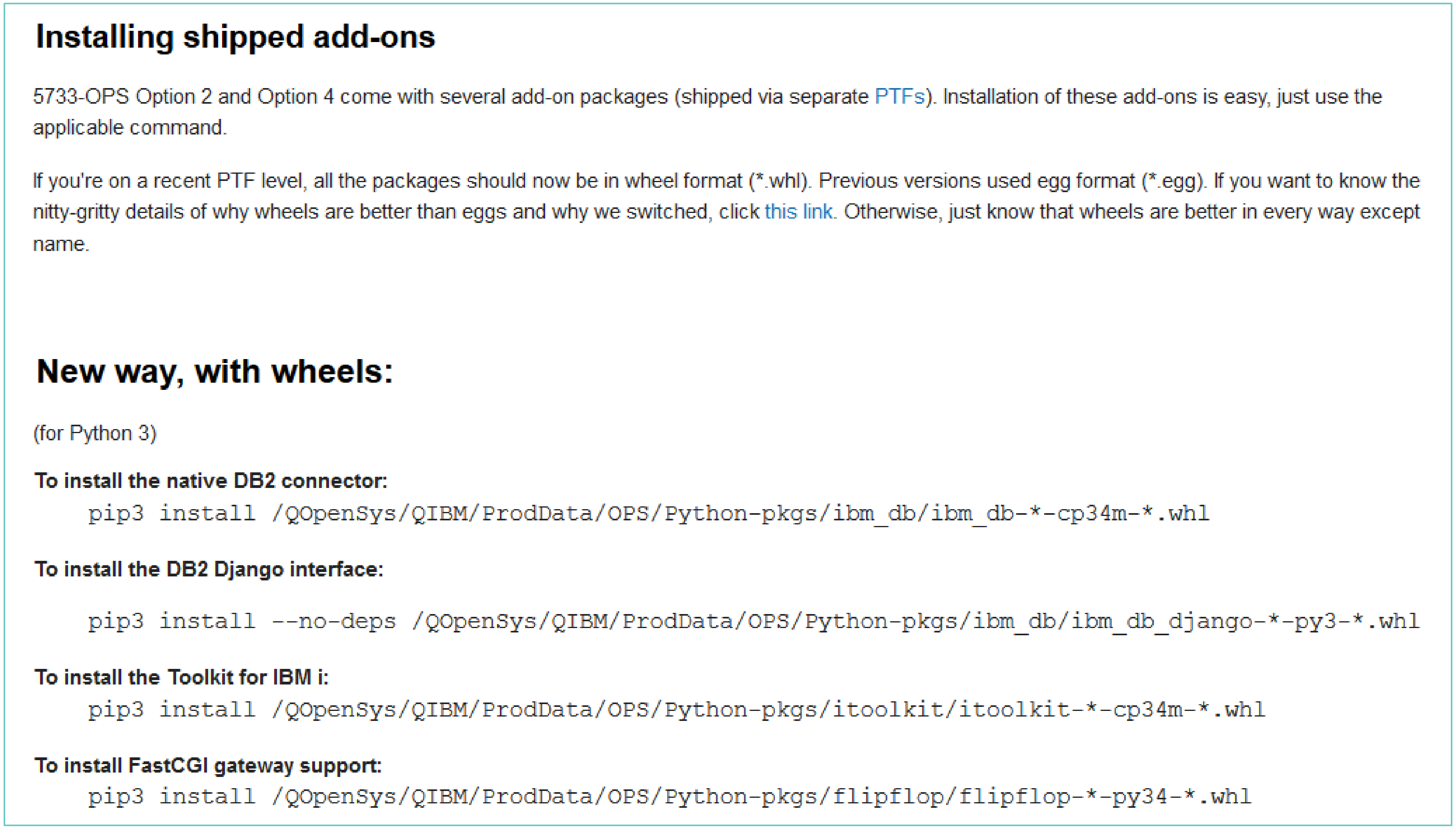
Locate the package or "wheel"


A - MVPOWER.MORAINEVALLEY.EDU
File Edit View Communication Actions Window Help

```
                              Display Attributes

Object . . . . . . :   /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/i >

 ..............................................................................
 :                             Object link                                    :
 : _  /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-2.0.5.5-cp34-c    :
 :    p34m-os400_powerpc.whl                                                  :
 :                                                                            :
 :                                                                            :
 :                                                                            :
 :                                                                            :
 :                                                                   Bottom   :
 :   F12=Cancel                                                               :
 :                                                                            :
 :............................................................................:
Need to archive (PC) . . . . . . . . . :   Yes
Need to archive (System) . . . . . . . :   Yes

                                                                      More...
Press Enter to continue.

F3=Exit   F12=Cancel   F22=Display entire field

MA     A                 MW                                           06/004
```

mvpower.morainevalley.edu:23

Install commands



Installing shipped add-ons

5733-OPS Option 2 and Option 4 come with several add-on packages (shipped via separate PTFs). Installation of these add-ons is easy, just use the applicable command.

If you're on a recent PTF level, all the packages should now be in wheel format (*.whl). Previous versions used egg format (*.egg). If you want to know the nitty-gritty details of why wheels are better than eggs and why we switched, click this link. Otherwise, just know that wheels are better in every way except name.

New way, with wheels:

(for Python 3)

To install the native DB2 connector:

```
    pip3 install /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-*-cp34m-*.whl
```

To install the DB2 Django interface:

```
    pip3 install --no-deps /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db_django-*-py3-*.whl
```

To install the Toolkit for IBM i:

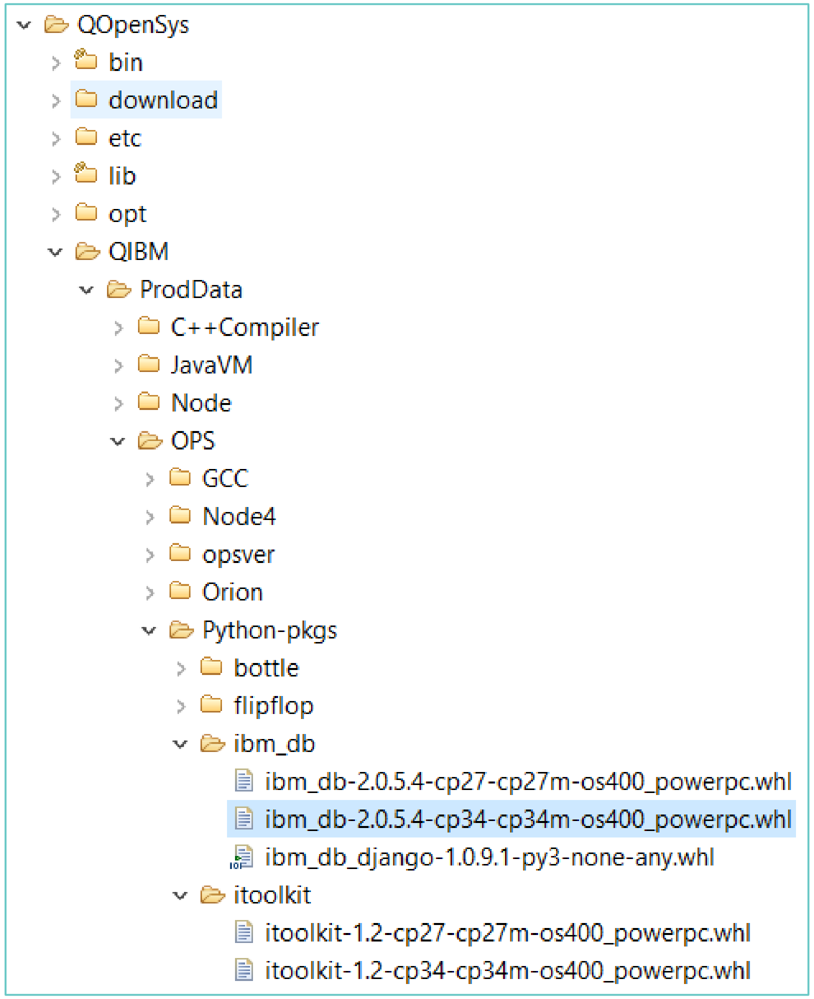
```
    pip3 install /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/itoolkit/itoolkit-*-cp34m-*.whl
```

To install FastCGI gateway support:

```
    pip3 install /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/flipflop/flipflop-*-py34-*.whl
```

Find the connector

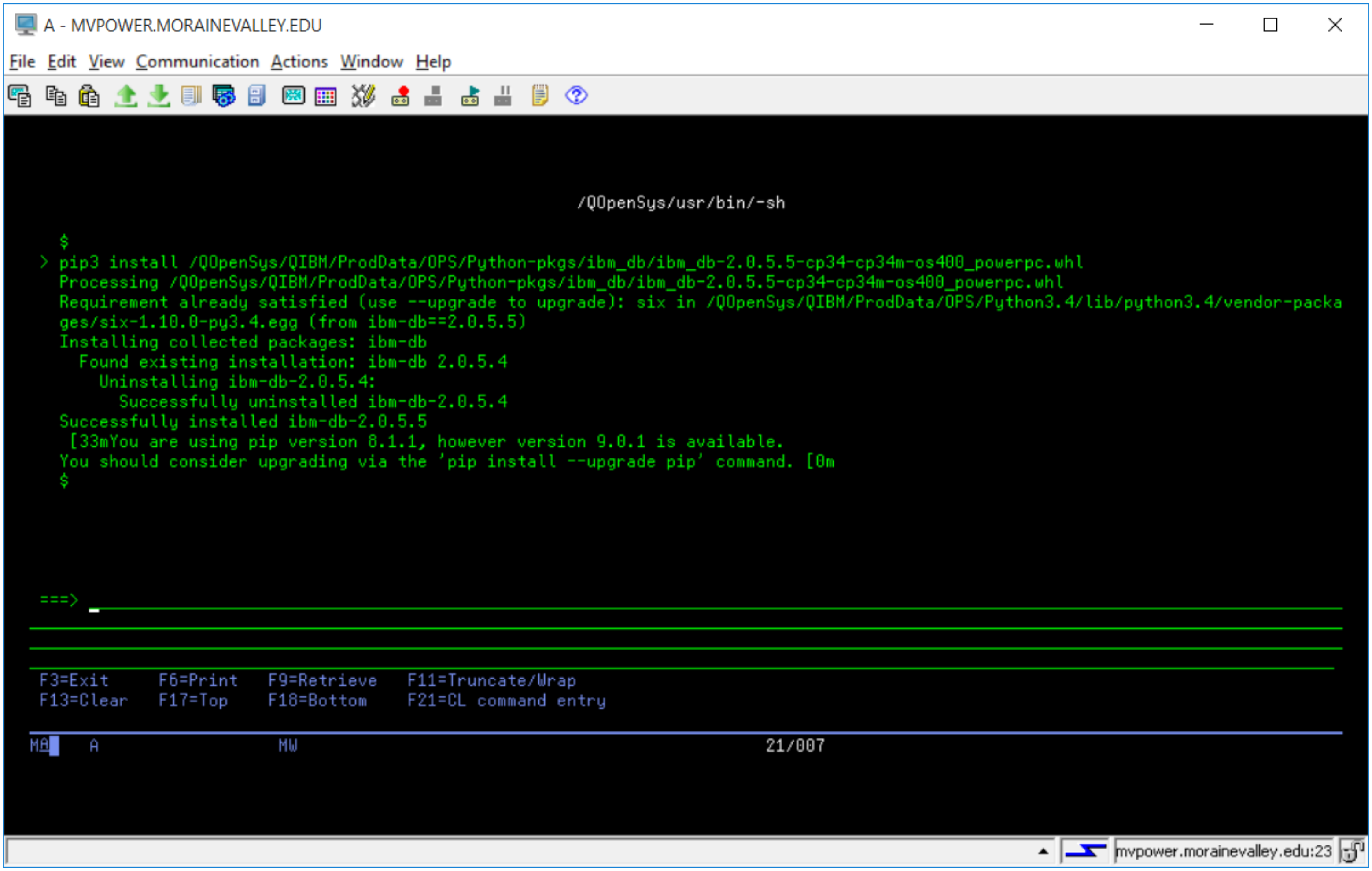
- YMMV
- With wheels



```
QOpenSys
  bin
  download
  etc
  lib
  opt
  QIBM
    ProdData
      C++Compiler
      JavaVM
      Node
      OPS
        GCC
        Node4
        opsver
        Orion
        Python-pkgs
          bottle
          flipflop
          ibm_db
            ibm_db-2.0.5.4-cp27-cp27m-os400_powerpc.whl
            ibm_db-2.0.5.4-cp34-cp34m-os400_powerpc.whl
            ibm_db_django-1.0.9.1-py3-none-any.whl
          itoolkit
            itoolkit-1.2-cp27-cp27m-os400_powerpc.whl
            itoolkit-1.2-cp34-cp34m-os400_powerpc.whl
```

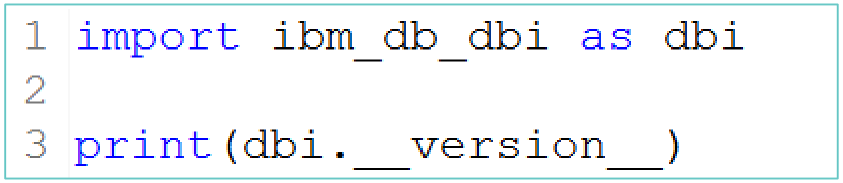
Run the pip install

- pip == Python installer program

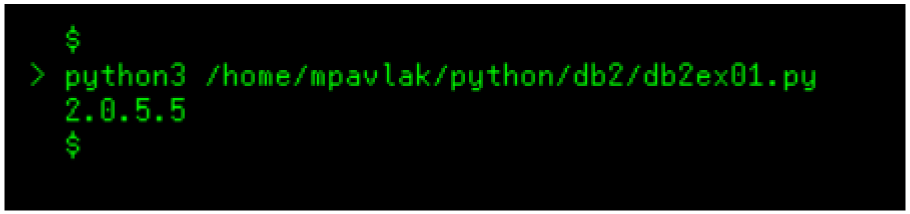

A - MVPOWER.MORAINEVALLEY.EDU
File Edit View Communication Actions Window Help

```
                                  /QOpenSys/usr/bin/-sh

   $
 > pip3 install /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-2.0.5.5-cp34-cp34m-os400_powerpc.whl
   Processing /QOpenSys/QIBM/ProdData/OPS/Python-pkgs/ibm_db/ibm_db-2.0.5.5-cp34-cp34m-os400_powerpc.whl
   Requirement already satisfied (use --upgrade to upgrade): six in /QOpenSys/QIBM/ProdData/OPS/Python3.4/lib/python3.4/vendor-packa
   ges/six-1.10.0-py3.4.egg (from ibm-db==2.0.5.5)
   Installing collected packages: ibm-db
     Found existing installation: ibm-db 2.0.5.4
       Uninstalling ibm-db-2.0.5.4:
         Successfully uninstalled ibm-db-2.0.5.4
   Successfully installed ibm-db-2.0.5.5
    [33mYou are using pip version 8.1.1, however version 9.0.1 is available.
   You should consider upgrading via the 'pip install --upgrade pip' command. [0m
   $




 ===> _



 F3=Exit      F6=Print    F9=Retrieve    F11=Truncate/Wrap
 F13=Clear    F17=Top     F18=Bottom     F21=CL command entry

MA     A                  MW                                         21/007
```

What version of the DB2 Extension?



```
import ibm_db_dbi as dbi

print(dbi.__version__)
```

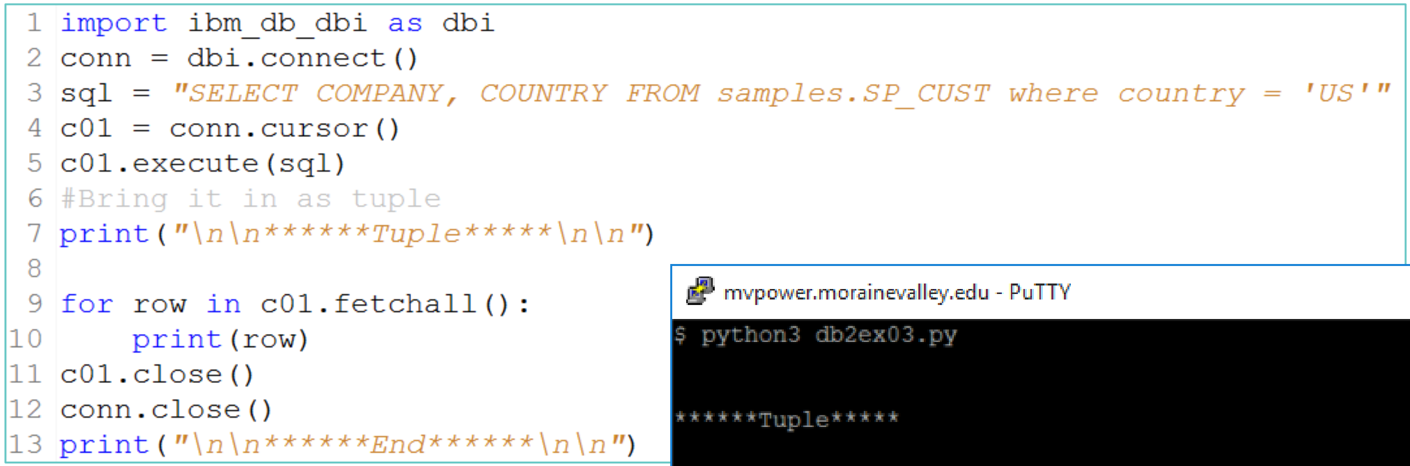


```
  $
> python3 /home/mpavlak/python/db2/db2ex01.py
  2.0.5.5
  $
```

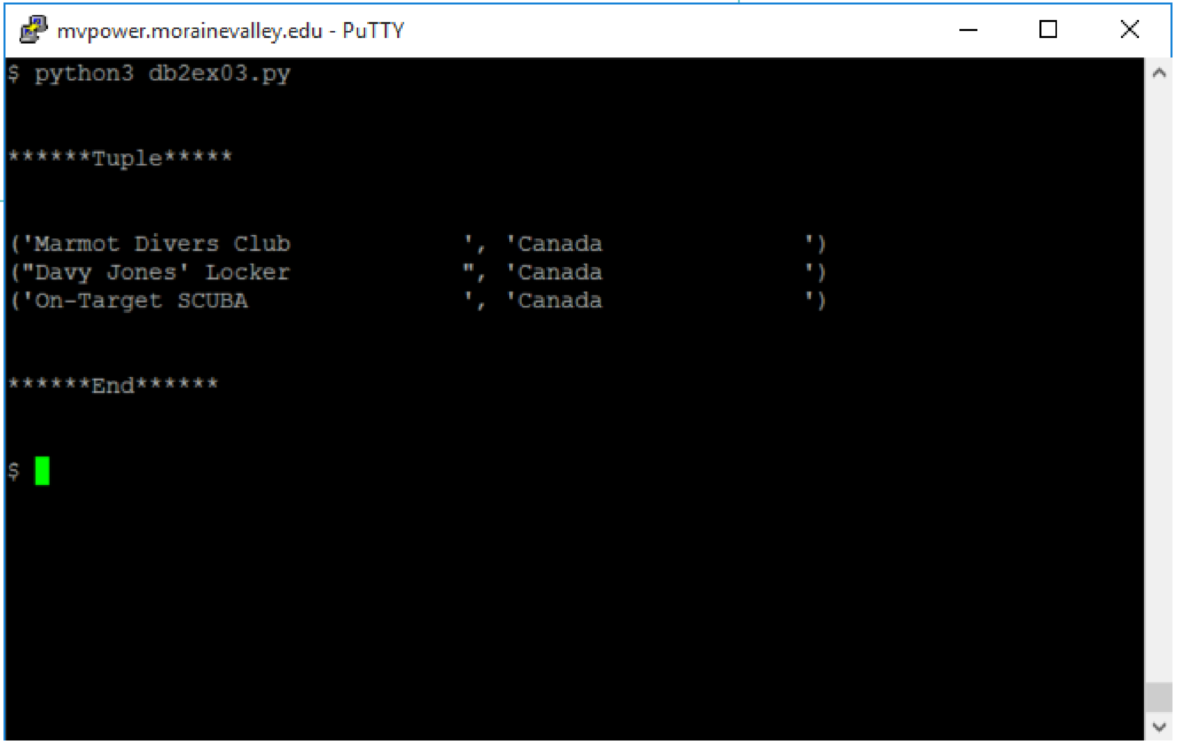
Steps for simple database Access

- Import the class
- Connect (with or without options
- Open the cursor
- Set the SQL
- Read

Simple database access



```python
import ibm_db_dbi as dbi
conn = dbi.connect()
sql = "SELECT COMPANY, COUNTRY FROM samples.SP_CUST where country = 'US'"
c01 = conn.cursor()
c01.execute(sql)
#Bring it in as tuple
print("\n\n******Tuple*****\n\n")

for row in c01.fetchall():
    print(row)
c01.close()
conn.close()
print("\n\n******End******\n\n")
```

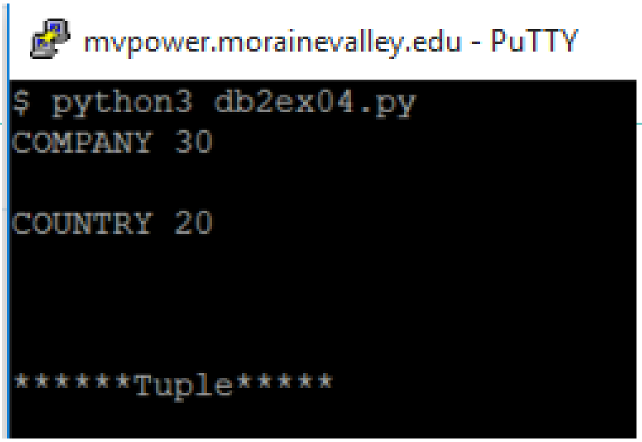


mvpower.morainevalley.edu - PuTTY

```
$ python3 db2ex03.py


******Tuple*****


('Marmot Divers Club              ', 'Canada               ')
("Davy Jones' Locker              ", 'Canada               ')
('On-Target SCUBA                 ', 'Canada               ')


******End******


$ 
```

Table info

```python
import ibm_db_dbi as dbi
conn = dbi.connect()
sql = "SELECT COMPANY, COUNTRY FROM ZENDSVR6.SP_CUST where country = 'Canada'"
c01 = conn.cursor()
c01.execute(sql)
desc = c01.description
print(desc[0][0], desc[0][4], "\n")
print(desc[1][0], desc[1][4], "\n")

#Bring it in as tuple
print("\n\n******Tuple*****\n\n")
for row in c01.fetchall():
    print(row)
c01.close()
conn.close()
print("\n\n******End******\n\n")
```

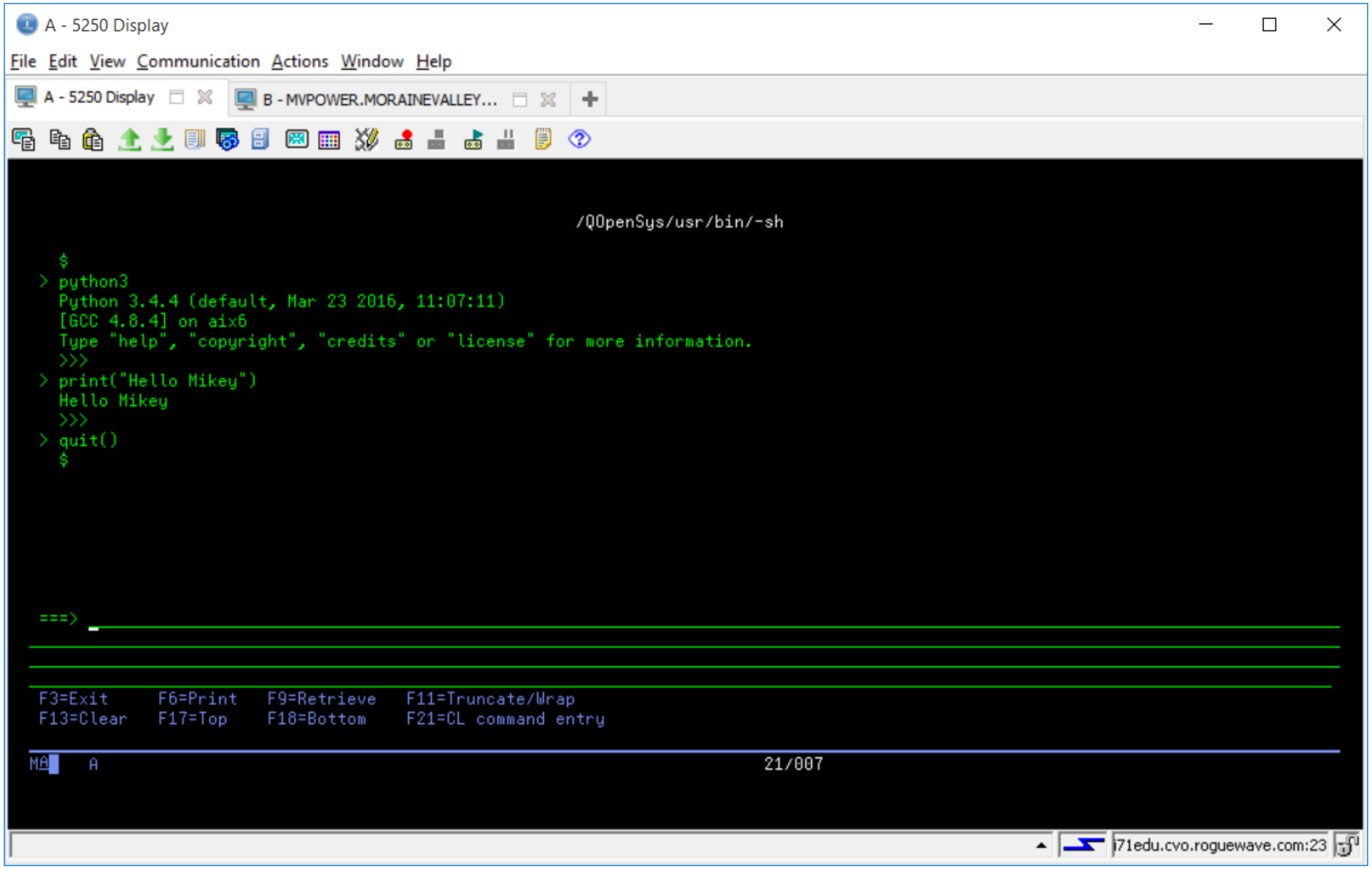


```
$ python3 db2ex04.py
COMPANY 30

COUNTRY 20



******Tuple*****
```

Summary – Why Python

- Lot’s of libraries
- Make it easy to do stuff
- OPC / OPO
- Education

End the session


A - 5250 Display
File Edit View Communication Actions Window Help
A - 5250 Display
B - MVPOWER.MORAINEVALLEY...

```
                                        /QOpenSys/usr/bin/-sh

   $
 > python3
   Python 3.4.4 (default, Mar 23 2016, 11:07:11)
   [GCC 4.8.4] on aix6
   Type "help", "copyright", "credits" or "license" for more information.
   >>>
 > print("Hello Mikey")
   Hello Mikey
   >>>
 > quit()
   $




 ===> _

 F3=Exit      F6=Print    F9=Retrieve    F11=Truncate/Wrap
 F13=Clear    F17=Top     F18=Bottom     F21=CL command entry

MA      A                                              21/007
```

i71edu.cvo.roguewave.com:23

THANK YOU

Mike.Pavlak@freschesolutions.com

FRESCHE SOLUTIONS