

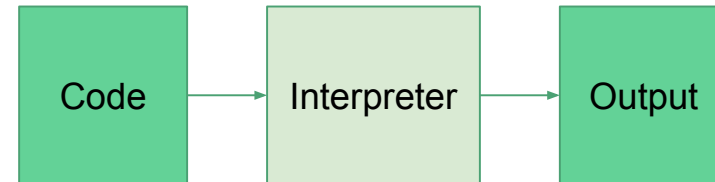
BU CodeBreakers 2017

Python I: Introduction

WHAT IS PYTHON?

Python

- Interpretive programming language
- Interpreter reads each line and executes them
- Invented by Guido Van Rossum in the Netherlands the early 90's
- Advantages:
 - Easy to learn (syntax)
 - Open-sourced [source-code is freely available]
 - Object-oriented
 - Extensive libraries



pycrypto 2.6.1

Cryptographic modules for Python.



Running Python

Modes:

1. Interactive Mode

```
>>> 2 + 2
```

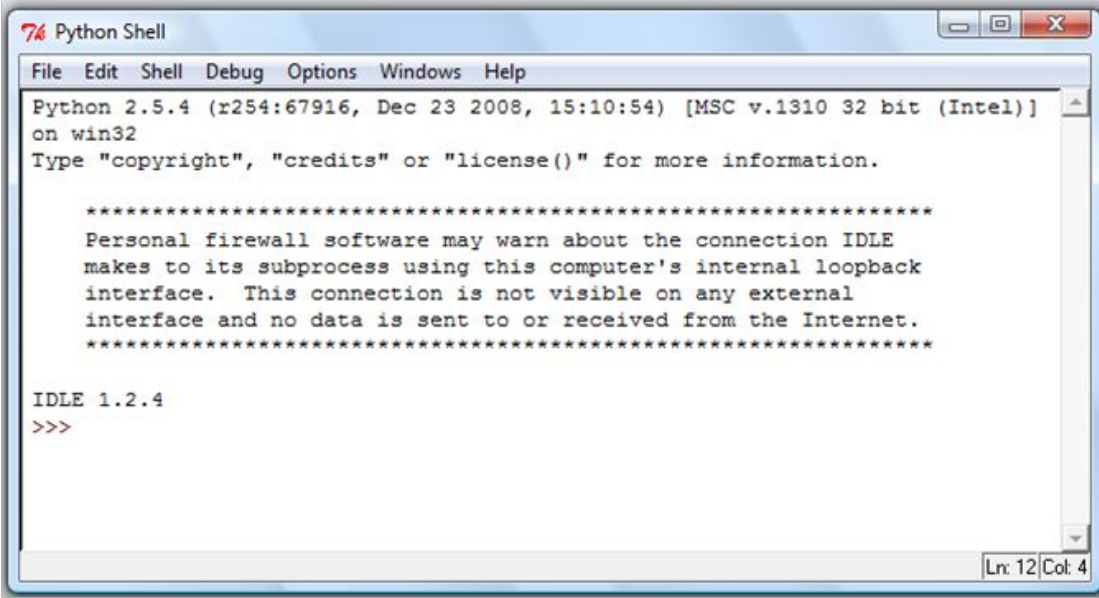
```
4
```

2. Script Mode



Python: Interactive Mode

1. Open Python IDLE [Integrated DeveLopment Environment] program
2. Directly type and enter commands after `>>>`



```
Python Shell
File Edit Shell Debug Options Windows Help
Python 2.5.4 (x254:67916, Dec 23 2008, 15:10:54) [MSC v.1310 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.

*****
Personal firewall software may warn about the connection IDLE
makes to its subprocess using this computer's internal loopback
interface. This connection is not visible on any external
interface and no data is sent to or received from the Internet.
*****

IDLE 1.2.4
>>>
```


Exercise

```
>>> print ('Hello world!!!')
```

Hello world!!!

```
>>> 10 * 3
```

30



What are the different
types of data we used
here?



Datatypes

- What are the **types** of data you can use?

Integers <int>:	12	100	000000000000
Decimals <float>:	1.345	0.0	24512.0
Strings <str>:	'Hello!!'	"M1a\$trInG?"	'123'
Lists <list>:	[1, 4, 6]	['a', 'd', '!??']	[1.4, 'str', 0]
Boolean <bool>:	True	/1	False/0



What <type> am I?

```
>>> type (1.342)
```

```
<class 'float'>
```

```
>>> type (2)
```

```
<class 'int'>
```

```
>>> type("2.14")
```

```
<class 'str'>
```

```
>>> type([1, 2.3, "hello"])
```

```
<class 'list'>
```

Can I change my <type>? : Typecasting

```
>>> int(3.134)
```

```
3
```

```
>>> int("sd")
```

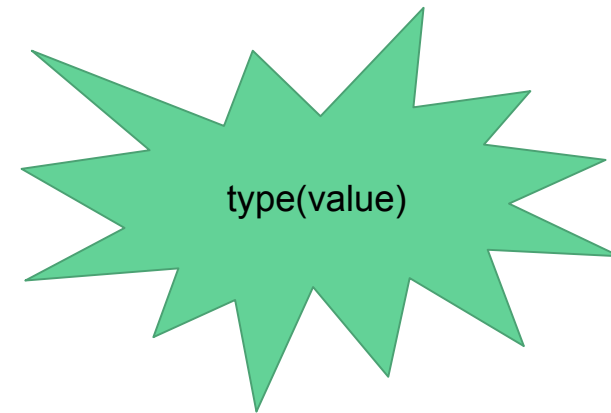
```
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in <module>
```

```
ValueError: invalid literal for int() with base 10: 'sd'
```

```
>>> str(123)
```

```
'123'
```



Variable

- Name that refers to a value
- Use assignment statement to create new variables

Example: What's the variable? What's the type? What's the value?

Message = 'this message will expire in 3, 2, 1,

N = 34

Pi = 3.14159

How to do you find their types?

Variable: types

```
>>> type (Message)
```

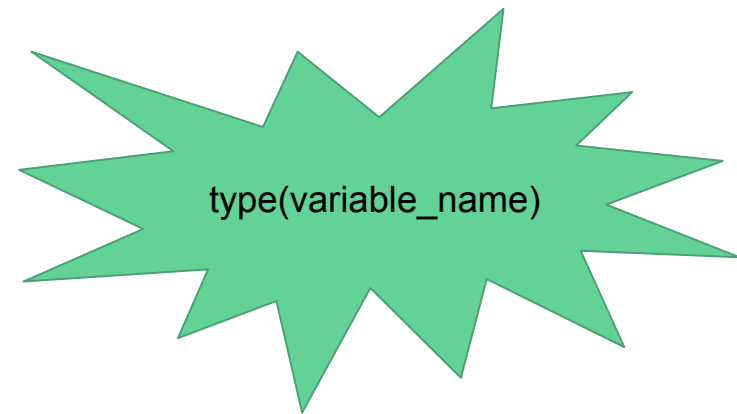
```
<class 'str'>
```

```
>>> type (N)
```

```
<class 'int'>
```

```
>>> type(Pi)
```

```
<class 'float'>
```



Variable Naming

Naming Rules:

1. Case-sensitive
Example: MyName \neq myname
2. Consists of characters
_ A-Z a-z 0-9
3. Cannot start with digits: 0-9
4. Cannot use **keywords**
Example: True, False, not, for, etc..



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

Variable name?

Value?

Variable type?

hello	=	"Hello!"
---------	---	----------



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

Variable name?

Value?

Variable type?

n_3_1

=

3100



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

1list

=

[1,2,3]



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- **Cannot start with digits: 0-9**
- Cannot use **keywords**

```
File "<stdin>", line 1
1list = [1,3,'one']
  ^
SyntaxError: invalid syntax
```

1list

=

[1,2,3]



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

li\$t

=

[1,2,3]



Is this correct?

Naming Rules:

- Case-sensitive
- **Consists of characters: A-Z, a-z, 0-9, _**
- Cannot start with digits: 0-9
- Cannot use **keywords**

```
File "<stdin>", line 1
  li$t = [1,3,'one']
    ^
SyntaxError: invalid syntax
```

li\$t

=

[1,2,3]

Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

Variable name?

Value?

Variable type?

Is this a good
variable name?

false

=

True



Is this correct?

Naming Rules:

- Case-sensitive
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- Cannot use **keywords**

False

=

True



Is this correct?

Naming Rules:

- **Case-sensitive**
- Consists of characters: A-Z, a-z, 0-9, _
- Cannot start with digits: 0-9
- **Cannot use keywords**

```
File "<stdin>", line 1  
SyntaxError: can't assign to  
keyword
```

False

=

True



Keywords

- Reserved words in Python
- Cannot be used as Identifiers!!
- Case-sensitive



<https://www.programiz.com/python-programming/keywords-identifier>

Operators Review

- What are Operators?

Special symbols to represent various **computations**

Example: $2 * 3 - 2 = 4$

What is $*$?

What is $-$?

What is $=$?

Arithmetic Operators?

- | | | | |
|----|----|----------------|----------------------------------|
| 1. | + | Addition | $2 + 3 \rightarrow 5$ |
| 2. | - | Subtraction | $2 - 3 \rightarrow -1$ |
| 3. | * | Multiplication | $2 * 3 \rightarrow 6$ |
| 4. | ** | Exponent | $2 ** 3 \rightarrow 8$ |
| 5. | % | Modulus | $2 \% 3 \rightarrow 2$ |
| 6. | / | Division | $2 / 3 \rightarrow 0.6666666666$ |



SEQUENTIAL DATA-TYPES

Strings and Lists: [Sequence types]

- Strings: Sequence of characters
- List: Sequence/Collection of numbers, strings, booleans, etc..

String	'	H	e	l	!	o	'
List	['hello'	1	3	4.5	'bye']



Strings and Lists: [Sequence types]

- Strings: Sequence of characters
- List: Sequence/Collection of numbers, strings, booleans, etc..
- **Index: indicates the position of each element in a string/list**

String	'	H	e	l	!	o	'
List	['hello'	1	3	4.5	'bye']
Index		0	1	2	3	4	



Accessing sequential elements

```
>>> my_name = "This is my name..."
```



What is my_name?



Accessing sequential elements

```
>>> my_name = "This is my name..."
```

```
>>> my_name[0]
```

```
'T'
```

```
>>> a = my_name[4]
```

```
>>> print (a)
```

```
','
```

```
>>> my_name[35]
```

```
IndexError: string index out of range
```

Single Index
[index]

Accessing sequential elements

```
>>> my_list = ["This", "is", "my", "list", ".", ".."]
```



What is my_list?



Accessing sequential elements

```
>>> my_list = ['This', 'is', 'my', 'list', '.', '..']
```

```
>>> my_list[0]
```

```
'This'
```

```
>>> my_list[4]
```

```
'.'
```

Single Index
[index]



Accessing sequential elements

```
>>> my_list = ['This', 'is', 'my', 'list', '.', '..']
```

```
>>> my_list[1:4]
```

```
['is', 'my', 'list']
```

```
# Watch out! The first index is inclusive, but the last index is exclusive
```

```
>>> my_list[4:]
```

```
['.', '..']
```

Index Range
[start_index : end_index]



Accessing sequential elements

```
>>> my_name = 'This is my name...'  
>>> my_list = ['This', 'is', 'my', 'list', '.', '..']
```

```
>>> my_name[-5]
```

```
'm'
```

```
>>> my_list[-3:5]
```

```
['list', '.']
```

```
>>> my_list[-4:]
```

```
['my', 'list', '.', '..']
```

Negative Index

[0	1	2	3	4]
[-5	-4	-3	-2	-1]



Accessing sequential elements

```
>>> my_name = "This is my name..."
```

```
>>> my_list = ["This", "is", "my", "list", ".", ".."]
```

```
>>> len(my_name)
```

```
18
```

```
>>> len(my_list)
```

```
6
```

Finding length

len (variable)



Accessing sequential elements

```
>>> my_name = "This is my name..."
```

```
>>> my_list = ["This", "is", "my", "list", ".", ".."]
```

```
>>> len(my_name)
```

```
18
```

```
>>> len(my_list)
```

```
6
```

Finding length

len () → function

We'll learn more about functions later..

Function



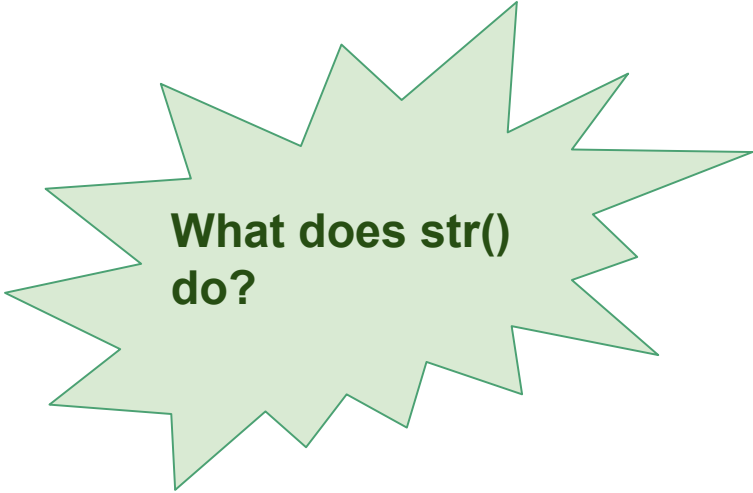
More Functions

```
>>> str(23)
```

```
'23'
```

```
>>> str([1,4,5])
```

```
'[1,4,5]'
```



**What does str()
do?**



More Functions

```
>>> list("hello")
```

```
['h', 'e', 'l', 'l', 'o']
```

```
>>> list(3)
```

```
TypeError: 'int' object is not iterable
```



**What does list()
do?**



Empty Sequential Types?

- Empty String?

“” ‘’

- Empty List?

[]



Concatenation: link together

```
>>> 'hello' + 'bye'
```

```
'hellobye'
```

```
>>> [1, 3, 5] + [3.4]
```

```
[1, 3, 5, 3.4]
```

```
>>> ['yello'] + []
```

```
['yello']
```

Concatenation Limitation

Cannot concat elements of different types

```
>>> 'hello' + 3
```

```
TypeError: Can't convert 'int' object to str implicitly
```

```
>>> 3 + []
```

```
TypeError: unsupported operand type(s) for +: 'int' and 'list'
```

More Concatenation

```
>>> 3.4 + 1
```

4.4

#Note: + here is used as addition not concatenation

How to concatenate 3.4 and 1 ?

Convert them to string...

```
>>> str(3.4) + str(1)
```

```
'3.41'
```



Note: String vs. List

- Strings are immutable while lists aren't

```
>>> my_list = [1, 2, 3]
```

```
>>> my_list[0] = 0
```

```
>>> print (my_list)
```

```
[0, 2, 3]
```

```
>>> my_string = 'string'
```

```
>>> my_string[0] = 'k'
```

```
TypeError: 'str' object does not support item assignment
```

in Operator

- Gives **True** if an element is in a list else gives **False**

```
>>> my_list = [1, 2, 3]
```

```
>>> 5 in my_list
```

False

- Gives **True** if a substring is in a string else gives **False**

```
>>> my_string = 'string'
```

```
>>> 'st' in my_string
```

True

Syntax Errors

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

```
SyntaxError: invalid syntax
```

```
>>> [2]+[[2]
```

```
...
```

```
... ]
```

```
[2, [2]]
```

EXERCISE 1

Additional Practice

- <http://codingbat.com/python>

