Data Processing and Analysis in Python
Lecture 1
Introduction

UNIVERSITY OF
MARYLAND

ROBERT H. SMITH
SCHOOL OF BUSINESS

Dr. Adam Lee

Information Processing

- Information is also commonly referred to as **data**
- In carrying out the instructions of an algorithm, computing agent manipulates information:
  - Starts with **input**
  - Transforms **information** according to well-defined rules
  - Produces **output**
- The algorithms that describe information processing can also be represented as information
- Computer scientists recently discovered how to represent many other things, such as:
  - Images, music, human speech, and video

Computer Software

- A program stored in computer memory must be represented in **binary** digits, or machine **code**
- A **loader** takes a set of machine language instructions as input and loads them into the memory locations
- The most important example of system software is a computer's **operating system (OS)**
- **Applications** include Web browsers, word processors, spreadsheets, database managers, games, etc…
- Scientists have developed **high-level** programming languages for expressing algorithms:

Computer Software

  - Resemble English and allow the author to express algorithms in a form that other people can understand

- Programmers usually start by writing high-level language statements in a **text editor**:
  - Runs another program called a **translator** to convert program code into executable code
  - Translator checks for **syntax errors**
- If no errors are found, program can be executed by the **run-time system**:
  - Might execute program directly, or
  - Run another program called **interpreter** or **virtual machine** to execute the program

Computer Software

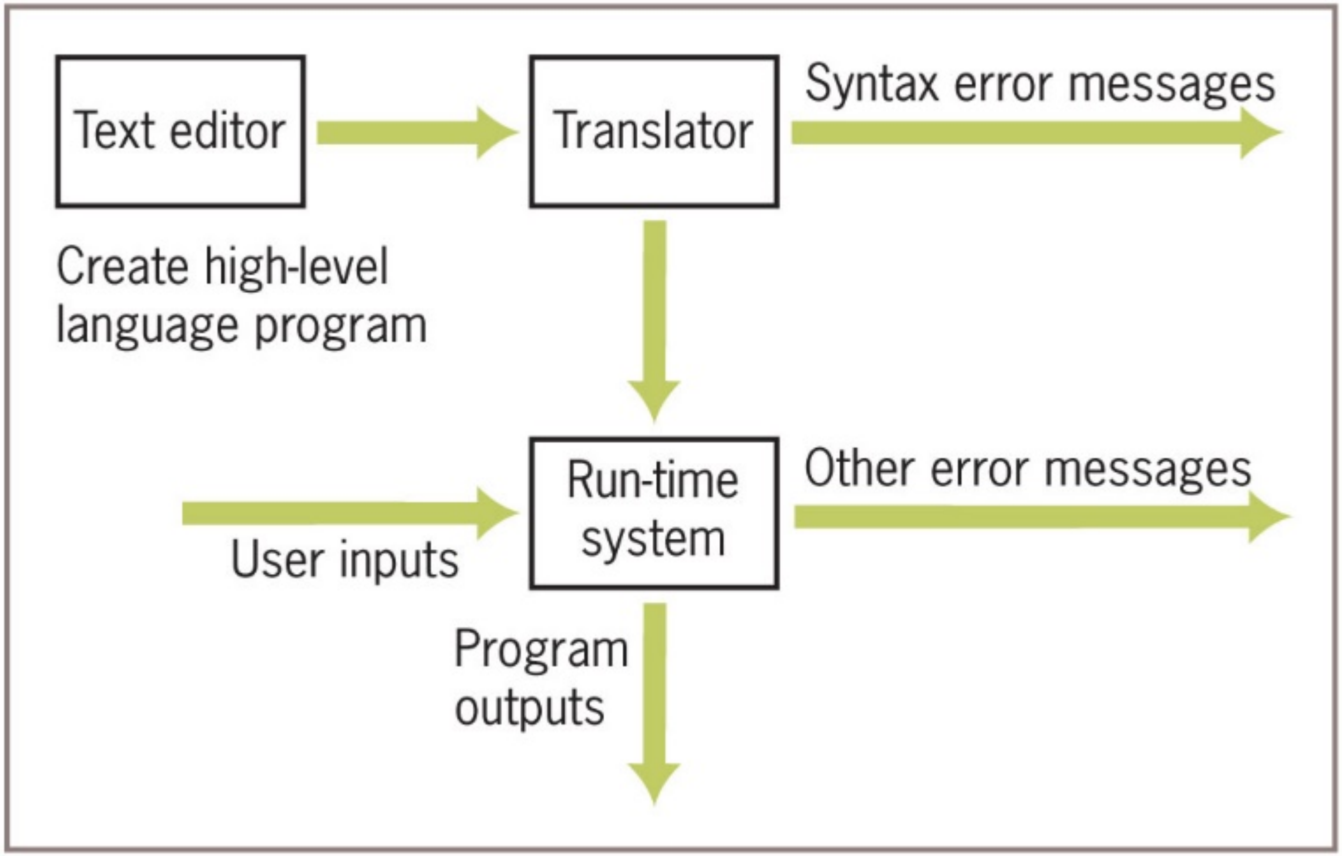
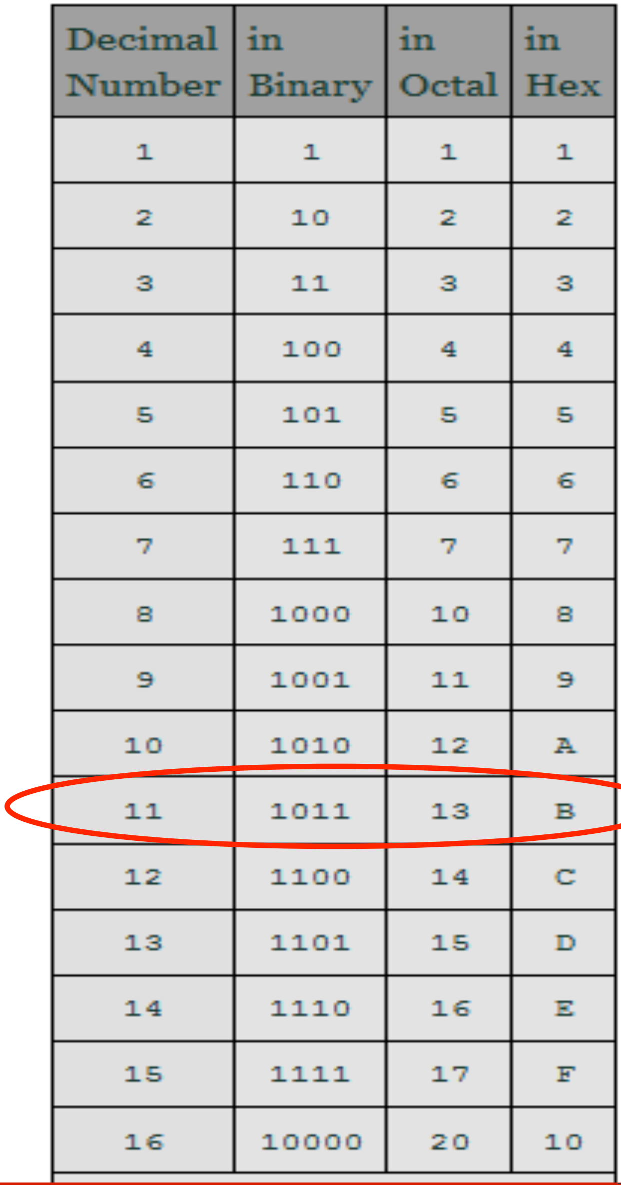


Figure 1-3 Software used in the coding process

Data Representation in Computers

- Computers are full of zillions of bits that are either on or off
  - 1 if the bit is on
  - 0 if the bit is off

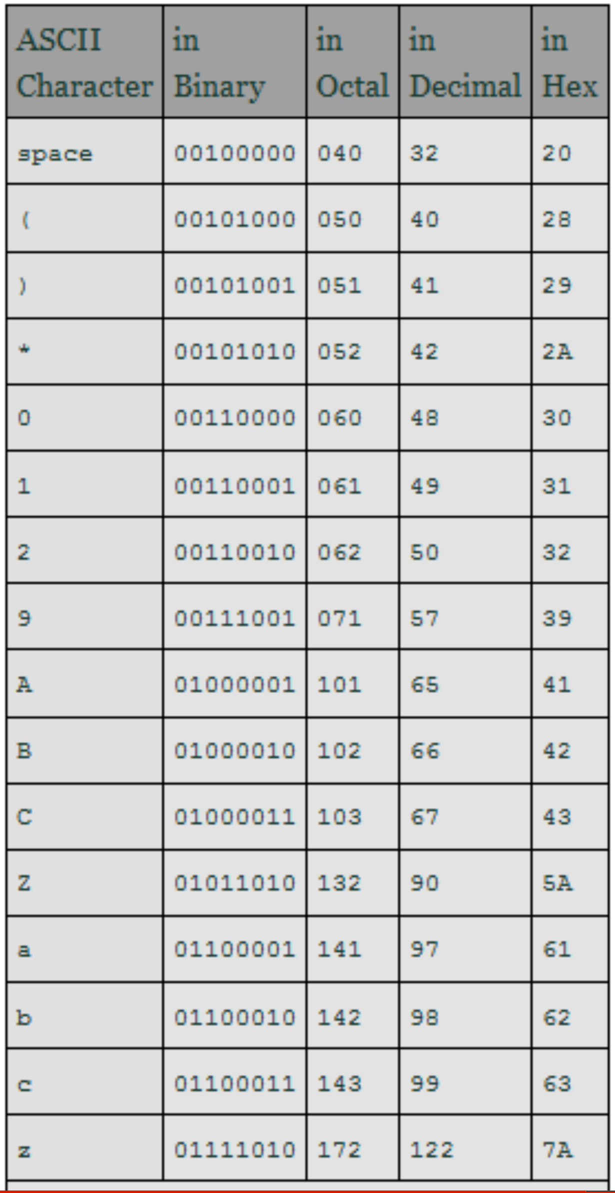
- Decimal numeral system
  - $123 = 1\times10^2 + 2\times10^1 + 3\times10^0$
- Binary numeral system
  - $1011 = 1\times2^3 + 0\times2^2 + 1\times2^1 + 1$ $= 11$ (decimal)



Decimal Number	in Binary	in Octal	in Hex
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F
16	10000	20	10

Data Representation in Computers

- **Symbols as bits – ASCII Characters**
  - 1 Byte = 8 bits
  - American Standard Code for Information Interchange (ASCII) standard www.asciitable.com
- Universal Coded Character Set (UCS) or **Unicode**
  - Unicode Transformation Format – UTF-7, UTF-8, UTF-16, UTF-32



ASCII Character	in Binary	in Octal	in Decimal	in Hex
space	00100000	040	32	20
(	00101000	050	40	28
)	00101001	051	41	29
*	00101010	052	42	2A
0	00110000	060	48	30
1	00110001	061	49	31
2	00110010	062	50	32
9	00111001	071	57	39
A	01000001	101	65	41
B	01000010	102	66	42
C	01000011	103	67	43
Z	01011010	132	90	5A
a	01100001	141	97	61
b	01100010	142	98	62
c	01100011	143	99	63
z	01111010	172	122	7A

History

- Guido Van Rossum (Netherland) invented/designed the Python programming language in 1989-1991.
- Succeeded from ABC Programming Language.
- Logos before 2006 and since 2006.

- Python was developed by Python Software Foundation.
- Useful resources at www.python.org

Releases

- Python 0.9.0 released in February 1991:
  - Classes with inheritance
  - Exception handling
  - Functions
  - Core data types of list, dict, str and so on
  - A module system borrowed from Modula-3; Van Rossum describes the module as "one of Python's major programming units"
- Python 1.0 was released in January 1994.
- Python 2.0 was released on October 16, 2000.
- Python 3.0 was released on December 3, 2008.

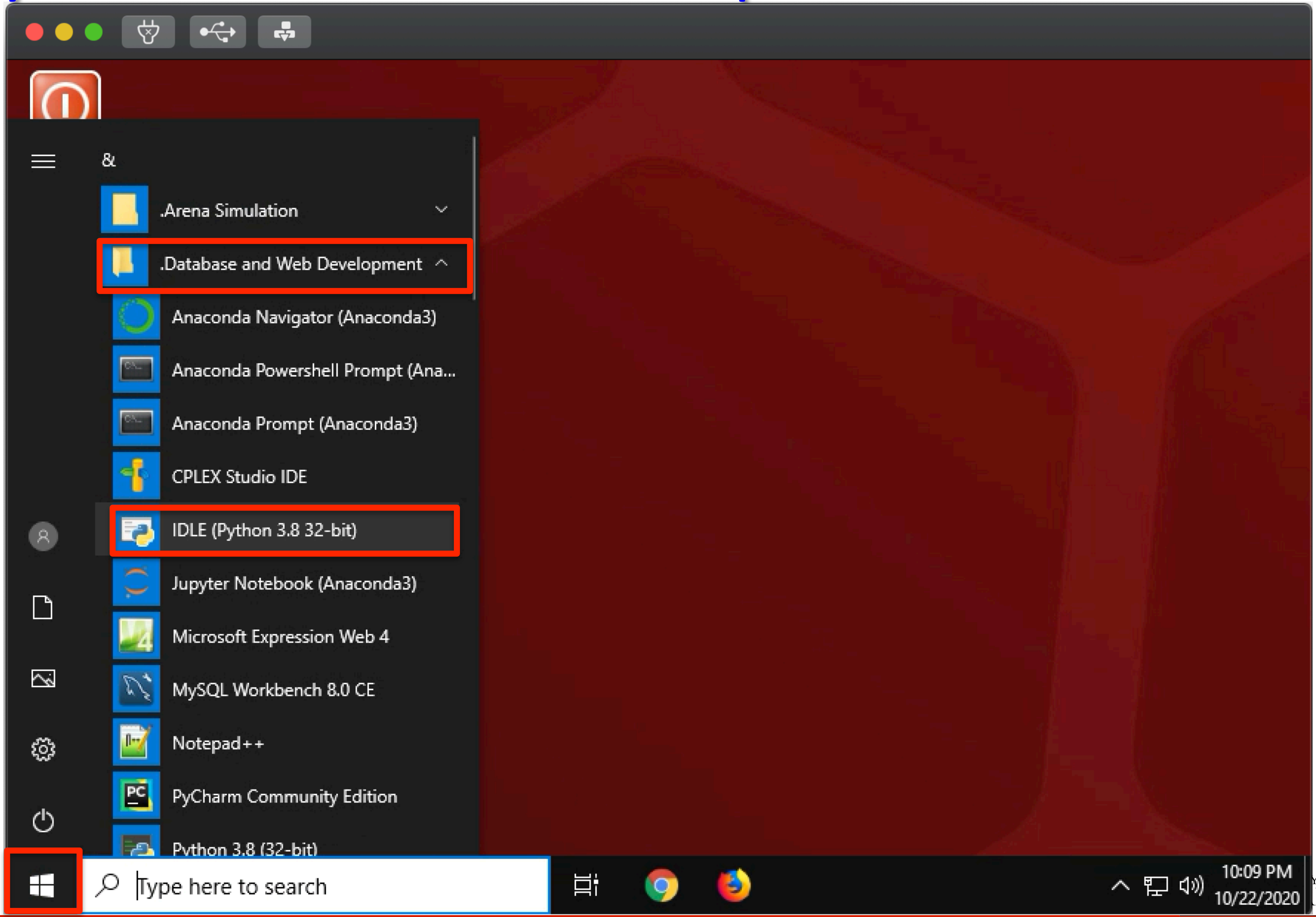
Releases

- Python 0.9.0 released in February 1991.
- Python 1.0 was released in January 1994:
  - Van Rossum stated "Python acquired lambda, reduce(), filter() and map(), courtesy of a Lisp hacker who missed them and submitted working patches"
  - The Modula-3 inspired keyword arguments (similar to Common Lisp's keyword arguments)
  - Built-in support for complex numbers
- Python 2.0 was released on October 16, 2000.
- Python 3.0 was released on December 3, 2008.

Releases

- Python 0.9.0 released in February 1991.
- Python 1.0 was released in January 1994.
- Python 2.0 was released on October 16, 2000:
  - Cycle-detecting garbage collector
  - Support for Unicode
  - Development process with a shift to a more transparent and community-backed process
- Python 3.0 was released on December 3, 2008:
  - Backwards-incompatible
  - Major features have also been backported to the backwards-compatible Python 2.6 and 2.7

Python on Student Desktop



&

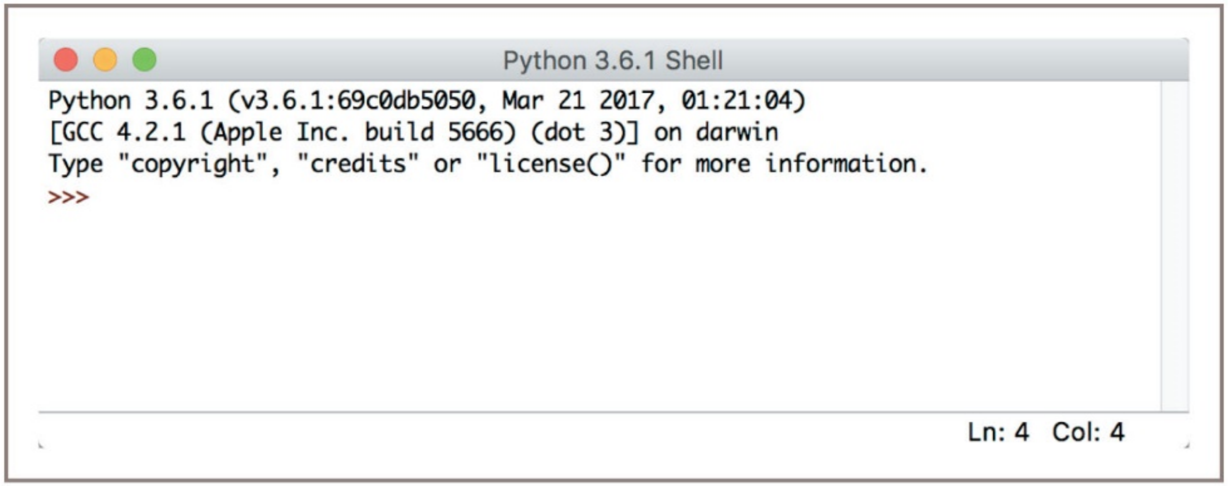
.Arena Simulation

.Database and Web Development

- Anaconda Navigator (Anaconda3)
- Anaconda Powershell Prompt (Ana...
- Anaconda Prompt (Anaconda3)
- CPLEX Studio IDE
- IDLE (Python 3.8 32-bit)
- Jupyter Notebook (Anaconda3)
- Microsoft Expression Web 4
- MySQL Workbench 8.0 CE
- Notepad++
- PyCharm Community Edition
- Python 3.8 (32-bit)

Running Python in the Interactive Shell

- Python is an **interpreted** language
- Shell **IDLE** is useful for:
  - Experimenting with short expressions or statements
  - Consulting the documentation
  - To quit, select the window's close button or press Ctrl+D



```
Python 3.6.1 (v3.6.1:69c0db5050, Mar 21 2017, 01:21:04)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>>
```

Figure 1-6 Python shell window

Input, Processing, and Output

- Programs usually accept inputs from a source, process them, and output results to a destination
- In Python, inputs are Python expressions or statements
- Outputs are the results displayed in the shell:

```
>>> 1 + 2
3
```

- Programmers can also force output of a value by using the **print** function:

```
>>> print("Hi there")
Hi there
```

Input, Processing, and Output

- The following example receives an input string from the user and saves it for further processing:

```
>>> name = input("Enter your name:")
Enter your name: Ken Lambert
>>> name
'Ken Lambert'
>>> print(name)
Ken Lambert
```

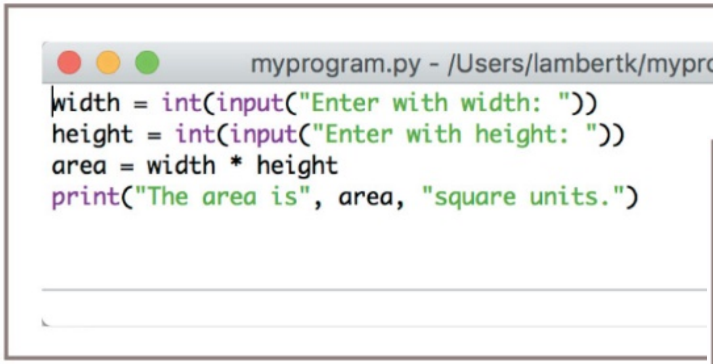
Input, Processing, and Output

- The **input** function always returns a string
- Strings that represent numbers must be converted from strings to appropriate number types:
  - **int** (for integers)
  - **float** (for floating-point numbers)

```
>>> first = int(input("Enter the first number: "))
Enter the first number: 23
>>> second = int(input("Enter the second number: "))
Enter the second number: 44
>>> print("The sum is", first + second)
The sum is 67
```

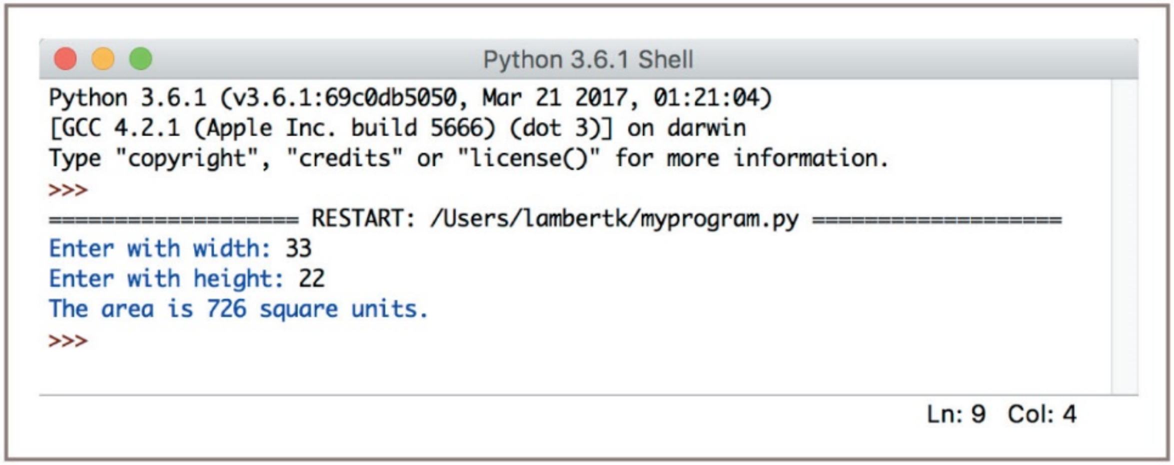
Editing, Saving, and Running a Script

- We can run Python program files, use **.py** extension, or scripts within IDLE or from the OS's command prompt
  - Create and edit new .py file
  - IDLE uses menu option F5 (Windows) or Ctrl+F5 (Mac)



```
width = int(input("Enter with width: "))
height = int(input("Enter with height: "))
area = width * height
print("The area is", area, "square units.")
```

Figure 1-7 Python script in an IDLE window



```
Python 3.6.1 (v3.6.1:69c0db5050, Mar 21 2017, 01:21:04)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>>
=================== RESTART: /Users/lambertk/myprogram.py ===================
Enter with width: 33
Enter with height: 22
The area is 726 square units.
>>>
```

Figure 1-8 Interaction with a script in a shell window

Detecting and Correcting Syntax Errors

- Programmers inevitably make typographical errors when editing programs, called **syntax errors**
- When Python encounters a syntax error in a program, it halts execution with an error message

```
>>> 1+
SyntaxError: invalid syntax
>>> print 1
SyntaxError: Missing parentheses in call to
'print'. Did you mean print(1)?
>>> 1: 1
SyntaxError: illegal target for annotation
>>>   print(1 + 2)
SyntaxError: unexpected indent
```

Detecting and Correcting Other Errors

```
>>> 1 / 0
Traceback (most recent call last):
   File "<pyshell#1>", line 1, in <module>
      1/0
ZeroDivisionError: devision by zero
>>> '1' + 1

...
TypeError: can only concatenate str (not
"int") to str
>>> int('i')

...
ValueError: invalid literal for int() with
base 10: 'i'
>>> print(one)

...
NameError: name 'one' is not defined
```