Chapter 1

An introduction to Python programming

Objectives

Applied

1. Use IDLE to test Python expressions and statements in the interactive shell.
2. Use IDLE to open, compile, and run a Python source file.

Knowledge

1. List three reasons why Python is a good first language for new programmers.
2. Describe a console program.
3. Explain how Python compiles and runs a program in terms of source code, bytecode, and the virtual machine.
4. Explain how main memory and disk storage work together when a program is running.

Objectives (cont.)

5. Distinguish between systems software and application software.
6. Distinguish between testing and debugging.
7. Distinguish between syntax errors and runtime errors.
8. Describe an exception.

Four general-purpose programming languages

- C++
- Java
- C#
- Python

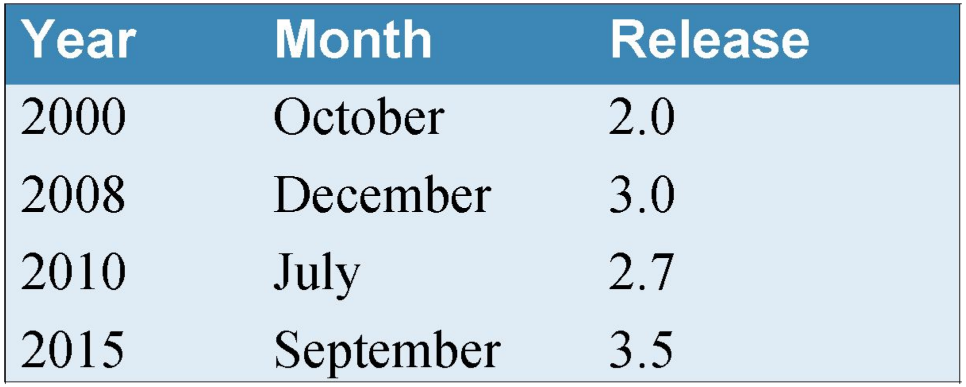
1990’s – Guido Van Rossum

simple

intuitive

powerful

The Python timeline



Year	Month	Release
2000	October	2.0
2008	December	3.0
2010	July	2.7
2015	September	3.5

Syntax differences between Python and Java

Some Java code

```
private static double calculateFutureValue(
    double monthlyInvestment,double monthlyRate, int months) {
    double futureValue = 0.0;
    for (int i = 1; i <= months; i++) {
        futureValue =
            (futureValue + monthlyInvestment) * (1 + monthlyRate);
    }
    return futureValue;
}
```

Python code that works the same

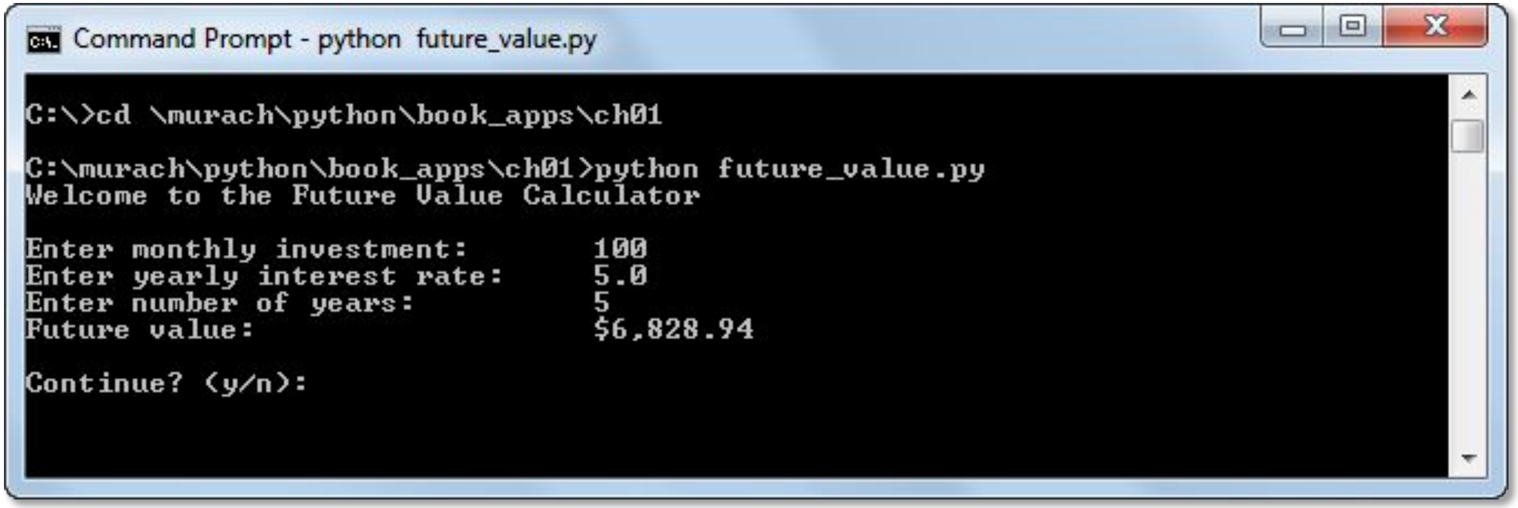
```
def calculateFutureValue(monthlyInvestment, monthlyRate, months):
    futureValue = 0.0
    for i in range(months):
        futureValue =
            (futureValue + monthlyInvestment) * (1 + monthlyRate)
    return futureValue
```

Why Python is a great first language

- Python has a simple syntax that's easier to read and use than most other languages.
- Python has most of the features of traditional programming languages. As a result, you can use Python to learn concepts and skills that apply to those languages too.
- Python supports the development of a wide range of programs, including games, web applications, and system administration.
- Python is used by many successful companies, including Google, IBM, Disney, and EA Games. As a result, knowing Python is a valuable skill.
- Python is *open source*. There are many advantages to being open source.

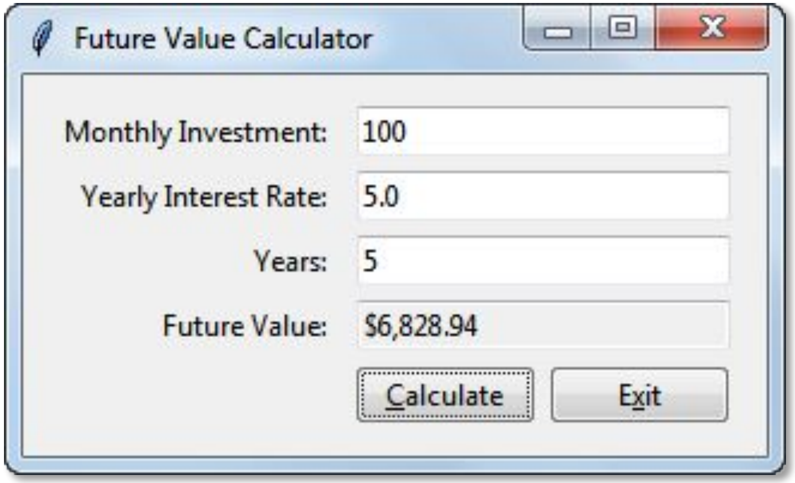
Types of Python Applications

A console application



```
C:\>cd \murach\python\book_apps\ch01

C:\murach\python\book_apps\ch01>python future_value.py
Welcome to the Future Value Calculator

Enter monthly investment:       100
Enter yearly interest rate:     5.0
Enter number of years:          5
Future value:                   $6,828.94

Continue? (y/n):
```

A GUI application



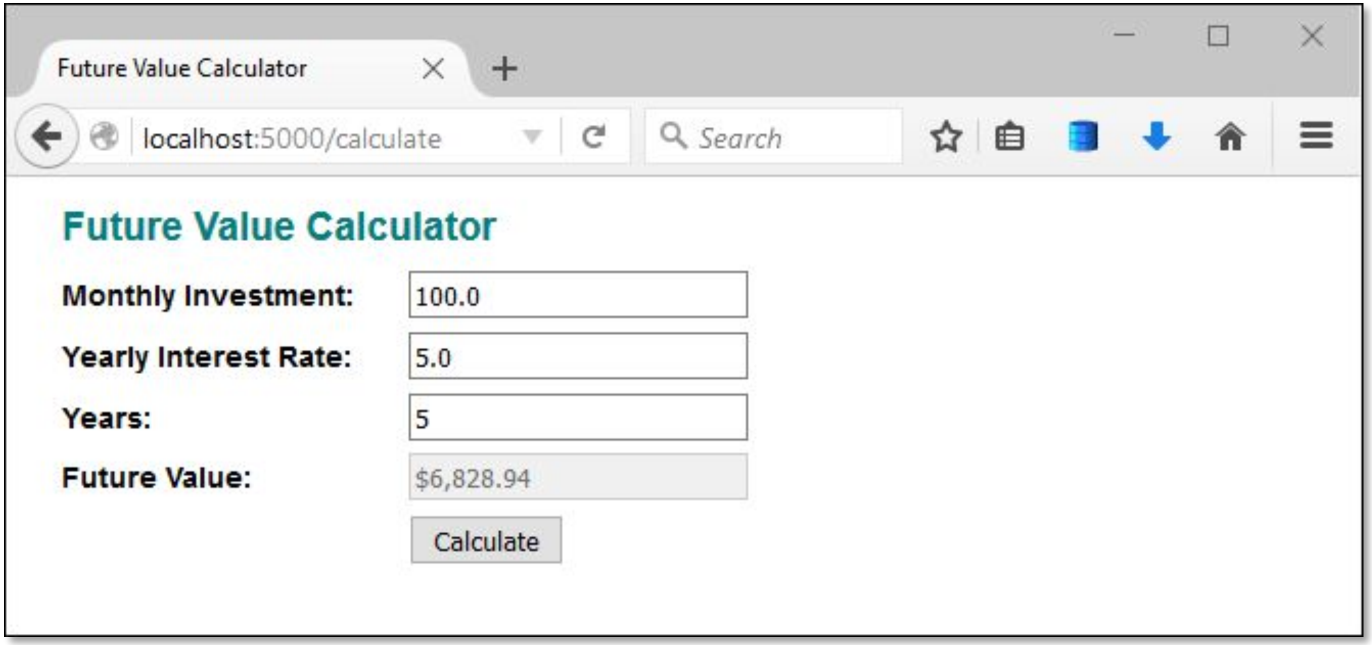
Monthly Investment: 100

Yearly Interest Rate: 5.0

Years: 5

Future Value: $6,828.94

A web application



Future Value Calculator

Monthly Investment:	100.0
Yearly Interest Rate:	5.0
Years:	5
Future Value:	$6,828.94

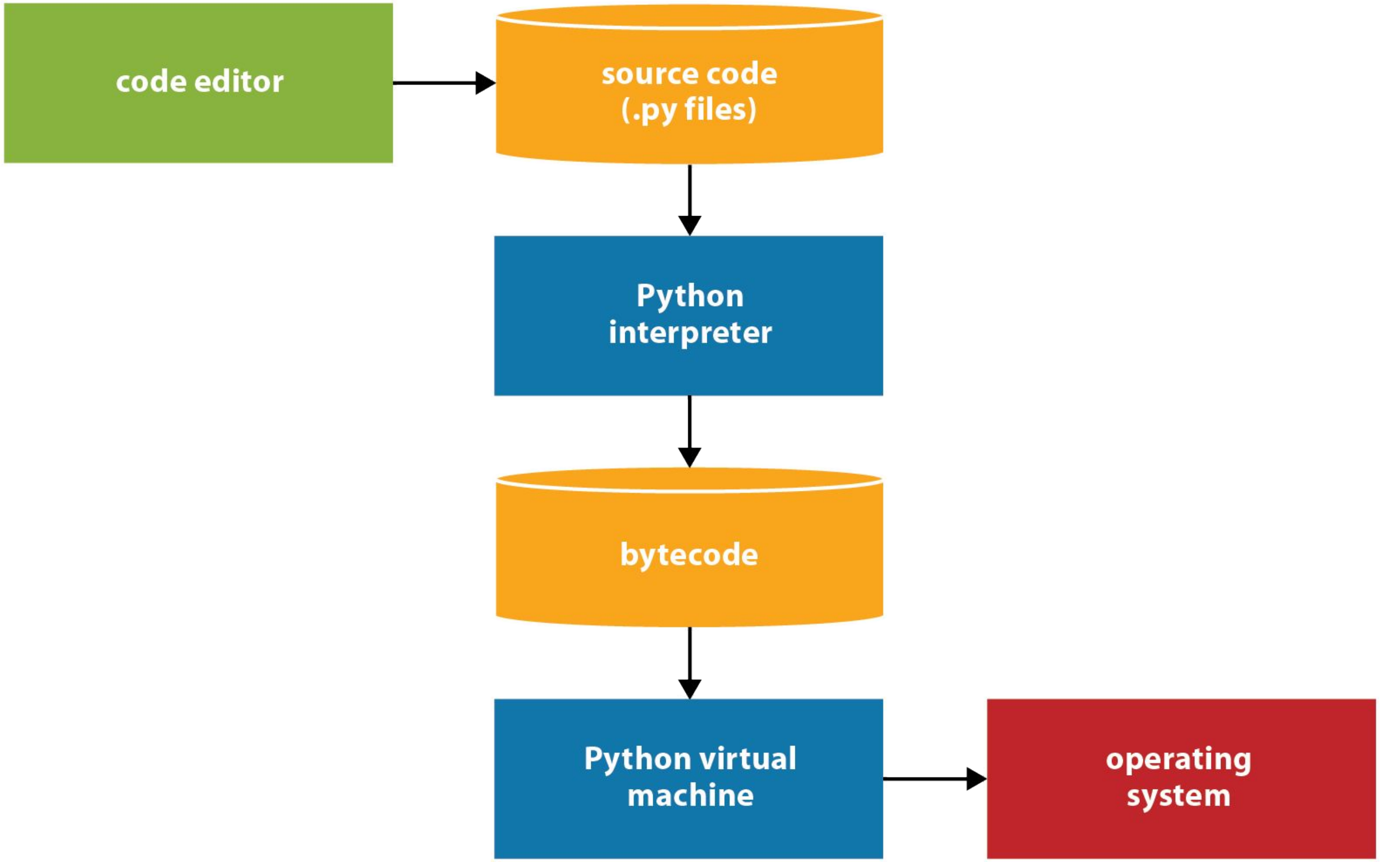
The source code for a console application

```
#!/usr/bin/env python3

import locale

# set the locale for use in currency formatting
result = locale.setlocale(locale.LC_ALL, '')
if result == 'C':
    locale.setlocale(locale.LC_ALL, 'en_US')

# display a welcome message
print("Welcome to the Future Value Calculator")
print()

choice = "y"
while choice.lower() == "y":
    # get input from the user
    monthly_investment = float(input("Enter monthly investment:\t"))
    yearly_interest_rate = float(input("Enter yearly interest rate:\t"))
    years = int(input("Enter number of years:\t\t"))

    # convert yearly values to monthly values
    monthly_interest_rate = yearly_interest_rate / 12 / 100
    months = years * 12
```

The source code for a console application (cont.)

```
    # calculate the future value
    future_value = 0
    for i in range(months):
        future_value = future_value + monthly_investment
        monthly_interest_amount = future_value * monthly_interest_rate
        future_value = future_value + monthly_interest_amount

    # format and display the result
    print("Future value:\t\t\t" + locale.currency(
        future_value, grouping=True))
    print()

    # see if the user wants to continue
    choice = input("Continue? (y/n): ")
    print()

print("Bye!")
```

How Python compiles and runs source code



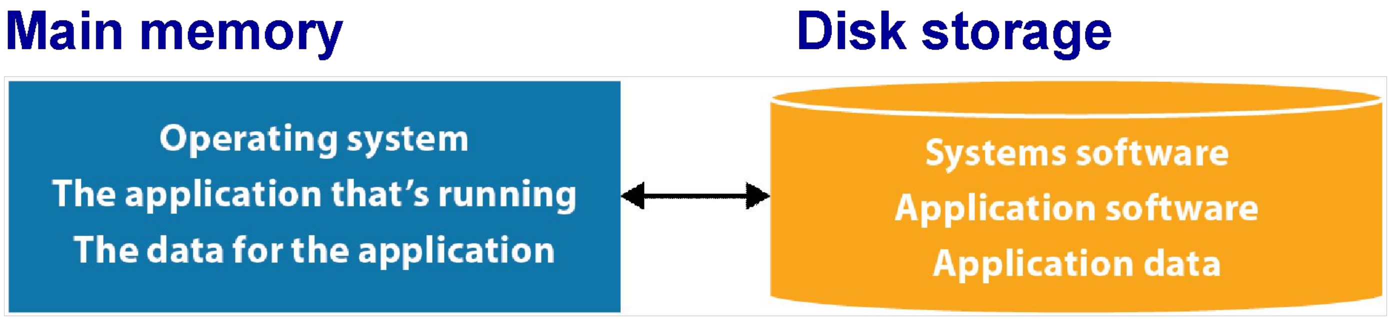
Procedure

Step 1 The programmer uses a *text editor* or *IDE* to enter and edit the *source code*. Then, the programmer saves the source code to a file with a .py extension.

Step 2 The source code is *compiled* by the Python *interpreter* into *bytecode*.

Step 3 The bytecode is translated by the Python *virtual machine* into instructions that can interact with the operating system of the computer.

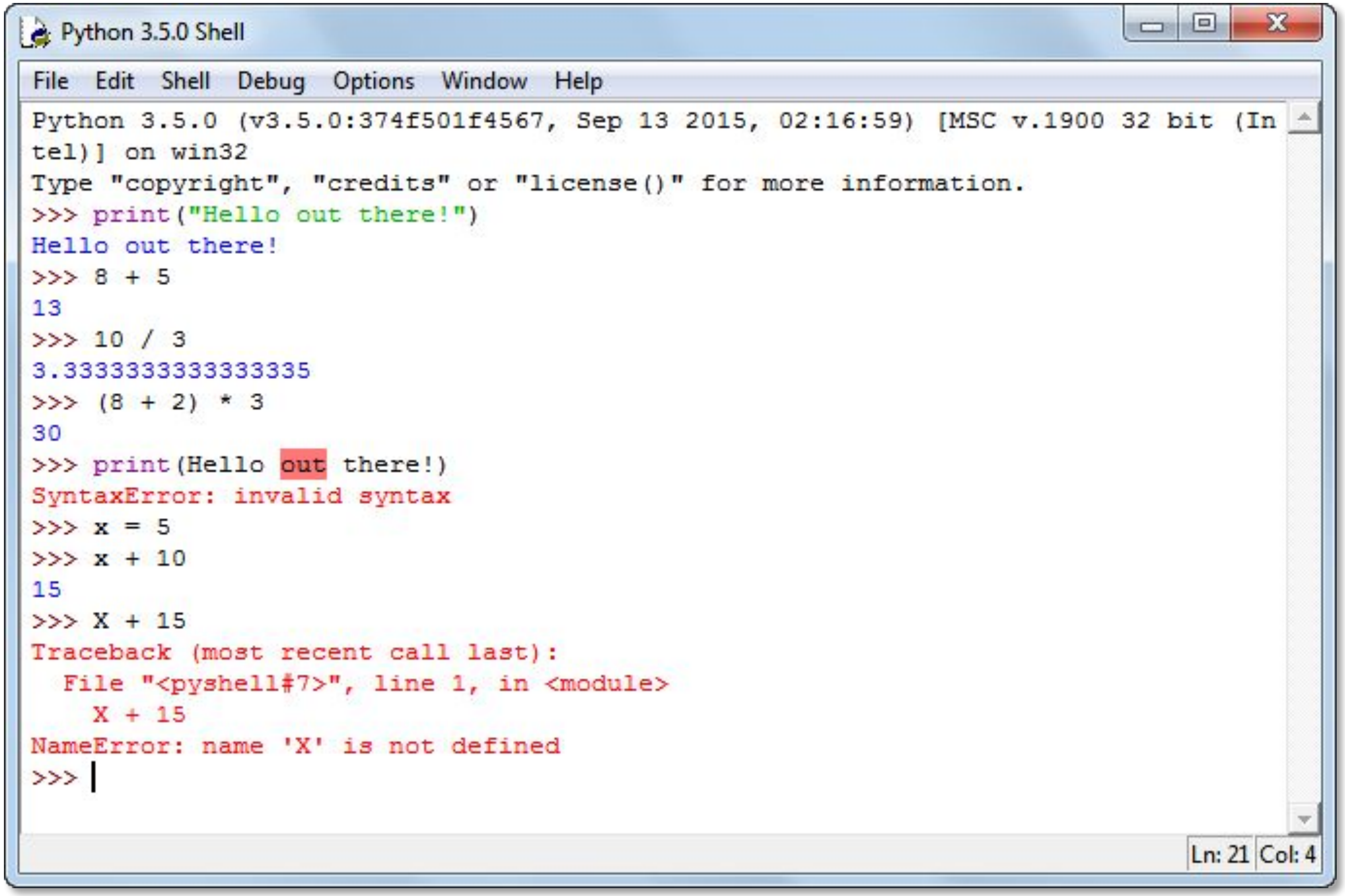
Main memory and disk storage as an application runs



How disk storage and main memory work together

- When you start the computer, it loads the operating system into main memory. Then, you use the features of the operating system to start an application.
- When you start an application, the operating system loads it into main memory. Then, it runs the application.
- As the application runs, it may read data from disk storage into main memory or write data from main memory to disk storage.

IDLE's interactive shell



```
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("Hello out there!")
Hello out there!
>>> 8 + 5
13
>>> 10 / 3
3.3333333333333335
>>> (8 + 2) * 3
30
>>> print(Hello out there!)
SyntaxError: invalid syntax
>>> x = 5
>>> x + 10
15
>>> X + 15
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    X + 15
NameError: name 'X' is not defined
>>> 
```

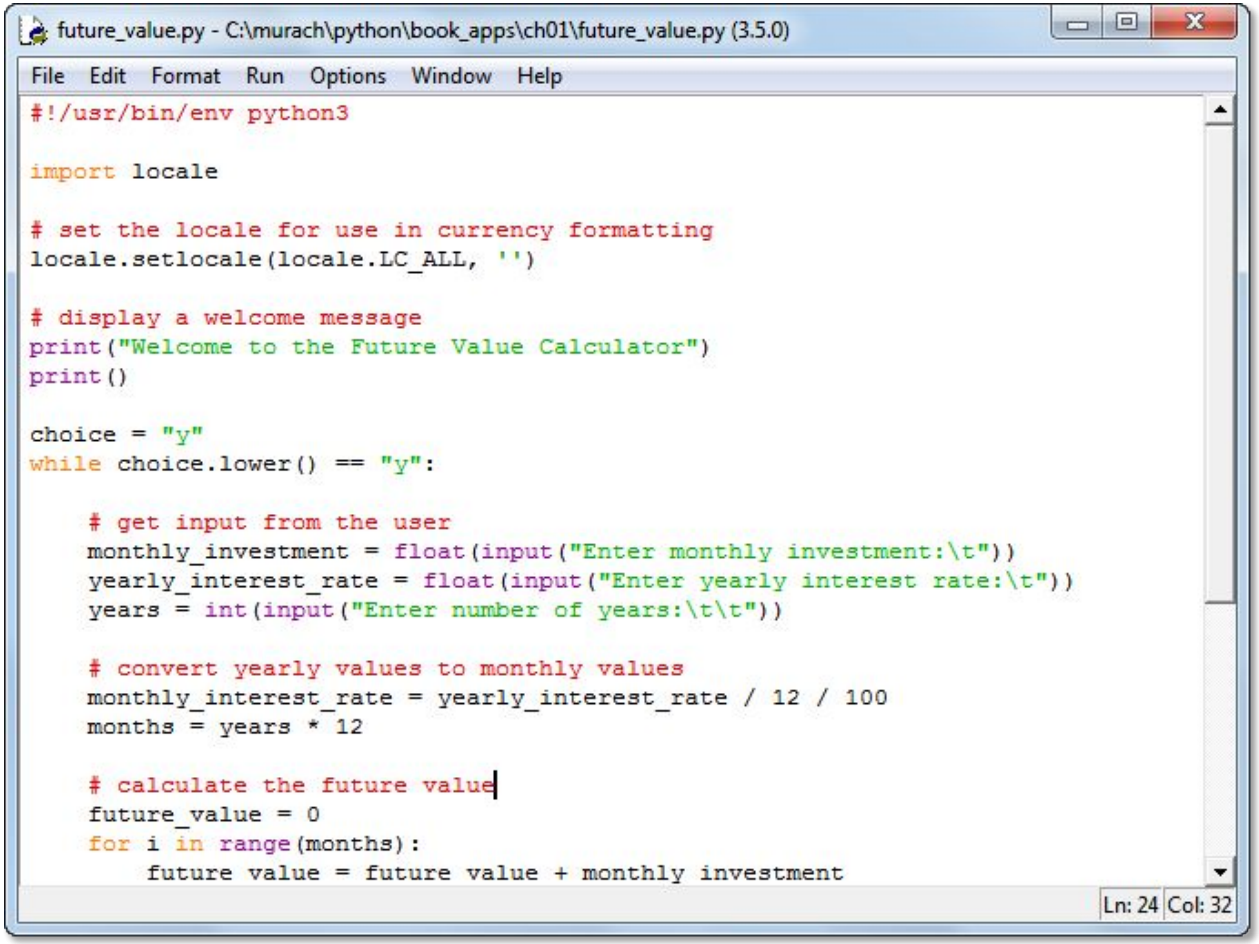
How to open, close, and restart the shell

- To start IDLE, use the features of your operating system. This opens an interactive shell.
- To close the interactive shell, click on its close button or select File→Close.
- To restart an interactive shell, select Run→Python Shell from an interactive shell window.

How to use the interactive shell

- Enter Python code after the >>> prompt. Then, press Enter.
- If you enter valid code that produces a result, the shell displays the result.
- If you enter an invalid code, the shell displays an error message.

IDLE’s editor with a source file displayed


future_value.py - C:\murach\python\book_apps\ch01\future_value.py (3.5.0)
File
Edit
Format
Run
Options
Window
Help
Ln: 24
Col: 32

```
#!/usr/bin/env python3

import locale

# set the locale for use in currency formatting
locale.setlocale(locale.LC_ALL, '')

# display a welcome message
print("Welcome to the Future Value Calculator")
print()

choice = "y"
while choice.lower() == "y":

    # get input from the user
    monthly_investment = float(input("Enter monthly investment:\t"))
    yearly_interest_rate = float(input("Enter yearly interest rate:\t"))
    years = int(input("Enter number of years:\t\t"))

    # convert yearly values to monthly values
    monthly_interest_rate = yearly_interest_rate / 12 / 100
    months = years * 12

    # calculate the future value
    future_value = 0
    for i in range(months):
        future_value = future_value + monthly_investment
```

How to create, open, save, and close source files

- Use IDLE’s File menu and common techniques for your operating system.

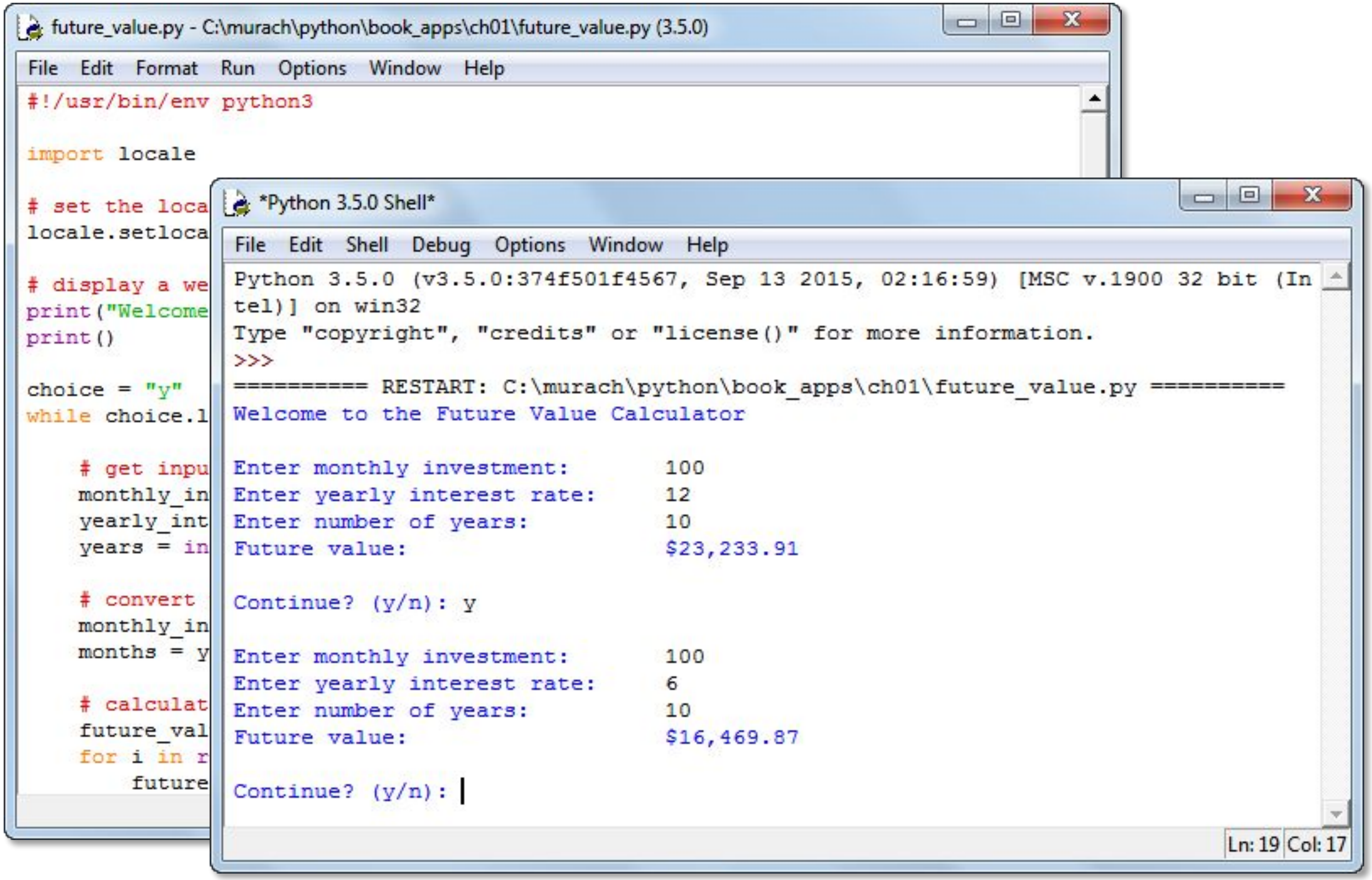
How to switch between source and shell windows

- Use IDLE’s Window menu.

How to enter and edit Python code

- Use IDLE’s Edit and Format menus, common editing keystrokes, and context menus.

A console application that’s being run in the shell


future_value.py - C:\murach\python\book_apps\ch01\future_value.py (3.5.0)
File Edit Format Run Options Window Help
Python 3.5.0 Shell
File Edit Shell Debug Options Window Help
Ln: 19 Col: 17

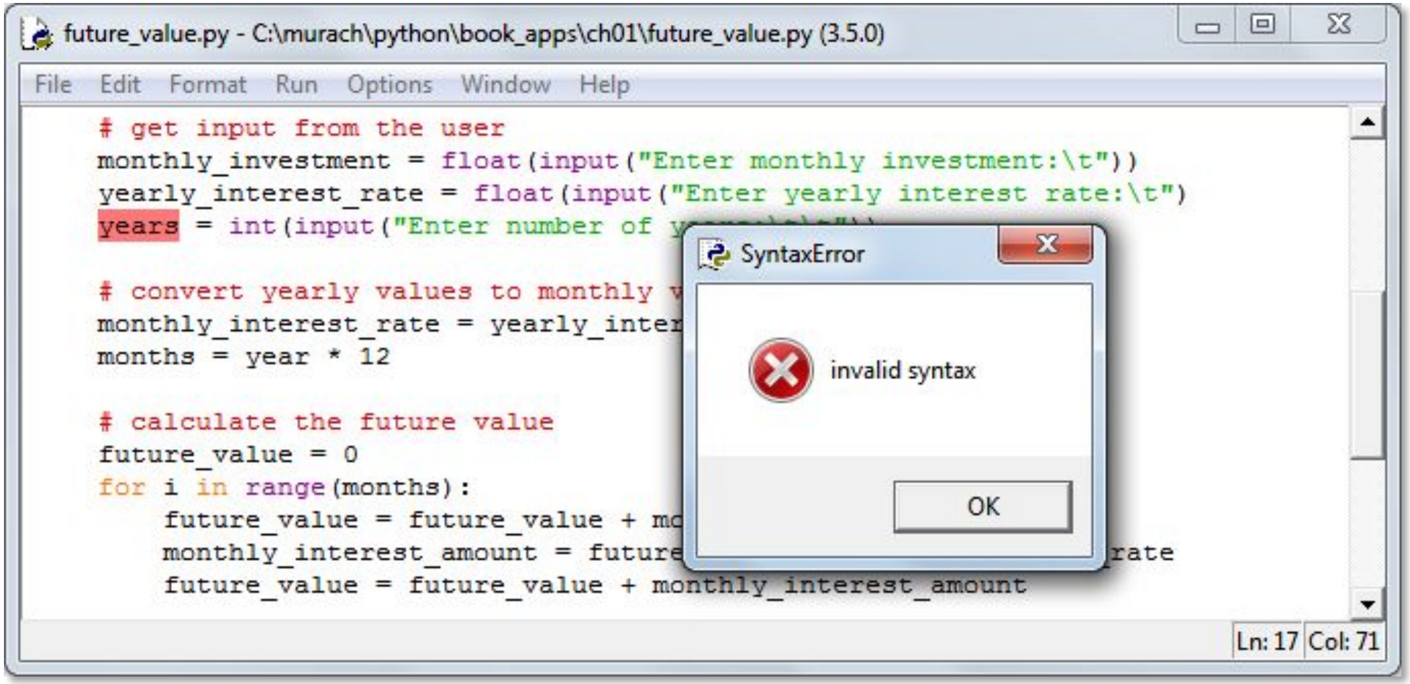
```
#!/usr/bin/env python3

import locale

# set the loca
locale.setloca

# display a we
print("Welcome
print()

choice = "y"
while choice.l

    # get inpu
    monthly_in
    yearly_int
    years = in

    # convert
    monthly_in
    months = y

    # calculat
    future_val
    for i in r
        future
```

```
Python 3.5.0 (v3.5.0:374f501f4567, Sep 13 2015, 02:16:59) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: C:\murach\python\book_apps\ch01\future_value.py ==========
Welcome to the Future Value Calculator

Enter monthly investment:       100
Enter yearly interest rate:     12
Enter number of years:          10
Future value:                   $23,233.91

Continue? (y/n): y

Enter monthly investment:       100
Enter yearly interest rate:     6
Enter number of years:          10
Future value:                   $16,469.87

Continue? (y/n):
```

How to compile and run a Python program

- From the editor window, press the F5 key or select Run→Run Module.
- If IDLE displays a dialog box that indicates that you must save the program first, click Yes to save it. Then, if the program doesn't have any errors, IDLE runs the program in the interactive shell.

A dialog box for a syntax error



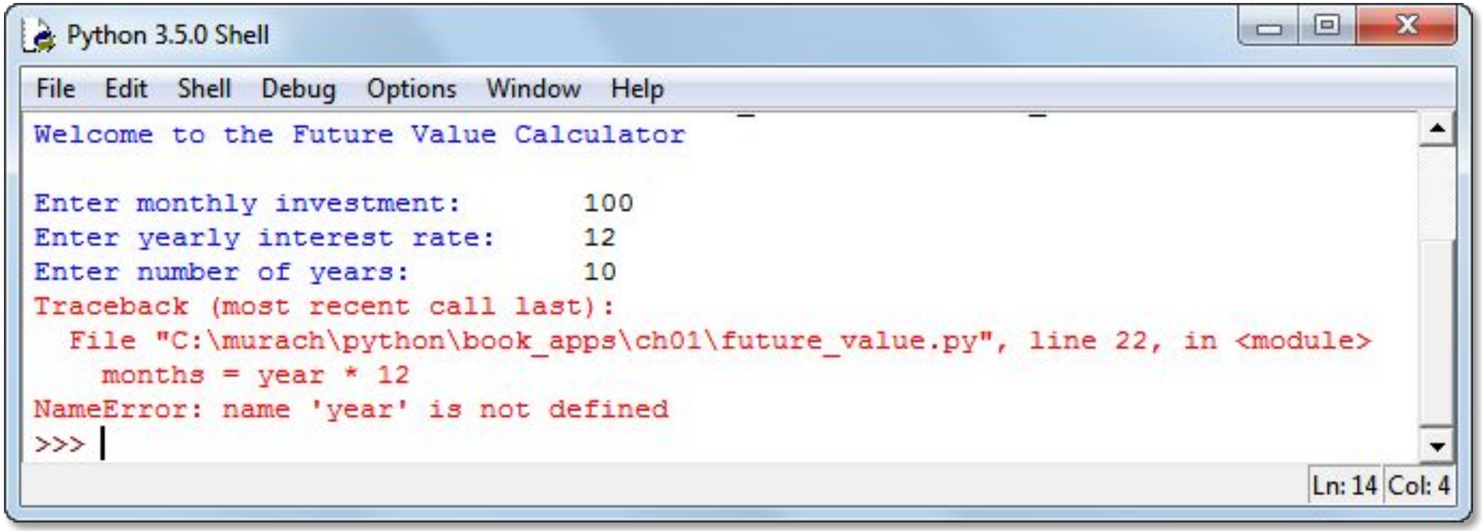
future_value.py - C:\murach\python\book_apps\ch01\future_value.py (3.5.0)

```
# get input from the user
monthly_investment = float(input("Enter monthly investment:\t"))
yearly_interest_rate = float(input("Enter yearly interest rate:\t")
years = int(input("Enter number of y

# convert yearly values to monthly v
monthly_interest_rate = yearly_inter
months = year * 12

# calculate the future value
future_value = 0
for i in range(months):
    future_value = future_value + mo
    monthly_interest_amount = future                          rate
    future_value = future_value + monthly_interest_amount
```

SyntaxError

invalid syntax

A message that’s displayed for a runtime error



```
Welcome to the Future Value Calculator

Enter monthly investment:        100
Enter yearly interest rate:      12
Enter number of years:           10
Traceback (most recent call last):
  File "C:\murach\python\book_apps\ch01\future_value.py", line 22, in <module>
    months = year * 12
NameError: name 'year' is not defined
>>> 
```