Errata/Corrigenda

Last updated:**$Date: 2008/06/11 08:51:12 $** *(PDFs are only from 2008 May 20... new ones pending)*

There will inevitably be errors when writing a book no matter how hard we try to keep it clean. In other words, "Don't (necessarily) believe everything you read!" A lot of hands touch the text before the book finally ends up in your hands, so here is where we try to clean it up. Even some of my own corrections found in the draft were not implemented due to lack of time or priority during the highly-compressed editing phase.

With regards to the corrections to the source code for the Examples in the book, MINOR problems do not prevent the script from running, but CRITICAL bugs means that the code printed as-is will not execute. We will provide changes or differences ("diffs") whenever possible. The diffs between the files are `-u` **context diffs** for those of you who know what they are. Because there is no other source for getting the code, all of the scripts for download have already been corrected.

If you find a new problem, send it to us at **`corepython (at) yahoo.com`**. Each errata item follows the format below... submit yours using the same format so that we can upload it faster to this page. We are also interested in hearing comments, suggestions, and general feedback, both good and bad. We want to continue improving the book for future readers and editions! MANY THANKS!!

Page(s) :: Section (may also have Figure, Table, or Example number) :: Subsection (if appl.) : *correction*

• Text which is in red indicates a change.
• Text which is in gray indicates an erratum that is of lower priority that is not normally displayed. Use the Toggle button below to swap between showing and hiding all minor errata.
• Errata denoted by **NEW** or **UPDATED** were added/changed between Feb 2008 and May 2008.

FrontMatter

xxx :: Preface : Chapter Guide : Chapter 8 : In the first sentence, `for`, `while`, and `if` should be in `Courier Bold` font, as all keywords are.

xxxiv :: Preface : Book Resources : In the second sentence, add a comma between "errata" and "source code".

Chapter 1

1-2 :: 1 : The book is misnumbered... page "1" is missing. The final page of the Preface is xxxvii and jumps immediately to page 2 of the main body of text.

11 :: 1.4 :: Build It Yourself : The third step, `make install` will install the Python you just compiled and set it as the default Python interpreter. Should you wish to install multiple versions of the interpreter or do not wish to replace the existing version, use `make altinstall`. For example, if you have an existing Python 2.4.3 installtion then build and

install 2.5 with `make altinstall`, calling `python` from the command-line still launches the 2.4.3 interpreter whereas to start 2.5, you would need to use `python2.5`.

17 :: 1.5.2 :: Unix : In the final paragraph of this section, we should also point out that executing `$ script.py` directly from the command-line only works if the current directory is part of your PATH environment variable. If it is not, you have to use `./` in front of the script name so that it can be found, e.g., `$ ./script.py`.

21 :: 1.5.4 :: Open Source : We inadvertently left out another popular IDE called **Stani's Python Editor** (SPE). The link to it: http://pythonide.stani.be/

NEW
24 :: 1.7 : The last sentence of the second paragraph needs to insert "of its", as in: "... Python has several of its own Web...". In the first sentence of the next paragraph, the word "shares" appears twice. Replace the second one with "with", as in: "... shares similarities with Python."

Chapter 2

33 :: 2.1 :: Core Note : In the final part of this sidebar, when dumping the previous expression using the underscore, the output should be in quotes: `'Hello World!'` (since the string itself is the expression and not its contents).

33 :: 2.1 : In the middle paragraph, delete the superfluous sentence that begins with, "Python is fairly flexible...."

34 :: 2.5 : Just about the Core Note, we should warn the reader by adding the following as the final sentence of that paragraph: "Readers should be cautioned that users need to enter a valid Python integer when prompted for a number as anything other than such will cause `int()` to raise an error because it cannot convert the input into a valid Python integer."

37 :: 2.5 : In the second sentence of the final paragraph at the bottom of the page, change the `--` to the unary negation sign `-` (single hyphen instead of two).

38-39 :: 2.6 : In the section preceding the call to `decimal.Decimal()`, we mention that "[you] must import the `decimal` module to use these types of numbers," but we didn't do so at the top of p. 39, so add the following at the top: `>>> import decimal`.

41 :: 2.10 : In the 2nd (and final) paragraph, change "***" to "device".

NEW
41 :: 2.11 : In the first sentence of the final paragraph, replace "then the statement" with "the". Also replace the word "otherwise" with "then".

42 :: 2.11 : In the first **`print`** statement at the top of the page, there should be a space to separate "at" and "least".

43 :: 2.12 : The sequence in the first **`for`** loop should not wrap.

47 :: 2.15 : How to Open a File : Replace the code snippet near the top of the page with:

```
filename = raw_input('Enter file name: ')
fobj = open(filename, 'r')
data = fobj.readlines()
fobj.close()
for eachLine in data:
    print eachLine,
```

52 :: 2.18 :: How to Create Class Instances : In the (first [John Doe]) sample output for calling `foo1.showname()`, the output should not have "`__main__.`" in it. In other words, it should just be:

```
>>> foo1.showname()
Your name is John Doe
My name is FooClass
```

Chapter 3

NEW
63 :: 3.1.4 :
- In the first sentence of the paragraph immediately beneath the Core Style sidebar, the word "and" has an extra space in it that should be removed.
- In the first sentence of the next paragraph, replace the last part of the sentence so that it reads as, "...more elegant and contributes to the goal of making the code easier to read.

68 :: 3.3.2 :: Table 3.1 : The keywords **`as`** and **`assert`** should be in **`Courier Bold`**, like they are in the equivalent table in Appendix A on p. 1022.

69 :: 3.3.3 : Add the following as the final sentence of this subsection: "Also see the Core Tip on p. 258 in Chapter 7."

80 :: 3.6 :: Example 3.1 : **`makeTextFile.py`**
- The code for line 10 is missing. It should be `fname = raw_input('Enter file name: ')` and indented at the same level line 11 below it.
- The text markers for lines 12 and 31, i.e, the "12" and "31" themselves, should **not** be in Courier/mono. Line 12 should be indented at the same level as line 14.
- Change line 6 to be a blank line (removing the `os.linesep()` reference). (Yes, both lines 6 and 7 are blank now.)
- Replace line 30 with `fobj.write('\n'.join(all))`

* Description of the 3rd and 4th changes for those who are curious: our goal for keeping the text file platform-independent was defeated by the `fopen()` C library call on Win32 platforms. For each NEWLINE (`\n`) character encountered, it is replaced by CARRIAGE RETURN/NEWLINE (`\r\n`). This means that the original program created text files with `\r\r\n` between lines, a minor text file corruption, *not* our original intention. The fix is to change the Python line terminator back to a solitary `\n` on line 30 and let the C library "do the right thing" for its respective platform. Also, rather than adding line terminators to each line in the file (string in the list) ourselves and sending those to the file with the `writelines()` method, the common best practice is to use the string `join()` method along with a single call to the file's `write()` method -- it performs faster and takes up less memory. We apologize for any confusion caused for readers new to Python and/or cross-platform issues.

(major patch [diff])

81 :: 3.6 : Core Tip : The word "lookup" should be in normal roman font, not `Courier`.

82 :: 3.6 : Lines 28-32 : Due to the change in the code for Example 3.1 above on p. 80, these two paragraphs need to be replaced with:

> Now that we have the entire contents in memory, we need to dump it to the text file. Line 29 opens the file for write, and line 30 writes each line to the file. Every file requires a line terminator (or termination character[s]). The `'\n'.join(all)` takes each line of `all` (list of input strings), and `join()`s them together into one large string delimited by the NEWLINE character. That single string is then written out to disk with the file object's `write()` method.

85 :: 3.8 : Exercises 3-6(b) : Shorten part (b) to read, "What do each contain after...."

85 :: 3.8 : Exercises 3-7 : The spacing gap between the 1st row of potential identifiers is greater than for the remaining rows. It should be reduced to match the rest.

Chapter 4

93 :: 4.3.2 : Core Note : Boolean values : The bullet points for the numeric types (4 thru 7) should be indented.

NEW
99 :: 4.5.2 : In the first paragraph of this page, replace "Sections 3.5.5-3.5.7" with "Section 3.5.4".

100 :: 4.5.2 : Core Note : Interning : Change `range(-1, 100)` to `range(-5, 257)`.

102 :: 4.6 : Table 4.4 : All instances of *obj* in the "Function" column should be in italics.

104 :: 4.6.3 : Core Note : It is likely that the back ticks (` `) will be deprecated in a future version of Python (say 3.0).

105 :: 4.6.4 : At the bottom of this page, there are 2 comments that both read as "`# new-style class`". Change the first one to "`# classic class`".

110 :: 4.6.5 :: Table 4.5 : `isinstance()` is missing from the built-in functions section of this table. The two possible syntaxes are: `isinstance(obj, cls)` and `isinstance(obj, (cls1, cls2,... clsN))`. The description should read, "Determines whether an object is of a certain type or class of types." Finally, it returns a `bool`. Also, the description for `type()` should be changed to: "Returns given object's type object".

NEW
113 :: 4.8.1 : The top of this page should refer instead to Section 4.9, not 4.8.

UPDATED
116 :: 4.9 :: `char` ~~`or byte`~~ : Remove "or `byte`" from both the section title and in the sentence. (Python 3.0 has a byte type.)

118 :: 4.10 : Exercise 4-9 : Change the middle pair of statements in the code snippet to:

```
c = 1000
d = 1000
```

Chapter 5

132-133 :: 5.5.3 : Modulus : We neglected to give a proper example of using the modulus operator. Add the following to the end of the subsection: "The modulus operator in Python is like that of C, the percent symbol (%), and here are some examples of using it:

```
>>> 5 % 2
1
>>> 8 % 4
0
>>> 6j-7 % 3
(-1+6j)
```

NEW
133 :: 5.5.3 :: Exponentiation : The final part of this subsection requires 2 changes:

- The last text sentence, beginning with, "Note that 1 / 4 as an integer...", needs to be replaced with, "Although 1 / 4 can be viewed strictly as an integer operation, (beginning with 2.2) Python correctly coerces both to `floats` so that the operation can succeed:".
- The code snippet immediately below should be replaced with:

```
>>> 4 ** -1
0.25
```

NEW
139 :: 5.6.2 :: Operational : It may be more clear in the `math.sqrt()` reference of the 2nd paragraph to use Python symbols instead of superscripts, so change it to: `math.sqrt(num.real ** 2 + num.imag ** 2)`.

NEW
143 :: 5.6.2 :: Table 5.6 : Add footnote "b" for `coerce()`: "Deprecated in Python 3.0." Place the footnote marker "b" just after the closing parenthesis in "... num2)".

NEW

147 :: 5.7.2 : In the final paragraph, add "example" between "the" and "below", as in: "... Note in the example below...."

NEW
151 :: 5.8 :: Table 5.9 : In the last row of the table (for the bitwise OR operator), there are tiny superscripted "v" and "vv" characters next to the bullets that should be removed. (Oddly, these do not show up in Table B.3 on p. 1025.)

153-154 :: 5.9 :: Exercises 5-9 and 5-17 : There is an extra space in exercise 17 between the `2**31` and the `-1`. It should be `2**31-1`. The extra space needs to be put between the `017` and the + in the 2nd example of exercise 9(a), as in `>>> 017 + 32`.

Chapter 6

UPDATED
167 :: 6.1.3 :: Table 6.3 :

- Footnote "d" in the first column marking `zip()` should be in roman font, not Courier/mono.
- For `sorted()`, add the following to the end of its description: "... built-in method (see Table 6.11 on p. 221)"

NEW
172 :: 6.3.2 :: Membership (**`in, not in`**) :

- The 2nd sentence uses a wrong word and should read as: "... if that substring appears in the...."
- The 3rd sentence uses a wrong word and should read as: "... used to determine where a substring is...."

173 :: 6.3.2 :: Membership (**`in, not in`**) : The `string` data attribute names have changed. `string.uppercase, string.lowercase, string.letters` all have Unicode characters now. To get the pure ASCII versions, use `string.ascii_uppercase, string.ascii_lowercase, string.ascii_letters`.

173 :: 6.3.2 :: Example 6.1 : Line 1 is missing the (root directory) first forward slash in the start-up directive; it should be, `#!/usr/bin/env python`, and line 5 should refer to `string.ascii_letters`.

173-174, 246, 1014 :: 6.3.2, 6.2.2, Appendix A :: Example 6.1, Exercise 6-2, Answer to Exercise 6-2 : An extraordinary amount of emphasis is placed requiring possible identifier strings to be longer than one character in length when it is not absolutely necessary (as you are probably aware). The first part of Exercise 6-2 is not a "trick question" nor is changing the length check artificial. More of an emphasis should be placed on the 2nd half of the exercise instead. The solution given in Appendix A can be replaced by a simpler solution (which can be downloaded here).

```
#!/usr/bin/env python

from keyword import kwlist
import string

ALPHAS = string.ascii_letters + '_'
NUMS = string.digits

def main():
    print 'Welcome to the Identifier Checker v2.0'
    myInput = raw_input('Identifier to test? ').strip()

    if len(myInput) == 0:
        print "ERROR: no identifier candidate entered"
        return

    if myInput in kwlist:
        print "ERROR: %r is a keyword" % myInput
        return

    alnums = ALPHAS + NUMS
    for i, c in enumerate(myInput):
        if i == 0 and c not in ALPHAS:
            print 'ERROR: first symbol must be alphabetic'
            break
        if c not in alnums:
```

```
                print 'ERROR: remaining symbols must be alphanumeric'
                break
        else:
            print "okay as an identifier"

    if __name__ == '__main__':
        main()
```

NEW
184 :: 6.4.4 : This section refers to Section 6.7.4 as that for Unicode when it should refer to Section 6.8 instead.

NEW
186 :: 6.5.3 :: `raw_input()` : Remove the first part of the first sentence in the final paragraph of this subsection. It should read as: "Strings in Python do not...."

UPDATED
188-190 :: 6.22 :: Table 6.6 :

- *`string`*`.center()`, *`string`*`.ljust()`, and *`string`*`.rjust()` methods are missing a parameter. They should be updated like this:

```
string.center(width, fillchar=' ')
string.ljust(width, fillchar=' ')
string.rjust(width, fillchar=' ')
```

Each description should be appended with, "; *`fillchar`* is the padding character". All 3 entries should have a (new) footnote "g": "New or changed in in Python 2.4."

- The entry for *`string`*`.join(`*`seq`*`)` should be: "Merges (concatenates) a sequence *`seq`* of strings into a single string delimited by separator *`string`*."
- The *`string`*`.rsplit(`*`str`*`=' ', `*`num`*`=`*`string`*`.count(`*`str`*`))` method is missing: "Same as `split()`, but search backward in *`string`*." It should have a (new) footnote "g": "New or changed in in Python 2.4."
- The signature for *`string`*`.splitlines()` is incorrect. It should be: *`string`*`.splitlines(`*`keepends`*`=False)`. The description should be revised as: "Splits *`string`* at all line breaks and returns a list of each line without them unless *`keepends`*`=True`."

NEW
193-194 :: 6.7.1 : There is case inconsistency in the hexadecimal value for zero. Change one occurrence of "0X00" on each page to "0x00".

NEW
195 :: 6.7.3 : The first sentence should refer to "Section 4.8.2".

199 :: 6.8.3 : In the first complete paragraph (starting with, "In order to make Unicode...), the last four sentences (starting with, "Both have string...") need to be replaced as follows:

> ... Both behave similarly. Due to many issues users are having with regards to moving between Unicode and ASCII strings, Python 3.0 will adopt a new standard where all strings are Unicode. A new `bytes` type will be added for those needing to manipulate binary data and encoded text.

NEW
200 :: 6.8.4 : The 1st sentence of the 2nd paragraph should say, "... may use one or two bytes to represent...."

NEW
202 :: 6.8.5 : The filename "unicode.txt" in the 2nd paragraph should be in Courier/mono font.

NEW
202 :: 6.8.6 : The 4th bullet point should say, "... encoding and decoding of Unicode strings...."

NEW
203 :: 6.8.6 : The 3rd bullet point in the middle of the page has 2 typos:

- "Plane" should be "Plone"

- Missing a comma right after "Django"

204 :: 6.8.8 :: Unicode Type : "basestring" should be in Courier/mono font.

NEW
206, 351 :: 6.9, 9.10 :: Tables 6.10 and 9.7 : Replace "`c/StringIO`" with "`{c,}StringIO`"

NEW
207 :: 6.9 :: Table 6.10 : Remove the row for the no-longer-available `rotor` module.

NEW
214-215 :: 6.12.2 :: Concatenation (+) : Replace both references to Section "6.13" to "6.14".

NEW
218 :: 6.13.2 :: `max()` and `min()` : The code snippet at the top of the page should be replaced with:

```
>>> max(str_list)
'over'
>>> max(num_list)
619000.0
>>> min(str_list)
'candlestick'
>>> min(num_list)
-2
```

219 :: 6.13.2 :: `sum()` : In the code snippet on the line that makes a call to the `reduce()` function, add a comment so the line looks like this:

```
>>> reduce(operator.add, a) # see Section 11.7.2
```

NEW
219 :: 6.13.2 :: `list()` and `tuple()` : The 2nd section has a wrong word: "list" should be "tuple", as in: "... the list you passed to `tuple()` does not turn into a tuple, and the tuple...."

UPDATED
232 :: 6.16 :: How to Create and Assign Tuples : Change "empty tuples" in the first paragraph to "tuples with only one element" -- see the example below with `emptiestPossibleTuple` which is also inappropriately-named since it is a tuple of a single element, not very empty at all -- change the name to `singleItemTuple`. Empty tuples are specified with just a pair of parentheses: ( )

237 :: 6.18.3 : The colons (":") in `foo1()`, `foo2()`, and `foo3()` serve only as ellipses to indicate some random code for each of these functions... they are not valid Python syntax.

240 :: 6.19 :
- Fix this typo in the final paragraph of this section at the top of the page: "... also contains the same **for** sequence types." (This refers to the functional equivalents to sequence type operators.)
- The final three sentences (beginning with, "Finally, the...") need to be replaced as follows:

> ... including sequence types. The `re` module is for regular expressions, the `StringIO` modules allow you to manipulate strings like files, `textwrap` is useful for wrapping/filling text fields, and `collections` has high-peformance specialized containers.

NEW
243 :: 6.21 : In the last sentence, add some text to explain the table shading: ".. function and methods (shaded in order of precedence) for sequence...."

247-248 :: 6.22 :: Exercise 6-7 and Example 6.4 :
- Example 6.4 is really 6.5. This should be fixed in the Exercise as well as at the top of p. 248.
- On lines 10-27, change all occurrences of `fac_list` to `non_fac_list`.
- On lines 11 and 27, change `'non_fac_list'` to `repr(non_fac_list)` (removing single quotes).

250 :: 6.22 :: Exercise 6-18 : This exercise requires corrections and should be replaced as follows:

6-18. `zip()` Built-in Function. In the `zip()` example of Section 6.13.2, what Python object does `zip (fn, ln)` return that can be iterated over by the **`for`** loop?

Chapter 7

NEW
256 :: 7.1 :

- Move the entire paragraph starting with, "Dictionary `dict1` defined above...", to near the top of p. 255 immediately preceding the paragraph beginning with, "In Python version 2.2 and newer...".
- In the 2nd-to-final paragraph, delete the middle portion of the sentence, as in: "... has a specific key is to use ~~the dictionary's `has_key()` method, or better yet,~~ the...".
- The next sentence should be changed this way: "... method is obsolete and removed in Python 3.0, so it is...".
- The next paragraph is reduced down to a single sentence:
  1. The first sentence and part of the second should be removed so that the only sentence in this paragraph now starts with: The **`in`** and **`not in`** operators...". It should end with a colon (":") instead of a period (".") because...
  2. The parenthesized sentence that begins with, "(In Python version preceding...", should be moved to the bottom of the Core Note on p. 93.
- Remove both comments in the code snippet at the bottom of p. 256 and fix the reference in the 2nd (from `dict` to `dict2`) so that it reads as:

```
>>> 'server' in dict2
False
>>> 'name' in dict2
True
```

NEW
258 :: 7.1 :: Core Tip : Change part of the last sentence to read as: "Do NOT create variables with built-in names like:...".

NEW
259 :: 7.2.2 :: Dictionary Key-Lookup Operator : In the code snippet at the end of this subsection, change the comments so they read as:

```
d[k] = v    # set value in dictionary
d[k]        # lookup value in dictionary
```

We should add one final paragraph in this section: "Although we can use `d[k]` to perform looking up a value in a dictionary using its key, this is not the safest way of doing so, as users will encounter `KeyError` if a key is not found in the dictionary. Better alternatives include using a dictionary's `get()` method -- this is an improvement because it allows the programmer to specify a default value to return in case the key doesn't exist. Another alternative is the dictionary `setdefault()` method -- similar to `get()`, it doesn't just return the default value but also adds that value to the dictionary using the (previously non-existent) key, hence its name of "setdefault".

NEW
261 :: 7.3.1 :: Dictionary Comparison Algorithm :

- The first sentence that begins at the top of this page has swapped the references for `dict1` and `dict2`. It should read as: "... when we get to the `'port'` key, `dict1` is deemed ... that of `dict2`'s `'port'` key...."
- The final sentence in this section should state that, "[our] final example reminds us that...".

266 :: 7.4 :: Table 7.2 :

- The second entry should be *`dict`*`.copy()` not *`dict`*`.clear()`.
- These should be in Courier/mono font:

  1. The "*" in *`dict`*`.iter*()` (1st col)
  2. The "`None`" in *`dict`*`.setdefault()` (1st col)

3. The "*dict*" in `dict.has_key()` (2nd col)
4. Swap the rows for `dict.keys()` and `dict.iter*()` so as to correctly alphabetize by method name.

Many of these errors are actually correct in (the equivalent entries in) Table B.9 on p. 1035.

- **NEW** Add a new footnote "f" for the `dict.has_key(key)` entry: "Deprecated in Python 2.2 and removed in Python 3.0; use **`in`** instead."
- Change footnote "b" to refer to Section 6.20
- For `dict.items()`, `dict.keys()`, `dict.values()`, replace "a list" in all 3 descriptions with "an iterable", and create a new footnote "g" for all 3 that reads as: "The iterable is a set view starting in Python 3.0 and a list in all previous versions."
- Change footnote "d" to read as: "New in Python 2.2 but removed in 3.0 because their non-iterator equivalents will return an iterable beginning with that release (also see "g")."
- Finally, ensure all necessary changes are made to Table B.9 on p. 1035.

UPDATED
267 :: 7.4 :

- In the first sentence of the second paragraph (right below the code snippet), "directory" is the wrong word and should be changed: "... add the contents of one dictionary to another."
- In the paragraph above the final code snippet replace the reference to Section 6.19 to 6.20.

NEW
270-271 :: 7.5.2 :: Example 7.1 :

- This source code is missing all boldfacing of Python keywords, i.e., all occurrences of **`def, while, if, continue, else, break, print, not, try, except, in`** (outside of quotes, i.e., "try"). Specifically, they are lines 5, 7, 9, 11-13, 17, 21-24, 26, 35, 38-39, 41, 43-46, 49-51, and 53.
- Lines 34, 49, 50, 51 should be indented to the same level as line 35.
- Lines 41, 43, 44, 46 should be indented to the same level as line 39.

NEW
274 :: 7.6 : The top of the page features a sentence ending in "... integrated into Python 2.4." Insert this sentence immediately following that one: "The `sets` module is deprecated in Python 2.6." Also, change the text in the graphic logo in the margin to "2.3-2.6".

NEW
276 :: 7.7.1 :: Membership (`in, not in`) : In the 1st sentence, both "in" and "not in" should be in Courier/mono bold.

NEW
286 :: 7.12 :: Exercise 7-5 : In exercise 5(c), the parenthesized list should remove reference to the now-obsolete `rotor` module and add in a few new ones. It should be replaced with: "(see the `getpass`, `md5`, `crypt` [Unix-only], `hashlib` [2.5], and other cryptographic modules)."

288-289 :: 7.12 :: Exercises 7-13 - 7-15 : In all 3 problems, all references to sets `A` and `B` should be in Courier/mono font, including any operators in between.

Chapter 8

295 :: 8.4 : Guido's name was messed up in the 1st and 3rd paragraphs during the final editing phase of the manuscript. Read as only "Guido" or "van Rossum", not "van Rossum Guido"[sic].

303 :: 8.6.5 : There is a misspelling of "returning" in the final sentence of this paragraph. Also, so readers don't have to hunt for the reference, the final sentence should end with, "... similar to views as discussed in the previous chapter at the end of Section 7.4."

NEW
306 :: 8.8 :

- In the code snippet, remove "`# (or valid == 0)`" on the line that starts with, "**`if not`** `valid:`".
- The last sentence of this section should be edited as follows: "... variable remains `False`, which presumably...."

NEW
307 :: 8.10 :: Example 8.1 : Line 1 of the code should be in Courier/mono

NEW
308 :: 8.10 : Remove the last sentence in the paragraph starting with, "Likewise, a...".

NEW
317 :: 8.13 :: Cross-Product Pairs Example : Change "like" to "unlike".

NEW
323 :: 8.15 :: Exercise 8-13 : Change the beginning sentence of this exercise to, "In Section 8.6.2, we...."

Chapter 9

327 :: 9.2 :: Table 9.1 : Footnote "A" : Universal NEWLINE Support (PEP 278) was added in Python 2.3 not 2.5.

NEW
334 :: 9.3.6 : A colon (:) is missing after "doublespaced" near the top of the page. It should read as: "... display would be doublespaced: one NEWLINE...."

338 :: 9.6 : In the first code snippet, the call to `str()` to convert `sys.argv` to a string is superfluous. **`print`** does it automagically.

NEW
341 :: 9.7 :: Table 9.5 :: File Descriptor Operations : In the description for `open()`, the "s" in the final word should be a closing square bracket, as in: "... built-in function]"

Chapter 10

NEW
362 :: 10.2 :: `ZeroDivisionError` : "Our examples above used integers, but in general...."

NEW
374 :: 10.3.6 : In the 1st sentence of the 2nd-to-final paragraph on this page, the Chapter 13 title is wrong, so we can just replace it with a comma, as in: "... discover in Chapter 13, the special...."

376 :: 10.3.7 :: Example 10.1 : **`cardrun.py`**
The original installed script was out-of-date. Please download a new one if your file is older than 2007 Jan 1. We also installed the example `carddata.txt` file as listed on p. 375 as well as updated the alternate version in the `alt` directory. (critical patch)

377 :: 10.3.7 :: Line-by-Line Explanation :: Lines 36-37 : The `cat cardlog.txt` command-line has an extra space that should not be there.

NEW
379 :: 10.3.9 : The `except` in the 1st sentence of the final paragraph of this section should be in Courier/mono **BOLD**.

380-381 :: 10.3.10 : In the pair of examples using **`try-finally`**, if the `open()` call fails, `ccfile` will be undefined, resulting in a `NameError` exception being thrown, an insult to injury on top of the `IOError` exception we received for the failure in `open()`. The usual idiom to avoid this situation is by setting `ccfile = None` before the **`try-finally`**, then add an **`if`** `ccfile:` before calling `close()`, as in this replacement for the code on p. 380:

```
ccfile = None
try:
    try:
        ccfile = open('carddata.txt', 'r')
        txns = ccfile.readlines()
    except IOError:
```

```
            log.write('no txns this month\n')
    finally:
        if ccfile:
            ccfile.close()
```

and this replacement for p. 381:

```
    ccfile = None
    try:
        try:
            ccfile = open('carddata.txt', 'r')
            txns = ccfile.readlines()
        finally:
            if ccfile:
                ccfile.close()
    except IOError:
        log.write('no txns this month\n')
```

384 :: 10.4.2 :: Context Expression, Context Manager : Italicize *`with_suite`* in the only paragraph on this page.

386, 394 :: 10.5, 10.8 : In the final paragraph for section 10.5, we describe the obsolescence of string exceptions. The raising exceptions in Python 2.5 resulting in a warning can be further clarified that it will no longer be allowed in 2.6. Similarly, the catching of string exceptions results in a warning beginning in 2.6 but will not be allowed in 2.7.
This clarification can also be carried to the second-to-last paragraph of section 10.8. It is erroneously stated that raising of string exceptions is not allowed in 2.5, which it is allowed but generates a warning. Similarly for the catching of string exceptions, it's really only disallowed starting in 2.7; in 2.6, a warning is generated.

387-389 :: 10.6.1 including Table 10.1 : Italicize all the variables, such as *`with_suite, args, traceback, instance, exclass, tb`*.

390 :: 10.7.1 : With regards to the `__debug__` system variable, after Python 2.2, rather than having values of 1 and 0, it is a Boolean, thus have values of `True` and `False`, respectively.

391-393 :: 10.8 : Table 10.2 : Various editing errors exist in the table of standard exceptions:
- (391) The 2nd entry for `SystemExit` should have a footnote of "b" just like it is in the entry above. The entire row should also be grayed out, indicating the change that occurred back when Python 2.5 was released.
- (392) (The 2nd entry for) `KeyboardInterrupt` should have a footnote of "c" just like it is in the entry on the previous page. The entire row should also be grayed out, indicating the change that occurred back when Python 2.5 was released.
- (392) `TabError` should be indented... it is derived from `IndentationError`.
- (393) `UnicodeTranslateError` should have a footnote of "i" (not "f").
- (393) `PendingDeprecationWarning` should be one word.
- (393) Footnotes "b" and "c": the word "subclassed" should be "derived from".
- (393) Footnote "f" is a duplicate of "h" and SHOULD BE REMOVED. All remaining footnotes and corresponding tagged text should be adjusted accordingly.

UPDATED
396-397 :: 10.9 :: Example 10.2 :

- Line 77 should read as: "`file = tempfile.mktemp()`
- In (most releases of) Python 2.x, `file()` is a factory function, so rename all variables named `file` between lines 20-95 to `fn`.
- On line 103, replace `'deli'` with `eachHost` (remove single quotes too)
- Enter your own hostnames on line 101 rather than using our test lab machine names. (critical patch)

NEW
404 :: 10.13 : Table 10.3 : Add a 4th row to this table for the `traceback` module whose description is: "Formatted display of traceback objects"

UPDATED
405 :: 10.14 : Exercise 10-5 : All five subproblems [parts (a)-(e)] should be aligned (flush) towards the left so that part (b) does not wrap. Also, the Python prompt (">>>") for part (e) should not be in **`bold`** and should line up flush with the line below it.

Chapter 11

413 :: 11.2.2 : Near the top of the page, in the calls to `foo`, both the standard calls as well as the keyword calls should be aligned properly.

414 :: 11.2.4 :: Example : In the 2nd paragraph, remove the space in "binary operators/ operations".

UPDATED
417 :: 11.2.4 :: Line-by-Line Explanation :: Lines 30-41 : Replace the middle section of the code snippet on the 2nd half of this page as follows:

```
$ easyMath.py
7 - 2 = 5
correct
Again? [y]
8 + 1 = 9
correct
Again? [y]
10 - 7 = 4
incorrect... try again
10 - 7 = 2
incorrect... try again
10 - 7 = 5
sorry... the answer is
10 - 7 = 3
10 - 7 = 3
correct
Again? [y]
7 - 5 = 2
correct
Again? [y] n
```

NEW
432 :: 11.5.2 :: Default Function Object Argument Example : The end of this subsection is missing an entire Line-by-Line Explanation subsection. At the bottom of this page, append the following:

Line-By-Line Explanation

Lines 1-3, 29-30

In this short application, we import the `urllib.urlretrieve()` function. The only thing this program does when invoked is to execute `download()`.

Lines 5-10

`firstNonBlank()` looks for and returns the first non-blank line in the list of `lines` passed in.

Lines 12-18

`firstLast()` uses `firstNonBlank()` to find the first and last non-blank lines in the set of lines of a downloaded web page. We can make the `open()` call more clear by passing in a 2nd parameter `'r'`, as a flag to open the file for read. It uses `firstNonBlank()` to find the first non-blank line and then flips the web page upside down by calling the `lines.reverse()` method and uses `firstNonBlank()` again to find the last non-blank line.

Lines 20-27

`download()` takes a URL and procesing function object `process` that it will execute later on. The URL has a default of `http://www` that will only work if you have a host named "www" on your network. It is a

default because it allows you to change your call to `download()` on line 30 so that you can pass in alternative URLs to download. Now when passing in a function object as `process`, be sure to leave OFF the "`()`". This is because you only want to pass in the function object, not *call* it. You will see the parentheses when you are calling it (as in line 27). Although we default to passing in `firstLast()`, it is also replaceable with something else if you so desire.

NEW
434 :: 11.6.1 : In the 1st code snippet on this page, the 2nd **`print`** statement should be changed to:

```
print 'formal arg2:', arg2
```

NEW
442 :: 11.7.2 : Table 11.2 : Add footnote "c" for `reduce()`: "Moved to `functools` module in Python 3.0." Place the footnote marker "c" just after the closing parenthesis in "...init])". Replace footnote "a" with: "Effectively deprecated in Python 1.6 and removed in Python 3.0."

443 :: 11.7.2 :: Figure 11-1 : The "0" and "1"s inside the box representing the boolean function `bool_func()` should instead be represented by the Boolean values `False` and `True`, respectively. Those values will be integers only prior to Python 2.3.

444 :: 11.7.2 :: `filter()` : The 2nd paragraph (right beneath the code snippet) mistakenly refers to having two functions, one of which is `main()`. (We removed this function, pushed the body out to the global part of the code but did not update the text. Rewrite this paragraph as:

> This code snippet contains a function `odd()` which returns `True` if the numeric argument that was passed to it was odd and `False` otherwise. The main portion of the snippet creates a list of nine random numbers in the range 1-99. [Note that `random.randint()` is inclusive and that we could have used the equivalent `random.randrange(1, 100)` which is recommended because it has the same calling convention as the built-in function `range()`.] Then `filter()` does its work: call `odd()` on each element of the list `allNums` and return a new list containing just those items of `allNums` for which `odd()` returned `True`, and that new list is then displayed to the user with **`print`**.

446 :: 11.7.2 :: `map()` : In the 1st sentence of the 3rd-to-last paragraph, "map()" should be in Courier/mono.

NEW
459 :: 11.8.4 :: Example 11.7 : Indent line 7 to be at the same level as line 9.

NEW
460 :: 11.8.4 :: Lines 6-9, 26-31 : Remove the extra word "a" at the top of the page.

UPDATED
465 :: 11.8.5 : Replace both **`print`** statements without a parameter... like this:

```
print bar()
```

UPDATED
469 :: 11.10.1 : Replace the 4-line code snippet in the middle of the page with:

```
from random import randrange
def randGen(aList):
    while aList:
        yield aList.pop(randrange(len(aList)))
```

Finally, all three of the code snippets used in this subsection can now be downloaded **here** (as of 2006 Dec 6).

470 :: 11.10.1 : In the final sentence of the 2nd paragraph, "you" is the wrong word and should be changed: "Without generators, your application code...".

474 :: 11.11 :: Exercise 11-13 : Reword part (c) to: Discard `mult()` and use **`lambda`** instead.

NEW
475 :: 11.11 :: Exercise 11-19 : For some reason, this new exercise was not added, so pls do so here:

11-19. *Variable Scope*. Earlier in the chapter (see Example 11.9 on p. 466), we left determining the output of `scope.py` as an exercise for the reader.
a) Write down your best guess, then download the code and run it. Were you close? Explain where you were wrong and why (if applicable). Hint: first determine the total number of lines output by counting how many **`print`** statements will execute before trying to figure out the output.
b) Line 11 in `proc2()` is currently blank... insert this statement (indented properly): **`global`** `j`. How does this affect the output (and why)?

Chapter 12

NEW
484 :: 12.4.1 : In the 2nd line of the code snippet, remove the extra "[".

485 :: 12.4.4 : In the section title, "Import" should be in regular roman (italicized) font, not `Courier Italic`.

488 :: 12.5.4 : The final sentence of the second paragraph features a misspelling of `impter.py` filename.

494 :: 12.7.1 : Towards the end of this section, the reference to `__init__.py` is horribly misspelled. It is supposed to be "init" surrounded by a pair of double-underscores.

495 :: 12.7.4 : At the bottom of the page when importing `G3.dial`, there is an extra trailing "." that SHOULD BE REMOVED.

502 :: 12.10 : Exercise 12-6 : Resolving various spacing issues in the example call, we get: `newname = importAs ('mymodule')`.

Chapter 13

508 :: 13.1 :: Classes and Instances : Missing a word in the 3rd paragraph which should read, "...because there would be no references...."

NEW
509 :: 13.1 :: Methods : The reference to Section 13.7 on page 540 in the 2nd-to-last paragraph on this page should refer to page 541 instead.

515 :: 13.2.1 : The 2nd to last sentence in this section should end with "OOD" instead of "OOP", otherwise it reads like infinite recursion, or "Mr. Bunny's Guide To Java". `:-)`

520 :: 13.4.1 : Since this section is optional (as per the previous section), there should be a "*" in front of the subsection title.

524 :: 13.4.4 : Under Table 13.1, the first paragraph should start with the following: "In addition to the `__dict__` attribute of the class `MyClass` we just defined above, we have the following:". Also, the **`print`** statement for `MyClass.__dict__` SHOULD BE REMOVED since we already displayed the contents of the dictionary for both new-style and classic classes in the immediately preceding subsection (on the previous page).

NEW
530 :: 13.5.4 :: Core Note : All references to `InstCt` should be changed to `InstTrack` to match the output below.

NEW
531 :: 13.6.1 :: Core Note : In the last sentence of the 2nd paragraph of this Core Note, change "much" to "must".

NEW
537 :: 13.6.5 :: Access to Class Attributes : In the 2nd-to-last sentence of the paragraph, change "free" to "tree".

NEW
538 :: 13.6.5 :: Use Caution When Accessing Class Attribute with Instance : In the last sentence of the final

paragraph, change "`c.version`" to "`foo.x`".

543 :: 13.8.1 : In the `TestStaticMethod` example, the call `foo = staticmethod(foo)` should be dedented to the same level as the definition of the `foo()` function, e.g.,

```
class TestStaticMethod:
    def foo():
        print 'calling static method foo()'
    foo = staticmethod(foo)
```

553 :: 13.11.3 : In the `SortedKeyDict` example output, the supposedly sorted list when using `keys()` is out-of-order due to an error correction: "xin-yi" was misspelled as "hsin-yi", and the edit was made without reordering the list correctly. it should be:

```
By keys(): ['hui-jun', 'xin-yi', 'zheng-cai']
```

575 :: 13.13.2 :: Example 13.3 : **`time60.py`**

- Line 7 needs to be indented (right) so it is lined up with lines 8 and 9.
- Also change the documentation string for `__init__.py` to say "`Time60 initializer` since it's more of an initializer than a constructor. (minor patch)

UPDATED
586 :: 13.14 :: Single Underscore (_) : Remove the "we" at the bottom of this page: "...of the new-style ~~we~~ attribute access...."

NEW
591 :: 13.15.2 :: Simple Example Wrapping Any Object : In the final code snippet in the middle of the page: change the middle part of the quote in the **`print`** statement to "... `file %r, mode %r at` ...". On the next 2 lines, remove the single quotes enclosing both `f.name` and `f.mode`, and also put "..." on the code line that starts with `f.mode`: `... f.mode, id(f.get()))`

603 :: 13.16.4 :: Descriptor Examples : In the middle of the page, the sentence that starts with, "Our final example...," should refer to Example 13.9 on the next page. Otherwise, you may think that it refers to the previous code snippet.

603 :: 13.16.4 :: Descriptor Examples and Example 13.9 :: Lines 28-38 : The final sentence of this paragraph has several problems. Rewrite as: "Note that if you are using Python older than 2.5, you cannot merge the **`try-except`** and **`try-finally`** statements together (lines 30-38)."

604 :: 13.16.4 :: Example 13.9 : **`descr.py`**
For this code to work with Python 2.5 and newer, delete line 31. For this code to work with any version of Python, indent line 34 one level (four spaces to the right). For the `__set__()` method, the `open()` statement on line 29 should really be moved to the innermost `try` block immediately below it (between lines 31-32), and the succeeding **`except`** should also be monitoring for `IOError` — just like in the above `__get__()` method. Less importantly, `name` should not not have a default value as it does now on line 9. Below, we post two patches, one for users of Python 2.5 and newer as well as one for those using Python 2.4.3 and older.
(critical patch [diff]; pre-2.5 patch [diff])

607 :: 13.16.4 : Properties and `property()` Built-in Function : The last part of the final sentence in the first complete paragraph at the top of the page before the definition of the `ProtectAndHideX` class should be changed to the following: "... encrypting it by using the bitwise complement operator:". (The XOR operator is binary and not used in this example.)

UPDATED
609 :: 13.16.4 : Properties and `property()` Built-in Function :

- DELETE the final two sentences at the bottom of the page, starting with the sentence beginning with, "We also use a function decorator...."
- The code snippet above this paragraph needs to be corrected, the most important is the DELETION of the "`@property`" line (see below):

```
#!/usr/bin/env python
'ProtectAndHideX.py -- protect the "x" attribute'
```

```
class ProtectAndHideX(object):
    def __init__(self, x):
        self.x = x

        @property
        def x():
            def fget(self):
                return ~self.__x

            def fset(self, x):
                assert isinstance(x, int), \
                    '"x" must be an integer!'
                self.__x = ~x

            return locals()

        x = property(**x())
```

We have made this change plus additional lines testing this code available **here**.

618 :: 13.18 :: Exercise 13-3 : In the preliminary work paragraph, there is a missing single closing quote for the string `'$1,234,567.89'`, and that string should be in Courier/mono font.

UPDATED
619 :: 13.18 :: Example 13.11 : **`moneyfmt.py`**
Line 13 of `__repr__()` should be replaced with a call to `repr()`, i.e.,

```
def __repr__(self):
    return repr(self.value)
```

The original code, including the source in the download area, has **`return`** `` `self.value` ``, which is synonymous for now. The single backticks will be deprecated in Python 3.0.

621 :: 13.18 :: Exercise 13-7 : The last part of the final sentence of this exercise should read as: "... this class in to Exercise 6-15.

UPDATED
622 :: 13.18 :: Exercise 13-9 :

- In the descriptions of the `enqueue()` and `dequeue()` functions, replace both occurrences of "list" with "queue."
- The last sentence of this problem should refer to Example 6.4.

624-625 :: 13.18 :: Exercise 13-20 :
- Part (c): Add a 4th bullet: "A pair of keyword arguments (`hr=10, min=30`)"
- Part (e): Delete the last sentence that reads as "Make it so."
- Part (f): The first and last lines of the code snippet are in a slightly larger font... make all 4 lines have the same sized font:

```
>>> thu = Time60(10, 30)              [ed. reduce font size]
>>> fri = Time60(8, 45)
>>> thu + fri
18:75                                 [ed. reduce font size]
```

- (Add a) Part (g): "Can we move the math-oriented code out of `__add__()` and into one of the other methods of this class? If so, which one and why would it make sense to do so?"
- (Add a) Part (h): 'Add new code to support the "subtraction" of `Time60` objects. Extra Credit: also support in-place subtraction.'
- (Add a) Part (i): "Implement an `__int__()` method which returns an `int` representing the total number of minutes expressed by a `Time60` object."

625 :: 13.18 :: Exercise 13-21 : Like the errata above for the code on p. 609, the use of `property()` as a decorator

does not work for *any* version of Python. There is also multiple typos in the "alternate syntax" (which turns out to be the only syntax that can be used). This problem should be completely rephrased as:

> 13-21. *Decorators and Function Call Syntax.* Towards the end of section 13.16.4, we used the syntax "`x = property(**x())`" to create a property for the x attribute. How does this idiom, including the enclosed call to `x()`, actually create the property?

Chapter 14

630 :: 14.1 :: "`lambda` Expressions" : In the (sub)section title, "`lambda`" should be in normal roman font, not `Courier`.

640 :: 14.3.4 : Add footnote "a" for section title right after "**`exec`**": "Changed to `exec()` built-in function in Python 3.0."

641 :: 14.3.5 : Add footnote "a" for section title right after "`input()`": "Because of its security flaws, `input()`'s functionality is replaced by `raw_input()`'s (and `raw_input()` removed) in Python 3.0."

644-645 :: 14.3.6 :: Creating Code at Runtime and Executing It : For those unfamiliar with the C shell and its descendants, you can change the % prompt to that of sh/bash: $. The first one is misindented too far... move it to the left to align it with the rest of the program output.

648 :: 14.3.6 :: Conditionally Executing Code :: Example 14.2 :
- Line 24 should be indented at the same level as line 25
- Line 29 should be indented at the same level as line 21

649 :: 14.3.6 :: Line-by-Line Explanation :: Lines 18-29 : There is an extra blank line in the program output... remove it.

650 :: 14.4.1 :: Core Note : Near the end of the 1st paragraph, replace "**`def`** `foo`" with "**`def`** `bar`".

651 :: 14.4.2 : Add footnote "a" for section title right after "`execfile()`": "Deprecated in Python 3.0; use `exec()` instead."

651 :: 14.4.1 : The **`print`** statement for `import2.py` needs to be indented to the right one more space.

652 :: 14.4.1 : In the final paragraph `eval()` should not be italacized.

657 :: 14.5.2 : In the (sub)section title `os.popen()` should be in `Courier` font.

661 :: 14.5.5 : The reference to `os.system()` at the top of the page should not be in boldface.

661 :: 14.5.5 :: Replacing `os.system()` : Remove the **`import`** `os` line completely.

662 :: 14.5.6 :: Table 14.7 : (Python 2.6) All text labeled as "`/popen2`" should be removed, i.e., "`os/popen2.popen2()`" should be changed to "`os.popen2()`" and maintain footnote "a". Apply this to all three such that only "`os.popen3()`" and "`os.popen4()`" appear in the 1st column of the table. Also delete footnote "b", replacing it with footnote "c", changing the footnote "c" for "`subprocess.call()`" to "b". The original footnote "b" for "`os.popen4()`" should revert to "a".

663 :: 14.7.1 : `SystemExit` in the second-to-last sentence on this page should be one word.

664 :: 14.7.1 :: Example "14.4" : Replace both references to "14.4" with "14.3". (There is no Example 14.4.)

NEW
668 :: 14.8 :: Table 14-9 : Add new footnote "d" for `popen2`: "Deprecated in Python 2.6; use `subprocess` instead."

UPDATED
668-669 :: 14.10 :: Exercise 14-1 : There should be 13 exercises in this chapter:

- The first two problems are accidentally merged into Exercise 14-1. The third is missing.
- A new Exercise 14-2 begins labeled: "**exec** *versus* `eval()`". Only ***exec*** should be **bolded**, not the remainder of the problem.
- A missing Exercise 14-3 follows: "*Statements vs. Expressions.* What are the differences between statements and expressions?"
- Renumber the remaining problems accordingly, i.e., the printed Exercise "14-2" should be Exercise "14-4" instead.

669 :: 14.10 :: Exercise 14-9 (14-11) :

- Extra Credit 1: The "`os`" module name should be in `Courier` font.
- Extra Credit 2: The "`-ef`" argument of the functional style call should be parenthesized, as in: `sort(grep(ps(-ef), root), -n, +1)`.

669 :: 14.10 :: Exercise 14-11 (14-13) : Should refer to Example 14.2, not 14.4.

Chapter 15

697 :: 15.3.13 :: Example 15.1 : **`rewho.py`**
`f.readlines()` in line 7 can be shortened to the file iterator `f`, as in:

```
for eachLine in f:
```

(minor patch)

NEW
699-700 :: 15.4 :: Example 15.2 and Line-By-Line Explanation (Lines 14-22) : In lines 4, 17, 22 of the code, change "`lowercase`" to "`ascii_lowercase`". Correspondingly, change the last word of the first paragraph of this subsection from "`string.lowercase`" to "`string.ascii_lowercase`".

Chapter 16

719-720 :: 16.3.2 :: Table 16.1 : It would be nice if the 2nd column were reduced in size so that the methods in the 1st column don't wrap.

UPDATED
729 :: 16.3.7 :: Example 16.4 : Line 20 in `tsUclnt.py` should simply be "**`print`** `data`", as in:

```
print data~~udpliSock.close()~~
```

(critical patch)

734 :: 16.4.1 :: Example 16.5 : Line 13 has a font size issue.

740 :: 16.5.2 : In the 2nd paragraph, remove the hyphen from the call to `clientConnectionFailed()`.

742 :: 16.6 : Replace the final part of the final sentence with, "... we strongly encourage you to read ahead to Chapters 17 and 20."

742 :: 16.7 :: Exercise 16-5 : The URL for the socket example has changed to
http://docs.python.org/lib/socket-example.html

745 :: 16.7 :: Exercise 16-14 : In the final portion of the exercise, change "i.e." to "e.g.".

Chapter 17

749 :: 17.2.2 : Figure 17-1 : The diagram was not drawn correctly to specification. Here are some fixes to repair it:

a. Make the FTP client box bigger and make the FTP server box smaller so they are nearly the same size... there is no size differentiation between client and server, so the diagram should reflect that.
b. Move "M (> 1023)" left just above the top solid line connecting the FTP client box and the Internet cloud, symmetric to where "M + 1" is below the bottom solid line.
c. Move "ctrl/cmd" down so it's just above the top dashed line inside the Internet cloud
d. Move "data" to be centered just below the bottom dashed line inside the Internet cloud
e. Change "20 or" to "20 [active] or"
f. Change "N (> 1023)" to "N (> 1023) [passive]"
g. Delete "(Active)(Passive)"

754 :: 17.2.6 : Example 17.1 : Line-by-Line Explanation : Lines 11-44 :

a. (middle paragraph) Add one more sentence: "After the transfer has completed, we would then call `loc.close()`."
b. (final paragraph) Strike "if it is there": "...we remove the empty file ~~if it is there~~ to avoid clutter...".
c. (final paragraph) Add a new second-to-last sentence: "We should probably put some error-checking around that call to `os.unlink(FILE)` in case the file does not exist."
d. (final paragraph) Change the final sentence to: "Finally, to avoid another pair of lines (lines 43-44) that close the FTP connection and return, we use an **`else`** clause (lines 35-42)."

755 :: 17.2.7 : In the first paragraph after the bullet points, "creating" should be changed to "created".

757 :: 17.3.3 : The first paragraph after the list of steps is missing a word: "...carbon copy of using the FTP protocol."

758 :: 17.3.4 : Table 17.2 : In the entry for "xhdr", move the "[" back two characters, to, `xhdr(`*`hdr, artrg`*`[,` *`ofile`*`])`

759 :: 17.3.5 : In the line of code that starts with, ">> Rounding like this...", remove one ">" so that it is "> Rounding like this..."; also, it appears either this line or the line above it has the wrong paragraph tag as there is a gap between these two lines... it should be single-spaced like all the others.

762 :: 17.3.6 : Example 17.2 : Line 60 : Not really a correction but a note that Generator Expressions were added in Python 2.4, so if you are using 2.0-2.3.x, you will need to switch it to a List Comprehension by changing the parentheses to square brackets.

765 :: 17.3.6 : Example 17.2 : Line-by-Line Explanation : Lines 57-80 : At the top of this page following the first paragraph of this section which began on the previous page, insert the following new paragraph which explains line 60.

> Line 60 is a generator expression. It works like a list comprehension only it does lazy evalution and does not build the entire results list when this line is executed. (The entire dataset, `data`, is already taking enough memory as it is!) Rather, each line is only processed when its turn comes up in the loop below on line 62. Generator expressions were added in Python 2.4, so if you are using an earlier release, the easiest fix while minimizing memory use is to add a new statement in between lines 62 and 63: `line = line.rstrip()`. If memory is not a concern, then just use a list comprehension instead (change the parentheses to square brackets).

UPDATED
772 :: 17.4.8 : Figure 17-3 : "POP3 (read)" should be changed to "POP3 (receive)"

773 :: 17.4.9 : Add a mention of secure mail retrieval by splitting up and updating the final sentence: "Before we take a look at a real example, we should mention that there is also a `poplib.POP3_SSL` class which will perform mail transfer over a secure connection provided the appropriate credentials are supplied. Let's take a look...."

784 :: 17.6 : Exercise 17-28(a) : Not really a correction, but add a... "Hint: see the `mailbox` module and `email` package."

Chapter 18

801 :: 18.5 :: Core Tip : In the 3rd paragraph, the `setDaemon()` method name should not have a hyphen/dash.

802 :: 18.5 :: Table 18.3 : There shouldn't be any spaces around the = for the `join()` method's arguments.

NEW
809 :: 18.5.4 :: Example 18.8 : In `mtfacfib.py`, we create a function named `sum()` which hides/shadows the built-in function of the same name. If you do not wish this, then just rename it to something else, like `mySum()`, for example, and update lines 16 and 21 accordingly.

811 :: 18.5.6 :: Table 18.5 : The table describing the attributes of the `Queue` module needs to be updated... stay tuned for the exact corrections/updates. The most important correction at this time is that the first entry should be categorized as "`Queue` Module Class", not "Function". Also, the "q" should be capitalized, as in `Queue(`*`size`*`)`. For more details, see the module documentation at **http://docs.python.org/lib/module-Queue.html**

Chapter 19

NEW
833 :: 19.3.5 :: Line-by-Line Explanation :: Lines 1-18 : The `functional` module is renamed to `functools`.

834-835 :: 19.3.6 :: Example 19.6 : Lines 15 and 17 should be indented like the rest of the lines.

851 :: 19.6 :: Exercise 19-2 : Replace "windows" with "widgets".

851 :: 19.6 :: Exercise 19-4 : HINT: you will need 3 separate handlers or, customize one handler with arguments preset (still 3 function objects).

Chapter 20

860 :: 20.2.1 :: Table 20.2 : In the header for the first column, ***`net_loc`*** should be in ***`Courier Bold Italic`***.

UPDATED
868 :: 20.2.4 :: Example 20.1 :

- Change the name of this cile in the Example header to "`urlopen`Auth`.py`".
- Change the login name from `wesc` to `wesley` so it matches up with the other examples in this chapter.
- Adjust line 20 so it is flush with the other lines of code.

869 :: 20.2.4 :: Line-by-Line Explanation :: Lines 24-29 :

- Change the filename in the code snippet `urlopenAuth.py` (The current name [`urlopen-auth.py`] will not allow the module to be imported since '-' is not a valid Python identifier.)

871 :: 20.3 :: Example 20.2 : The code for the `__init__()` and `filename()` methods (lines 16-17, 20-34) are not indented properly... there should be 4 spaces of indentation for these methods.

878 :: 20.5.1 : The final line of code on this page is missing an optional parameter... it should read as: `$ python -m CGIHTTPServer [port]`.

879 :: 20.5.1 : In the final paragraph of this section (above the URL links), `cgi-bin` is accidentally titlecased twice.

880, 890-892, 897-898 :: 20.5.2, 20.5.5, 20.7.4 : Figures 20-4, 20-9, 20-10, 20-11, 20-14, 20-15 : The captions should be centered so they fit on a single line below each figure.

NEW

885 :: 20.5.4 :: Example 20.5 : The **`if`** clause (line 49) should be indented at the same level as the **`else`** clause (line 51).

896 :: 20.7.3 : In the first complete paragraph at the top of the page, `string.split()` should be *`str`*`.split()` to indicate usage of the `split()` method of strings rather than the `split()` function of the `string` module.

911 :: 20.9 :: Table 20.7 : In the description for the `wsgiref` module, "application" should be in plural, e.g., "applications."

913 :: 20.10 :: Exercise 20-3 : Remove the extra comma in the reference to Example 11.4.

914 :: 20.10 : Exercises 20-10 and 20-11 : Exercises 20-7 through 20-10 are related and grouped together; there should be an extra blank line separating Exercise 20-10 and 20-11.

Chapter 21

928 :: 21.2.2 : Methods : Table 21.5 : Each of the arguments of `errorhandler()` should be in *`Courier Italic`*. There should also be some text just above this table which describes the error-handling mechanism further, such as, "An option for database adapter writers to allow for user-defined error handlers is available. Both `Connection` objects as well as `Cursor` objects (see next section) allow module authors to define an `errorhandler()` method for this purpose.

929 :: 21.2.3 :: Table 21.6 : For the `description` attribute description, both *`name`* and *`type_code`* at the end should be in *`Courier Italic`*.

UPDATED
935-936 :: 21.2.7 :: MySQL : Remove all text starting at, "Keep in mind...", at the bottom of p. 935, all the way to the end of this section at the top of p. 936.

937 :: 21.2.7 :: SQLite : The pysqlite database adapter should not be in `Courier`.

939-945 :: 21.2.7 :: (Example 21.1) : There is no output for Example 21.1; however, we do provide the output of the equivalent ORM versions (twice) later in the chapter. The output of those applications will be the virtually the same, given the natural variations of the random numbers generated.

940-953 :: 21.2.7-21.3.3 :: Examples 21.1-21.3 : In various places, the font was shrunk so as to not wrap lines. This may look strange in the text. These include the following:

- Example 21.1: 54, 107-115, 136
- Example 21.2: 56, 64, 75, 99
- Example 21.3: 71

941 :: 21.2.7 :: Example 21.1 : Line 55 should be flush with the others in this **`elif`** clause.

947 :: 21.3.3 :: Line-by-Line Explanation :: Lines 33-44 : **`try-except`** should be in bold.

953 :: 21.3.3 :: Line-by-Line Explanation : In the section for lines 79-80, there should not be a space between the function name and parameters in the `getattr(orm, 'drop')` call, and further on, `self.users` should be in Courier/mono font. Finally, there are too many blank lines above and below the next section (lines 82-84).

Chapter 23

987 :: 23.1.1 : Table 23.1 : Footnote "b" can be worded better: "Some values come back (additionally) quoted although all are returned as part of a single CSV string from the server."

988 :: 23.1.1 :: Example 23.1 : **`stock.py`**
Line 20 should be `u.close()`.
(critical patch)

998 :: 23.2.5 :: Example 23.5 : **`olook.pyw`**
For running the **demo**), the `outlook()` function should be named `olook()` (to match the name of the module) because that is what the demo expects. Since we are lazy, we just simply create an alias to the original function by inserting a new line 31 which reads: `olook = outlook`. If you're not going to run it, this change is not necessary. (minor patch)

1001 :: 23.2.6 :: Example 23.6 : **`estock.pyw`**
Line 43 should be `u.close()`. Also, on line 25, "as" should not be **bolded**.
(critical patch)

Appendix A

NEW
1012 :: Chapter 2 :: Exercise 2-7 : The 3rd solution for this problem, using a **`while`** needs to increment the variable otherwise the loop is infinite. Add the following line immediately following and indented at the same level as the **`print`** statement, i.e.,

```
i = 0
slen = len(s)
while i < slen:
    print i, s[i]
    i += 1
```

Appendix B

NEW
1025 :: Table B.3 : Footnote "b" should refers erroneously to Table 5.2... it should be Table 5.3.

NEW
1038 :: File Object Methods and Data Attributes : The section title has "Attributes" misspelled... remove the extra "o".

CREDITS

Thanks to the many readers who have provided invaluable feedback and corrections in this section. They include but are not limited to: David Librik, Sebastian Strempel, Loek Engels, Steven Wayne, Viraj Alankar, Johan Neven, Dave Hines, Barrett Shiff, Harold Trammel, Marie-Cécile Baland, Greg Nofi, Mark Ackerman, Tony Tang, Adil Lotia, Satoshi Tanabe, Raven Erebus, Tim Heaney, Victor Cosby, Dimitris Segounis, Paul Kippes, David Håsäther, Mike Driscoll, Dick Moores, Art Rathjen/Mary Bos, Boyd Brown, Paolo Palumbo, David Drysdale, and others.

corepython (at) yahoo (dot) com