Errata for *Impractical Python Projects* (updated to 6th printing)

Page 33: In the list of palingrams, "dairy raid" should now read "welsh slew"

Page 68: The first two lines of the second code snippet that read:

```
for i in range(len(list_of_lists)):
print(list_of_lists[i])
```

should now read:

```
for nested_list in list_of_lists:
print(nested_list)
```

Page 79: In the sentence before the last code block, we changed "Figure 4-2" to "Figure 4-3"

Page 85: In Listing 4-9, the code at number ball (5) and the following line that read:

```
    row1 = (message[:row_1_len])
    row2 = (message[row_1_len:])
```

should now read:

```
    row1 = (message[:row_1_len]).lower()
    row2 = (message[row_1_len:]).lower()
```

and the code immediately below number ball (8) that reads:

```
        plaintext.append(r1.lower())
        plaintext.append(r2.lower())
```

should now read:

```
        plaintext.append(r1)
        plaintext.append(r2)
```

Page 100: The output printout near the bottom of page that reads:

```
Panel at east end of chapel slides
```

should now read:

```
Panelateastendofchapelslides
```

Page 103: The cipher example that reads:

The cold tea didn't please the old finicky woman.

should now read:

So, the cold tea didn't please the old finicky woman.

Page 111: In the 4th and 5th paragraphs, we changed all instances of "property" and "properties" to "attribute" and "attributes"

Page 116: In the second line of the first paragraph, we changed "property" to "attribute"

Page 141: In Listing 7-10, the line labeled with number ball (7) and the preceding line that read:

```
lock_wheel = int(randrange(0, len(combo)))
next_try[lock_wheel] = randint(0, len(combo)-1)
```

should now read:

```
lock_wheel = randrange(0, len(combo))
next_try[lock_wheel] = randint(0, 9)
```

Page 156: In the note, the part that reads:

. . . and adding the key/value pair (at any location, since dictionaries are unordered).

should now read:

. . . and adding the key/value pair at any location.

Page 164: In the second paragraph, the line that reads:

Because of the very short training corpus, the moon is the only word pair with multiple keys.

should now read:

Because of the very short training corpus, the moon is the only word pair with multiple values.

Page 171: In the last sentence before the section "The Code":

This is a far better solution than manually finding and commenting out `print()` statements!

should now read:

This is a far better solution than manually finding and commenting out calls to `print()`!

Page 182: In the last paragraph, the poem that reads:

Cool stars enter the

Window this hot evening all

Heaven and earth ache

should now read:

A line flap-flapping

Across the dark crimson sky

On this winter pond

Page 205: The line and following equation that read:

The transformation to generate points over a unit disc is as follows:

x= √r*cos

should now read:

The transformation to generate points evenly over a unit disc is as follows:

x= √r*cosθ

and the line that reads:

The equations yield (x, y) values between 0 and 1.

should now read:

The equations yield (x, y) values between –1 and 1.

Page 218: The indentation for the listing on the page should be as follows:

```
>>> from random import randint
>>> trials = 100000
>>> success = 0
>>> for trial in range(trials):
           faces = set()
           for rolls in range(6):
               roll = randint(1, 6)
               faces.add(roll)
           if len(faces) == 6:
               success += 1
>>> print("probability of success = {}".format(success/trials))
probability of success = 0.01528
```

Page 250: The indentation of the three lines labeled number ball (8), number ball (9), and number ball (10) should now be indented by one more level.

Page 252: The last line on page that reads:

Set the default to `'sbc_blend'`, since this is theoretically the most stable mix of the four choices.

should now read:

Set the default to `'bonds'`, in order to see how this supposedly 'safe' choice performs.

Page 259: The first line in last paragraph that reads:

. . . a 4 percent withdrawal rate (equal to $80,000 per year), a 30-year retirement, and 50,000 cases.

should now read:

. . . a 4 percent withdrawal rate (equal to $80,000 per year), a 29-30-31 retirement range, and 50,000 cases.

Page 261: The indentation of the last two lines in Listing 12-9:

```
    investments -= withdraw_infl_adj
          investments = int(investments * (1 + i))
```

should now be:

```
          investments -= withdraw_infl_adj
          investments = int(investments * (1 + i))
```

Page 305: In the last paragraph, `transform_rotate()` should now read `transform.rotate()`

Page 329: The first sentence under "The Shell Utilities Model" that reads:

The shell utilities module, `shutil`, provides high-level functions for working with files and folders, such as copying, moving, renaming, and deleting.

should now read:

The shell utilities module, `shutil`, provides high-level functions for working with files and folders, such as copying, moving, and deleting.

Page 356: Between the lines labeled with number balls (5) and (6) in Listing 16-2, we inserted the following new lines of code:

```
    # check for missing digits
    keys = [str(digit) for digit in range(1, 10)]
    for key in keys:
        if key not in first_digits:
            first_digits[key] = 0
```

Page 357: In the first line of the second paragraph, we deleted the line: “Like all Python dictionaries, `first_digits` is unordered.”

Page 360: In the 5th line of the 6th paragraph, “property” should now read “attribute”

Page 368: The code for “Dictionary Cleanup” that reads:

```
"""Remove single-letter words from list if not 'a' or 'i'."""
word_list = ['a', 'nurses', 'i', 'stack', 'b', 'cats', 'c']

permissible = ('a', 'i')

# remove single-letter words if not "a" or "i"
for word in word_list:
      if len(word) == 1 and word not in permissible:
            word_list.remove(word)
print("{}".format(word_list))
```

should now read:

```
"""Remove single-letter words from list if not 'a' or 'i'."""
word_list = ['a', 'nurses', 'i', 'stack', 'b', 'c', 'cat']
word_list_clean = []

permissible = ('a', 'i')

for word in word_list:
    if len(word) > 1:
        word_list_clean.append(word)
    elif len(word) == 1 and word in permissible:
        word_list_clean.append(word)
    else:
        continue
```

```
print("{}".
```

Page 369: Line 17 of the code in "Finding Digrams" which reads:

```
print(*digrams, sep='\n')
```

should now read:

```
print(*sorted(digrams), sep='\n')
```

and line 28 which reads:

```
for k in mapped:
```

should now read:

```
for k in sorted(mapped):
```