Errata for
Python and Algorithmic Thinking for the Complete Beginner

1.2 What it Hardware?

- **The Central Processing Unit (CPU)**

  This is the part of a computer that actually performs all the tasks defined in a program **(basic arithmetic, logical, and input/output operations.)**

5.10 Review Exercises

2. Match each element from the first column with one element from the second column.

Value	Data Type
1. The name of a person	**a.** Boolean
2. The age of a person	**b.** Real
3. The result of the division 5/2	**c.** Integer
4. Is it black or is it white?	**d.** String

13.1 Introduction

Exercise 13.1-2 Finding the Sum of Digits

First Approach

Step	Statement	Notes	number	digit1	digit2	digit3	digit4	r	total
...	...		...	...	...	...	...	...	...
8	`total = digit1 + digit2 + digit3 + digit4`		6753	6	7	5	3	53	**21**
9	`print(total)`	Value **21** is displayed							

Second Approach

Once more, let's try to understand the second approach using an arithmetic example. Take the same number, 6753, for example.

Fourth digit = 3	The fourth digit can be isolated if you divide the given number by 10 to get the integer remainder `digit4 = 6753 % 10`
Remaining digits = 675	The remaining digits can be isolated if you divide the given number by 10 to get the integer quotient `r = 6753 // 10`

Third digit = 5	The third digit can be isolated if you divide the remaining digits by 10 to get the integer remainder `digit3 = 675 % 10`
Remaining digits = 67	The remaining digits are now `r = 675 // 10`
Second digit = 7	The second digit can be isolated if you divide the remaining digits by 10 to get the integer remainder `digit2 = 67 % 10`
First digit = 6	The last remaining digit, which happens to be the first digit, is `digit1 = 67 // 10`

13.2 Review Exercises

5. Write a Python program that prompts the user to enter an integer representing an elapsed time in seconds and then displays it in the format "WW weeks DD days HH hours MM minutes and SS seconds." For example, if the user enters the number 2000000, the message "3 weeks 2 days 3 hours 33 minutes and 20 seconds" should be displayed.

14.3 Retrieving a Portion from a String

Exercise 14.3-1 Displaying a String Backwards

Third Approach

file_14_3_1c

```
s = input("Enter a word with four letters: ")

s_rev = s[::-1]

print(s_rev)
```

15.5 Python's Membership Operators

- `s not in ["a", "b"]`. This can be read as "test if ***s*** is **not** equal to letter "a", nor equal to letter "b".

22.5 Exercises of a General Nature with Decision Control Structures

Exercise 22.5-3 Is the Number a Palindrome?

Second Approach

file_22_5_3b

```
import re
IS_NUMERIC = "^[-+]?\\d+(\\.\\d+)?$"

inp = input()

if not re.match(IS_NUMERIC, inp):
    print("You entered non-numeric characters")
else:
    x = int(inp)
    if x < 10000:
        print("You entered less than five digits")
```

```
    elif x > 99999:
        print("You entered more than five digits")
    else:
        inp_reversed = inp[::-1]
        if inp == inp_reversed:
            print("Palindrome")
        else:
            print("Not Palindrome")
```

24.1 The Pre-Test Loop Structure

Exercise 24.1-3 Designing the Flowchart and Counting the Total Number of Iterations

Now, let's create a trace table to observe the flow of execution.

Step	Statement	Notes	i	
1	`i = 1`		1	
2	`while i != 6:`	This evaluates to `True`		1st Iteration
3	`i += 2`		3	
4	`while i != 6:`	This evaluates to `True`		2nd Iteration
5	`i += 2`		5	
6	`while i != 6:`	This evaluates to `True`		3rd Iteration
7	`i += 2`		7	
8	`while i != 6:`	This evaluates to `True`		.
9	…	…		.
10	…	…		

Exercise 24.1-5 Finding the Sum of 4 Numbers

Using a pre-test loop structure, write a Python program that lets the user enter 4 numbers and then calculates and displays their sum.

24.6 Review Exercises

3. Design the corresponding flowchart and create a trace table to determine the values of the variables in each step of the next Python program. How many iterations does this Python program perform?

```
i = -5
while i < 10:
    i -= 1
print(i)
```

27.8 Converting from a While-Loop to a For-loop

Exercise 27.8-3 Converting the Python Program

file_27_8_3b

```
s = 0
i = 1
for i in range(1, 6):
    s = s + (i + 1) ** 2
```

```
print(s)
```

29.6 Exercises of a General Nature with Loop Control Structures

Exercise 29.6-4 Is the Message a Palindrome?

file_29_6_4d

```
message = input("Enter a message: ").lower()

#Create a new string which contains letters except
#spaces, commas, periods or question marks
message_clean = message
for c in " ,.?":
    message_clean = message_clean.replace(c, "")

if message_clean == message_clean[::-1]:
    print(message_clean, "The message is palindrome")
```

35.2 Data Validation with Lists

Exercise 35.2-3 Displaying Odds in Reverse Order – Validation with Individual Error Messages

file_35_2_3

```
import re
IS_NUMERIC = "^[-+]?\\d+(\\.\\d+)?$"
ELEMENTS = 20


…
```

35.4 Sorting Lists

Exercise 35.4-1 The Bubble Sort Algorithm – Sorting One-Dimensional Lists with Numeric Values

Fifth pass

1st Compare

Elements at index positions 4 and 5 are compared. Since the value 49 is ***not*** *less than the value* ***25****,* ***no*** *swapping is done.*

38.3 Formal and Actual Arguments

> ***Remember!*** *There is a one-to-one match between the formal and the actual arguments. The value of argument* `a` *is passed to argument* `n1`*, the value of argument* `b` *is passed to argument* `n2`*, and so on.*

41.1 Simple Exercises with Subprograms

Exercise 41.1-5 How Many Times Does Each Number of the Dice Appear?

```
#Variable n1 is assigned the number of times that value 1 exists in list a
n1 = search_and_count(1, a)

#Variable n2 is assigned the number of times that value 2 exists in list a
n2 = search_and_count(2, a)
.
```

```
.
.
#Variable n6 is assigned the number of times that value 6 exists in list a
n6 = search_and_count(6, a)

#Display how many times each of the six numbers appears in list a
print(n1, n2, n3, n4, n5, n6)

#Find maximum of n1, n2,… n6
maximum = n1
max_i = 1

if n2 > maximum:
    maximum = n2
    max_i = 3

if n3 > maximum:
    maximum = n3
    max_i = 3
.
.
.
```

41.2 Exercises of a General Nature with Subprograms

Exercise 41.2-3 Progressive Rates and Electricity Consumption

file_41_2_3

```
…

def find_amount(kwh):
    if kwh <= 400:
        amount = kwh * 0.08
    elif kwh <= 1500:
        amount = 400 * 0.08 + (kwh - 400) * 0.22
    elif kwh <= 2000:
        amount = 400 * 0.08 + 1100 * 0.22 + (kwh - 1500) * 0.35
    else:
        amount = 400 * 0.08 + 1100 * 0.22 + 1500 * 0.35 + (kwh - 3000) * 0.5

    amount += 0.26 * amount
    return amount

…
```