



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

3.9 Application: Input Validation	110
Chapter Summary	123
Interactive Review and Practice	124
End-of-Chapter Exercises	EX3-1
4. Loops	125
<b>4.1 The while Loop</b>	126
4.2 Problem Solving: Hand-Tracing	133
4.3 Application: Processing Sentinel Values	135
4.4 Problem Solving: Storyboards	139
4.5 Common Loop Algorithms	141
4.6 The for Loop	145
4.7 Nested Loops	152
4.8 Processing Strings	159
4.9 Application: Random Numbers and Simulations	164
4.10 Graphics: Digital Image Processing	169
4.11 Problem Solving: Solve a Simpler Problem First	174
Chapter Summary	180
Interactive Review and Practice	182



SELF CHECK

- 1. What values are printed by this while loop? (For this exercise, assume that all output is printed on one line.)

```
number = 1
while number <= 5 :
    value = number * 3
    print(str(value) + ", ")
    number = number + 1
```

- 1, 2, 3, 4, 5,
- 3, 3, 3, 3, 3,
- 3, 6, 9, 12, 15,
- 3, 6, 9, 12,

One correct, 0 errors, 100%

- 2. Suppose you change the initial balance to a million dollars in the investment program. How many years does it take for the investment to double?

- Much longer than with \$10,000.
- A little longer than with \$10,000.
- Exactly the same as with \$10,000.

It is exactly the same duration: The initial balance, target balance, and interest payments are all multiplied by 100.

- Less than with \$10,000.

One correct, 0 errors, 100%

- 3. What do these code fragments print? Note that the code fragments print one value per line, but for this exercise, assume all output is printed on one line, with each value separated by a blank space (i.e., 1 2 3 4).

GOOD JOB! ✓

Loop	Output	Explanation
for i in range(2, 5): print(i)	2 4 8 16	Note that 16 is printed after the loop has completed.



Back to Page

126 / 554 >



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

3.9 Application: Input Validation	110
Chapter Summary	123
Interactive Review and Practice	124
End-of-Chapter Exercises	EX3-1
4. Loops	125
4.1 The while Loop	126
4.2 Problem Solving: Hand-Tracing	133
4.3 Application: Processing Sentinel Values	135
4.4 Problem Solving: Storyboards	139
4.5 Common Loop Algorithms	141
4.6 The for Loop	145
4.7 Nested Loops	152
4.8 Processing Strings	159
4.9 Application: Random Numbers and Simulations	164
4.10 Graphics: Digital Image Processing	169
4.11 Problem Solving: Solve a Simpler Problem First	174
Chapter Summary	180
Interactive Review and Practice	182

- 3. What do these code fragments print? Note that the code fragments print one value per line, but for this exercise, assume all output is printed on one line, with each value separated by a blank space (i.e., 1 2 3 4).

GOOD JOB! ✓

Loop	Output	Explanation
<pre>n = 2 while n &lt; 10 :     print(n)     n = 2 * n print(n)</pre>	2 4 8 16	Note that 16 is printed <i>after</i> the loop has completed.
<pre>n = 2 while n &lt; 10 :     n = 2 * n     print(n)     print(n)</pre>	4 8 16 16	In the third iteration, n is set to 16, and that value is printed. It is printed again after the loop has completed.
<pre>n = 1 while n &lt; 100 :     print(n)     n = 10 * n</pre>	1 10	n is set to 100 in the third loop iteration. The loop exits because the condition n < 100 is no longer true.
<pre>n = 1 while n &lt;= 100 :     print(n)     n = 10 * n</pre>	1 10 100	n is set to 100 in the third loop iteration. The loop executes one more time because the condition n <= 100 is still true. When n is set to 100, the loop exits.
<pre>n = 1 while n &gt;= 100 :     print(n)     n = 10 * n print(n)</pre>	1	This loop is never entered because the condition n >= 100 is false at the outset. Thus, n still has a value of 1 when it is printed.

5 correct, 0 errors, 100%, 31 seconds

Start over

- 4. The half life of Cesium 90, a dangerous radioactive isotope released in the Fukushima reactor accident, is 30 years. That is, half of its radioactive material will decay every 30 years. Rearrange the following lines to produce a program that computes when 99 percent of the material is gone.

Back to Page

126 / 554



	Python For Everyone, Enhanced eText	...
	Cay S. Horstmann; Rance D. Necaise	
<a href="#">Expand</a>   <a href="#">Collapse</a>		
▼	3.9 Application: Input Validation	110
	Chapter Summary	123
▼	Interactive Review and Practice	124
▼	End-of-Chapter Exercises	EX3-1
▲	4. Loops	125
▼	4.1 The while Loop	126
▼	4.2 Problem Solving: Hand-Tracing	133
▼	4.3 Application: Processing Sentinel Values	135
▼	4.4 Problem Solving: Storyboards	139
▼	4.5 Common Loop Algorithms	141
▼	4.6 The for Loop	145
▼	4.7 Nested Loops	152
▼	4.8 Processing Strings	159
▼	4.9 Application: Random Numbers and Simulations	164
▼	4.10 Graphics: Digital Image Processing	169
▼	4.11 Problem Solving: Solve a Simpler Problem First	174
	Chapter Summary	180
▼	Interactive Review and Practice	182

Start over

- 4. The half life of Cesium 90, a dangerous radioactive isotope released in the Fukushima reactor accident, is 30 years. That is, half of its radioactive material will decay every 30 years. Rearrange the following lines to produce a program that computes when 99 percent of the material is gone.

GOOD JOB! ✓

```
percentage = 100
year = 0
while percentage > 1.0 :
    percentage = percentage * 0.5
    year = year + 30
print(year)
```

6 correct, 0 errors, 100%, 157 seconds

Start over

- 5. Complete the following program that computes and prints the sum of the first 100 positive integers.

intsum.py

```
1 number = 1
2 intSum = 0
3 while number <= 100:
4     intSum = intSum + number
5     number = number + 1
6
7 print(intSum)
```

[CodeCheck](#) [Reset](#)

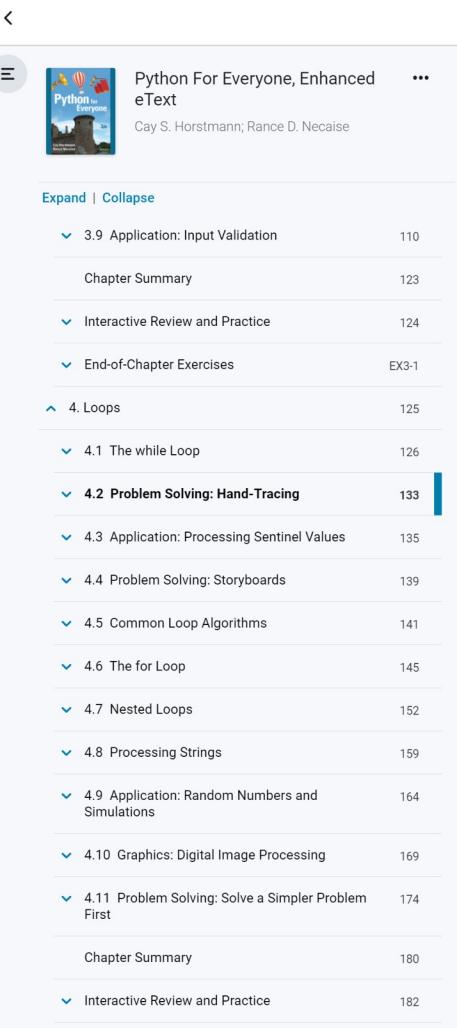
Running intsum.py

5050

pass

Score

1/1



**SELF CHECK**

- 1. Trace the following code segment. How many values are printed?

```
first = 1
second = 1
while second <= 10 :
    print(second)
    temp = first + second
    first = second
    second = temp
```

- 1
- 4
- 5
- 6

One correct, 0 errors, 100%

- 2. Trace through the following loop.

GOOD JOB! ✓

```
n = 1796
count = 0;
while n > 0 :
    digit = n % 10
    if digit == 6 or digit == 7 :
        count = count + 1
    n = n // 10
print(count)
```

<i>n</i>	<i>count</i>	<i>digit</i>	<i>Output</i>
1796	0	6	
	+		
174		4	
17		7	
	2		
4		1	
0			2

22 correct, 0 errors, 100%, 302 seconds

**Start over**

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 3.9 Application: Input Validation 110
- Chapter Summary 123
- Interactive Review and Practice 124
- End-of-Chapter Exercises EX3-1
- 4. Loops 125
- 4.1 The while Loop 126
- 4.2 Problem Solving: Hand-Tracing 133
- 4.3 Application: Processing Sentinel Values 135
- 4.4 Problem Solving: Storyboards 139
- 4.5 Common Loop Algorithms 141
- 4.6 The for Loop 145
- 4.7 Nested Loops 152
- 4.8 Processing Strings 159
- 4.9 Application: Random Numbers and Simulations 164
- 4.10 Graphics: Digital Image Processing 169
- 4.11 Problem Solving: Solve a Simpler Problem First 174
- Chapter Summary 180
- Interactive Review and Practice 182

- 3. What did the loop of the preceding problem do?

- Compute the sum of all digits in n
- Compute the sum of all digits in n that are 6 or 7
- Count all digits in n
- Count all digits in n that are 6 or 7

One correct, 0 errors, 100%

- 4. Trace through the following loop. For this exercise, you do not need to enclose string values in quotes.

GOOD JOB! ✓

```
s = "Fred"
r = ""
i = 0
while i < length of s:
    c = ith character of s
    r = c + r
    i = i + 1
```

r	i	c	Output
(empty string)	0	F	
F	1	r	
Fr	2	e	
fre	3	d	
derF	4		derF

Print r

19 correct, 0 errors, 100%, 95 seconds

Start over

- 5. What did the loop of the preceding problem do?

- Print all characters in the string s
- Print the string s in reverse order
- Print every other character in the string s
- Count the number of characters in s

One correct, 0 errors, 100%



Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise

Expand | Collapse

4. Loops	125
4.1 The while Loop	126
4.2 Problem Solving: Hand-Tracing	133
<b>4.3 Application: Processing Sentinel Values</b>	<b>135</b>
4.4 Problem Solving: Storyboards	139
4.5 Common Loop Algorithms	141
4.6 The for Loop	145
4.7 Nested Loops	152
4.8 Processing Strings	159
4.9 Application: Random Numbers and Simulations	164
4.10 Graphics: Digital Image Processing	169
4.11 Problem Solving: Solve a Simpler Problem First	174
Chapter Summary	180
Interactive Review and Practice	182
End-of-Chapter Exercises	EX4-1
5. Functions	183
6. Lists	245
7. Files and Exceptions	299



Back to Page

🔍 A A ⌂ ⌂ ⌂



- 1. Suppose the initialization of the salary variable in sentinel.py was changed to salary = -1. What output would be displayed for the inputs 1 2 4 -1?

No data was entered.

The while loop would never be entered. The user would never be prompted for input. Because count stays 0, the program would then print "No data was entered". For that reason, salary can be initialized with any value other than the sentinel -1.

- Average salary is -1  
 Average salary is 0.0  
 The program would abort with an exception.

One correct, 0 errors, 100%

- 2. Walk through the following code, where the input is 1 2 3 -1.

GOOD JOB! ✓

```
sum = 0
value = 0
while value != -1 :
    value = int(input("Enter a value: "))
    sum = sum + value
print(sum)
```

value	sum
0	0
1	1
2	3
3	6
-1	5

15 correct, 0 errors, 100%

Start over

- 3. Suppose the program segment of [the preceding problem](#) was intended to compute the sum of the inputs, where -1 is used as the sentinel. Which of the following best describes the bug in the code?

- The sentinel should not be included in the sum.  
 A sentinel of 0 should be used instead.

◀ 135 / 554 ▶



	Python For Everyone, Enhanced eText	...
	Cay S. Horstmann, Rance D. Necaise	
	<a href="#">Expand</a>   <a href="#">Collapse</a>	
	4. Loops	125
	4.1 The while Loop	126
	4.2 Problem Solving: Hand-Tracing	133
	<b>4.3 Application: Processing Sentinel Values</b>	<b>135</b>
	4.4 Problem Solving: Storyboards	139
	4.5 Common Loop Algorithms	141
	4.6 The for Loop	145
	4.7 Nested Loops	152
	4.8 Processing Strings	159
	4.9 Application: Random Numbers and Simulations	164
	4.10 Graphics: Digital Image Processing	169
	4.11 Problem Solving: Solve a Simpler Problem First	174
	Chapter Summary	180
	Interactive Review and Practice	182
	End-of-Chapter Exercises	EX4-1
	5. Functions	183
	6. Lists	245
	7. Files and Exceptions	299

- 3. Suppose the program segment of [the preceding problem](#) was intended to compute the sum of the inputs, where -1 is used as the sentinel. Which of the following best describes the bug in the code?

- The sentinel should not be included in the sum.
- A sentinel of 0 should be used instead.
- The loop does not terminate when the sentinel is read.
- The variable `input` should have been initialized.

One correct, 0 errors, 100%

- 4. Rearrange the following lines so that the code properly reads a sequence of integers and computes and prints their sum, using -1 as a sentinel. Not all lines are useful.

GOOD JOB! ✓

```
total = 0
value = int(input("Enter a value or -1 to finish: "))
while value != -1:
    total = total + value
    value = int(input("Enter a value or -1 to finish: "))
print(total)
```

```
while value >= -1 :
while value != 0 :
total = total + 1
```

4 correct, 0 errors, 100%

[Start over](#)

- 5. This program reads a sequence of floating-point values and computes their average. A value of zero is used as the sentinel. Improve the loop so that the loop is only exited if the user enters the letter Q.

```
computesum.py
1 total = 0.0
2 count = 0
3
4 # TODO: Fix this program so that it only stops reading numbers
5 # when the user enters Q.
6
7 value = input("Enter a real value or Q to quit: ")
8 while value != "0":
```

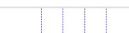


	Python For Everyone, Enhanced eText	...
	Cay S. Horstmann, Rance D. Necaise	
<a href="#">Expand</a>   <a href="#">Collapse</a>		
▲ 4. Loops	125	
▼ 4.1 The while Loop	126	
▼ 4.2 Problem Solving: Hand-Tracing	133	
▼ 4.3 Application: Processing Sentinel Values	135	
▼ 4.4 Problem Solving: Storyboards	139	
▼ 4.5 Common Loop Algorithms	141	
▼ 4.6 The for Loop	145	
▼ 4.7 Nested Loops	152	
▼ 4.8 Processing Strings	159	
▼ 4.9 Application: Random Numbers and Simulations	164	
▼ 4.10 Graphics: Digital Image Processing	169	
▼ 4.11 Problem Solving: Solve a Simpler Problem First	174	
Chapter Summary	180	
▼ Interactive Review and Practice	182	
▼ End-of-Chapter Exercises	EX4-1	
▼ 5. Functions	183	
▼ 6. Lists	245	
▼ 7. Files and Exceptions	299	



[Back to Page](#)

🔍 ⚡ ⌂ ⋮



4 correct, 0 errors, 100%

[Start over](#)

\*\*\* 5. This program reads a sequence of floating-point values and computes their average. A value of zero is used as the sentinel. Improve the loop so that the loop is only exited if the user enters the letter Q.

**computesum.py**

```
1 total = 0.0
2 count = 0
3
4 # TODO: Fix this program so that it only stops reading numbers
5 # when the user enters Q.
6
7 value = input("Enter a real value or Q to quit: ")
8 while value != "Q":
9     total += float(value)
10    count = count + 1
11    value = input("Enter a real value or Q to quit: ")
12
13 avg = total / count
14 print(avg)
```

[CodeCheck](#) [Reset](#)

**Running computesum.py**

**Test 1**

```
Enter a real value or Q to quit: 1
Enter a real value or Q to quit: 2
Enter a real value or Q to quit: 3
Enter a real value or Q to quit: 4
Enter a real value or Q to quit: 0
Enter a real value or Q to quit: 5
Enter a real value or Q to quit: 6
Enter a real value or Q to quit: 7
Enter a real value or Q to quit: 8
Enter a real value or Q to quit: 0
4.0
```

**pass**

**Test 2**

```
Enter a real value or Q to quit: 1
Enter a real value or Q to quit: 2
Enter a real value or Q to quit: 3
Enter a real value or Q to quit: 4
Enter a real value or Q to quit: 0
Enter a real value or Q to quit: 0
2.0
```

**pass**

**Score**

2/2

◀ 135 / 554 ▶



Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise



Expand | Collapse

▲ 4. Loops	125
▼ 4.1 The while Loop	126
▼ 4.2 Problem Solving: Hand-Tracing	133
▼ 4.3 Application: Processing Sentinel Values	135
▼ 4.4 Problem Solving: Storyboards	139
▼ 4.5 Common Loop Algorithms	141
▼ 4.6 The for Loop	145
▼ 4.7 Nested Loops	152
▼ 4.8 Processing Strings	159
▼ 4.9 Application: Random Numbers and Simulations	164
▼ 4.10 Graphics: Digital Image Processing	169
▼ 4.11 Problem Solving: Solve a Simpler Problem First	174
Chapter Summary	180
▼ Interactive Review and Practice	182
▼ End-of-Chapter Exercises	EX4-1
▼ 5. Functions	183
Chapter Summary	180
▼ Interactive Review and Practice	182
▼ End-of-Chapter Exercises	EX4-1
▼ 5. Functions	183
▼ 6. Lists	245
▼ 7. Files and Exceptions	299



Back to Page



- 1. For which of the following applications would a storyboard be most useful?

- A program to compute the sum of the first 1000 positive integers.
- A program to display the amount earned by a \$1000 investment at 3% interest after 1, 2, ..., 20 years.
- A program to display a table of Fahrenheit temperatures and their equivalent Celsius temperatures.
- A program to compute the maximum and minimum values from a sequence of test scores to be entered by the user.

One correct, 0 errors, 100%

- 2. Some unit conversions are not possible; for example, from meters to grams. Arrange the following lines into a storyboard illustrating what happens when an illegal conversion is attempted. Not all lines are used in the solution.

GOOD JOB! ✓

From unit (in, ft, mi, mm, cm, m, km, oz, lb, g, kg, tsp, tbsp, pint, gal): cm

To unit: oz

Sorry, incompatible units

From unit (in, ft, mi, mm, cm, m, km, oz, lb, g, kg, tsp, tbsp, pint, gal): cm

To unit: ft

Enter values, terminated by zero

Sorry, unknown unit

6 correct, 0 errors, 100%

Start over

- 3. How could one improve the user experience of the unit conversion program so that users are not frustrated by incompatible units?

- Terminate the program when two units are incompatible.
- Return a result of 0 when two units are incompatible:  $30 \text{ in} = 0 \text{ oz}$
- Only provide conversions between compatible units.
- List compatible units in the To unit prompt:  $\text{To unit (ft, mi, mm, cm, m, km)}$ :

One correct, 0 errors, 100%

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 3.7 Boolean Variables and Operators 101
- 3.8 Analyzing Strings 106
- 3.9 Application: Input Validation 110
- Chapter Summary 123
- Interactive Review and Practice 124
- End-of-Chapter Exercises EX3-1
- 4. Loops 125
  - 4.1 The while Loop 126
  - 4.2 Problem Solving: Hand-Tracing 133
  - 4.3 Application: Processing Sentinel Values 135
  - 4.4 Problem Solving: Storyboards 139
  - 4.5 Common Loop Algorithms 141
    - 4.6 The for Loop 145
    - 4.7 Nested Loops 152
    - 4.8 Processing Strings 159
    - 4.9 Application: Random Numbers and Simulations 164
    - 4.10 Graphics: Digital Image Processing 169
    - 4.11 Problem Solving: Solve a Simpler Problem First 174

while inputStr != "":  
 value = float(inputStr)  
 total = total + value  
 count = count + 1  
 inputStr = input("Enter value: ")  
  
if count > 0:  
 average = total / count  
else:  
 average = 0.0

 SELF CHECK

• 1. What does the variable total contain after the loop in the algorithm in Section 4.5.1 terminates when the user presses Enter at the first prompt without entering a value?

In the algorithm in Section 4.5.1, if the user presses Enter at the first prompt without entering a value, what does the variable total contain after the loop terminates?

- The program aborts with an exception.
- The empty string
- 1
- 0

When the user simply presses the Enter key at the first prompt, the loop is never executed. Because the variable total is initialized to 0 before the loop, that is the value it will contain after the loop.

One correct, 0 errors, 100%

• 2. What is wrong with this code?

```
total = 0
inputStr = input("Enter value: ")
while inputStr != "":
    value = float(inputStr)
    total = total + value
```

- The total variable should be initialized inside the loop.
- The total variable should be initialized with -1.
- The modification read is missing at the bottom of the loop.
- The input string should be converted to an integer instead of a floating-point value.

One correct, 0 errors, 100%

Back to Page

141 / 554



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 4.3 Application: Processing Sentinel Values 135
- 4.4 Problem Solving: Storyboards 139
- 4.5 Common Loop Algorithms 141
- 4.6 The for Loop 145
- 4.7 Nested Loops 152
- 4.8 Processing Strings 159
- 4.9 Application: Random Numbers and Simulations 164
- 4.10 Graphics: Digital Image Processing 169
- 4.11 Problem Solving: Solve a Simpler Problem First 174

Chapter Summary 180

- Interactive Review and Practice 182
- End-of-Chapter Exercises EX4-1

- 5. Functions 183
- 6. Lists 245

**SELF CHECK**

• 1. What output does this for loop generate?

```
for j in range(5) :  
    value = j * 2  
    print("%d, " % value, end="")
```

0, 1, 2, 3, 4,  
 0, 2, 4, 6, 8,  
 2, 4, 6, 8, 10,  
 0, 2, 4, 6, 8, 10,

One correct, 0 errors, 100%

• 2. What do these loops print? (Note, the end="" argument suppresses the new line so all values print on one line.)

GOOD JOB! ✓

Loop	Output	Explanation
<pre>for i in range(4) :     print("%d " % i, end="")</pre>	0 1 2 3	In the loop for i in range(n), i is set to 0, 1, 2, ..., up to (but not including) n.
<pre>for i in range(4) :     print("%d " % (3 - i), end="")</pre>	3 2 1 0	When i goes from 0 to 3, 3 - i goes down from 3 to 0.



< 🔍 A ⌂ ...

 Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

[Expand](#) | [Collapse](#)

- 4.3 Application: Processing Sentinel Values 135
- 4.4 Problem Solving: Storyboards 139
- 4.5 Common Loop Algorithms 141
- 4.6 The for Loop 145
- 4.7 Nested Loops 152
- 4.8 Processing Strings 159
- 4.9 Application: Random Numbers and Simulations 164
- 4.10 Graphics: Digital Image Processing 169
- 4.11 Problem Solving: Solve a Simpler Problem First 174

Chapter Summary 180

Interactive Review and Practice 182

End-of-Chapter Exercises EX4-1

- 5. Functions 183
- 6. Lists 245

•• 2. What do these loops print? (Note, the end="" argument suppresses the new line so all values print on one line.)

GOOD JOB! ✓

Loop	Output	Explanation
<pre>for i in range(4) :     print("%d" % i, end="")</pre>	0 1 2 3	In the loop for <code>i</code> in <code>range(n)</code> , <code>i</code> is set to 0, 1, 2, ..., up to (but not including) <code>n</code> .
<pre>for i in range(4) :     print("%d" % (3 - i), end="")</pre>	3 2 1 0	When <code>i</code> goes from 0 to 3, <code>3 - i</code> goes down from 3 to 0.
<pre>name = "Fred" for ch in name :     print(ch, end="")</pre>	Fred	The <code>for</code> loop is used to iterate over containers, one element at a time.
<pre>name = "Fred" for i in range(len(name) - 1, -1, -1) :     print(name[i], end="")</pre>	derF	This is one way of traversing the characters backwards.
<pre>for i in range(0, 8, 2) :     print("%d" % i, end="")</pre>	0 2 4 6	When passing three arguments to the <code>range</code> function, the third argument is the step value. This is how you implement a step size of 2.
<pre>for i in range(3, 0, -1) :     print("%d" % i, end="")</pre>	3 2 1	This is how you step backwards.

6 correct, 0 errors, 100%, 225 seconds

[Start over](#)

•• 3. Trace through the following code. Do not include quotes when entering string values.

< 145 / 554 >

<      Q      Aa      ⌂      ...

 Python For Everyone, Enhanced eText    ...  
Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 4.3 Application: Processing Sentinel Values    135
- 4.4 Problem Solving: Storyboards    139
- 4.5 Common Loop Algorithms    141
- 4.6 The for Loop**    145
- 4.7 Nested Loops    152
- 4.8 Processing Strings    159
- 4.9 Application: Random Numbers and Simulations    164
- 4.10 Graphics: Digital Image Processing    169
- 4.11 Problem Solving: Solve a Simpler Problem First    174
- Chapter Summary    180
- Interactive Review and Practice    182
- End-of-Chapter Exercises    EX4-1
- 5. Functions    183
- 6. Lists    245
- 7. Files and Exceptions    299
- 8. Sets and Dictionaries    357
- 9. Objects and Classes    393
- 10. Inheritance    443

---

\*\* 3. Trace through the following code. Do not include quotes when entering string values.

GOOD JOB! ✓

```
message = "Hello Fred"
result = ""
n = len(message)
for i in range(0, n, 2):
    ch = message[i]
    result = result + ch
print(result)
```

message	n	i	ch	result
Hello Fred				(empty string)
	10	0	H	H
		2	I	HI
		4	O	HO
		6	F	HOF
		8	e	HOFe

22 correct, 0 errors, 100%, 186 seconds

Start over

---

\*\* 4. Rearrange the following lines to produce a loop that prints all characters of a string, one per line. Not all lines are useful.

GOOD JOB! ✓

```
message = "Hello, World!"
for i in range(len(message)) :
    ch = message[i]
    print(ch)
```

```
for i in range(0, len(message) + 1) :
for i in range(1, len(message)) :
    ch = message[i + 1]
    print(i)
```

4 correct, 0 errors, 100%, 60 seconds

Start over

---

\*\*\* 5. Write a program that prints a countdown of the form

W    145 / 554 >



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

[Expand](#) | [Collapse](#)

▼ 4.3 Application: Processing Sentinel Values	135
▼ 4.4 Problem Solving: Storyboards	139
▼ 4.5 Common Loop Algorithms	141
<b>4.6 The for Loop</b>	145
▼ 4.7 Nested Loops	152
▼ 4.8 Processing Strings	159
▼ 4.9 Application: Random Numbers and Simulations	164
▼ 4.10 Graphics: Digital Image Processing	169
▼ 4.11 Problem Solving: Solve a Simpler Problem First	174
Chapter Summary	180
▼ Interactive Review and Practice	182

4 correct, 0 errors, 100%, 60 seconds

[Start over](#)

••• 5. Write a program that prints a countdown of the form

10...9...8...7...6...5...4...3...2...1...0...Liftoff

**countdown.py**

```
1 n = int(input("Enter a positive integer: "))
2 for i in range(n,-1,-1):
3     print("%d..." % i, end="")
4 print("Liftoff")
```

[CodeCheck](#) [Reset](#)

**Running countdown.py**

**Test 1**

Enter a positive integer: 10  
10...9...8...7...6...5...4...3...2...1...0...Liftoff

**pass**

**Test 2**

Enter a positive integer: 5  
5...4...3...2...1...0...Liftoff

**pass**

**Score**

2/2



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

  4.1 The while Loop      126

  4.2 Problem Solving: Hand-Tracing      133

  4.3 Application: Processing Sentinel Values      135

  4.4 Problem Solving: Storyboards      139

  4.5 Common Loop Algorithms      141

  4.6 The for Loop      145

4.7 Nested Loops      152

  4.8 Processing Strings      159

  4.9 Application: Random Numbers and Simulations      164

  4.10 Graphics: Digital Image Processing      169

  4.11 Problem Solving: Solve a Simpler Problem First      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Files and Exceptions      266

SELF CHECK

1. What output do these nested for loops generate?

```
for j in range(1, 4) :  
    for k in range(2, 4) :  
        print("%d, " % k, end="")
```

1, 2, 3, 1, 2, 3,  
 2, 3, 4, 2, 3, 4,  
 2, 3, 2, 3,  
 2, 3, 2, 3, 2, 3,

One correct, 0 errors, 100%

2. Trace through the following code:

GOOD JOB! ✓

```
for n in range(2, 4) :  
    p = 1  
    for i in range(1, 4) :  
        p = p * n  
        print("%2d" % p, end="")  
    print()
```

n	i	p
2	1	1
	2	2
	3	4
3	1	3
	2	9
	3	27

18 correct, 0 errors, 100%, 130 seconds

Start over

3. Rearrange the following lines of code so that they print the following shape. Not all lines are useful.

```
[] [] [] [] []  
[] [] [] [] []  
[] [] [] [] []
```

Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

4.1 The while Loop      126

4.2 Problem Solving: Hand-Tracing      133

4.3 Application: Processing Sentinel Values      135

4.4 Problem Solving: Storyboards      139

4.5 Common Loop Algorithms      141

4.6 The for Loop      145

4.7 Nested Loops      152

4.8 Processing Strings      159

4.9 Application: Random Numbers and Simulations      164

4.10 Graphics: Digital Image Processing      169

4.11 Problem Solving: Solve a Simpler Problem First      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Files and Exceptions      266

• 3. Rearrange the following lines of code so that they print the following shape. Not all lines are useful.

```
[] [] [] []
[] [] [] []
[] [] [] []
```

GOOD JOB! ✓

```
for i in range(0, 3) :
    for j in range(0, 5) :
        print("[]", end="")
    print()
```

4 correct, 0 errors, 100%, 68 seconds

Start over

• 4. Which of the following changes must be made so that the [powertable.py](#) program displays all powers from  $x^0$  to  $x^{99}$ ?

True     False Change line 6 to NMAX = 5

True     False Change lines 10 and 14 to for n in range(0, NMAX + 1) :

True     False Change line 20 to for x in range(0, XMAX + 1) :

True     False Change line 22 to for n in range(0, NMAX + 1) :

4 correct, 0 errors, 100%

• 5. If you make the change(s) identified in the preceding problem, how many values are displayed?

40

50

60

The outer loop is executed 10 times, and the inner loop 6 times.

66

One correct, 0 errors, 100%

Back to Page

152 / 554

Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

4.1 The while Loop      126

4.2 Problem Solving: Hand-Tracing      133

4.3 Application: Processing Sentinel Values      135

4.4 Problem Solving: Storyboards      139

4.5 Common Loop Algorithms      141

4.6 The for Loop      145

4.7 Nested Loops      152

4.8 Processing Strings      159

4.9 Application: Random Numbers and Simulations      164

4.10 Graphics: Digital Image Processing      169

4.11 Problem Solving: Solve a Simpler Problem First      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Errors and Exceptions      266

SELF CHECK

Note the use of the `lower` method in the logical expression. This method is used to convert each uppercase letter to its corresponding lowercase letter before checking to see if it is a vowel. That way, we limit the number of characters that must be specified in the literal string.

• 1. Assuming that `string` is a variable that contains a string, what is the purpose of the code segment below?

```
count = 0
for char in string :
    if char != "*":
        count = count + 1
```

To count the number of characters in `string` that are asterisks.  
 To count the number of characters in `string` that are not asterisks.  
 To count the number of characters in `string` that are punctuation symbols.  
 To count the number of characters in `string` that are not punctuation symbols.

One correct, 0 errors, 100%

••• 2. Complete this program that counts how many times the letter "e" appears in a given string. For example, if the string is "redeemed", then the count is 4. Do not use the `count` string method. Instead, use a `for` loop.

```
strings.py
1 word = input("Enter a word: ")
2 count = 0
3 for e in word:
4     if e == "e":
5         count = count + 1
6 print("count:", count)
```

CodeCheck    Reset

Running strings.py

Test 1

```
Enter a word: Hello
count: 1
```

pass

Test 2

```
Enter a word: Mississippi
count: 0
```

pass

Test 3

```
Enter a word: nevertheless
count: 4
```

pass

Score

3/3

Back to Page      159 / 554

The screenshot shows a digital book interface for "Python For Everyone, Enhanced eText" by Cay S. Horstmann and Rance D. Necaise. The left sidebar contains a table of contents with chapters 1 through 7 and their page numbers. The main content area is titled "4.9 Application: Random Numbers and Simulations". A sub-section "4.9.1 Generating Random Numbers" is shown with a text block explaining simulations and a code editor window displaying a Python script named "sec09\_01/randomtest.py". The script prints ten random numbers between 0 and 1. Below the code is an "Output" section showing the generated numbers. A note at the bottom states that the numbers are pseudorandom.

Python For Everyone, Enhanced eText  
Cay S. Horstmann, Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

4.1 The while Loop      126

4.2 Problem Solving: Hand-Tracing      133

4.3 Application: Processing Sentinel Values      135

4.4 Problem Solving: Storyboards      139

4.5 Common Loop Algorithms      141

4.6 The for Loop      145

4.7 Nested Loops      152

4.8 Processing Strings      159

4.9 Application: Random Numbers and Simulations      164

4.10 Graphics: Digital Image Processing      169

4.11 Problem Solving: Solve a Simpler Problem First      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Errors and Exceptions      266

4.9 Application: Random Numbers and Simulations

In a simulation, you use the computer to simulate an activity in the real world (or an imaginary one). Simulations are commonly used for predicting climate change, analyzing traffic, picking stocks, and many other applications in science and business. In many simulations, one or more loops are used to modify the state of a system and observe the changes. You will see examples in the following sections.

You can introduce randomness by calling the random number generator.

A simulation program uses the computer to simulate an activity in the real world (or an imaginary one). Simulations are commonly used for predicting climate change, analyzing traffic, picking stocks, and many other applications in science and business. In many simulations, one or more loops are used to modify the state of a system and observe the changes. You will see examples in the following sections.

Many events in the real world are difficult to predict with absolute precision, yet we can sometimes know the average behavior quite well. For example, a store may know from experience that a customer arrives every five minutes. Of course, that is an average—customers don't arrive in five minute intervals. To accurately model customer traffic, you want to take that random fluctuation into account. Now, how can you run such a simulation in the computer?

The Python library has a *random number generator* that produces numbers that appear to be completely random. Calling `random()` yields a random floating-point number that is  $\geq 0$  and  $< 1$ . Call `random()` again, and you get a different number. The `random` function is defined in the `random` module.

The following program calls `random()` ten times.

**sec09\_01/randomtest.py**

```
1 # This program prints ten random numbers between 0 and 1.
2 #
3 # from random import random
4 #
5 for i in range(10):
6     value = random()
7     print(value)
```

Run | Reset

**Output**

```
0.06220949771583904
0.9177602169207496
0.06427651921708388
0.8637764232095644
0.750379438559716
0.3437419402228793
0.9621804977177587
0.851191039261271
0.7916719886617386
0.44586234324476315
```

Actually, the numbers are not completely random. They are drawn from sequences of numbers that don't repeat for a long time. These sequences are actually computed from fairly simple formulas; they just behave like random numbers (see [Exercise • P4.27](#)). For that reason, they are often called **pseudorandom** numbers.

Back to Page | 164 / 554

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

  4.1 The while Loop      126

  4.2 Problem Solving: Hand-Tracing      133

  4.3 Application: Processing Sentinel Values      135

  4.4 Problem Solving: Storyboards      139

  4.5 Common Loop Algorithms      141

  4.6 The for Loop      145

  4.7 Nested Loops      152

  4.8 Processing Strings      159

  4.9 Application: Random Numbers and Simulations      164

  4.10 Graphics: Digital Image Processing      169

  4.11 Problem Solving: Solve a Simpler Problem First      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Errors and Exceptions      288

SELF CHECK

•• 1. Suppose you are asked to find all words from a list of words in which no letter is repeated. What simpler problem could you try first?

✓ True False

Print the words in which the first letter is not repeated.

If you can find out whether the first letter is not repeated, you can next check whether the second letter is not repeated, and so on.

✓ True False

Read one word and print it if no letter is repeated.

This is a simplification, but it is not as useful. The difficult part of the problem is to find repeated letters, not to read multiple words.

✓ True False

Find all words in which all letters are repeated.

Checking whether all letters are repeated is a different problem that is no easier than the original problem.

✓ True False

Check whether the first word occurs again in the list of words.

This may be simple to do, but it has no relationship to the original problem.

4 correct, 0 errors, 100%

•• 2. Consider the task of finding numbers in a string. For example, the string “In 1987, a typical personal computer cost \$3,000 and had 512 kilobytes of RAM.” has three numbers. Break this task down into a sequence of simpler tasks.

Drag the statements to the left, in increasing order of complexity. Not all suggestions are useful.

GOOD JOB!

Find the position of the first digit in a string

Find the position of the last digit in a string

Find the position of the first non-digit after a given position

Extract the first integer from a string

Find the positions of all commas in a string

Print all integers from a string

Back to Page      174 / 554

Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise

Expand | Collapse

Quick Reference      xiii

1. Introduction      1

2. Programming with Numbers and Strings      23

3. Decisions      73

4. Loops      125

  4.1 The while Loop      126

  4.2 Problem Solving: Hand-Tracing      133

  4.3 Application: Processing Sentinel Values      135

  4.4 Problem Solving: Storyboards      139

  4.5 Common Loop Algorithms      141

  4.6 The for Loop      145

  4.7 Nested Loops      152

  4.8 Processing Strings      159

  4.9 Application: Random Numbers and Simulations      164

  4.10 Graphics: Digital Image Processing      169

**4.11 Problem Solving: Solve a Simpler Problem First**      174

Chapter Summary      180

Interactive Review and Practice      182

End-of-Chapter Exercises      EX4-1

5. Functions      183

6. Lists      245

7. Files and Exceptions      260

•• 3. You want to remove “red eyes” from images and are looking for red circles. What simpler problem can you start with?

✓ True False

You can change all red pixels to black.

That is easy to do but it doesn’t address the problem of finding eyes.

✓ True False

You can look for circles of any color.

That does not simplify the problem.

✓ True False

You can look for a single red pixel.

That’s certainly a start.

✓ True False

You can look for a block of nine neighboring red pixels.

That would be a good second step.

**4 correct, 0 errors, 100%**

•• 4. Suppose that a.png, b.png, c.png, and d.png have dimensions 130 × 104, 260 × 120, 100 × 128, 84 × 128. Walk through the following loop that arranges them in a row.

**GOOD JOB!** ✓

i	x
1	140
2	410
3	520
4	614

```
GAP = 10
NAMES = "abcd"
NUM PICTURES = len(NAMES)

win = GraphicsWindow()
canvas = win.canvas()

pic = GraphicsImage("a.gif")
canvas.drawImage(0, 0, pic)
x = pic.width() + GAP
for i in range(1, NUM PICTURES) :
    pic2 = GraphicsImage(NAMES[i] + ".gif")
    canvas.drawImage(x, 0, pic2)
    x = x + pic2.width() + GAP
    wait()
```

**7 correct, 0 errors, 100%, 160 seconds**

**Start over**

Back to Page

174 / 554