

The screenshot shows a digital textbook 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 and page numbers. A "SELF CHECK" section is open, featuring a clock icon. It displays three programming questions with code snippets, multiple-choice answers, and feedback sections. The first question asks about variable assignment after an if statement. The second asks for the output of a print statement. The third is a general statement analysis. A "GOOD JOB!" message with a checkmark is shown at the bottom right.

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

Title Page i

Getting the Most from Your eText iii

Quick Reference xiii

1. Introduction 1

2. Programming with Numbers and Strings 23

2.1 Variables 24

2.2 Arithmetic 31

2.3 Problem Solving: First Do It By Hand 39

2.4 Strings 41

2.5 Input and Output 48

2.6 Graphics: Simple Drawings 58

Chapter Summary 71

Interactive Review and Practice 72

End-of-Chapter Exercises EX-1

3. Decisions 73

3.1 The If Statement 74

3.2 Relational Operators 79

3.3 Nested Branches 87

3.4 Multiple Alternatives 91

3.5 Problem Solving: Flowcharts 96

3.6 Problem Solving: Test Cases 99

3.7 Boolean Variables and Operators 101

3.8 Analyzing Strings 106

3.9 Application: Input Validation 110

Chapter Summary 123

**SELF CHECK**

• 1. Assuming that variable `balance` has a value of 1200, what is the value in variable `rate` after the code segment below is executed?

```
rate = 0.5
if balance > 1000 :
    rate = rate + 0.25
```

0.25  
 0.5  
 0.75  
 1.0

One correct, 0 errors, 100%

• 2. Assuming that variable `temperature` has a value of 88, what output is generated by the code segment below?

```
if temperature > 90 :
    print("It is")
else :
    print("It is not")
print("a hot day.")
```

It is  
 It is not  
 It is a hot day.  
 It is not a hot day.

One correct, 0 errors, 100%

• 3. Consider the following statement:

```
if originalPrice > 100 :
    discountedPrice = originalPrice - 20
else :
    discountedPrice = originalPrice - 10
```

What is the discounted price for each of the following original prices?

GOOD JOB! ✓

originalPrice	discountedPrice	Explanation
100	80	This original value is less than 100. Therefore the statement is false so discountedPrice is not assigned.

74 / 554



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

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

Title Page	i
Getting the Most from Your eText	iii
Quick Reference	xiii
1. Introduction	1
2. Programming with Numbers and Strings	23
2.1 Variables	24
2.2 Arithmetic	31
2.3 Problem Solving: First Do It By Hand	39
2.4 Strings	41
2.5 Input and Output	48
2.6 Graphics: Simple Drawings	58
Chapter Summary	71
Interactive Review and Practice	72
End-of-Chapter Exercises	EX2-1
3. Decisions	73
3.1 The if Statement	74
3.2 Relational Operators	79
3.3 Nested Branches	87
3.4 Multiple Alternatives	91
3.5 Problem Solving: Flowcharts	96
3.6 Problem Solving: Test Cases	99
3. Decisions	73
3.1 The if Statement	74
3.2 Relational Operators	79
3.3 Nested Branches	87
3.4 Multiple Alternatives	91
3.5 Problem Solving: Flowcharts	96
3.6 Problem Solving: Test Cases	99
3.7 Boolean Variables and Operators	101
3.8 Analyzing Strings	106
3.9 Application: Input Validation	110
Chapter Summary	123

• 3. Consider the following statement:

```
if originalPrice > 100 :
    discountedPrice = originalPrice - 20
else :
    discountedPrice = originalPrice - 10
```

What is the discounted price for each of the following original prices?

**GOOD JOB!** ✓

originalPrice	discountedPrice	Explanation
95	85	The original price is less than 100. Thus, the statement in the <code>else</code> branch is executed.
100	90	The original price is not greater than 100. Again, the statement in the <code>else</code> branch is executed.
105	85	Because the original price is greater than 100, the statement in the first branch is executed.

**3 correct, 0 errors, 100%, 22 seconds**

[Start over](#)

• 4. Consider the following statement:

```
if hour < 21 :
    response = "Goodbye"
else :
    response = "Goodnight"
```

Determine the value of `response` when `hour` has the values given in the table below. (You do not need to put quotes around the response strings.)

**GOOD JOB!** ✓

hour	response	Explanation
20	Goodbye	20 < 21, and the first branch of the statement executes.
22	Goodnight	It is not true that 22 < 21, so the <code>else</code> clause executes.
21	Goodnight	It is also not true that 21 < 21.
3	Goodbye	Because 3 < 21, the first branch executes, even though one might want to wish a good night at 3 am.

**4 correct, 0 errors, 100%, 21 seconds**

[Start over](#)

• 5. For this activity, trace through the code by clicking on the line that will be executed next.

Observe the inputs as they appear in the table below. The variables `fuelCapacity` and `fuelAmount` hold the size of a vehicle's fuel tank and the actual amount of fuel in the tank. If less than 20 percent is remaining in the tank, a status light should show a red color; otherwise it shows a green color.

**GOOD JOB!** ✓

userInput	fuelAmount	fuelCapacity	Output
"10"	10		
"14"		14	green
"2.5"		2.5	
"2.5"		2.5	
"14"		14	red
"4"	4		
"18"		18	green

**3 correct, 0 errors, 100%, 113 seconds**

[Start over](#)

3.1 (p.2)

3.1 (p.2) - The if Statement (3, 4, 5)



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

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

- [Title Page](#)
- [Getting the Most from Your eText](#)
- [Quick Reference](#)
- [1. Introduction](#)
- [2. Programming with Numbers and Strings](#)
  - [2.1 Variables](#)
  - [2.2 Arithmetic](#)
  - [2.3 Problem Solving: First Do It By Hand](#)
  - [2.4 Strings](#)
  - [2.5 Input and Output](#)
  - [2.6 Graphics: Simple Drawings](#)
- [Chapter Summary](#)
- [Interactive Review and Practice](#)
- [End-of-Chapter Exercises](#)
- [3. Decisions](#)
  - [3.1 The if Statement](#)
  - [3.2 Relational Operators](#)
  - [3.3 Nested Branches](#)
  - [3.4 Multiple Alternatives](#)
  - [3.5 Problem Solving: Flowcharts](#)
  - [3.6 Problem Solving: Test Cases](#)
  - [3.7 Boolean Variables and Operators](#)
  - [3.8 Analyzing Strings](#)
  - [3.9 Application: Input Validation](#)
- [Chapter Summary](#)



• 1. Assuming that variable `dollars` has a value of 50 and variable `cost` has a value of 100, which of the following conditions is False?

`cost <= dollars + 75`  
 `cost != dollars`  
 `dollars + 50 >= cost`  
 `dollars + 50 != cost`

**One correct, 0 errors, 100%**

• 2. Which expression is the correct condition for the `if` statement in the following code segment? The code is intended to add 5 to variable `shippingCost` if the value of variable `weight` is at least 10.

```
if _____:
    shippingCost = shippingCost + 5
```

`weight >= 10`  
 `weight ==> 10`  
 `weight == 10`  
 `weight > 10`

**One correct, 0 errors, 100%**

• 3. Check whether the following conditions evaluate to True or False, or whether they contain an error. Answer with True, False, or error.

**GOOD JOB!** ✓

Condition	Outcome	Explanation
<code>2 &lt;= 3</code>	True	2 is less than 3.
<code>3 &lt;= 3</code>	True	3 is equal to 3.
<code>4 &lt;= 3</code>	False	4 is greater than 3.
<code>4 ==&gt; 3</code>	error	The "greater than or equal" symbol is >=.
<code>4 != 4</code>	False	4 is equal to 4, so it is false to state that it isn't.
<code>3 + 1 = 4</code>	error	The symbol for equality testing is ==.
<code>4.35 * 100 == 435</code>	False	Due to roundoff, 4.35 * 100 is 434.9999999999999. It does not usually make sense to use == with floating-point numbers.
<code>"Hello" &lt; 5</code>	error	You cannot compare strings with integers.

**8 correct, 0 errors, 100%, 54 seconds**

[Start over](#)



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

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

Title Page	i
Getting the Most from Your eText	iii
Quick Reference	xiii
1. Introduction	1
2. Programming with Numbers and Strings	23
2.1 Variables	24
2.2 Arithmetic	31
2.3 Problem Solving: First Do It By Hand	39
2.4 Strings	41
2.5 Input and Output	48
2.6 Graphics: Simple Drawings	58
Chapter Summary	71
Interactive Review and Practice	72
End-of-Chapter Exercises	EX2-1
3. Decisions	73
3.1 The If Statement	74
<b>3.2 Relational Operators</b>	<b>79</b>
3.3 Nested Branches	87
3.4 Multiple Alternatives	91
3.5 Problem Solving: Flowcharts	96
3.6 Problem Solving: Test Cases	99
3.7 Boolean Variables and Operators	101
3.8 Analyzing Strings	106
3.9 Application: Input Validation	110
Chapter Summary	123

"Hello" < 5	error	with floating-point numbers. You cannot compare strings with integers.
-------------	-------	---

8 correct, 0 errors, 100%, 54 seconds

[Start over](#)

• 4. Which of the following conditions evaluate to true, provided a is 3 and b is 4?

✓  True  False a + 1 <= b  
 Because a + 1 is 4, the condition reduces to 4 <= 4, which is true.

✓  True  False a > b - 1  
 Because b - 1 is 3, the condition reduces to 3 > 3, which is not true. The two sides are equal.

✓  True  False a >= b - 1  
 Both sides of the relational operator have a value of 3. Thus, they are equal.

✓  True  False a \* 2 != b + 2  
 Both a \* 2 and b + 2 are equal to 6. Thus both sides of the relational operator are equal.

4 correct, 0 errors, 100%

• 5. Provide conditions for the given tasks.

**GOOD JOB!** ✓

Task	Condition	Explanation
Give the opposite of the condition floor > 13	floor <= 13	Note that the opposite of > is <= and not <.
A programmer wrote the condition floor = 13	floor == 13	You need to use ==, not =, to compare whether two values are equal.
to test whether the floor is 13. Fix the error.		
Supply a condition in this if statement to test whether the user entered a Y:	response == "Y"	You can compare strings with ==.
<pre>response = input("Enter Y to quit.") if ...:     print("Goodbye")</pre>		
How do you test that a string response is the empty string?	response == ""	You can use response == "" or len(response) == 0.

4 correct, 0 errors, 100%, 52 seconds

[Start over](#)

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1
- 3. Decisions 73
- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96
- 3.6 Problem Solving: Test Cases 99
- 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
- 5. Functions 183
- 6. Lists 245
- 7. Files and Exceptions 299
- 8. Sets and Dictionaries 357
- 9. Objects and Classes 393

**SELF CHECK**

\*\* 1. Consider the taxes.py program above. Which of the following statements are true?

True  False

The program result is unchanged if the first nested if statement in the tax calculator program is changed from

```
if income <= RATE1_SINGLE_LIMIT :  
    to  
    if income < RATE1_SINGLE_LIMIT :
```

After the change, the computation is  $0.10 \times 32000 + 0.25 \times (32000 - 32000)$ .

True  False

If Harry and Sally each make \$40,000 per year, they would save taxes if they married each other.

Their individual tax is \$5,200 each, and if they married, they would pay \$10,400. Actually, taxpayers in higher tax brackets (which our program does not model) may pay higher taxes when they marry, a phenomenon known as the marriage penalty.

True  False

When you work hard and get a raise, you might end up with less money after taxes.

The higher tax rate is only applied on the income in the higher bracket. Suppose you are single and make \$31,900. Should you try to get a \$200 raise? Absolutely: you get to keep 90 percent of the first \$100 and 75 percent of the next \$100.

**3 correct, 0 errors, 100%**

• 2. Assuming that variable number has a value of 777, what output is generated by the code segment below?

```
if number > 1000 :  
    if number % 2 == 0 :  
        print("A large even number")  
    else :  
        print("A large odd number")  
else :  
    if number % 2 == 0 :  
        print("A small even number")  
    else :  
        print("A small odd number")
```

A large even number  
 A large odd number  
 A small even number  
 A small odd number

**One correct, 0 errors, 100%**

\*\* 3. In this activity, trace through the tax calculation with four different sets of inputs. For each input set, click on the conditions of the if statement that are tested and then on the statement that is executed (which may be another if statement).



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

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

- [2.4 Strings](#) 41
- [2.5 Input and Output](#) 48
- [2.6 Graphics: Simple Drawings](#) 58
- [Chapter Summary](#) 71
- [Interactive Review and Practice](#) 72
- [End-of-Chapter Exercises](#) EX2-1

---

- [3. Decisions](#) 73
- [3.1 The if Statement](#) 74
- [3.2 Relational Operators](#) 79
- [3.3 Nested Branches](#) 87
- [3.4 Multiple Alternatives](#) 91
- [3.5 Problem Solving: Flowcharts](#) 96
- [3.6 Problem Solving: Test Cases](#) 99
- [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
- [5. Functions](#) 183
- [6. Lists](#) 245
- [7. Files and Exceptions](#) 299
- [8. Sets and Dictionaries](#) 357
- [9. Objects and Classes](#) 393

**.. 3.** In this activity, trace through the tax calculation with four different sets of inputs. For each input set, click on the conditions of the if statement that are tested and then on the statement that is executed (which may be another if statement).

**GOOD JOB!** ✓

```
income = float(input("Income: "))
maritalStatus = input("Marital status: ")
if maritalStatus == "s" :
    if income <= 32000 :
        tax = 0.10 * income
    else :
        tax = 3200 + 0.25 * (income - 32000)
else :
    if income <= 64000 :
        tax = 0.10 * income
    else :
        tax = 6400 + 0.25 * (income - 64000)
```

maritalStatus	income	tax
s	30000	3000
s	40000	4200
s	40000	4200
m	70000	7000
m	79000	7900

**12 correct, 0 errors, 100%, 98 seconds**

[Start over](#)

---

**.. 4.** Consider the code segment below:

```
if average >= 60 :
    print("Passing")
    if average < 70 :
        print("but just barely")
    else :
        print("Failing")
```

and answer each of the following questions.

**GOOD JOB!** ✓

**4 correct, 0 errors, 100%, 11 seconds**

[Start over](#)

---

**.. 5.** Rearrange the lines of code below to produce a program fragment that reads user input and prints whether a time (such as 9 am) is in the morning, afternoon, or evening. Prompt the user for the hour before the suffix.

**GOOD JOB!** ✓



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

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

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1
- 3. Decisions 73
- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96
- 3.6 Problem Solving: Test Cases 99
- 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
- 5. Functions 183
- 6. Lists 245
- 7. Files and Exceptions 299
- 8. Sets and Dictionaries 357
- 9. Objects and Classes 393

•• 4. Consider the code segment below:

```
if average >= 60 :
    print("Passing")
    if average < 70 :
        print("but just barely")
    else :
        print("Failing")
```

and answer each of the following questions.

GOOD JOB! ✓

Question	Answer	Explanation
If the variable <code>average</code> contains a value of 45.0, what output is generated?	Failing	The first condition ( <code>average &gt;= 60</code> ) is false, which results in the execution of the statement in the <code>else</code> branch.
If the variable <code>average</code> contains a value of 87.4, what output is generated?	Passing	The first condition ( <code>average &gt;= 60</code> ) is true, but the second ( <code>average &lt; 70</code> ) is not.
What is the smallest value for <code>average</code> that will result in both <code>Passing</code> and <code>but just barely</code> being printed?	60.0	<code>Passing but just barely</code> is printed for values greater than or equal to 60 and less than 70.
What is the largest value for <code>average</code> (to one decimal place) that will result in <code>Failing</code> being printed?	59.9	Floating-point values are not rounded before they are compared. Thus, 59.9 is the largest value to one decimal place that is less than 60.

4 correct, 0 errors, 100%, 11 seconds Start over

•• 5. Rearrange the lines of code below to produce a program fragment that reads user input and prints whether a time (such as 9 am) is in the morning, afternoon, or evening. Prompt the user for the hour before the suffix.

GOOD JOB! ✓

```
hour = int(input("Hour: "))
suffix = input("Suffix: ")
if suffix == "pm" :
    if hour < 6 :
        time = "afternoon"
    else :
        time = "evening"
else :
    time = "morning"
print("Good", time)
```

10 correct, 0 errors, 100%, 275 seconds Start over



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

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

2.4 Strings	41
2.5 Input and Output	48
2.6 Graphics: Simple Drawings	58
Chapter Summary	71
Interactive Review and Practice	72
End-of-Chapter Exercises	EX2-1
3. Decisions	73
3.1 The if Statement	74
3.2 Relational Operators	79
3.3 Nested Branches	87
<b>3.4 Multiple Alternatives</b>	<b>91</b>
3.5 Problem Solving: Flowcharts	96
3.6 Problem Solving: Test Cases	99
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
5. Functions	183
6. Lists	245
7. Files and Exceptions	299
8. Sets and Dictionaries	357
9. Objects and Classes	393

**SELF CHECK**

• 1. Which expression is correct for the missing condition in the following code segment? The code is intended to calculate the cost of shipping a package, as follows: \$5 for packages weighing less than ten pounds, \$10 for packages weighing ten pounds or more but less than twenty pounds, and \$15 for packages weighing twenty pounds or more.

```
if weight < 10 :
    shippingCost = 5
elif weight >= 10 :
    shippingCost = 10
else :
    shippingCost = 15
```

weight > 10  
 weight >= 10  
 weight < 20  
 weight <= 20

**One correct, 0 errors, 100%**

• 2. In this activity, observe the inputs. They denote hours in "military time" between 0 and 23. For each input, click on the conditions of the if statement that are tested and, when a test is successful, on the statement that is executed.

**GOOD JOB! ✓**

```
hour = int(input("Hour: "))
if hour < 6 :
    time = "night"
elif hour < 12 :
    time = "morning"
elif hour < 18 :
    time = "afternoon"
else :
    time = "evening"
```

hour	time
4	morning
15	afternoon
18	evening

**8 correct, 0 errors, 100%, 57 seconds**

**Start over**

• 3. In this activity, observe what happens when the alternatives are tested in the wrong order. For each input, click on the conditions of the if statement that are tested and, when a test is successful, on the statement that is executed. Follow the actual execution flow, even though it will produce the wrong results.

**GOOD JOB! ✓**

```
hour = int(input("Enter hour: "))
if hour < 18 :
    time = "afternoon"
elif hour < 12 :
    time = "morning"
else :
    time = "evening"
```

hour	time
11	afternoon
15	afternoon
5	afternoon
22	morning

Back to Page

91 / 554

3.4 (p.1)

3.4 (p.1) - Multiple Alternatives (1, 2)

Python For Everyone, Enhanced eText

Cay S. Horstmann, Rance D. Necaise

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

2.4 Strings 41

2.5 Input and Output 48

2.6 Graphics: Simple Drawings 58

Chapter Summary 71

Interactive Review and Practice 72

End-of-Chapter Exercises EX2-1

3. Decisions 73

3.1 The if Statement 74

3.2 Relational Operators 79

3.3 Nested Branches 87

**3.4 Multiple Alternatives** 91

3.5 Problem Solving: Flowcharts 96

3.6 Problem Solving: Test Cases 99

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

5. Functions 183

6. Lists 245

7. Files and Exceptions 299

8. Sets and Dictionaries 357

9. Objects and Classes 393

0 correct, 0 errors, 100%, 37 seconds

[Start over](#)

• 3. In this activity, observe what happens when the alternatives are tested in the wrong order. For each input, click on the conditions of the if statement that are tested and, when a test is successful, on the statement that is executed. Follow the actual execution flow, even though it will produce the wrong results.

**GOOD JOB!** ✓

hour	time
11	afternoon
15	afternoon
5	afternoon
23	evening
12	time = "evening"

6 correct, 0 errors, 100%, 30 seconds

[Start over](#)

• 4. Which of the following fixes the problem that was displayed in the preceding problem?

- Set time to "morning" before the if statement.
- Swap the first two tests.
- Test the values in increasing order.
- Use  $\geq$  instead of  $<$  in the tests.

One correct, 0 errors, 100%

• 5. In this activity, observe what happens when a sequence of separate if statements is used instead of an if statement with multiple alternatives. For each input, click on the conditions of the if statement that are tested and, when a test is successful, on the statement that is executed. Follow the actual execution flow, even though it will produce the wrong results.

**GOOD JOB!** ✓

hour	Output
4	morning
10	afternoon
11	evening
15	afternoon
23	evening
12	print("evening")

11 correct, 0 errors, 100%, 48 seconds

[Start over](#)

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

Expand | Collapse

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1

3. Decisions 73

- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96
- 3.6 Problem Solving: Test Cases 99
- 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

5. Functions 183

6. Lists 245

7. Files and Exceptions 299

8. Sets and Dictionaries 357

9. Objects and Classes 393

**SELF CHECK**

• 1. In a flowchart, a decision is represented by

- a rectangle.
- a square.
- a diamond with two outcomes.
- a diamond with three outcomes.

One correct, 0 errors, 100%

• 2. Consider this flow chart for computing travel time between two towns.

```
graph TD; A{Via Bay Bridge?} -- N --> B[travelTime = 45]; A -- Y --> C{Rush hour?}; C -- N --> D[travelTime = 30]; C -- Y --> E[travelTime = 60];
```

Which code segment represents this flow chart?

- if rushHour == "Y" :  
if route == "bay bridge" :  
travelTime = 60  
else :  
travelTime = 30  
else :  
travelTime = 45
- if route == "bay bridge" :  
if rushHour == "Y" :  
travelTime = 60  
else :  
travelTime = 30  
else :  
travelTime = 45
- if route == "bay bridge" :  
travelTime = 45

The second and fourth choices both compute the same answer, but the fourth choice doesn't follow the flow chart.

96 / 554 < >

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1
- 3. Decisions 73
- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96**
- 3.6 Problem Solving: Test Cases 99
- 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
- 5. Functions 183
- 6. Lists 245
- 7. Files and Exceptions 299
- 8. Sets and Dictionaries 357
- 9. Objects and Classes 393

• 2. Consider this flow chart for computing travel time between two towns.

```

graph TD
    A{Via Bay Bridge?} -- N --> B[travelTime = 45]
    A -- Y --> C{Rush hour?}
    C -- N --> D[travelTime = 30]
    C -- Y --> E[travelTime = 60]
    
```

Which code segment represents this flow chart?

`if rushHour == "Y" :  
 if route == "bay bridge" :  
 travelTime = 60  
 else :  
 travelTime = 30  
else :  
 travelTime = 45`

`if route == "bay bridge" :  
 if rushHour == "Y" :  
 travelTime = 60  
 else :  
 travelTime = 30  
else :  
 travelTime = 45`

The second and fourth choices both compute the same answer, but the fourth choice doesn't follow the flow chart.

`if route == "bay bridge" :  
 travelTime = 45  
elif rushHour == "Y" :  
 travelTime = 60  
else :  
 travelTime = 30`

`if rushHour == "Y" :  
 if route == "bay bridge" :  
 travelTime = 60  
 else :  
 travelTime = 45  
else :  
 if route == "bay bridge" :  
 travelTime = 30  
 else :  
 travelTime = 45`

**One correct, 0 errors, 100%**

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

SELF CHECK
SEARCH
AA
...

Expand | Collapse

2.4 Strings
41

2.5 Input and Output
48

2.6 Graphics: Simple Drawings
58

Chapter Summary
71

Interactive Review and Practice
72

End-of-Chapter Exercises
EX2-1

3. Decisions
73

3.1 The if Statement
74

3.2 Relational Operators
79

3.3 Nested Branches
87

3.4 Multiple Alternatives
91

3.5 Problem Solving: Flowcharts
96

3.6 Problem Solving: Test Cases
99

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

5. Functions
183

6. Lists
245

7. Files and Exceptions
299

8. Sets and Dictionaries
357

9. Objects and Classes
393

**1.** Consider this code segment for computing income tax:

```
income = float(input("Income: "))
maritalStatus = input("Marital status: ")
if maritalStatus == "s": # Condition 1
    if income <= 30000 : # Condition 2
        tax = 0.10 * income # Branch 1
    else:
        tax = 3000 + 0.25 * (income - 30000) # Branch 2
else:
    if income <= 60000 : # Condition 3
        tax = 0.10 * income # Branch 3
    else:
        tax = 6000 + 0.25 * (income - 60000) # Branch 4
```

and answer each of the following questions.

**GOOD JOB!** ✓

Which branch is tested by the input Income: 40000 Marital status: s	Branch 2	Condition 1 is fulfilled, but condition 2 is not because the input is > 30000.
Which branch is tested by the input Income: 70000 Marital status: n	Branch 4	Both conditions 1 and 3 are not fulfilled, leading to branch 4.
Which branch is tested by the input Income: 30000 Marital status: s	Branch 1	The tested value is at the boundary of condition 2.
Which branch is not tested by any of these test cases?	Branch 3	No values were entered where the marital status was n and the income was less than or equal to 60,000.

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

**Start over**

**2.** The program segment below has been tested by giving variable number the values 777, 1000, and 1035. What additional value should be used in order to achieve coverage of all decision points?

```
if number > 1000 :
    if number % 2 == 0 :
        print("A large even number")
    else:
        print("A large odd number")
else:
    if number % 2 == 0 :
        print("A small even number")
    else:
        print("A small odd number")
```

7  
 778  
 1005



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

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

- [2.4 Strings](#) 41
- [2.5 Input and Output](#) 48
- [2.6 Graphics: Simple Drawings](#) 58
- [Chapter Summary](#) 71
- [Interactive Review and Practice](#) 72
- [End-of-Chapter Exercises](#) EX2-1

---

- [3. Decisions](#) 73
- [3.1 The if Statement](#) 74
- [3.2 Relational Operators](#) 79
- [3.3 Nested Branches](#) 87
- [3.4 Multiple Alternatives](#) 91
- [3.5 Problem Solving: Flowcharts](#) 96
- [3.6 Problem Solving: Test Cases](#) 99
- [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
- [5. Functions](#) 183
- [6. Lists](#) 245
- [7. Files and Exceptions](#) 299
- [8. Sets and Dictionaries](#) 357
- [9. Objects and Classes](#) 393

GOOD JOB! ✓

Which branch is tested by the input Income: 40000 Marital status: s	Branch 2	Condition 1 is fulfilled, but condition 2 is not because the input is > 30000.
Which branch is tested by the input Income: 70000 Marital status: m	Branch 4	Both conditions 1 and 3 are not fulfilled, leading to branch 4.
Which branch is tested by the input Income: 30000 Marital status: s	Branch 1	The tested value is at the boundary of condition 2.
Which branch is not tested by any of these test cases?	Branch 3	No values were entered where the marital status was m and the income was less than or equal to 60,000.

4 correct, 0 errors, 100%
[Start over](#)

---

• 2. The program segment below has been tested by giving variable number the values 777, 1000, and 1035. What additional value should be used in order to achieve coverage of all decision points?

```
if number > 1000 :
    if number % 2 == 0 :
        print("A large even number")
    else :
        print("A large odd number")
else :
    if number % 2 == 0 :
        print("A small even number")
    else :
        print("A small odd number")
```

7
  778
  1005
  1006

One correct, 0 errors, 100%
[Start over](#)

---

•• 3. Consider the elevator simulation program from Section 3.1. Which of the following inputs would be an appropriate set of test cases?

11 and 12
  14 and 15
  12 and 14

You want to test floors above and below floor 13. The specification isn't clear what should happen with an input of 13. In Section 3.9, you will see a version of the program that reports an error for that input.

12, 13, 14

One correct, 0 errors, 100%
[Start over](#)

3.6 (p.2)

3.6 (p.2) - Problem Solving: Test Cases (2, 3)

99 / 554



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

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

- [2.4 Strings](#) 41
- [2.5 Input and Output](#) 48
- [2.6 Graphics: Simple Drawings](#) 58
- [Chapter Summary](#) 71
- [Interactive Review and Practice](#) 72
- [End-of-Chapter Exercises](#) EX2-1
- [3. Decisions](#) 73
- [3.1 The If Statement](#) 74
- [3.2 Relational Operators](#) 79
- [3.3 Nested Branches](#) 87
- [3.4 Multiple Alternatives](#) 91
- [3.5 Problem Solving: Flowcharts](#) 96
- [3.6 Problem Solving: Test Cases](#) 99
- [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
- [5. Functions](#) 183
- [6. Lists](#) 245
- [7. Files and Exceptions](#) 299
- [8. Sets and Dictionaries](#) 357
- [9. Objects and Classes](#) 393

**SELF CHECK**

• 1. Assuming that variable  $n$  has a value of 10 and variable  $valid$  has a value of `False`, which of the following expressions is True?

`n > 2 and valid`  
 `n < 2 and valid`  
 `n > 2 and not valid`  
 `n < 2 or valid`

**One correct, 0 errors, 100%**

• 2. Assume that  $n = 5$  and  $k = 2$ . What are the values of the following Boolean expressions?

`True`  `False` `0 <= n or n < k`  
 The first condition is True.

`True`  `False` `n >= 0 and k > 0`  
 Both conditions are True.

`True`  `False` `n and k > 0`  
 Both conditions are True because Python treats a non-zero value as True.

`True`  `False` `5 < n and n < k or k < 10`  
 The `and` operator has a higher precedence than `or`. The expression is evaluated in the order  $((5 < n \text{ and } n < k) \text{ or } (k < 10))$ .

`True`  `False` `not (n <= 5)`  
 $n <= 5$  is True.

`True`  `False` `not (n < 5)`  
 $n < 5$  is False.

`True`  `False` `not (0 <= n and n <= k)`  
 This condition can be simplified to  $0 > n$  or  $n > k$ . Because  $n > k$ , the second condition, and therefore the "or" expression, is True.

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

• 3. Suppose  $x$  and  $y$  contain integer values. Give Boolean expressions, using the `and` and `or` operators, for the following tests. Do not use the Boolean `not` operator. But you can use `!=` to check whether a number is not equal to zero. Only use parentheses if needed to override the order of operator precedence.

**GOOD JOB!** ✓

Test	Expression	Explanation
Both $x$ and $y$ are zero	<code>x == 0 and y == 0</code>	This <code>and</code> operator indicates that both conditions must be true.



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

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

- [2.4 Strings](#) 41
- [2.5 Input and Output](#) 48
- [2.6 Graphics: Simple Drawings](#) 58
- [Chapter Summary](#) 71
- [Interactive Review and Practice](#) 72
- [End-of-Chapter Exercises](#) EX2-1

---

- [3. Decisions](#) 73
- [3.1 The If Statement](#) 74
- [3.2 Relational Operators](#) 79
- [3.3 Nested Branches](#) 87
- [3.4 Multiple Alternatives](#) 91
- [3.5 Problem Solving: Flowcharts](#) 96
- [3.6 Problem Solving: Test Cases](#) 99
- [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
- [5. Functions](#) 183
- [6. Lists](#) 245
- [7. Files and Exceptions](#) 299
- [8. Sets and Dictionaries](#) 357
- [9. Objects and Classes](#) 393

n <= 5 is True.

True  False not (n < 5)

n < 5 is False.

True  False not (0 <= n and n <= k)

This condition can be simplified to  $0 > n$  or  $n > k$ . Because  $n > k$ , the second condition, and therefore the "or" expression, is True.

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

• 3. Suppose x and y contain integer values. Give Boolean expressions, using the and and or operators, for the following tests. Do not use the Boolean not operator. But you can use != to check whether a number is not equal to zero. Only use parentheses if needed to override the order of operator precedence.

**GOOD JOB!**

Test	Expression	Explanation
Both x and y are zero.	$x == 0 \text{ and } y == 0$	The and operator indicates that both conditions must be true.
At least one of them is zero.	$x == 0 \text{ or } y == 0$	The or operator indicates that at least one condition must be true.
At most one of them is zero.	$x != 0 \text{ or } y != 0$	If at most one of them is zero, at least one of them must be nonzero.
Neither of them is zero.	$x != 0 \text{ and } y != 0$	Both of them must be nonzero.
x is zero, but y is not zero.	$x == 0 \text{ and } y != 0$	Both conditions must be true, thus, the and is required.

**5 correct, 0 errors, 100%**

**Start over**

• 4. How do you test whether exactly one of two integers is zero? Complete the program, filling in only the condition.

**problem.py**

```

1 x = int(input("Enter x: "))
2 y = int(input("Enter y: "))
3 if x == 0 and y != 0 or x != 0 and y == 0:
4     print("Exactly one of", x, "and", y, "is zero.")
5 else:
6     print("Neither or both of", x, "and", y, "are zero.")

```

**CodeCheck** **Reset**

Score: 4/4

• 5. How do you test whether: 1) exactly one of two integers is zero, 2) both integers are zero, or 3) neither of two integers is zero? Complete the program, filling in only the conditions.

**problem.py**

```

1 x = int(input("Enter x: "))
2 y = int(input("Enter y: "))
3 if x == 0 and y == 0:
4     print("Both of", x, "and", y, "are zero.")
5 elif x == 0 and y != 0:
6     print("Exactly one of", x, "and", y, "is zero.")
7 elif x != 0 and y == 0:
8     print("Exactly one of", x, "and", y, "is zero.")
9 else:
10    print("Neither of", x, "and", y, "are zero.")

```

**CodeCheck** **Reset**

Score: 4/4

3.7 (p.2)

3.7 (p.2) - Boolean Variables and Operators (3, 4, 5)

101 / 554



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

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

- [2.4 Strings](#) 41
- [2.5 Input and Output](#) 48
- [2.6 Graphics: Simple Drawings](#) 58
- [Chapter Summary](#) 71
- [Interactive Review and Practice](#) 72
- [End-of-Chapter Exercises](#) EX2-1
- [3. Decisions](#) 73
- [3.1 The if Statement](#) 74
- [3.2 Relational Operators](#) 79
- [3.3 Nested Branches](#) 87
- [3.4 Multiple Alternatives](#) 91
- [3.5 Problem Solving: Flowcharts](#) 96
- [3.6 Problem Solving: Test Cases](#) 99
- [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
- [5. Functions](#) 183
- [6. Lists](#) 245
- [7. Files and Exceptions](#) 299
- [8. Sets and Dictionaries](#) 357
- [9. Objects and Classes](#) 393



SELF CHECK

•• 1. What is the result of each of the conditions?

**True**  **False** "Smith" == "smith"  
 For two strings to be equal they must be identical, including the case of the letters.

**True**  **False** "B" < "K"  
 Based on lexicographical ordering, "B" comes before "K".

**True**  **False** "char" in "Characters"  
 The substring must match exactly.

**True**  **False** "Hwy66".isalpha()  
 The character "6" is not a letter of the alphabet.

**True**  **False** "Hwy66".isdigit()  
 The string does not consist of only digits.

**True**  **False** "Hwy66".isalnum()  
 ... but it does consist of both letters and digits.

**6 correct, 0 errors, 100%**

•• 2. Assuming the variable name contains the string "Robert Smith", determine the result of each of the following statements.

**GOOD JOB!** ✓

Statement	Result	Explanation
<code>name.count("t")</code>	2	There are two occurrences of the letter "t".
<code>name.count("R")</code>	1	The count method distinguishes between upper- and lowercase letters.
<code>name.count("smith")</code>	0	The substring must match exactly.
<code>name.find("th")</code>	10	The first character is the eleventh, but that character is at index position 10.
<code>name.find("smith")</code>	-1	The substring must match exactly.

**5 correct, 0 errors, 100%, 13 seconds**

Start over

•• 3. In each of the following, does the given solution solve the indicated problem?

**Yes**  **No**  
 Determine the number of blank spaces contained in a string.

3.8 (p.1)

3.8 (p.1) - Analyzing Strings (1, 2)

106 / 554

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

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

2.4 Strings
41

2.5 Input and Output
48

2.6 Graphics: Simple Drawings
58

Chapter Summary
71

Interactive Review and Practice
72

End-of-Chapter Exercises
EX2-1

3. Decisions
73

3.1 The if Statement
74

3.2 Relational Operators
79

3.3 Nested Branches
87

3.4 Multiple Alternatives
91

3.5 Problem Solving: Flowcharts
96

3.6 Problem Solving: Test Cases
99

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

5. Functions
183

6. Lists
245

7. Files and Exceptions
299

8. Sets and Dictionaries
357

9. Objects and Classes
393

.. 3. In each of the following, does the given solution solve the indicated problem?

✓ Yes No

Determine the number of blank spaces contained in a string.

```
numSpace = myString.count(" ")
```

A blank space is indicated as " ". Here, we have specified the empty string.

✓ Yes No

Test whether the first character of a string is an uppercase letter.

```
firstChar = myString[0]
if firstChar.isupper() :
```

The `isupper` string method checks all letters in the string. The test only needs to check whether the first character is uppercase, not the others in the string.

✓ Yes No

Suppose `userStr` contains a string. Test whether the string contains only lowercase letters.

```
if userStr.isalpha() and userStr.islower() :
```

The `islower` string method only tests whether the string contains at least one letter and that all letters in the string are lowercase. To ensure that the string contains only letters, you must also use the `isalpha` string method.

✓ Yes No

Test whether a file contains a JPEG image, which is identified by a filename (given as a string) with an extension of either `.jpg` or `.jpeg`.

```
if filename.endswith(".jpg") and filename.endswith(".jpeg") :
```

A filename can only have one extension; it can not be `.jpg` and `.jpeg` at the same time. To test for one or the other, you must use the `or` Boolean operator.

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

.. 4. In this program, you will check whether two playing cards match. Complete the `if` statement in the following code with the required condition.

```
problem.py
1 print("Enter first card: ")
2 value1 = input("value (#, Ace, Jack, Queen, King): ")
3 suit1 = input("suit (hearts, spades, clubs, diamonds): ")
4
5 print("Enter second card: ")
6 value2 = input("value (#, Ace, Jack, Queen, King): ")
7 suit2 = input("suit (hearts, spades, clubs, diamonds): ")
8
9 if value1 != value2 or suit1 != suit2:
10     print("Different cards")
11 else:
12     print("Same card")
```

CodeCheck Reset

Running `problem.py`

**Test 1**

```
Enter first card:
value (#, Ace, Jack, Queen, King): Ace
suit (hearts, spades, clubs, diamonds): hearts
Enter second card:
value (#, Ace, Jack, Queen, King): Ace
suit (hearts, spades, clubs, diamonds): hearts
Same card
```

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1
- 3. Decisions 73
  - 3.1 The if Statement 74
  - 3.2 Relational Operators 79
  - 3.3 Nested Branches 87
  - 3.4 Multiple Alternatives 91
  - 3.5 Problem Solving: Flowcharts 96
  - 3.6 Problem Solving: Test Cases 99
  - 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
- 5. Functions 183
- 6. Lists 245
- 7. Files and Exceptions 299
- 8. Sets and Dictionaries 357
- 9. Objects and Classes 393

• 4. In this program, you will check whether two playing cards match. Complete the if statement in the following code with the required condition.

```
problem.py
1 print("Enter first card:")
2 value1 = input("#, Ace, Jack, Queen, King: ")
3 suit1 = input("suit (hearts, spaces, clubs, diamonds): ")
4
5 print("Enter second card:")
6 value2 = input("#, Ace, Jack, Queen, King: ")
7 suit2 = input("suit (hearts, spaces, clubs, diamonds): ")
8
9 if value1 != value2 or suit1 != suit2:
10     print("Different cards")
11 else:
12     print("Same card")
```

**CodeCheck** **Reset**

**Running problem.py**

**Test 1**

Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Same card

**pass**

**Test 2**

Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): 9  
suit (hearts, spaces, clubs, diamonds): hearts  
Different cards

**pass**

**Test 3**

Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): spades  
Different cards

**pass**

**Test 4**

Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Queen  
suit (hearts, spaces, clubs, diamonds): clubs  
Different cards

**pass**

**Score**

4/4

• 5. What value is displayed by the following code segment?

```
str = "Computer Science"
where = str.find("TFR")
```

Back to Page 106 / 554

Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 2.4 Strings 41
- 2.5 Input and Output 48
- 2.6 Graphics: Simple Drawings 58
- Chapter Summary 71
- Interactive Review and Practice 72
- End-of-Chapter Exercises EX2-1
- 3. Decisions 73
- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96
- 3.6 Problem Solving: Test Cases 99
- 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
- 5. Functions 183
- 6. Lists 245
- 7. Files and Exceptions 299
- 8. Sets and Dictionaries 357
- 9. Objects and Classes 393

1 **else:**  
1 print("Same card")

**CodeCheck** **Reset**

**Running problem.py**

**Test 1**

```
Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Same card
```

**pass**

**Test 2**

```
Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): 9  
suit (hearts, spaces, clubs, diamonds): hearts  
Different cards
```

**pass**

**Test 3**

```
Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): spades  
Different cards
```

**pass**

**Test 4**

```
Enter first card:  
value (#, Ace, Jack, Queen, King): Ace  
suit (hearts, spaces, clubs, diamonds): hearts  
Enter second card:  
value (#, Ace, Jack, Queen, King): Queen  
suit (hearts, spaces, clubs, diamonds): clubs  
Different cards
```

**pass**

**Score**

4/4

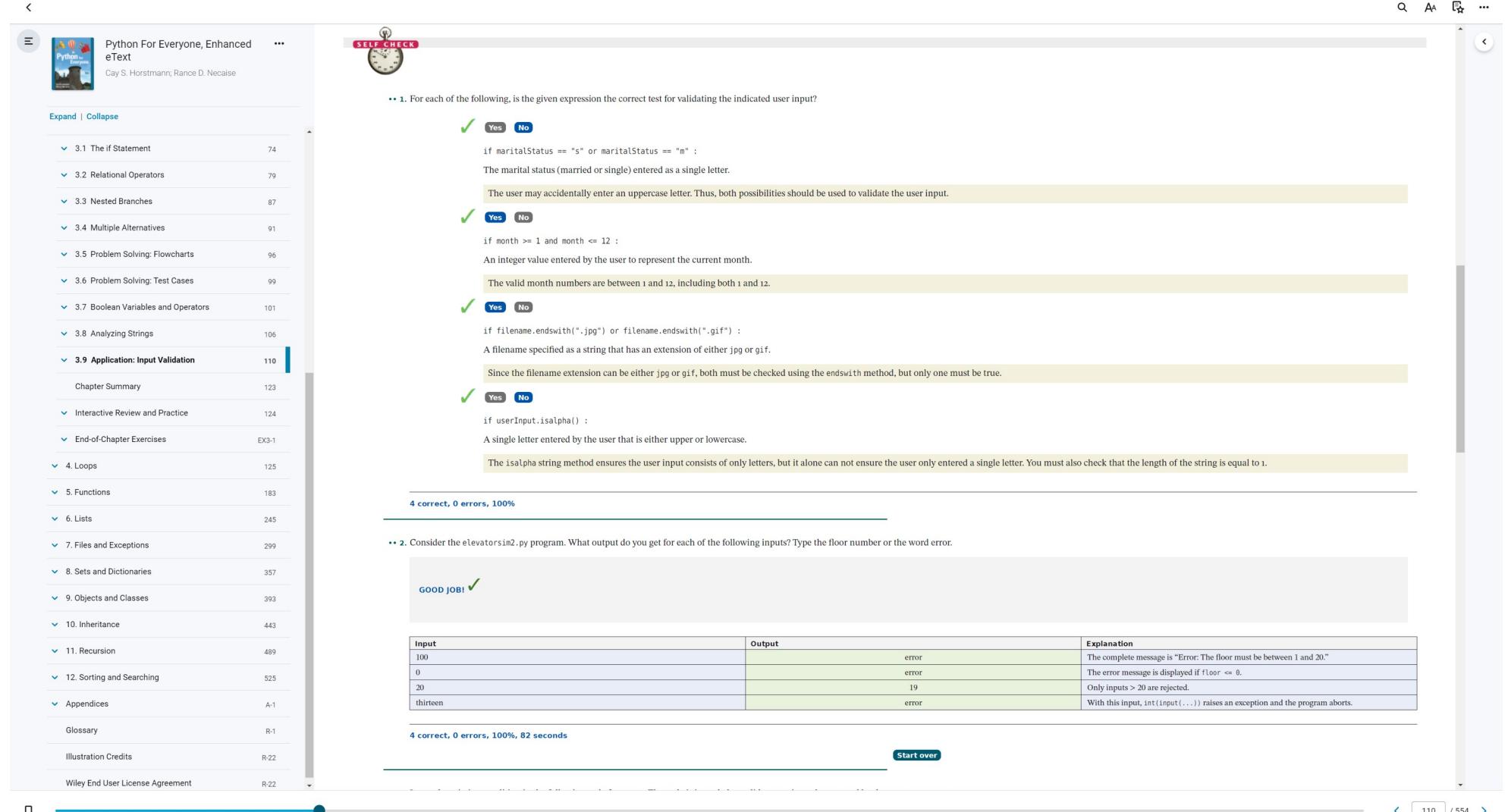
• 5. What value is displayed by the following code segment?

```
str = "Computer Science"  
where = str.find("TER")  
print(where)
```

-1  
 0  
 5  
 6

**One correct, 0 errors, 100%**

Back to Page 106 / 554



Python For Everyone, Enhanced eText

Cay S. Horstmann; Rance D. Necaise

Expand | Collapse

- 3.1 The if Statement 74
- 3.2 Relational Operators 79
- 3.3 Nested Branches 87
- 3.4 Multiple Alternatives 91
- 3.5 Problem Solving: Flowcharts 96
- 3.6 Problem Solving: Test Cases 99
- 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

5. Functions 183

6. Lists 245

7. Files and Exceptions 299

8. Sets and Dictionaries 357

9. Objects and Classes 393

10. Inheritance 443

11. Recursion 489

12. Sorting and Searching 525

Appendices A-1

Glossary R-1

Illustration Credits R-22

Wiley End User License Agreement R-22

.. 3. Insert the missing condition in the following code fragment. The code is intended to validate a string value entered by the user to represent a yes or no response.

```
if _____:  
    valid = True  
else:  
    valid = False
```

response == "y" or response == "n"  
 response == "y" and response == "n"  
 response == "y" or response == "Y" or response == "n" or response == "N"

If the user is not instructed to specifically enter a lowercase or uppercase letter for the response, so you must check for both cases.  
 response == "y" or response == "Y" and response == "n" or response == "N"

**One correct, 0 errors, 100%**

.. 4. In the Sherlock Holmes story "The Adventure of the Sussex Vampire", the inimitable detective uttered these words: "Matilda Briggs was not the name of a young woman, Watson, ... It was a ship which is associated with the giant rat of Sumatra, a story for which the world is not yet prepared." Over a hundred years later, researchers found giant rats in Western New Guinea, another part of Indonesia.



© jeannas85/Stockphoto.

Suppose you are charged with writing a program that processes rat weights. We don't know for sure how giant a rat could be, but the New Guinea rats weighed no more than 2 kg. A regular house rat (*rattus rattus*) weighs up to 0.2 kg, so we'll say that any weight > 10 kg was surely an input error, perhaps confusing grams and kilograms. Of course, inputs should be positive numbers. Rearrange the following lines of code to supply error handling. Not all lines are useful.

**GOOD JOB! ✓**

```
weight = float(input("Enter weight in kg: "))  
if weight <= 0 :  
    print("Error: Weight must be positive.")  
elif weight > 10 :  
    print("Error: Weight > 10 kg.")  
else :  
    ## Process weight . . .
```

**6 correct, 0 errors, 100%, 65 seconds**

**Start over**

.. 5. Modify lines 10 – 13 of the elevatorsim2.py program such that there is a single if with a complex condition that prints Error: Invalid floor number when the condition is true.

The screenshot shows a PythonAnywhere IDE interface. On the left is a sidebar with a book icon and the title "Python For Everyone, Enhanced eText" by Cay S. Horstmann and Rance D. Necaise. Below the title is a table of contents for Chapter 3, which includes sections like "3.1 The If Statement", "3.2 Relational Operators", and "3.9 Application: Input Validation". The "3.9 Application: Input Validation" section is currently selected. To the right of the sidebar is a code editor window containing Python code for "elevatorsim2.py". The code checks for an input floor number and prints an error if it's invalid. A status bar at the bottom of the code editor indicates "6 correct, 0 errors, 100%, 65 seconds". Below the code editor is a terminal window showing the output of running the program with various floor inputs. The terminal output includes "Floor: 13", "Error: Invalid floor number", "Floor: 8", "The elevator will travel to the actual floor 8", "Floor: 21", "Error: Invalid floor number", "Floor: -5", "Error: Invalid floor number", and "Floor: 16", "The elevator will travel to the actual floor 15". At the bottom of the terminal window, the score is listed as "Score 5/5". The bottom right corner of the interface shows navigation controls and a page number "110 / 554".