If Statement Solutions

10

Solution to Exercise 34: Even or Odd?

```
##
# Determine and display whether an integer entered by the user is even or odd.
#

# Read the integer from the user
num = int(input("Enter an integer: "))

# Determine whether it is even or odd by using the
# modulus (remainder) operator
if num % 2 == 1:
  print(num, "is odd.")
else:
  print(num, "is even.")
```

Dividing an even number by 2 always results in a remainder of 0. Dividing an odd number by 2 always results in a remainder of 1.

Solution to Exercise 36: Vowel or Consonant

```
##
# Determine if a letter is a vowel or a consonant.
#

# Read a letter from the user
letter = input("Enter a letter of the alphabet: ")

# Classify the letter and report the result
if letter == "a" or letter == "e" or \
   letter == "i" or letter == "o" or \
   letter == "u":
  print("It's a vowel.")
elif letter == "y":
  print("Sometimes it's a vowel... Sometimes it's a consonant.")
else:
  print("It's a consonant.")
```

This version of the program only works for lowercase letters. You can add support for uppercase letters by including additional comparisons that follow the same pattern.

B. Stephenson, *The Python Workbook*, DOI 10.1007/978-3-319-14240-1_10

Solution to Exercise 37: Name that Shape

```
##
#  Report the name of a shape from its number of sides.
#

# Read the number of sides from the user
nsides = int(input("Enter the number of sides: "))

# Determine the name, leaving it empty if an unsupported number of sides was entered
name = ""
if nsides == 3:
  name = "triangle"
elif nsides == 4:
  name = "quadrilateral"
elif nsides == 5:
  name = "pentagon"
elif nsides == 6:
  name = "hexagon"
elif nsides == 7:
  name = "heptagon"
elif nsides == 8:
  name = "octagon"
elif nsides == 9:
  name = "nonagon"
elif nsides == 10:
  name = "decagon"

# Display an error message or the name of the polygon
if name == "":
  print("That number of sides is not supported by this program.")
else:
  print("That's a", name)
```

The empty string is being used as a sentinel value. If the number of sides entered by the user is outside of the supported range then `name` will remain empty, causing an error message to be displayed later in the program.

Solution to Exercise 38: Month Name to Number of Days

```
##
#  Display the number of days in a month.
#

# Read input from the user
month = input("Enter the name of a month: ")

# Compute the number of days in the month
days = 31
```

Start by assuming that the number of days is 31. Then update the number of days if necessary.

```
if month == "April" or month == "June" or \
   month == "September" or month == "November":
  days = 30
elif month == "February":
  days = "28 or 29"

# Display the result
print(month, "has", days, "days in it.")
```

When `month` is February, the value assigned to `days` is a string so that we can represent 28 or 29 days.

Solution to Exercise 40: Name that Triangle

```
##
# Determine the name of a triangle from the lengths of its sides.
#

# Read the side lengths from the user
side1 = float(input("Enter the length of side 1: "))
side2 = float(input("Enter the length of side 2: "))
side3 = float(input("Enter the length of side 3: "))

# Determine the triangle's name
if side1 == side2 and side2 == side3:
  name = "equilateral"
elif side1 == side2 or side2 == side3 or \
     side3 == side1:
  name = "isoceles"
else:
  name = "scalene"

# Display the triangle's name
print("That's a", name, "triangle")
```

We could also check that `side1` is equal to `side3` as part of the condition for an equilateral triangle. However, that comparison isn't necessary because the == operator is transitive.

Solution to Exercise 41: Note to Frequency

```
##
# Convert the name of a note to its frequency.
#
C4_FREQ = 261.63
D4_FREQ = 293.66
E4_FREQ = 329.63
F4_FREQ = 349.23
G4_FREQ = 392.00
A4_FREQ = 440.00
B4_FREQ = 493.88

# Read the note name from the user
name = input("Enter the two character note name, such as C4: ")

# Store the note and its octave in separate variables
note = name[0]
octave = int(name[1])
```

```
# Get the frequency of the note, assuming it is in the fourth octave
if note == "C":
  freq = C4_FREQ
elif note == "D":
  freq = D4_FREQ
elif note == "E":
  freq = E4_FREQ
elif note == "F":
  freq = F4_FREQ
elif note == "G":
  freq = G4_FREQ
elif note == "A":
  freq = A4_FREQ
elif note == "B":
  freq = B4_FREQ

# Now adjust the frequency to bring it into the correct octave
freq = freq / 2 ** (4 - octave)

# Display the result
print("The frequncy of", name, "is", freq)
```

Solution to Exercise 42: Frequency to Note

```
##
#  Read a frequency from the user and display the note (if any) that it corresponds to.
#
C4_FREQ = 261.63
D4_FREQ = 293.66
E4_FREQ = 329.63
F4_FREQ = 349.23
G4_FREQ = 392.00
A4_FREQ = 440.00
B4_FREQ = 493.88
LIMIT = 1

# Read the frequency from the user
freq = float(input("Enter a frequency: "))

# Determine the note that corresponds to the entered frequency.  Set
# note equal to the empty string if there isn't a match.
if freq >= C4_FREQ - LIMIT and freq <= C4_FREQ + LIMIT:
  note = "C4"
elif freq >= D4_FREQ - LIMIT and freq <= D4_FREQ + LIMIT:
  note = "D4"
elif freq >= E4_FREQ - LIMIT and freq <= E4_FREQ + LIMIT:
  note = "E4"
elif freq >= F4_FREQ - LIMIT and freq <= F4_FREQ + LIMIT:
  note = "F4"
elif freq >= G4_FREQ - LIMIT and freq <= G4_FREQ + LIMIT:
  note = "G4"
```

```
elif freq >= A4_FREQ - LIMIT and freq <= A4_FREQ + LIMIT:
  note = "A4"
elif freq >= B4_FREQ - LIMIT and freq <= B4_FREQ + LIMIT:
  note = "B4"
else:
  note = ""

# Display the result, or an appropriate error message
if note == "":
  print("There is no note that corresponds to that frequency.")
else:
  print("That frequency is", note)
```

Solution to Exercise 46: Season from Month and Day

```
##
# Determine and display the season associated with a date.
#

# Read the date from the user
month = input("Enter the name of the month: ")
day = int(input("Enter the day number: "))

# Determine the season
if month == "January" or month == "February":
  season = "Winter"
elif month == "March":
  if day < 20:
    season = "Winter"
  else:
    season = "Spring"
elif month == "April" or month == "May":
  season = "Spring"
elif month == "June":
  if day < 21:
    season = "Spring"
  else:
    season = "Summer"
elif month == "July" or month == "August":
  season = "Summer"
elif month == "September":
  if day < 22:
    season = "Summer"
  else:
    season = "Fall"
elif month == "October" or month == "November":
  season = "Fall"
elif month == "December":
  if day < 21:
    season = "Fall"
  else:
    season = "Winter"
```

This solution to the season problem uses several `elif` statements so that the conditions remain as simple as possible. Another way of approaching this problem is to minimize the number of `elif` statements by making the conditions more complex.

```
# Display the result
print(month, day, "is in", season)
```

Solution to Exercise 48: Chinese Zodiac

```
##
#  Determine the animal associated with a year according to the Chinese zodiac.
#

# Read a year from the user
year = int(input("Enter a year: "))

# Determine the animal associated with that year
if year % 12 == 8:
  animal = "Dragon"
elif year % 12 == 9:
  animal = "Snake"
elif year % 12 == 10:
  animal = "Horse"
elif year % 12 == 11:
  animal = "Sheep"
elif year % 12 == 0:
  animal = "Monkey"
elif year % 12 == 1:
  animal = "Rooster"
elif year % 12 == 2:
  animal = "Dog"
elif year % 12 == 3:
  animal = "Pig"
elif year % 12 == 4:
  animal = "Rat"
elif year % 12 == 5:
  animal = "Ox"
elif year % 12 == 6:
  animal = "Tiger"
elif year % 12 == 7:
  animal = "Hare"

# Report the result
print("%d is the year of the %s." % (year, animal))
```

When multiple items are formatted all of the values are placed inside parentheses on the right side of the `%` operator.

Solution to Exercise 51: Letter Grade to Grade Points

```
##
# Convert from a letter grade to a number of grade points.
#
A       = 4.0
A_MINUS = 3.7
B_PLUS  = 3.3
B       = 3.0
B_MINUS = 2.7
C_PLUS  = 2.3
C       = 2.0
C_MINUS = 1.7
D_PLUS  = 1.3
D       = 1.0
F       = 0
INVALID = -1

# Read the letter grade from the user
letter = input("Enter a letter grade: ")
letter = letter.upper()
```

The statement `letter = letter.upper()` converts any lowercase letters entered by the user into uppercase letters, storing the result back into the same variable. Including this statement allows the program to work with lowercase letters without including them in the conditions of the `if` and `elif` statements.

```
# Convert from a letter grade to a number of grade points using -1 grade points as a sentinel
# value indicating invalid input
if letter == "A+" or letter == "A":
  gp = A
elif letter == "A-":
  gp = A_MINUS
elif letter == "B+":
  gp = B_PLUS
elif letter == "B":
  gp = B
elif letter == "B-":
  gp = B_MINUS
elif letter == "C+":
  gp = C_PLUS
elif letter == "C":
  gp = C
elif letter == "C-":
  gp = C_MINUS
elif letter == "D+":
  gp = D_PLUS
```

```
elif letter == "D":
  gp = D
elif letter == "F":
  gp = F
else:
  gp = INVALID

# Display the output
if gp == INVALID:
  print("That wasn't a valid number of grade points.")
else:
  print("That's", gp, "grade points.")
```

Solution to Exercise 53: Assessing Employees

```
##
# Report whether an employee's performance is unacceptable, acceptable
# or meritorious based on the rating entered by the user.
#
RAISE_FACTOR = 2400.00
UNACCEPTABLE = 0
ACCEPTABLE = 0.4
MERITORIOUS = 0.6

# Read the rating from the user
rating = float(input("Enter the rating: "))

# Classify the performance
if rating == UNACCEPTABLE:
  performance = "Unacceptable"
elif rating == ACCEPTABLE:
  performance = "Acceptable"
elif rating >= MERITORIOUS:
  performance = "Meritorious"
else:
  performance = ""

# Report the result
if performance == "":
  print("That wasn't a valid rating.")
else:
  print("Based on that rating, the performance is %s." % performance)
  print("You will receive a raise of $%.2f." % (rating * RAISE_FACTOR))
```

The parentheses around `rating * RAISE_FACTOR` on the final line are necessary because the `%` and `*` operators have the same precedence. Including the parentheses forces Python to perform the mathematical calculation before formatting the result.

Solution to Exercise 57: Is it a Leap Year?

```
##
#  Determine whether or not a year is a leap year.
#

# Read the year from the user
year = int(input("Enter a year: "))

# Determine if it is a leap year
if year % 400 == 0:
  isLeapYear = True
elif year % 100 == 0:
  isLeapYear = False
elif year % 4 == 0:
  isLeapYear = True
else:
  isLeapYear = False

# Display the result
if isLeapYear:
  print(year, "is a leap year.")
else:
  print(year, "is not a leap year.")
```

Solution to Exercise 59: Is a License Plate Valid?

```
## Determine whether or not a license plate is valid.  A valid license plate either consists of:
#    1)  3 letters followed by 3 numbers, or
#    2)  4 numbers followed by 3 numbers

# Read the plate from the user
plate = input("Enter the license plate: ")

# Check the status of the plate and display it.  It is necessary to check each of the 6 characters
# for an older style plate, or each of the 7 characters for a newer style plate.
if len(plate) == 6 and plate[0] >= "A" and plate[0] <= "Z" and \
  plate[1] >= "A" and plate[1] <= "Z" and \
  plate[2] >= "A" and plate[2] <= "Z" and \
  plate[3] >= "0" and plate[3] <= "9" and \
  plate[4] >= "0" and plate[4] <= "9" and \
  plate[5] >= "0" and plate[5] <= "9":
    print("The plate is a valid older style plate.")
elif len(plate) == 7 and plate[0] >= "0" and plate[0] <= "9" and \
  plate[1] >= "0" and plate[1] <= "9" and \
  plate[2] >= "0" and plate[2] <= "9" and \
  plate[3] >= "0" and plate[3] <= "9" and \
  plate[4] >= "A" and plate[4] <= "Z" and \
  plate[5] >= "A" and plate[5] <= "Z" and \
  plate[6] >= "A" and plate[6] <= "Z":
    print("The plate is a valid newer style plate.")
```

```
else:
  print("The plate is not valid.")
```

Solution to Exercise 60: Roulette Payouts

```
##
# Display the bets that pay out in a roulette simulation.
#
from random import randrange

# Simulate spinning the wheel, using 37 to represent 00
value = randrange(0, 38)
if value == 37:
  print("The spin resulted in 00...")
else:
  print("The spin resulted in %d..." % value)

# Display the payout for a single number
if value == 37:
  print("Pay 00")
else:
  print("Pay", value)

# Display the color payout
# The first line in the condition checks for 1, 3, 5, 7, 9
# The second line in the condition checks for 12, 14, 16, 18
# The third line in the condition checks for 19, 21, 23, 25, 27
# The fourth line in the condition checks for 30, 32, 34, 36
if value % 2 == 1 and value >= 1 and value <= 9 or \
   value % 2 == 0 and value >= 12 and value <= 18 or \
   value % 2 == 1 and value >= 19 and value <= 27 or \
   value % 2 == 0 and value >= 30 and value <= 36:
  print("Pay Red")
elif value == 0 or value == 37:
  pass
else:
  print("Pay Black")

# Display the odd vs. even payout
if value >= 1 and value <= 36:
  if value % 2 == 1:
    print("Pay Odd")
  else:
    print("Pay Even")

# Display the lower number vs. upper number payout
if value >= 1 and value <= 18:
  print("Pay 1 to 18")
elif value >= 19 and value <= 36:
  print("Pay 19 to 36")
```

The body of an `if`, `elif` or `else` must contain at least one statement. The `pass` statement can be used in situations where a statement is required but there is no work to be performed.