STUDENT SOLUTIONS MANUAL

to accompany

An Introduction to Programming Using Python

by David I. Schneider

```
69. tonsPerAcre = 18
    acres = 30
    totalTonsProduced = tonsPerAcre * acres
    print(totalTonsProduced)

71. distance = 233
    elapsedTime = 7 - 2
    averageSpeed = distance / elapsedTime
    print(averageSpeed)

73. gallonsPerPersonDaily = 1600
    numberOfPeople = 315000000
    numberOfDays = 365
    gallonsPerYear = gallonsPerPersonDaily * numberOfPeople * numberOfDays
    print(gallonsPerYear)

75. numberOfPizzarias = 70000
    percentage = .12
    numberOfRestaurants = numberOfPizzarias / percentage
    print(round(numberOfRestaurants))

77. nationalDebt = 1.68e+13
    population = 3.1588e+8
    perCapitaDebt = nationalDebt / population
    print(round(perCapitaDebt))
```

EXERCISES 2.2

```
1. Python    3. Ernie    5. "o"       7. "h"      9. "Pyt"    11. "Py"    13. "h"

15. "th"     17. "Python"     19. 2       21. -1      23. 10      25. 2       27. -1

29. 3        31. "8 ball"     33. "8 BALL"     35. "The Artist"         39. 5

41. 7        43. 2        45. "King Kong"      47. 12
                                                   MUNICIPALITY
                                                   city
                                                   6

49. flute    51. Your age is 21.     53. A ROSE IS A ROSE IS A ROSE

55. WALLAWALLA           57. goodbye 59. Mmmmmmm.      61. a     b

63. 76 trombones         65. 17      67. 8             69. The Great 9

71. s[:-1]               73. -8      75. True          77. True
```

79. `234-5678` should be surrounded with quotation marks.

81. *for* is a reserved word and cannot be used as a variable name.

83. The string should be replaced with `"Say it ain't so."`

85. `Upper` should be changed to `upper`.

57.
```
import pickle

def main():
    ## Determine states that were home to three or more presidents.
    presidents = getDictionary("USpresStatesDict.dat")
    states = createStatesDict(presidents)
    sortedStates = [state for state in states if states[state] > 2]
    sortedStates.sort(key=lambda state: states[state], reverse=True)
    print("States that produced three or")
    print("more presidents as of 2016:")
    for state in sortedStates:
        print(" ", state + ":", states[state])

def getDictionary(fileName):
    infile = open(fileName, 'rb')
    dictName = pickle.load(infile)
    infile.close()
    return dictName

def createStatesDict(presidents):
    states = {}
    for state in presidents.values():
        if not states.get(state, False):
            states[state] = 1
        else:
            states[state] += 1
    return states

main()
```

```
States that produced three or
more presidents as of 2016:
  Ohio: 6
  New York: 6
  Virginia: 5
  Massachusetts: 4
  Tennessee: 3
  California: 3
  Texas: 3
  Illinois: 3
```

59.
```
def main():
    ## Determine the day of the week for a date.
    calendar2015Dict = createDictionary("Calendar2015.txt")
    date = input("Enter a date in 2015: ")
    print(date, "falls on a", calendar2015Dict[date])

def createDictionary(fileName):
    infile = open(fileName, 'r')
    textList = [line.rstrip() for line in infile]
    infile.close()
    return dict([x.split(',') for x in textList])

main()
```

```
Enter a date in 2015: 2/14/2015
11/3/2015 falls on a Saturday
```

```
class Purchase:
    def __init__(self, description="", price=0, quantity=0):
        self._description = description
        self._price = price
        self._quantity = quantity

    def setDescription(self, description):
        self._description = description

    def getDescription(self):
        return self._description

    def setPrice(self, price):
        self._price = price

    def getPrice(self):
        return self._price

    def setQuantity(self, quantity):
        self._quantity = quantity

    def getQuantity(self):
        return self._quantity

class Cart:
    def __init__(self, items=[]):
        self._items = items

    def addItemToCart(self, item):
        self._items.append(item)

    def getItems(self):
       return self._items

    def calculateTotal(self):
        amount = 0
        for item in self._items:
            amount += item.getPrice() * item.getQuantity()
        return amount

main()
```

```
Enter description of article: shirt
Enter price of article: 35
Enter quantity of article: 3
Do you want to enter more articles (Y/N)? Y
Enter description of article: tie
Enter price of article: 15
Enter quantity of article: 2
Do you want to enter more articles (Y/N)? N

ARTICLE       PRICE   QUANTITY
shirt        $35.00       3
tie          $15.00       2

TOTAL COST: $135.00
```

```
    def checkAnswer(self):
        m = self._people.index(
                    self._lstPeople.get(self._lstPeople.curselection()))
        n = self._places.index(
                    self._lstPlaces.get(self._lstPlaces.curselection()))
        if m == n:
            self._conOFentAnswer.set("CORRECT")
        else:
            self._conOFentAnswer.set("INCORRECT")

Workplaces()
```