**HAUB SCHOOL OF BUSINESS**

**SAINT JOSEPH’S UNIVERSITY**

**DSS 615: Python Programming**

**Instructor: Michael Ghen**

**Assignment 4**

By:

Vinayak Suresh Tayshetye (10673718)

Section 3.1 pp 86-89 Exercises 2-86 even

**In Exercises 1 through 8, determine the output displayed.**

2. **print('C' + chr(35))**

[output]

C#

4. **print(chr(ord('B'))) # The ASCII value of B is 66**

[output]

B

6.

list1 = [17, 3, 12, 9, 10]

list1.sort()

print("Spread:", list1[-1] - list1[0])

[output]

Spread: 14

8.

letter = 'D'

spread = ord('a') - ord('A')

print(chr(ord(letter) + spread))

[output]

D

**In Exercises 9 through 20, assume the value of *a* is 2 and the value of *b* is 3, and determine whether the condition evaluates to True or False. Then, use a print function to confirm your answer.**

10. ((5 - a) \* b) < 7

[output]

False

12. a \*\* b == b \*\* a

[output]

False

14. 3e-2 < .01 \* a

[output]

False

16. (a \* a < b) or not(a \* a < a)

[output]

True

18. not(a < b) or not (a < (b + a))

[output]

False

20. ((a == b) or not (b < a)) and ((a < b) or (b == a + 1))

[output]

True

22. "Inspector" < "gadget"

[output]

True

24. 'J' >= 'J'

[output]

True

26. 'B' > '?'

[output]

True

28. "Duck" < "Duck" + "Duck"

[output]

True

30. "th" in "Python"

[output]

True

32. (7 < 34) and ("7" < "34")

[output]

False

34. isinstance(32., int)

[output]

False

36. isinstance(32, int)

[output]

True

38. "knight".startswith('n')

[output]

False

40. "flute".endswith('t')

[output]

False

42. True and False

[output]

False

44. not False

[output]

True

**In Exercises 45 through 54, determine whether or not the two conditions are equivalent - that is, whether they will both evaluate to True or both evaluate to False for any values of the variables appearing in them.**

46. not(a < b); a > b

[output]

False; False

48. not((a == b) or (a == c)); (a != b) and (a != c)

[output]

True; True

50. (a < b) and ((a > d) or (a > e));

((a < b) and (a > d)) or ((a < b) and (a > e))

[output]

False; False

52. not(a >= b); (a <= b) and not(a == b)

[output]

True; True

54. str1.upper() == str1; str1.isupper()

[output]

False; False

**In Exercises 55 through 60, write a condition equivalent to the negation of the given condition. (For example, a !**= **b is equivalent to the negation of a** = = **b.)**

56. (a == b) or (a == d)

[output]

(a != b) and (a != d)

58. not((a == b) or (a > b))

[output]

not((a != b) or (a <= b))

60. (a != "") and (a < b) and (len(a) < 5)

[output]

(a == "") or (a >= b) or (len(a) >= 5)

[Also, len(a) would give error as object of type 'int' has no len()]

**In Exercises 61 through 68, simplify the expression. (In Exercises 63 through 68, assume that the variable has an integer value.)**

62. (name == "Athos") or (name == "Porthos") or (name == "Aramis")

[output]

Name in ["Athos", "Porthos", "Aramis")

64. (n == 1) or (n == 2) or (n == 3) or (n == 4) or (n == 5) or (n == 6)

[output]

1 <= n <= 6

66. (n <= 22) and (n > 1)

[output]

1 < n <= 22

68. (n <= 200) and (n >= 100)

[output]

100 <= n <= 200

**In Exercises 69 through 84, determine whether True or False is displayed.**

70. print("colonel".startswith('k'))

[output]

False

72. str1 = "target"

str2 = "get"

print(str1.endswith(str2))

[output]

True

74. str1 = "Teapot"

print(str1.startswith(str1[0:4]))

[output]

True

76. str1 = "spam and eggs"

print(str1.endswith(str1[10:len(str1)]))

[output]

True

78. num = "1234.56"

print(isinstance(num, float))

[output]

False

80. print(isinstance("25", int))

[output]

False

82. letter = ord('M')

print(isinstaance(letter, str))

[output]

False

84. print("seven".isdigit())

[output]

False

**Section 3.2** pp 98-104 Exercises 2-44 even

**In Exercises 1 through 14, determine the output displayed.**

2.

gpa = 3.49

result = ""

if gpa >= 3.5:

result = "Honors"

print(result + "Student")

[output]

Student

4. print('A' < 'B' < 'c')

[output]

True

6.

change = 356

if change >= 100:

print("Your change contains", change // 100, "dollars.")

else:

print("Your change contains no dollars.")

[output]

Your change contains 3 dollars.

8.

length = eval(input("Enter length of cloth in yards: "))

if length < 1:

cost = 3.00 # cost in dollars

else:

cost = 3.00 + ((length - 1) \* 2.50)

result = "Cost of cloth is ${0:0.2f}.".format(cost)

print(result)

#(Assume the response is 6.)

[output]

Enter length of cloth in yards: 6

Cost of cloth is $15.50.

10.

isvowel = False

letter = input("Enter a letter: ")

letter = letter.upper()

if (letter in "AEIOU"):

isvowel = True

if isvowel:

print(letter, "is a vowel.")

elif (not(65 <= ord(letter) <= 90)):

print("You did not enter a letter.")

else:

print(letter, "is not a vowel.")

#(Assume the response is a.)

[output]

Enter a letter: a

A is a vowel.

12.

number = 5

if number < 0:

print("negative")

else:

if number == 0:

print("zero")

else:

print("positive")

[output]

Positive

14.

if "":

print("An empty string is true.")

else:

print("An empty string is false.")

[output]

An empty string is false.

**In Exercises 15 through 18, identify the errors, state the type of each error (syntax, runtime, or logic), and correct the block of code.**

16.

number = 6

if number > 5 and < 9:

print("Yes")

else:

print("No")

[output]

**SyntaxError:** invalid syntax

number = 6

if 5 < number < 9:

print("Yes")

else:

print("No")

.

18.

if a = b:

print("same")

[output]

**SyntaxError:** invalid syntax

a = 2

b = 2

if a == b:

print("same")

**In Exercises 19 through 24, simplify the code.**

20.

if (a == 7):

print("seven")

elif (a != 7):

print("eleven")

[output]

if (a == 7):

print("seven")

else:

print("eleven")

22.

if state == "CA":

if city == "LA" or city == "SD":

print("Large city!")

[output]

if (state == "CA") and ((city == "LA") or (city == "SD")):

print("Large city!")

24.

feet = eval(input("How tall (in feet) is the Statue of Liberty? "))

if (feet <= 141):

print("Nope")

if (feet > 141):

if (feet < 161):

print("Good")

else:

print("Nope")

print("The statue is 151 feet tall from base to torch.")

[output]

feet = eval(input("How tall (in feet) is the Statue of Liberty? "))

if (141 <= feet < 161):

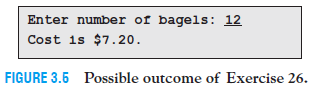
print("Good")

else:

print("Nope")

print("The statue is 151 feet tall from base to torch.")

26. **Cost of Bagels** A bagel shop charges 75 cents per bagel for orders of less than a half-dozen bagels and 60 cents per bagel for orders of a half-dozen or more. Write a program that requests the number of bagels ordered and displays the total cost. See Fig. 3.5.



[code]

bagels = int(input("Enter number of bagels: "))

if (bagels < 6):

cost = bagels \* 0.75

else:

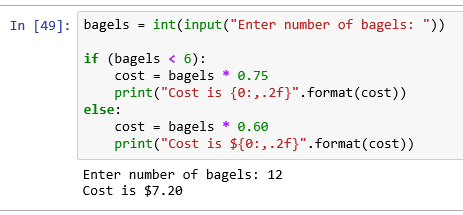
cost = bagels \* 0.60

print("Cost is ${0:,.2f}".format(cost))

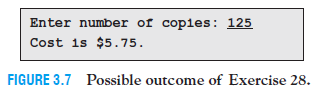
[run]

Enter number of bagels: 12

Cost is $7.20



28. **Cost of Copies** A copy center charges 5 cents per copy for the first 100 copies and 3 cents per copy for each additional copy. Write a program that requests the number of copies as input and displays the total cost. See Fig. 3.7.



[code]

copies = int(input("Enter number of copies: "))

if copies > 100:

cost = (100 \* 0.05) + ((copies - 100) \* 0.03)

else:

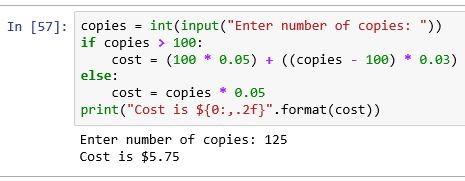
cost = copies \* 0.05

print("Cost is ${0:,.2f}".format(cost))

[run]

Enter number of copies: 125

Cost is $5.75

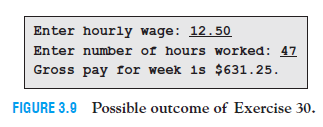


30.**Overtime Pay** Federal law requires that hourly employees be paid “time-and-a-half” for work in excess of 40 hours in a week. For example, if a person’s hourly wage is $12

and he or she works 60 hours in a week, the person’s gross pay should be

(40 \* 12) + (1.5 \* 12 \* (60 - 40)) = $840.

Write a program that requests the number of hours a person works in a given week and the person’s hourly wage as input, and then displays the person’s gross pay. See Fig. 3.9.



[code]

wage = eval(input("Enter hourly wage: "))

hours = eval(input("Enter number of hours worked: "))

if hours > 40:

pay = (40 \* wage) + (1.5 \* wage \* (hours - 40))

else:

pay = hours \* wage

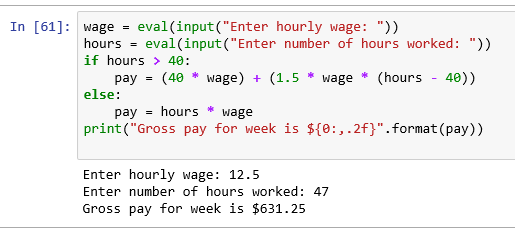
print("Gross pay for week is ${0:,.2f}".format(pay))

[run]

Enter hourly wage: 12.5

Enter number of hours worked: 47

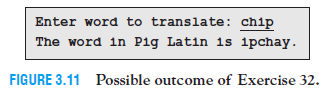
Gross pay for week is $631.25



32. **Pig Latin** Write a program that requests a word (in lowercase letters) as input and translates the word into Pig Latin. See Fig. 3.11. The rules for translating a word into Pig Latin are as follows:

**(a)** If the word begins with a group of consonants, move them to the end of the word and add *ay*. For instance, *chip* becomes *ipchay*.

**(b)** If the word begins with a vowel, add *way* to the end of the word. For instance, *else* becomes *elseway*.



[code]

word = str(input("Enter word to translate: "))

for i in range(0, len(word)-1, 1):

if (word[i] not in "aeiou"):

i= i+1

elif (word[0] in "aeiou"):

print("The word in Pig Latin is " + word + "way")

break

else:

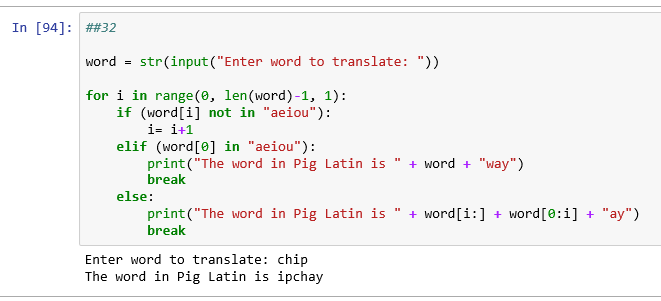
print("The word in Pig Latin is " + word[i:] + word[0:i] + "ay")

break

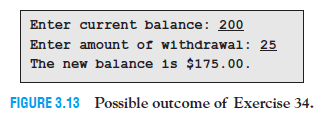
[run]

Enter word to translate: chip

The word in Pig Latin is ipchay



34. **Savings Account** Write a program to process a savings-account withdrawal. The program should request the current balance and the amount of the withdrawal as input and then display the new balance. If the withdrawal is greater than the original balance, the program should display “Withdrawal denied.” If the new balance is less than $150, the message “Balance below $150” should also be displayed. See Fig. 3.13.



[code]

cBal = eval(input("Enter current balance: "))

wdBal = eval(input("Enter amount of withdrawal: "))

if wdBal > cBal:

print("Withdrawal denied.")

elif (cBal - wdBal) < 150:

print("Balance below $150")

print("The new balance is ${0:.2f}".format(cBal - wdBal))

else:

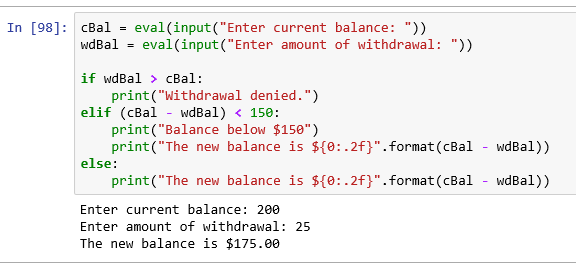
print("The new balance is ${0:.2f}".format(cBal - wdBal))

[run]

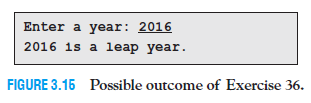
Enter current balance: 200

Enter amount of withdrawal: 25

The new balance is $175.00



36. **Year** The current calendar, called the Gregorian calendar, was introduced in 1582. Every year divisible by four was created to be a leap year, with the exception of the years ending in 00 (that is, those divisible by 100) and not divisible by 400. For instance, the years 1600 and 2000 are leap years, but 1700, 1800, and 1900 are not. Write a program that requests a year as input and states whether it is a leap year. See Fig. 3.15.



[code]

year = eval(input("Enter a year: "))

if (year % 4) == 0:

if (year % 400) == 0:

print("{0} is a leap year".format(year))

elif (year % 100) == 0:

print("{0} is not a leap year".format(year))

else:

print("{0} is a leap year".format(year))

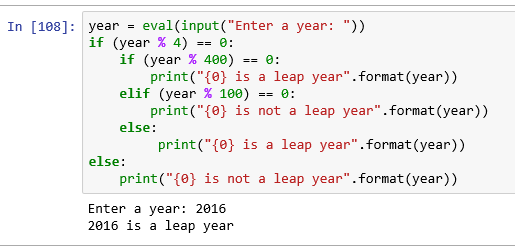
else:

print("{0} is not a leap year".format(year))

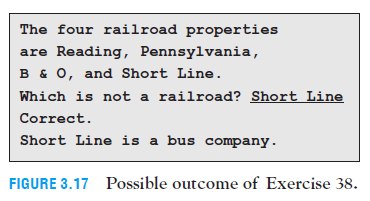
[run]

Enter a year: 2016

2016 is a leap year



38. **Railroad Properties** One of the four railroad properties in Monopoly is not an actual railroad. Write a program that displays the names of the four properties and asks the user to identify the property that is not a railroad. The user should be informed if the selection is correct or not. See Fig. 3.17.



[code]

##38

print("The four railroad properties are Reading, Pennsylvania, B & O, and Short Line.")

ans = str(input("Which is not a railroad? "))

if ans.title() == "Short Line":

print("Correct")

print("{0} is a bus company".format(ans.title()))

else:

print("{0} is a railroad property".format(ans.title()))

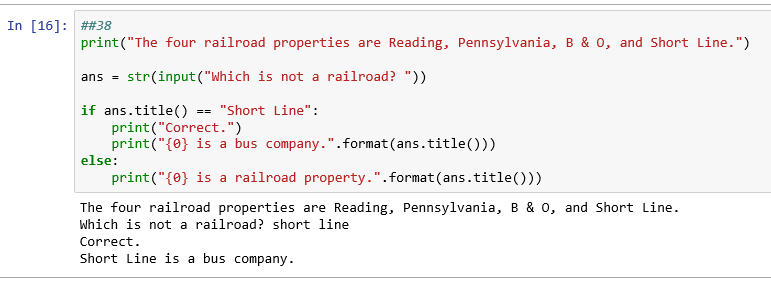
[run]

The four railroad properties are Reading, Pennsylvania, B & O, and Short Line.

Which is not a railroad? short line

Correct.

Short Line is a bus company.0



40. **Graduation Honors** Rewrite the program in Example 8 without **elif** clauses. That is, the task should be carried out with a sequence of simple **if** statements.

[code]

## Bestow graduation honors.

# Request grade point average.

gpa = eval(input("Enter your gpa: "))

# Determine if honors are warranted.

if gpa >= 3.9:

honors = " summa cum laude."

if 3.6 <= gpa < 3.9:

honors = " magna cum laude."

if 3.3 <= gpa < 3.6:

honors = " cum laude."

if gpa < 3.3:

honors = "."

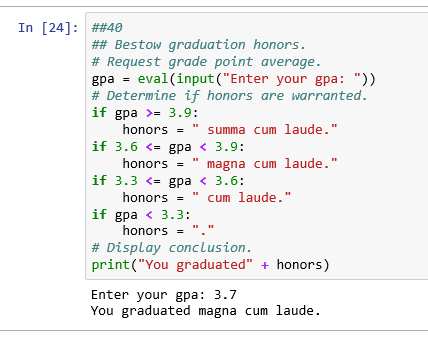
# Display conclusion.

print("You graduated" + honors)

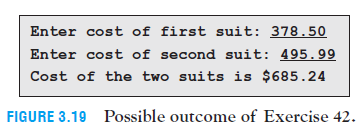
[run]

Enter your gpa: 3.7

You graduated magna cum laude.



42. **Second-Suit-Half-Off Sale** A men’s clothing store advertises that if you buy a suit, you can get a second suit at half-off. What they mean is that if you buy two suits, then the price of the lower-cost suit is reduced by 50%. Write a program that accepts the two costs as input and then calculates the total cost after halving the cost of the lowest price suit. See Fig. 3.19.



[code]

suit1 = eval(input("Enter cost of first suit: "))

suit2 = eval(input("Enter cost of second suit: "))

if suit1 <= suit2:

suit1, suit2 = suit2, suit1

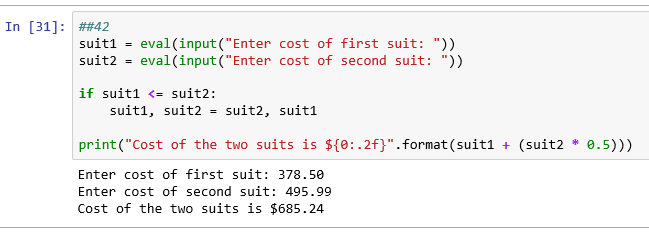
print("Cost of the two suits is ${0:.2f}".format(suit1 + (suit2 \* 0.5)))

[run]

Enter cost of first suit: 378.50

Enter cost of second suit: 495.99

Cost of the two suits is $685.24



Section 3.3 pp 111-117 Exercises 2-32 even

**In Exercises 1 through 8, determine the output displayed.**

2.

num = 3

while num < 15:

num += 5

print(num)

[output]

18

4.

total = 0

num = 1

while num < 5:

total += num

num += 1

print(total)

[output]

10

6.

oceans = ["Atlantic", "Pacific", "Indian", "Arctic", "Antarctic"]

i = len(oceans) - 1

while i >= 0:

if len(oceans[i]) < 7:

del oceans[i]

i = i - 1

print(", ".join(oceans))

[output]

Atlantic, Pacific, Antarctic

8.

numTries = 0

year = 0

while (numTries < 7) and (year != 1964):

numTries += 1

year = int(input( "Try #" + str(numTries) + ": In what year " +

"did the Beatles invade the U.S.? "))

if year == 1964:

print("\nYes. They performed on the Ed Sullivan show in 1964.")

print("You answered correctly in " + str(numTries) + " tries.")

elif year < 1964:

print("Later than", year)

else: # year > 1964

print("Earlier than", year)

if (numTries == 7) and (year != 1964):

print("\nYour 7 tries are up. The answer is 1964.")

#(Assume that the responses are 1950, 1970, and 1964.)

[output]

Try #1: In what year did the Beatles invade the U.S.? 1950

Later than 1950

Try #2: In what year did the Beatles invade the U.S.?

1970 Earlier than 1970

Try #3: In what year did the Beatles invade the U.S.? 1964

Yes. They performed on the Ed Sullivan show in 1964.

You answered correctly in 3 tries.

10.

## Display the numbers from 1 through 5.

num = 1

while (num <= 5):

print(num)

num += 1

[output]

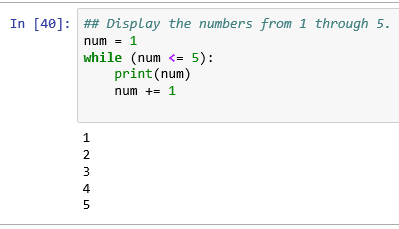
1

2

3

4

5



12.

## Display the elements from a list.

list1 = ['a', 'b', 'c', 'd']

i = 0

while i < len(list1):

print(list1[i])

i += 1

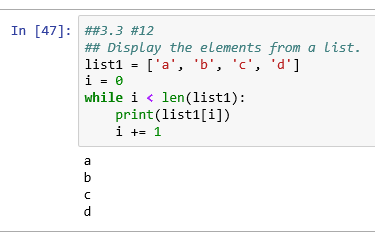
[output]

a

b

c

d



**In Exercises 13 and 14, write a simpler and clearer code that performs the same task as the given code.**

14.

L = [2, 4, 6, 8]

total = 0

i = 0

while i < len(L):

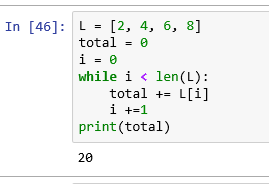
total += L[i]

i +=1

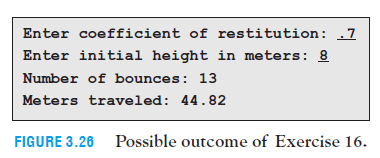
print(total)

[output]

20



16. **Bouncing Ball** The *coefficient of restitution* of a ball, a number between 0 and 1, specifies how much energy is conserved when the ball hits a rigid surface. A coefficient of .9, for instance, means a bouncing ball will rise to 90% of its previous height after each bounce. Write a program to input a coefficient of restitution and an initial height in meters, and report how many times a ball bounces when dropped from its initial height before it rises to a height of less than 10 centimeters. Also report the total distance traveled by the ball before this point. See Fig. 3.26. The coefficients of restitution of a tennis ball, basketball, super ball, and softball are .7, .75, .9, and .3, respectively.



coef = eval(input("Enter coefficient of restitution: "))

ht = eval(input("Enter initial height in meters: "))

bounce = 0

dist = 0

ht = ht \* 100

while ht > 10:

bounce += 1

dist += ht + (ht\* coef)

ht = ht \* coef

print("Number of bounces: {0} \nMeters travelled: {1:.2f}".format(bounce, (dist/100)) )

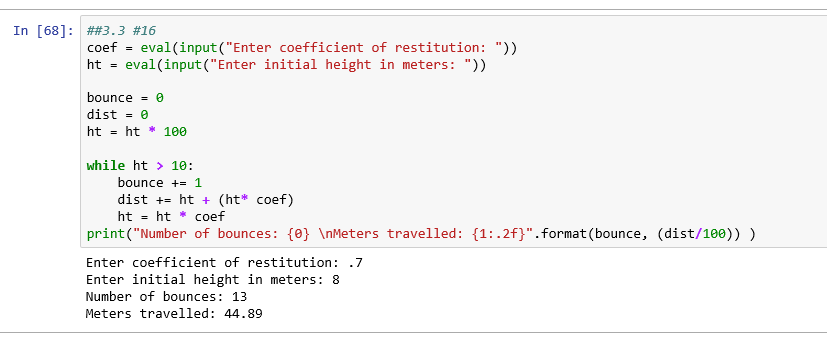
[output]

Enter coefficient of restitution: .7

Enter initial height in meters: 8

Number of bounces: 13

Meters travelled: 44.89



**In Exercises 17 and 18, write a program corresponding to the flowchart.**

18. **Factorization** The flowchart in Fig. 3.30 requests a whole number greater than 1 as input and factors it into a product of prime numbers. ***Note:*** A number is *prime* if its only factors are 1 and itself. See Fig. 3.28.

integer=int(input("Enter a positive integer(>1):"))

factor=2

prime=[]

while integer>1:

if integer%factor==0:

prime.append(factor)

integer=integer/factor

else:

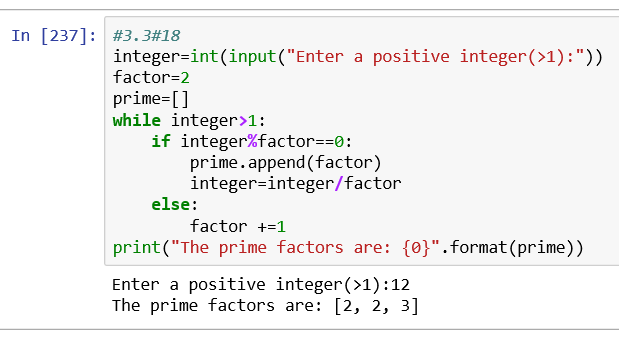
factor +=1

print("The prime factors are: {0}".format(prime))

[output]

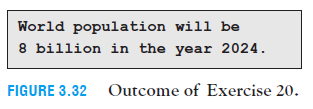
Enter a positive integer(>1):12

The prime factors are: [2, 2, 3]



**In Exercises 19 through 31, write a program to answer the question.**

20. **Population Growth** The world population reached 7 billion people on October 21, 2011, and was growing at the rate of 1.1% each year. Assuming that the population continues to grow at the same rate, approximately when will the population reach 8 billion? See Fig. 3.32.



popu = 7

gRate = 1.1

year = 2011

while popu < 8:

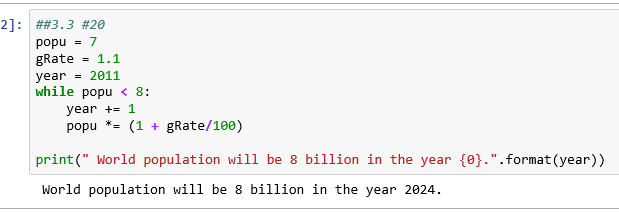
year += 1

popu \*= (1 + gRate/100)

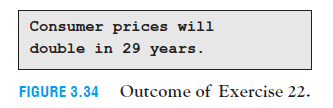
print(" World population will be 8 billion in the year {0}.".format(year))

[output]

World population will be 8 billion in the year 2024.



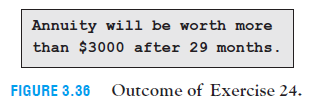
22. **Consumer Price Index** The *consumer price index (CPI)* indicates the average price of a fixed basket of goods and services. It is customarily taken as a measure of inflation and is frequently used to adjust pensions. The CPI was 9.9 in July 1913, was 100 in July 1983, and was 238.25 in July 2014. This means that $9.90 in July 1913 had the same purchasing power as $100.00 in July 1983, and the same purchasing power as $238.25 in July 2014. In 2009, the CPI fell for the first time since 1955. However, for most of the preceding 15 years it had grown at an average rate of 2.5% per year. Assuming that, the CPI will rise at 2.5% per year in the future, in how many years will the CPI have at least doubled from its July 2014 level? ***Note:*** Each year, the CPI will be 1.025 times the CPI for the previous year. See Fig. 3.34.



[output]

24. **Annuity** An *annuity* is a sequence of equal periodic payments. One type of annuity, called a *savings plan*, consists of monthly payments into a savings account in order to generate money for a future purchase. Suppose you decide to deposit $100 at the end of each month into a savings account paying 3% interest compounded monthly. The monthly interest rate will be .03/12 or .0025, and the balance in the account at the end of each month will be computed as

[balance at end of month] = (1.0025) \* [balance at end of previous month] + 100.



months = 1

depAmt = 100

while depAmt <= 3001:

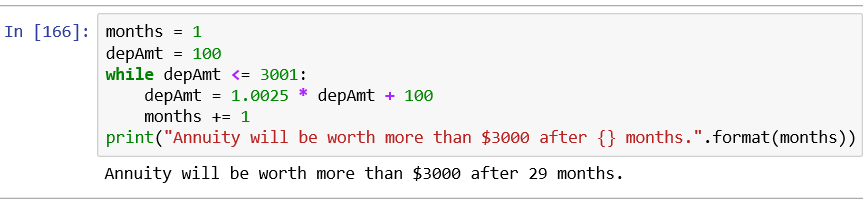
depAmt = 1.0025 \* depAmt + 100

months += 1

print("Annuity will be worth more than $3000 after {} months.".format(months))

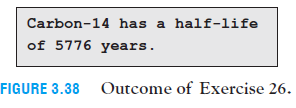
[output]

Annuity will be worth more than $3000 after 29 months.



26.

**Radioactive Decay** Carbon-14 is constantly produced in Earth’s upper atmosphere due to interactions between cosmic rays and nitrogen, and is found in all plants and animals. After a plant or animal dies, its amount of carbon-14 decreases by about .012% per year. Determine the half-life of carbon-14, that is, the number of years required for 1 gram of carbon-14 to decay to less than ½ gram. See Fig. 3.38.



halfLife = 0

gram = 1

while gram >= 0.50:

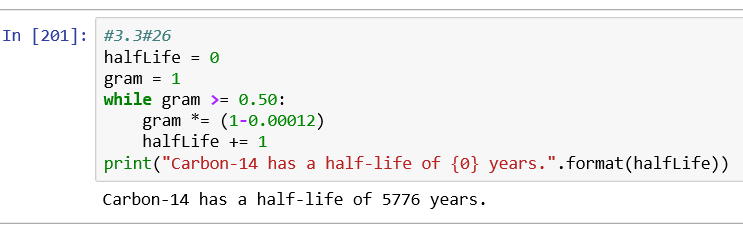
gram \*= (1-0.00012)

halfLife += 1

print("Carbon-14 has a half-life of {0} years.".format(halfLife))

[output]

Carbon-14 has a half-life of 5776 years.



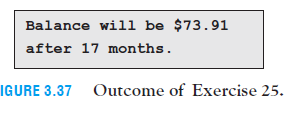
28. **Annuity** An *annuity* is a sequence of equal periodic payments. For one type of

annuity, a large amount of money is deposited into a bank account and then a fixed amount is withdrawn each month. Suppose you deposit $10,000 into such an account paying 3.6% interest compounded monthly, and then withdraw $600 at the end of each month. The monthly interest rate will be .036/12 or .003, and the balance in the account

at the end of each month will be computed as

[balance at end of month] = (1.003) \* [balance at end of previous month] - 600.

After how many months will the account contain less than $600, and what will be the amount in the account at that time? See Fig. 3.37.



fAmt = 10000

withdrawal = 600

months = 0

while fAmt > 600:

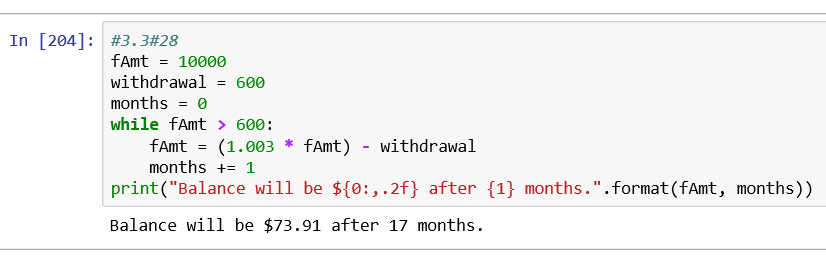
fAmt = (1.003 \* fAmt) - withdrawal

months += 1

print("Balance will be ${0:,.2f} after {1} months.".format(fAmt, months))

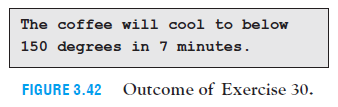
[output]

Balance will be $73.91 after 17 months.



30. **Cooling** *Newton’s Law of Cooling* states that when a hot liquid is placed in a cool room, each minute the decrease in the temperature is approximately proportional to the difference between the liquid’s temperature and the room’s temperature. That is, there is a constant *k* such that each minute the temperature loss is

*k* \* (liquid\_s temperature - room\_s temperature). Suppose a cup of 212°F coffee is placed in a 70°F room and that *k* = .079. Determine the number of minutes required for the coffee to cool to below 150°F. See Fig. 3.42.



coffTemp = 212

roomTemp = 70

k =0.079

mins=0

cooldown = 0

while coffTemp >= 150:

cooldown = k \* (coffTemp - roomTemp)

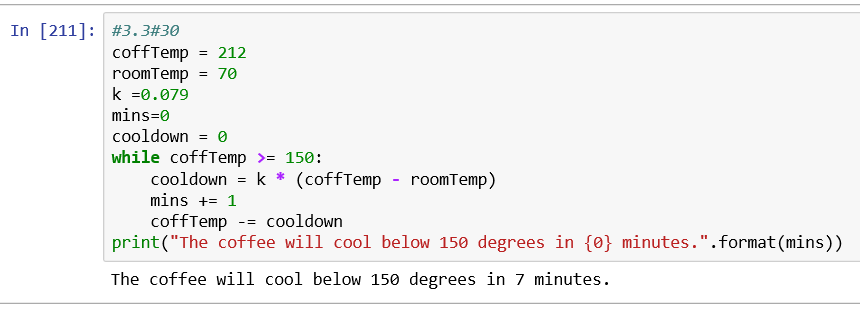
mins += 1

coffTemp -= cooldown

print("The coffee will cool below 150 degrees in {0} minutes.".format(mins))

[output]

The coffee will cool below 150 degrees in 7 minutes.



**Section 3.4 pp 127-136 Exercises 2-84 even**

**In Exercises 1 through 8, determine the sequence generated by the range function.**

2. **range(-11, -7)**

[output]

-11, -10, -9, -8

4. **range(2010, 2030, 5)**

[output]

2010, 2015, 2020, 2025

6. **range(1)**

[output]

0

8. **range(12, 2, -5)**

[output]

12, 7

**In Exercises 9 through 16, determine a range function that generates the sequence of numbers.**

10. **0, 1, 2, 3**

[output]

range(0,4)

12. **4, 3, 2, 1**

[output]

range(4,0,-1

14. **7**

[output]

range(7,8)

16. **-5, -3, -1, 1**

[output]

range(-5,2,1)

**In Exercises 17 through 40, determine the output displayed.**

18. for i in range(3, 7):

print(2 \* i)

[output]

6

8

10

12

20. for i in range(-9, 0, 3):

print(i)

[output]

-9

-6

-3

22. n = 3

total = 0

for i in range(1, n + 1):

total += i

print(total)

[output]

6

24. for countdown in range(10, 0, -1):

print(countdown)

[output]

10

9

8

7

6

5

4

3

2

1

26. numCaps = 0

name = "United States of America"

for ch in name:

if ch.isupper():

numCaps += 1

print(numCaps)

[output]

3

28. word = "cloudier"

newWord = ""

evenIndex = True

for ch in word:

if evenIndex:

newWord += ch

evenIndex = not evenIndex

print(newWord)

[output]

code

30. for ch in "Python":

break

print(ch)

[output]

P

32. list1 = [2, 9, 6, 7, 13, 3]

maxOfOdds = 0

for num in list1:

if (num % 2 == 1) and (num > maxOfOdds):

maxOfOdds = num

print(maxOfOdds)

[output]

13

34. numOfNumbers = 0

list1 = ["three", 4, 5.7, "six", "seven", 8, 3.1416]

for item in list1:

if isinstance(item, str):

continue

numOfNumbers += 1

print(numOfNumbers)

[output]

4

36. # I'm looking over a four leaf clover.

leaves = ("sunshine","rain", "the roses that bloom in the lane",

"somebody I adore")

number = 1

for leaf in leaves:

print("Leaf", str(number) + ':', leaf)

number += 1

[output]

Leaf 1: sunshine

Leaf 2: rain

Leaf 3: the roses that bloom in the lane

Leaf 4: somebody I adore

38.

[output]

40.

infile = open("States.txt", 'r')

for line in infile:

continue

infile.close()

print(line, end="")

[output]

Hawaii

**In Exercises 41 through 46, identify all errors.**

42. for i in range(1, 4):

print(i + " " + 2 \*\* i)

[output]

unsupported operand type(s) for +: 'int' and 'str'

44. list1 = ['a', 'b', 'c']

for letter in list1:

letter = letter.upper()

print(list1)

[output]

Output will be same as list1. Because upper case is assigned to letter but list1 is printed.

46. list1 = ["one", "two", "three", "four"]

for item in list1:

item = item.upper()

print(list1)

[output]

Output will be same as list1. Because upper case is assigned to item but list1 is printed.

**In Exercises 47 and 48, rewrite the program using a for loop.**

48. for i in range(4):

print("hello")

[output]

hello

hello

hello

hello

**Simplify the programs in Exercises 49 and 50.**

50. lakes = ["Erie", "Huron", "Michigan", "Ontario", "Superior"]

print(" | ".join(lakes))

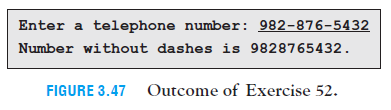
[output]

Erie | Huron | Michigan | Ontario | Superior

**In Exercises 51 through 65, write a program to carry out the stated task.**

52. **Phone Number** Remove the dashes from a telephone number input by the user. See

Fig. 3.47.



pNumber = input("Enter a telephone number:")

i=0

for i in range(0, len(pNumber)-1):

j = pNumber.find("-")

break

pNumber = pNumber[0:j] + pNumber[j+1:]

for i in range(0, len(pNumber)-1):

j = pNumber.find("-")

break

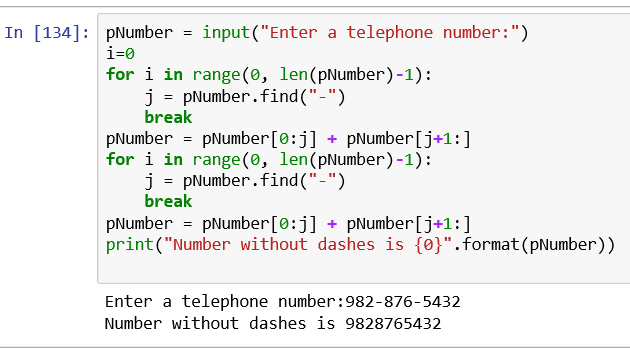
pNumber = pNumber[0:j] + pNumber[j+1:]

print("Number without dashes is {0}".format(pNumber))

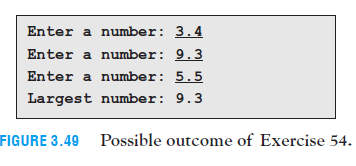
[output]

Enter a telephone number:982-876-5432

Number without dashes is 9828765432



54. **Largest Number** Without using a list, find the largest of three numbers obtained from the user. See Fig. 3.49.



number1 = eval(input("Enter a number: "))

number2 = eval(input("Enter a number: "))

number3 = eval(input("Enter a number: "))

largest = number3

if number1 > number3:

largest = number1

if number2 > largest:

largest = number2

print("Largest number: {0}".format(largest))

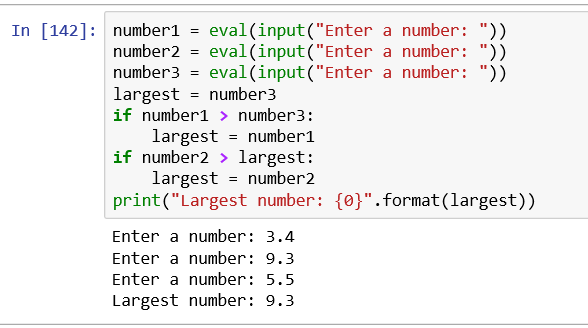
[output]

Enter a number: 3.4

Enter a number: 9.3

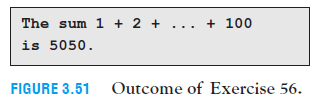
Enter a number: 5.5

Largest number: 9.3



56. **Sum of Numbers** Find the sum of the first one hundred positive integers. See

Fig. 3.51.



i = 0

sum = 0

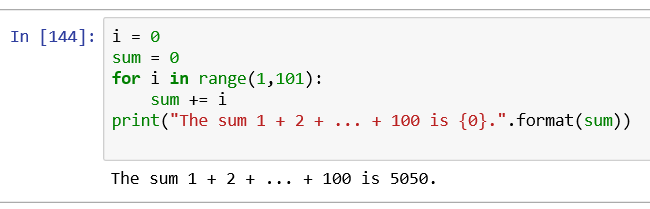
for i in range(1,101):

sum += i

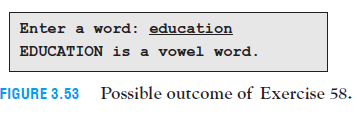
print("The sum 1 + 2 + ... + 100 is {0}.".format(sum))

[output]

The sum 1 + 2 + ... + 100 is 5050.



58. **Vowel Words** A **vowel word** is a word that contains every vowel. Some examples of vowel words are *sequoia*, *facetious*, and *dialogue*. Determine if a word input by the user is a vowel word. See Fig. 3.53 on the previous page.



word=input("Enter a word:")

vowels=['a', 'e', 'i', 'o', 'u']

vowel1=[]

for i in word.lower():

if i in vowels:

vowel1.append(i)

vowel1.sort()

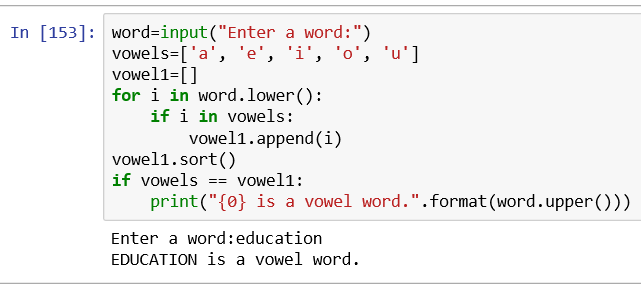
if vowels == vowel1:

print("{0} is a vowel word.".format(word.upper()))

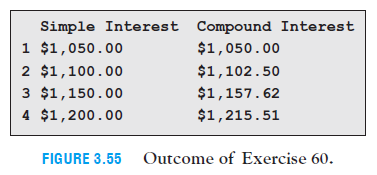
[output]

Enter a word:education

EDUCATION is a vowel word.



60. **Simple versus Compound Interest** When $1,000 is invested at 5% simple interest, the amount grows by $50 each year. When money is invested at 5% interest compounded annually, the amount at the end of each year is 1.05 times the amount at the beginning of that year. Display the amounts after the first four years for a $1,000 investment at 5% simple and compound interest. See Fig. 3.55.



siAmount = 1000

inr = 5

print(" Simple Interest\tCompound Interest")

ciAmount = siAmount

for i in range(4):

siAmount += (inr \* 10)

ciAmount \*= (1+ (inr/100))

print(i,"${0:.2f}\t\t${1:.2f}".format(siAmount, ciAmount))

[output]

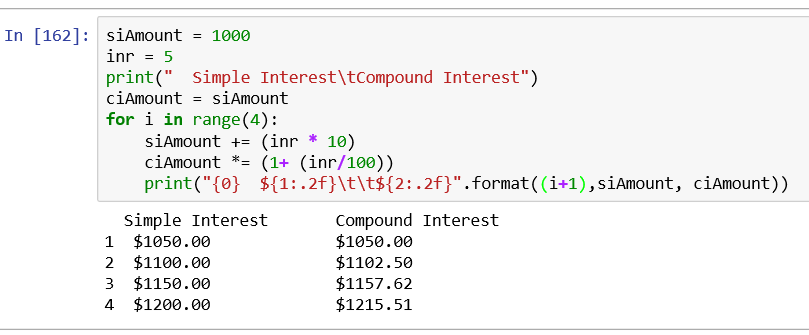
Simple Interest Compound Interest

1 $1050.00 $1050.00

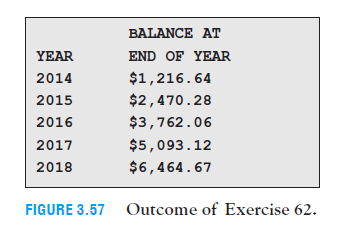
2 $1100.00 $1102.50

3 $1150.00 $1157.62

4 $1200.00 $1215.51



62. **Annuity** Refer to the annuity discussed in Exercise 24 of Section 3.3. Assume that the first deposit is made at the end of January 2014, and display the balance in the account at the end of each year from 2014 to 2018. See Fig. 3.57.



depAmt = 0

print("YEAR \tBALANCE AT\n\tEND OF YEAR")

for i in range(2014,2019,1):

j = 0

while j < 12:

depAmt = 1.0025 \* depAmt + 100

j += 1

print("{0}\t${1:,.2f}".format(i, depAmt))

[output]

YEAR BALANCE AT

END OF YEAR

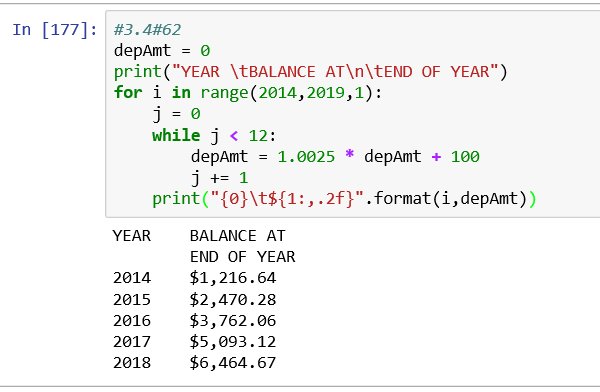
2014 $1,216.64

2015 $2,470.28

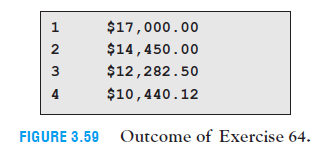
2016 $3,762.06

2017 $5,093.12

2018 $6,464.67



64. **Automobile Depreciation** A rule of thumb states that cars in personal use depreciate by 15% each year. Suppose a new car is purchased for $20,000. Produce a table showing the value of the car at the end of each of the next four years. See Fig. 3.59.



purchValue=20000

for i in range(4):

purchValue \*= 0.85

print("{0}\t${1:,.2f}".format((i+1), purchValue))

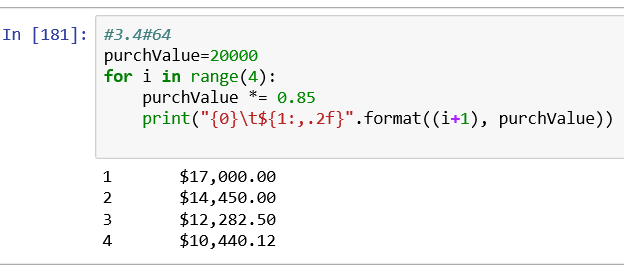
[output]

1 $17,000.00

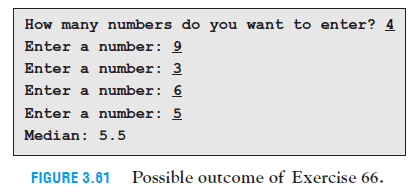
2 $14,450.00

3 $12,282.50

4 $10,440.12



66. **Median** The **median** of an ordered set of measurements is a number separating the lower half from the upper half. If the number of measurements is odd, the median is the middle measurement. If the number of measurements is even, the median is the average of the two middle measurements. Write a program that requests a number *n* and a set of *n* measurements (not necessarily ordered) as input and then displays the median of the measurements. See Fig. 3.61.



num = eval(input("How many numbers do you want to enter? "))

order=[]

for i in range(num):

s = eval(input("Enter a number:"))

order.append(s)

order.sort()

if num%2==0:

mid = (order[int(num/2)-1] + order[int(num/2)]) / 2

else:

mid = order[int((num)/2)]

print("Median:",mid)

[output]

How many numbers do you want to enter? 4

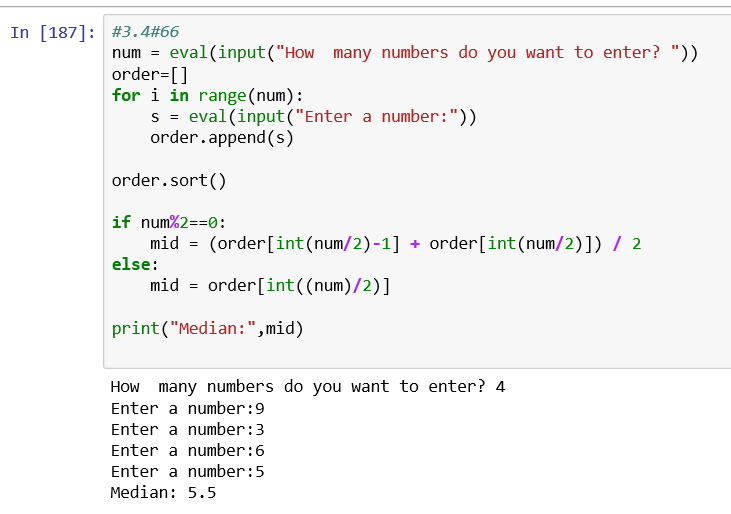
Enter a number:9

Enter a number:3

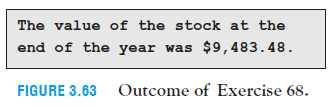
Enter a number:6

Enter a number:5

Median: 5.5

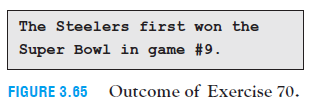


68. **Misleading Percentages** At the beginning of the year you purchased a stock for $10,000. At the end of the year you are told that your stock gained 18% during the past month and that the average monthly change was +1%. Sounds like good news, doesn’t it? Later you learn that your stock lost 16% during each of the first six months of the year and gained 18% during each of the last six months of the year. Write a program to determine the value of the stock at the end of the year. See Fig. 3.63.



[output]

70. **Super Bowl** Write a program to determine the number of the game in which the Steelers first won the Super Bowl. See Fig. 3.65.



List2 = []

count=0

infile = open("G:/SJU/10 - Python/Text File/SBWinners.txt", 'r')

for word in infile:

if word =="Steelers\n":

count+=1

print("The Steelers first won the Super Bowl in game #{0}".format(count))

break

else:

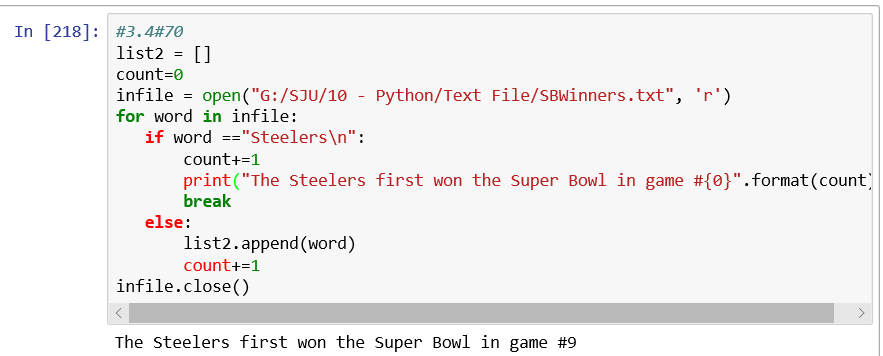
list2.append(word)

count+=1

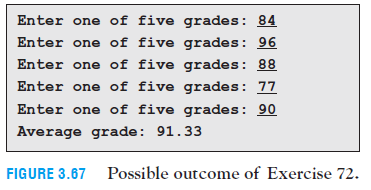
infile.close()

[output]

The Steelers first won the Super Bowl in game #9



72. **Average Grade** Write a program that requests five grades as input and then calculates the average after dropping the two lowest grades. See Fig. 3.67.



grade = []

for i in range(5):

g=eval(input("Enter one of five grades: "))

grade.append(g)

grade.sort(reverse=True)

sum1 = 0

for i in range(3):

sum1 += grade[i]

average = sum1 /3

print("Average grade: {0:.2f}".format(average))

[output]

Enter one of five grades: 84

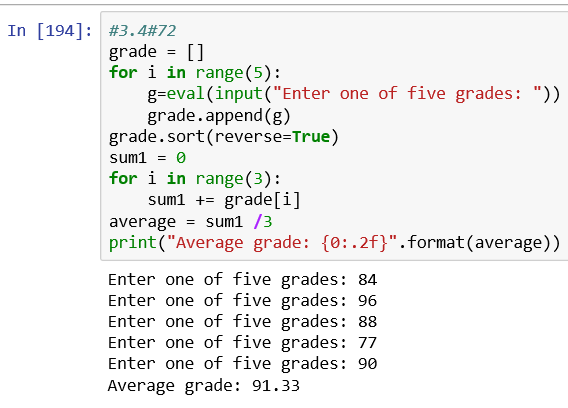
Enter one of five grades: 96

Enter one of five grades: 88

Enter one of five grades: 77

Enter one of five grades: 90

Average grade: 91.33



74.

word="NAISNIENLGELTETWEORRSD"

list1=list(word)

even=[]

odd=[]

for i in range(0,len(list1)):

if i%2==0:

even.append(list1[i])

#even="".join(even)

else:

odd.append(list1[i])

#odd="".join(odd)

print("Starting word:",word)

print("Crossed out letters: ", "".join(even))

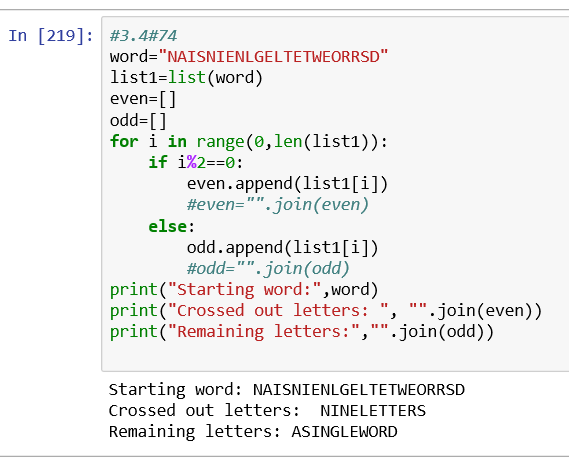
print("Remaining letters:","".join(odd))

[output]

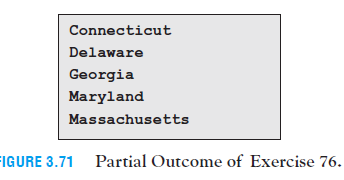
Starting word: NAISNIENLGELTETWEORRSD

Crossed out letters: NINELETTERS

Remaining letters: ASINGLEWORD



76. **Original U.S. States** The file **States.txt** contains the 50 U.S. states in the order in which they joined the union. Write a program to display the original 13 states in alphabetical order. Fig. 3.71 shows the first five lines of output.



List3 = []

infile = open("G:/SJU/10 - Python/Text File/States.txt", 'r')

count=0

for line in infile:

List3.append(line)

if len(List3)==13:

break

List3.sort()

for i in range(5):

print(List3[i])

infile.close()

[output]

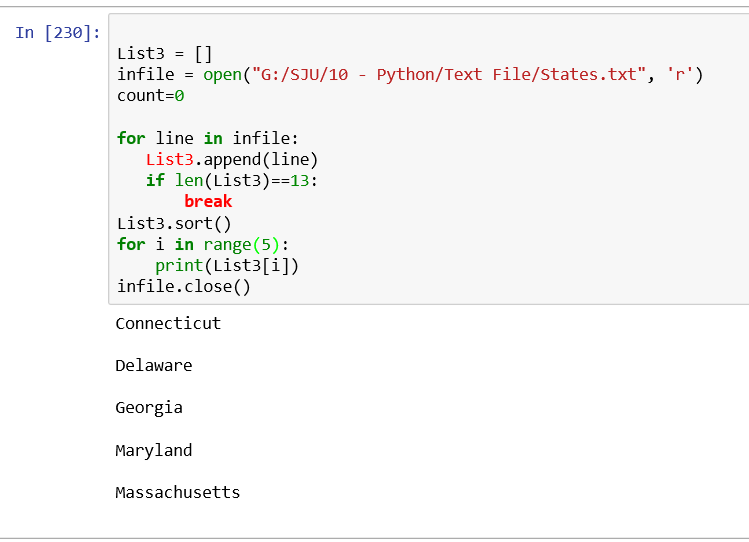
Connecticut

Delaware

Georgia

Maryland

Massachusetts



78.

#3.4#78

abcd = 2178

#special = not True

sNum = 4 \* 2178

for i in range(4):

if abcd[i] == sNum[i-len(snum)]:

special = isTrue

else:

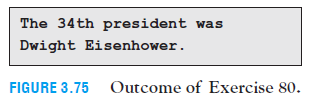
print("{0} is not a special number".format(abcd))

break

print("special")

[output]

80. **U.S. Presidents** Write a program that determines the name of the 34th president. Do not use a list in the program. See Fig. 3.75.



file = open("G:/SJU/10 - Python/Text File/USPres.txt", 'r')

count=0

for line in file:

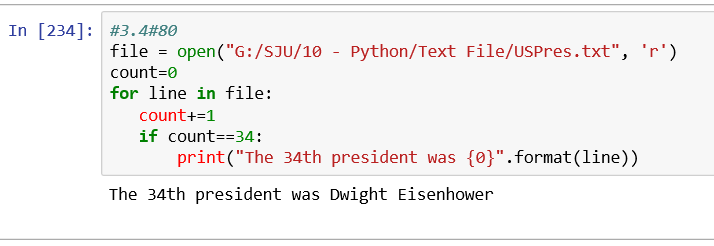
count+=1

if count==34:

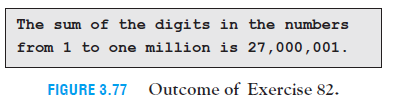
print("The 34th president was {0}".format(line))

[output]

The 34th president was Dwight Eisenhower



82. **Digit Sum** Write a program to calculate the total sum of the digits in the integers from 1 to a million. See Fig. 3.77.



#3.4#82

digits=0

for i in range(1,1000001):

digits+=1

print("The sum of the digits in the numbers from 1 to one million is","{0:,d}".format(digits))

[output]