Python for Tweens and Teens

Learn Computational and Algorithmic Thinking

Revised Second Edition

The Answers

By
Aristides S. Bouras

Python for Tweens and Teens – The Answers

Revised Second Edition

https://www.bouraspage.com

RCode: 220225

Warning and Disclaimer

This book is designed to provide information about learning “Computational and Algorithmic Thinking,” mainly through the use of Python programming language. Every effort has been taken to make this book compatible with all releases of Python 3.x, and it is almost certain to be compatible with any future releases of Python.

The information is provided on an “as is” basis. The author shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the files that may accompany it.

Table of Contents

How to Report Errata

Although I have taken great care to ensure the accuracy of the content of this book, mistakes do occur. If you find a mistake in this book, either in the text or the code, I encourage you to report it to me. By doing so, you can save other readers from frustration and, of course, help me to improve the next release of this book. If you find any errata, please feel free to report them by visiting the following address:

https://www.bouraspage.com/report-errata

Once your errata are verified, your submission will be accepted and the errata will be uploaded to my website, and added to any existing list of errata.

Chapter 1

1.7 Review Questions: True/False

1. False
2. False
3. True
4. False
5. False
6. True
7. True
8. False
9. False
10. False
11. True
12. True
13. False
14. False
15. False
16. True
17. True
18. False
19. False
20. True
21. False
22. False
23. True

1.8 Review Questions: Multiple Choice

1. b
2. d
3. c
4. g
5. d
6. c
7. b
8. c
9. b
10. a

Chapter 3

3.11 Review Questions: True/False

1. True
2. False
3. False
4. False
5. True
6. False
7. True
8. True
9. True
10. False
11. True
12. False
13. True
14. False
15. False
16. False
17. True
18. False
19. False
20. False
21. False
22. True

3.12 Review Questions: Multiple Choice

1. a
2. c
3. c
4. a
5. a
6. d

Chapter 4

4.5 Review Questions: True/False

1. False
2. True
3. False
4. True
5. False
6. True
7. False
8. True
9. False
10. True
11. True
12. True
13. False

4.6 Review Questions: Multiple Choice

1. e
2. a
3. b
4. c
5. c
6. d

4.7 Review Exercises

1. 1 – c, 2 – d, 3 – a, 4 – b
2. 1 – d, 2 – c, 3 – b, 4 – a
3.

Value	Data Type	Declaration and Initialization
The name of my friend	String	`name = "Mark"`
My address	String	`address = "254 Lookout Rd. Wilson, NY 27893"`
The average daily temperature	Float	`average = 70.3`
A telephone number	String	`phone_number = "1-891-764-2410"`
My Social Security Number (SSN)	String	`ssn = "123-45-6789"`
The speed of a car	Float	`speed = 90.5`
The number of children in a family	Integer	`children = 3`

Chapter 5

5.4 Review Questions: True/False

1. True
2. True
3. True
4. False
5. False

5.5 Review Questions: Multiple Choice

1. a
2. c
3. b
4. b

Chapter 6

6.6 Review Questions: True/False

1. False
2. True
3. False
4. False
5. False
6. False
7. False
8. False
9. False
10. False
11. False
12. True
13. False
14. False
15. True
16. False
17. False
18. False
19. False

6.7 Review Questions: Multiple Choice

1. c
2. c
3. b
4. d
5. d
6. b
7. d
8. a

6.8 Review Exercises

1. ii, iv, v, ix, x
2. i – String, ii – Boolean, iii – String, iv – String, v – Float, vi – Integer
3. i – b, ii – d, iii – c, iv – e
4. i – 26, ii – 28
5. i – 0, ii – 4
6. i – 2.0, ii – 40
7. My name is Alexander the Great
8. i – 3, ii – 1
9. California California California

Chapter 9

9.2 Review Exercises

1. Solution

```
b = float(input("Enter base: "))
h = float(input("Enter height: "))

area = b * h / 2

print(area)
```

2. Solution

```
f = float(input("Enter temperature in Fahrenheit: "))

k = (f + 459.67) / 1.8
```

```
print(k)
```

3. Solution

```
angle1 = float(input("Enter 1st angle: "))
angle2 = float(input("Enter 2nd angle: "))

angle3 = 180 - angle1 - angle2

print(angle3)
```

4. Solution

```
g1 = int(input("Enter 1st grade: "))
g2 = int(input("Enter 2nd grade: "))
g3 = int(input("Enter 3rd grade: "))
g4 = int(input("Enter 4th grade: "))

average = (g1 + g2 + g3 + g4) / 4.0

print(average)
```

5. Solution

```
PI = 3.14159

r = float(input("Enter radius: "))

perimeter = 2 * PI * r

print(perimeter)
```

6. Solution

```
PI = 3.14159

d = float(input("Enter diameter (in meters): "))

radius = d / 2

volume = 4 / 3 * PI * radius ** 3

print(volume)
```

7. Solution

Only a), e) and g) are syntactically correct. The latter is more user friendly.

8. Solution

```
PI = 3.14159

d = float(input("Enter diameter (in meters): "))

radius = d / 2
```

```
perimeter = 2 * PI * radius
area = PI * radius ** 2
volume = 4 / 3 * PI * radius ** 3

print(radius, perimeter, area, volume)
```

9. Solution

```
w = int(input("Enter weight in pounds: "))
h = int(input("Enter height in inches: "))

bmi = w * 703 / h ** 2

print(bmi)
```

10.Solution

```
m = int(input("Enter current month: "))
d = int(input("Enter current day: "))

days_passed = (m - 1) * 30 + d
days_left = 360 - days_passed

print(days_left)
```

11.Solution

```
first_name = input("First name: ")
middle_name = input("Middle name: ")
last_name = input("Last name: ")
title = input("Title: ")

print(title, first_name, middle_name, last_name)
print(first_name, middle_name, last_name)
print(last_name, ",", first_name)
print(last_name, ",", first_name, middle_name)
print(last_name, ",", first_name, middle_name, ",", title)
print(first_name, last_name)
```

Chapter 10

10.3 Review Questions: True/False

1. True
2. False
3. False
4. True
5. False
6. True
7. False
8. True
9. True
10. True
11. False
12. True
13. False

10.4 Review Exercises

1. 2
2. i – 2.5, ii – 2.2
3. i – 4, ii – 9
4. i – 5.5, ii – 3.5
5. i – 31, ii – 32

6. Solution

```
import math

a = float(input("Enter right angle side A of a right-angled triangle: "))
b = float(input("Enter right angle side B of a right-angled triangle: "))

hypotenuse = math.sqrt(a ** 2 + b ** 2)

print(hypotenuse)
```

Chapter 11

11.4 Review Questions: True/False

1. True
2. False
3. False
4. True
5. True
6. True
7. False
8. False
9. True
10. False
11. True

11.5 Review Questions: Multiple Choice

1. d
2. b
3. a
4. b
5. c
6. a
7. c

11.6 Review Exercises

1. Solution

```
import random

alphabet = "abcdefghijklmnopqrstuvwxyz"

random_word = alphabet[random.randrange(26)].upper() + alphabet[random.randrange(26)] +  \
              alphabet[random.randrange(26)] + alphabet[random.randrange(26)]

print(random_word)
```

2. Solution

```
import random
name = input("Enter name: ")
```

```
x = name.lower().replace(" ", "")  #Convert to lower case and remove spaces

secret_password = x[random.randrange(len(x))] + x[random.randrange(len(x))] +    \
                  x[random.randrange(len(x))] + str(random.randrange(1000, 10000))

print(secret_password)
```

3. Solution

First approach

```
number = int(input("Enter a three-digit integer: "))

#Convert the number to string
number_string = str(number)

#Reverse the string
reversed_string = 100 * int(number_string[2])   /
                + 10 * int(number_string[1])    /
                + int(number_string[0])

#Convert the reversed string to integer
reversed_number = int(reversed_string)

print(reversed_number)
```

Second approach

```
number = int(input("Enter a three-digit integer: "))

#Convert the number to string
number_string = str(number)

#Reverse the string
reversed_string = number_string[2] + number_string[1] + number_string[0]

#Convert the reversed string to integer
reversed_number = int(reversed_string)

print(reversed_number)
```

Third approach

```
number = int(input("Enter an integer: "))

#Convert the number to string
number_string = str(number)

#Reverse the string
reversed_string = number_string[::-1]

#Convert the reversed string to integer
reversed_number = int(reversed_string)
```

NOTICE

The advantage of this approach is that the user is allowed to enter any integer, no matter how small or large!

```
print(reversed_number)
```

Chapter 12

12.10 Review Questions: True/False

1. True
2. False
3. False
4. False
5. False
6. True
7. True
8. True
9. True
10. True
11. True
12. True
13. False
14. False
15. True
16. True
17. False
18. True
19. True

12.11 Review Questions: Multiple Choice

1. b
2. a
3. a
4. a
5. c

12.12 Review Exercises

1. Solution

i. b, d, f

ii. i

iii. c, e

iv. a, g, h

2. Solution

a	b	c	a != 1	b > a	c / 2 > 2 * a
3	–5	8	*True*	*False*	*False*
1	10	20	*False*	*True*	*True*
–4	–2	–9	*True*	*True*	*True*

3. Solution

BE1 **(Boolean Expression 1)**	***BE2*** **(Boolean Expression 2)**	***BE1* or *BE2***	***BE1* and *BE2***	**not(*BE2*)**
False	False	*False*	*False*	*True*
False	True	*True*	*False*	*False*
True	False	*True*	*False*	*True*
True	True	*True*	*True*	*False*

4. Solution

a	b	c	a > 3 or c > b and c > 1	a > 3 and c > b or c > 1
4	−6	2	*True*	*True*
−3	2	−4	*False*	*False*

5. Solution

Expression	Value
`(x + y) ** 3`	*8*
`(x + y) / (x ** 2 - 14)`	*1*
`(x - 1) == y + 5`	*True*
`x > 2 and y == 1`	*False*
`x == 1 or not(flag == False)`	*True*

6. Solution

a. `age < 12 and age != 8`

b. `6 <= age <= 9 or age == 11`

c. `age > 7 and age != 10 and age != 12`

d. `age == 6 or age == 9 or age == 11`

e. `6 <= age <= 12 and age != 8`

f. `age != 7 and age != 10`

Chapter 13

13.2 Review Questions: True/False

1. False
2. False
3. True
4. False

13.3 Review Questions: Multiple Choice

1. b
2. a
3. d
4. c

13.4 Review Exercises

1. Solution

```
x = float(input())

y = 5
if x * y / 2 > 20:
    y *= 2
    x = 4 * x ** 2
```

```
print(x, y)
```

2. Solution

i. 9 12 ii. 2 2

3. Solution

```
x = float(input("Enter a number: "))
if x > 0:
    print("Positive")
```

4. Solution

```
x = float(input("Enter a number: "))
y = float(input("Enter a second number"))

if x > 0 and y > 0:
    print("Positives")
```

5. Solution

```
x = int(input("Enter your age: "))

if x > 14:
    print("You can drive a car in Kansas (USA)")
```

6. Solution

```
s = input("Enter a string: ")

if s == s.upper():
    print("Uppercase")
```

7. Solution

```
s = input("Enter a string: ")

if len(s) > 20:
    print("Many characters")
```

8. Solution

```
n1 = float(input("Enter 1st number: "))
n2 = float(input("Enter 2nd number: "))
n3 = float(input("Enter 3rd number: "))

if n1 < 0 or n2 < 0 or n3 < 0:
    print("Among the given numbers, there is a negative one!")
```

9. Solution

```
t1 = float(input("Enter 1st temperature: "))
t2 = float(input("Enter 2nd temperature: "))
t3 = float(input("Enter 3rd temperature: "))

average = (t1 + t2 + t3) / 3
```

```python
if average > 60:
    print("Heat Wave")
```

10.Solution

First approach

```python
w1 = float(input("Enter the weight of the 1st person: "))
w2 = float(input("Enter the weight of the 2nd person: "))
w3 = float(input("Enter the weight of the 3rd person: "))
w4 = float(input("Enter the weight of the 4th person: "))

maximum = w1

if w2 > maximum:
    maximum = w2

if w3 > maximum:
    maximum = w3

if w4 > maximum:
    maximum = w4

print(maximum)
```

Second approach

```python
w1 = float(input("Enter the weight of the 1st person: "))
w2 = float(input("Enter the weight of the 2nd person: "))
w3 = float(input("Enter the weight of the 3rd person: "))
w4 = float(input("Enter the weight of the 4th person: "))

print(max(w1, w2, w3, w4))
```

11.Solution

```python
a1 = int(input("Enter the age of the 1st person: "))
n1 = input("Enter the name of the 1st person: ")

a2 = int(input("Enter the age of the 2nd person: "))
n2 = input("Enter the name of the 2nd person: ")

a3 = int(input("Enter the age of the 3rd person: "))
n3 = input("Enter the name of the 3rd person: ")

a4 = int(input("Enter the age of the 4th person: "))
n4 = input("Enter the name of the 4th person: ")

minimum = a1
m_name = n1

if a2 < minimum:
    minimum = a2
    m_name = n2

if a3 < minimum:
    minimum = a3
    m_name = n3

if a4 < minimum:
    minimum = a4
    m_name = n4

print("The youngest person is", m_name)
```

12.Solution

First approach

```
a1 = int(input("Enter the age of the 1st person: "))
a2 = int(input("Enter the age of the 2nd person: "))
a3 = int(input("Enter the age of the 3rd person: "))

minimum = a1
if a2 < minimum:
    minimum = a2
if a3 < minimum:
    minimum = a3

maximum = a1
if a2 > maximum:
    maximum = a2
if a3 > maximum:
    maximum = a3

middle = a1 + a2 + a3 - minimum - maximum

print(middle)
```

Second approach

```
a1 = int(input("Enter the age of the 1st person: "))
a2 = int(input("Enter the age of the 2nd person: "))
a3 = int(input("Enter the age of the 3rd person: "))

middle = a1 + a2 + a3 - min(a1, a2, a3) - max(a1, a2, a3)

print(middle)
```

Chapter 14

14.2 Review Questions: True/False

1. False
2. True
3. False
4. False

14.3 Review Questions: Multiple Choice

1. a
2. a
3. a
4. d
5. c

14.4 Review Exercises

1. Solution

i. 1 ii. 5

2. Solution

i. 7.0 18.0 ii. 0.5 3.5

3. Solution

```
num = float(input("Enter a number: "))
if num > 100:
    print("Given number is greater than 100")
else:
    print("Given number is less than or equal to 100")
```

4. Solution

```
num = float(input("Enter a number: "))
if num >= 0 and num <= 100:
    print("Given number is between 0 and 100")
else:
    print("Given number is not between 0 and 100")
```

5. Solution

```
num = int(input())
if num >= 1000 and num <= 9999:
    print(num, "is a four-digit integer")
else:
    print(num, "is not a four-digit integer")
```

6. Solution

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

if num1 < num2:
    print(num1)
else:
    print(num2)
```

7. Solution

```
name1 = input("Enter team name 1: ")
name2 = input("Enter team name 2: ")
goals1 = int(input("Enter goals " + name1 + " scored: "))
```

```
goals2 = int(input("Enter goals " + name2 + " scored: "))

if goals1 > goals2:
    print("Winner:", name1)
else:
    print("Winner:", name2)
```

8. Solution

```
a = float(input("Enter 1st jump in meters: "))
b = float(input("Enter 2nd jump in meters: "))
c = float(input("Enter 3rd jump in meters: "))

average = (a + b + c) / 3

if average >= 8:
    print("Qualified")
else:
    print("Disqualified")
```

Chapter 15

15.2 Review Questions: True/False

1. True
2. False
3. False
4. False
5. False
6. True
7. True

15.3 Review Exercises

1. Solution

i. 1   ii. 2   iii. 4   iv. 4

2. Solution

i. 0 5.0   ii. 10 90.0   iii. 20 160.0

3. Solution

```
name1 = input("Enter team name 1: ")
name2 = input("Enter team name 2: ")
goals1 = int(input("Enter goals " + name1 + " scored: "))
goals2 = int(input("Enter goals " + name2 + " scored: "))

if goals1 > goals2:
    print("Winner:", name1)
elif goals2 > goals1:
    print("Winner:", name2)
else:
    print("It's a tie!")
```

4. Solution

First approach

```
a = int(input("Enter an integer between -9999 and 9999: "))

if -9999 <= a <= -1000 or 1000 <= a <= 9999:
    n = 4
elif -999 <= a <= -100 or 100 <= a <= 999:
    n = 3
elif -99 <= a <= -10 or 10 <= a <= 99:
    n = 2
else:
    n = 1

print("You entered a ", n, "-digit integer", sep="")
```

Second approach

```
a = int(input("Enter an integer between -9999 and 9999: "))

#If variable a is negative, make it positive
if a < 0:
    a = (-1) * a

if 1000 <= a <= 9999:
    n = 4
elif 100 <= a <= 999:
    n = 3
elif 10 <= a <= 99:
    n = 2
else:
    n = 1

print("You entered a ", n, "-digit integer", sep="")
```

5. Solution

First approach

```
a = int(input("Enter an integer between -9999 and 9999: "))

if -9999 <= a <= -1000 or 1000 <= a <= 9999:
    print("You entered a 4-digit integer")
elif -999 <= a <= -100 or 100 <= a <= 999:
    print("You entered a 3-digit integer")
elif -99 <= a <= -10 or 10 <= a <= 99:
    print("You entered a 2-digit integer")
elif -9 <= a <= 9:
    print("You entered a 1-digit integer")
else:
    print("Error: Invalid value!")
```

Second approach

```
a = int(input("Enter an integer between -9999 and 9999: "))

#If variable a is negative, make it positive
if a < 0:
    a = (-1) * a

if 1000 <= a <= 9999:
    print("You entered a 4-digit integer")
elif 100 <= a <= 999:
    print("You entered a 3-digit integer")
elif 10 <= a <= 99:
    print("You entered a 2-digit integer")
elif 0 <= a <= 9:
    print("You entered a 1-digit integer")
else:
    print("Error: Invalid value!")
```

6. Solution

```
m = int(input("Enter the number of a month between 1 and 12: "))

if m <= 2 or m == 12:
    print("Winter")
elif m <= 5:
    print("Spring")
elif m <= 8:
    print("Summer")
else:
    print("Fall (Autumn)")
```

7. Solution

```
m = int(input("Enter the number of a month between 1 and 12: "))

if m < 1 or m > 12:
    print("Error: Invalid value!")
elif m <= 2 or m == 12:
    print("Winter")
elif m <= 5:
    print("Spring")
elif m <= 8:
    print("Summer")
else:
    print("Fall (Autumn)")
```

8. Solution

```
name = input("Enter the name of a month: ")

name = name.upper()
```

```
if name == "JANUARY":
    print(1)
elif name == "FEBRUARY":
    print(2)
elif name == "MARCH":
    print(3)
elif name == "APRIL":
    print(4)
elif name == "MAY":
    print(5)
elif name == "JUNE":
    print(6)
elif name == "JULY":
    print(7)
elif name == "AUGUST":
    print(8)
elif name == "SEPTEMBER":
    print(9)
elif name == "OCTOBER":
    print(10)
elif name == "NOVEMBER":
    print(11)
elif name == "DECEMBER":
    print(12)
else:
    print("Error")
```

9. Solution

```
letter = input("Enter a letter between A and F: ")

if letter == "A":
    print("90 - 100")
elif letter == "B":
    print("80 - 89")
elif letter == "C":
    print("70 - 79")
elif letter == "D":
    print("60 - 69")
else:
    print("0 - 59")
```

10.Solution

```
roman = input("Enter a Roman numeral between I and X: ")

if roman == "I":
    print(1)
```

```
elif roman == "II":
    print(2)
elif roman == "III":
    print(3)
elif roman == "IV":
    print(4)
elif roman == "V":
    print(5)
elif roman == "VI":
    print(6)
elif roman == "VII":
    print(7)
elif roman == "VIII":
    print(8)
elif roman == "IX":
    print(9)
elif roman == "X":
    print(10)
else:
    print("Error")
```

11.Solution

```
total = int(input("Enter the total number of CDs purchased in a month: "))

if total == 1:
    print("You are awarded 3 points")
elif total == 2:
    print("You are awarded 10 points")
elif total == 3:
    print("You are awarded 20 points")
else:
    print("You are awarded 45 points")
```

12.Solution

```
num = input("Enter a number (0 - 3) in words:")

if num == "zero":
    print(0)
elif num == "one":
    print(1)
elif num == "two":
    print(2)
elif num == "three":
    print(3)
else:
    print("I don't know this number!")
```

13.Solution

```
b = int(input("Enter Beaufort number: "))

if b == 0:
    print("Calm")
elif b == 1:
    print("Light Air")
elif b == 2:
    print("Light breeze")
elif b == 3:
    print("Gentle breeze")
elif b == 4:
    print("Moderate breeze")
elif b == 5:
    print("Fresh breeze")
elif b == 6:
    print("Strong breeze")
elif b == 7:
    print("Moderate gale")
elif b == 8:
    print("Gale")
elif b == 9:
    print("Strong gale")
elif b == 10:
    print("Storm")
elif b == 11:
    print("Violent storm")
elif b == 12:
    print("Hurricane force")
else:
    print("Invalid Beaufort number!")
```

14.Solution

```
wind = float(input("Enter wind speed (in miles/hour): "))

if wind < 0:
    print("Entered value is negative")
elif wind < 1:
    print("Beaufort: 0\nCalm")
elif wind < 4:
    print("Beaufort: 1\nLight air")
elif wind < 8:
    print("Beaufort: 2\nLight breeze")
elif wind < 13:
    print("Beaufort: 3\nGentle breeze")
elif wind < 18:
    print("Beaufort: 4\nModerate breeze")
elif wind < 25:
    print("Beaufort: 5\nFresh breeze")
elif wind < 31:
    print("Beaufort: 6\nStrong breeze")
elif wind < 39:
    print("Beaufort: 7\nModerate gale")
elif wind < 47:
    print("Beaufort: 8\nGale")
elif wind < 55:
    print("Beaufort: 9\nStrong gale")
elif wind < 64:
    print("Beaufort: 10\nStorm")
elif wind < 74:
    print("Beaufort: 11\nViolent storm")
else:
    print("Beaufort: 12\nHurricane force")
```

15.Solution

```
print("1. Convert Kelvin to Fahrenheit")
print("2. Convert Fahrenheit to Kelvin")
print("3. Convert Fahrenheit to Celsius")
print("4. Convert Celsius to Fahrenheit")

choice = int(input("Enter a choice: "))

if choice == 1:
    kelvin = float(input("Enter a temperature in degrees Kelvin: "))
    fahrenheit = 1.8 * kelvin - 459.67
    print(kelvin, "degrees Kelvin =", fahrenheit, "degrees Fahrenheit")
elif choice == 2:
    fahrenheit = float(input("Enter a temperature in degrees Fahrenheit: "))
    kelvin = (fahrenheit + 459.67) / 1.8
    print(fahrenheit, "degrees Fahrenheit =", kelvin, "degrees Kelvin")
elif choice == 3:
    fahrenheit = float(input("Enter a temperature in degrees Fahrenheit: "))
    celsius = 5 / 9 * (fahrenheit - 32)
    print(fahrenheit, "degrees Fahrenheit =", celsius, "degrees Celsius")
elif choice == 4:
    celsius = float(input("Enter a temperature in degrees Celsius: "))
    fahrenheit = 9 / 5 * celsius + 32
    print(celsius, "degrees Celsius =", fahrenheit, "degrees Fahrenheit")
else:
    print("Invalid choice!")
```

Chapter 16

16.2 Review Questions: True/False

1. True   2. True   3. False

16.3 Review Exercises

1. Solution

i. 25 6   ii. 10 9   iii. 50 2

2. Solution

First approach

```
age = int(input("Enter your age: "))

if age < 0:
    print("Error: Invalid age!")
else:
    if age < 16:
        print("You cannot drive either a small scooter or a car")
    else:
        if age < 18:
            print("You can drive a small scooter")
        else:
            print("You can drive a car and a small scooter")
```

Second approach

```
age = int(input("Enter your age: "))

if age < 0:
    print("Error: Invalid age!")
else:
    if age < 16:
        print("You cannot drive either a small scooter or a car")
    elif age < 18:
        print("You can drive a small scooter")
    else:
        print("You can drive a car and a small scooter")
```

Third approach

```
age = int(input("Enter your age: "))

if age < 0:
    print("Error: Invalid age!")
elif age < 16:
    print("You cannot drive either a small scooter or a car")
elif age < 18:
    print("You can drive a small scooter")
```

```
else:
    print("You can drive a car and a small scooter")
```

3. Solution

First approach

```
t = float(input("Enter temperature (in Fahrenheit): "))
w = float(input("Enter wind speed (in miles/hour): "))

if t > 75:
    if w > 12:
        print("The day is hot and windy")
    else:
        print("The day is hot and not windy")
else:
    if w > 12:
        print("The day is cold and windy")
    else:
        print("The day is cold and not windy")
```

Second approach

```
t = float(input("Enter temperature (in Fahrenheit): "))
w = float(input("Enter wind speed (in miles/hour): "))

if t > 75:
    message1 = "hot"
else:
    message1 = "cold"

if w > 12:
    message2 = "windy"
else:
    message2 = "not windy"

print("The day is", message1, "and", message2)
```

4. Solution

```
a = int(input("Enter age: "))
if a < 18:
    print("Invalid age")
else:
    w = int(input("Enter weight in pounds: "))
    h = int(input("Enter height in inches: "))

    bmi = w * 703 / h ** 2

    if bmi < 15:
        print("Very severely underweight")
    elif bmi < 16:
        print("Severely underweight")
    elif bmi < 18.5:
        print("Underweight")
    elif bmi < 25:
        print("Normal")
    elif bmi < 30:
        print("Overweight")
    elif bmi < 35:
        print("Severely overweight")
    else:
        print("Very severely overweight")
```

Chapter 17

17.3 Review Questions: True/False

1. True
2. True
3. False
4. True

Chapter 18

18.2 Review Questions: True/False

1. True
2. False
3. False
4. False
5. False
6. False
7. False

18.3 Review Questions: Multiple Choice

1. b
2. b
3. c
4. b
5. c
6. b

18.4 Review Exercises

1. Solution

```
i = 30.0
while i > 5:
   print(i)
   i /= 2
print("The end")
```

2. Solution

Four

3. Solution

Zero

4. Solution

It displays

2

14

6

and performs three iterations

5. Solution

i. −1

ii. 9

iii. 0.5

iv. −7

v. A value between 17 and 32

vi. 1.4

6. Solution

```
total = 0

i = 1
while i <= 20:
    x = float(input("Enter a number: "))
    total += x
    i += 1
print(total / 20)
```

7. Solution

```
n = int(input("Enter N: "))

p = 1
i = 1
while i <= n:
    x = float(input("Enter a number: "))
    if x > 0:
        p *= x
    i += 1
print(p)
```

8. Solution

```
total = 0
i = 1
while i <= 10:
    x = int(input("Enter an integer: "))
    if x >= 100 and x <= 200:
        total += x
    i += 1
```

```
print(total)
```

9. Solution

```
total = 0
i = 1
while i <= 20:
    x = int(input("Enter an integer: "))
    if x >= 100 and x <= 999:
        total += x
    i += 1

print(total)
```

10.Solution

```
p = 1
x = float(input("Enter a number: "))
while x != 0:
    p *= x
    x = float(input("Enter a number: "))

print(p)
```

Chapter 19

19.2 Review Questions: True/False

1. True
2. True
3. False
4. False
5. True
6. True
7. False
8. False
9. True

19.3 Review Questions: Multiple Choice

1. d
2. c
3. b
4. a
5. b
6. d
7. c
8. c
9. a

19.4 Review Exercises

1. Solution

It displays

12 3

and performs five iterations

2. Solution

It displays

```
10  4
19  20
28  32
```

3. Solution

i. 9

ii. A value between 17 and 20

iii. −7 (or −6)

iv. −1

4. Solution

```python
p = 1
total = 0
for i in range(20):
    x = float(input("Enter a number: "))
    p *= x
    total += x

print(p, total / 20)
```

5. Solution

```python
n = int(input("Enter N: "))

count = 0

for i in range(n):
    x = int(input("Enter an integer: "))
    if x > 0:
        count += 1

if count > 0:
    print(count)
else:
    print("You entered no positive integers")
```

6. Solution

```python
start = int(input("Enter value for start: "))
finish = int(input("Enter value for finish: "))

for i in range(start, finish + 1):
    print(i)
```

7. Solution

```python
b = float(input("Enter a value for base: "))
exp = int(input("Enter an integer for exponent: "))

p = 1
```

```
for i in range(exp):
    p *= b

print(p)
```

8. Solution

First approach

```
msg = input("Enter a message: ")

count = 0
for character in msg:
    if character == " ":
        count += 1

words = count + 1

print("The message entered contains", words, "words")
```

Second approach

```
msg = input("Enter a message: ")

count = 0
for i in range(len(msg)):
    if msg[i] == " ":
        count += 1

words = count + 1

print("The message entered contains", words, "words")
```

Chapter 20

20.2 Review Questions: True/False

1. True
2. False
3. True
4. False
5. True
6. True

20.3 Review Questions: Multiple Choice

1. b
2. c
3. d
4. a
5. b

20.4 Review Exercises

1. Solution

i. 10

ii. A value between 46 and 50

iii. −7 (or −8)

iv. 138 (or 139)

2. Solution

```
for hour in range(24):
    for minutes in range(60):
        print(hour, "\t", minutes)
```

3. Solution

```
for i in range(5, 0, -1):
    for j in range(i):
        print(i, end = "")
    print()
```

4. Solution

```
for i in range(6):
    for j in range(i + 1):
        print(j, end = "")
    print()
```

5. Solution

First approach (The amateur way!!!)

```
print("*  *  *  *  *  *  *  *  *  *")
print("*  *  *  *  *  *  *  *  *  *")
print("*  *  *  *  *  *  *  *  *  *")
print("*  *  *  *  *  *  *  *  *  *")
```

Second approach

```
for i in range(4):
    for j in range(10):
        print("* ", end = "")
    print()
```

Third approach

```
print("*  " * 10)
print("*  " * 10)
print("*  " * 10)
print("*  " * 10)
```

Fourth approach

```
print((("*  " * 10) + "\n") * 4)
```

6. Solution

First approach

```
y = int(input("Enter an integer between 3 and 20: "))

for i in range(y):
    for j in range(y):
        print("* ", end = "")
    print()
```

Second approach

```
y = int(input("Enter an integer between 3 and 20: "))

for i in range(y):
    print("*  " * y)
```

Third approach

```
y = int(input("Enter an integer between 3 and 20: "))

print((("*  " * y) + "\n") * y)
```

7. Solution

```
for i in range(1, 6):
    for j in range(i):
        print("* ", end = "")
    print()
```

8. Solution

```
for i in range(1, 6):
    for j in range(i):
        print("* ", end = "")
    print()

for i in range(4, 0, -1):
    for j in range(i):
        print("* ", end = "")
    print()
```

Chapter 21

21.7 Review Questions: True/False

1. False
2. False
3. False
4. True
5. False
6. False

21.8 Review Questions: Multiple Choice

1. b
2. c
3. b
4. a
5. d

21.9 Review Exercises

1. Solution

```
count_not_johns = 0
count_names = 0
~~name = ""~~
name = input("Enter a name: ")
while name != "STOP":
    ~~name = input("Enter a name: ")~~
    count_names += 1
    if name != "John":
        count_not_johns += 1
    name = input("Enter a name: ")

print("Names other than John entered", count_not_johns, "times")
print(count_names, "names entered")
```

2. Solution

```
text = input("Enter a text: ")

found = False
for character in text:
    if character == " ":
        found = True
        break

if found == False:
    print("One Single Word")
else:
    print("Complete Sentence")
```

3. Solution

First approach

```
sentence = input("Enter a sentence: ")

found = False
for character in sentence:
    if character in "0123456789":
        found = True
        break

if found == True:
    print("The sentence contains a number")
```

Second approach

```
sentence = input("Enter a sentence: ")

found = False
for digit in "0123456789":
    if digit in sentence:
        found = True
        break

if found == True:
    print("The sentence contains a number")
```

4. Solution

```
print("Printing all integers from 1 to 100")
i = 1
while i < 101:
    print(i)
    i += 1
```

5. Solution

```
print("Printing odd integers from 1 to 99")
i = 1
while not(i > 100):
    print(i)
    i += 2
```

6. Solution

```
for i in range(1, 5):
    for j in range(1, 5):
        print(i, "x", j, "=", i * j)
```

7. Solution

```
print("\t|\t", end = "")
for i in range(1, 13):
    print(i, "\t", end = "")
print()

for i in range(1, 13):
    print("---------", end = "")
print()

for i in range(1, 13):
    print(i, "\t|\t", end = "")
    for j in range(1, 13):
        print(i * j, end = "\t")
    print()
```

8. Solution

```
n = int(input("Enter an integer: "))

print("\t|\t", end = "")
for i in range(1, n + 1):
    print(i, "\t", end = "")
print()

for i in range(n + 1):
    print("--------", end = "")
print()

for i in range(1, n + 1):
    print(i, "\t|\t", end = "")
    for j in range(1, n + 1):
        print(i * j, end = "\t")
    print()
```

Chapter 22

22.2 Review Exercises

1. Solution

```
total = 0
for i in range(1, 101, 2):
    total += i

print(total)
```

2. Solution

```
n = int(input("Enter N: "))

total = 0
for i in range(2, 2 * n + 1, 2):
    total += i

print(total)
```

3. Solution

```
count_pos = 0
count_neg = 0
total_pos = 0
total_neg = 0

for i in range(50):
    x = int(input("Enter an integer: "))
    if x > 0:
        count_pos += 1
        total_pos += x
    elif x < 0:
        count_neg += 1
        total_neg += x

if count_pos > 0:
    print(total_pos / count_pos)

if count_neg > 0:
    print(total_neg / count_neg)
```

4. Solution

```
n = int(input("Enter total number of students: "))

total = 0
count = 0
for i in range(n):
    grade = int(input("Enter grade: "))
    if grade >= 90 and grade <= 100:
        total += grade
        count += 1

if count > 0:
    print(total / count)
else:
    print("There are no students that got an A")
```

5. Solution

```
total = 0
count = 0
while total <= 3000:
    x = float(input("Enter a number: "))
    if x == 0:
        count += 1

    total += x

print(count)
```

6. Solution

```
answer = "YES"
while answer.upper() == "YES":
    r = float(input("Enter the length of a radius of a circle: "))

    area = 3.141 * r ** 2
    print("The area is:", area)

    answer = input("Would you like to repeat? ")
```

7. Solution

```
x = 1
while x <= 1073741824:
    print(x)
    x *= 2
```

8. Solution

```
for i in range(1, 101):
    print(-i, "\n", i)
```

9. Solution

First approach

```python
value = 0
for i in range(8):
    offset = 10 ** i
    value += offset
    print(value)
```

Second approach

```python
value = "1"
for i in range(8):
    print(value)
    value += "1"
```

10.Solution

```python
t = float(input("Enter temperature for day 1: "))
maximum = t
total = t
for i in range(2, 32):
    t = float(input("Enter temperature for day " + str(i) + ": "))

    total += t
    if t > maximum:
        maximum = t

print(total / 31, maximum)
```

11.Solution

```python
level = float(input("Enter level: "))
hour = int(input("Enter hour: "))

maximum = level
minimum = level
max_hour = hour
min_hour = hour

for i in range(23):
    level = float(input("Enter level: "))
    hour = int(input("Enter hour: "))

    if level > maximum:
        maximum = level
        max_hour = hour

    if level < minimum:
        minimum = level
        min_hour = hour
```

```
print(maximum, max_hour, minimum, min_hour)
```

12.Solution

```
import random

for i in range(2):
    secret_number = random.randrange(1, 101)

    attempts = 1

    guess = int(input("Enter a guess: "))
    while guess != secret_number:
        if guess > secret_number:
            print("Your guess is bigger than my secret number. Try again.")
        else:
            print("Your guess is smaller than my secret number. Try again.")

        attempts += 1

        guess = int(input("Enter a guess: "))

    print("You found it!")
    print("Attempts:", attempts)

    if i == 0:
        attempts_1st_player = attempts
    else:
        attempts_2nd_player = attempts

if attempts_1st_player < attempts_2nd_player:
    print("First Player Wins")
elif attempts_2nd_player < attempts_1st_player:
    print("Second Player Wins")
else:
    print("It's a draw")
```

13.Solution

```
n = int(input("Enter total number of students: "))

total = 0
total_a = 0
count_a = 0
total_b = 0
count_b = 0
total_a_boys = 0
count_a_boys = 0
count_cdef_girls = 0
```

```
for i in range(n):
    grade = int(input("Enter grade for student No" + str(i + 1) + ": "))

    gender = input("Enter gender for student No" + str(i + 1) + " (M/F): ")

    if 90 <= grade <= 100:
        total_a += grade
        count_a += 1
        if gender == "M":
            total_a_boys += grade
            count_a_boys += 1
    elif 80 <= grade <= 89:
        total_b += grade
        count_b += 1
    else:
        if gender == "F":
            count_cdef_girls += 1

    total += grade

if count_a > 0:
    print("Average value of those who got an 'A': ")
    print(total_a / count_a)

if count_b > 0:
    print("Average value of those who got a 'B': ")
    print(total_b / count_b)

if count_a_boys > 0:
    print("Average value of boys who got an 'A': ")
    print(total_a_boys / count_a_boys)

print("Total number of girls that got less than 'B':", count_cdef_girls)
print("Average grade of the whole class:", total / n)
```

14.Solution

```
answer = "yes"
while answer.upper() == "YES":
    amount = float(input("Enter amount: "))

    if amount < 20:
        discount = 0
    elif amount < 50:
        discount = 3
    elif amount < 100:
        discount = 5
    else:
        discount = 10

    print("Discount:", discount, "%", sep="")

    answer = input("Would you like to repeat? ")
```

Chapter 23

23.12 Review Exercises

1. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.forward(200)
george.left(130)
george.forward(50)

george.penup()
george.goto(200, 0)
george.pendown()

george.setheading(230)
george.forward(50)

wn.exitonclick()
```

2. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.forward(200)
george.left(90 - 20)
george.forward(100)
george.left(90 + 20)
george.forward(200)
george.left(90 - 20)
george.forward(100)

wn.exitonclick()
```

3. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.left(70)
george.forward(100)
george.left(40)
george.forward(100)
george.left(140)
george.forward(100)
george.left(40)
george.forward(100)

wn.exitonclick()
```

4. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.penup()
george.goto(-200, 0)
george.pendown()

george.forward(300)
george.left(90 + 45)
george.forward(141)
george.left(45)
george.forward(100)
george.left(45)
george.forward(141)

wn.exitonclick()
```

5. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for k in range(2):
    for i in range(4):
        george.forward(100)
        george.left(90)

    george.penup()
    george.forward(100)
    george.pendown()

wn.exitonclick()
```

6. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for m in range(2):
    for n in range(2):
        for i in range(4):
            george.forward(100)
            george.left(90)

        george.penup()
        george.forward(200)
        george.pendown()

    george.penup()
    george.goto(0, 150)
    george.pendown()

wn.exitonclick()
```

7. Solution

```
import turtle

size = int(input("Enter pen size: "))
length = int(input("Enter length: "))
height = int(input("Enter height: "))

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.pensize(size)

george.forward(length)
george.left(90)
george.forward(height)
george.left(90)
george.forward(length)
george.left(90)
george.forward(height)

wn.exitonclick()
```

8. Solution

```
import turtle

length = int(input("Enter length of the side: "))

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

george.forward(length)
george.left(120)
george.forward(length)
george.left(120)
george.forward(length)
george.left(120)
george.forward(length)

wn.exitonclick()
```

9. Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for i in range(12):
    george.forward(100)
    george.penup()
    george.backward(100)
    george.left(30)
    george.pendown()

wn.exitonclick()
```

10.Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")
```

```
for k in range(0, 180, 60):
    for i in range(5):
        george.forward(150 + k)
        george.right(180 / 5 * 4)

    george.penup()
    george.backward(30)
    george.left(90)
    george.forward(10)
    george.right(90)
    george.pendown()

wn.exitonclick()
```

11.Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for k in range(3):
    for i in range(4):
        george.forward(100)
        george.left(90)

    george.left(30)

wn.exitonclick()
```

12.Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for k in range(12):
    for i in range(4):
        george.forward(100)
        george.left(90)

    george.left(30)

wn.exitonclick()
```

13.Solution

```
import turtle
```

```
wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")

for k in range(8):
    for i in range(4):
        george.forward(100)
        george.left(90)

    george.left(45)

wn.exitonclick()
```

14.Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")
george.pensize(3)

#Poll position
george.penup()
george.goto(-300, 0)
george.pendown()

#Draw a blue rectangle
george.color("blue")
george.forward(200)
george.left(90)
george.forward(100)
george.left(90)
george.forward(200)
george.left(90)
george.forward(100)

#Move George to the top left corner of the rectangle
george.penup()
george.backward(100)
george.pendown()

#Draw the red roof
george.setheading(45)
george.color("red")
george.forward(141)
george.right(90)
george.forward(141)
```

```
#Draw the windows
george.color("brown")
george.penup()
george.setheading(0)
george.backward(180)
george.right(90)
george.forward(50)
george.left(90)
george.pendown()

for k in range(2):
    for i in range(4):
        george.forward(40)
        george.left(90)

    george.penup()
    george.forward(20)
    george.left(90)
    george.pendown()
    george.forward(40)

    george.penup()
    george.backward(20)
    george.right(90)
    george.backward(20)
    george.pendown()
    george.forward(40)

    george.penup()
    george.forward(80)
    george.right(90)
    george.forward(20)
    george.left(90)
    george.pendown()

#Draw the door
george.penup()
george.backward(180)
george.right(90)
george.forward(50)
george.left(90)
george.pendown()

george.forward(40)
george.left(90)
george.forward(70)
george.left(90)
george.forward(40)
george.left(90)
george.forward(70)

wn.exitonclick()
```

15.Solution

```
import turtle

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")
george.pensize(3)

#Poll position
george.penup()
george.goto(-300, 0)
george.pendown()

for m in range(3):
    #Draw a blue rectangle
    george.color("blue")
    george.forward(200)
    george.left(90)
    george.forward(100)
    george.left(90)
    george.forward(200)
    george.left(90)
    george.forward(100)

    #Move George to the top left corner of the rectangle
    george.penup()
    george.backward(100)
    george.pendown()

    #Draw the red roof
    george.setheading(45)
    george.color("red")
    george.forward(141)
    george.right(90)
    george.forward(141)

    #Draw the windows
    george.color("brown")
    george.penup()
    george.setheading(0)
    george.backward(180)
    george.right(90)
    george.forward(50)
    george.left(90)
    george.pendown()

    for k in range(2):
        for i in range(4):
            george.forward(40)
            george.left(90)

        george.penup()
        george.forward(20)
        george.left(90)
        george.pendown()
        george.forward(40)

        george.penup()
        george.backward(20)
        george.right(90)
        george.backward(20)
        george.pendown()
        george.forward(40)

        george.penup()
        george.forward(80)
        george.right(90)
        george.forward(20)
        george.left(90)
        george.pendown()

    #Draw the door
    george.penup()
    george.backward(180)
    george.right(90)
    george.forward(50)
    george.left(90)
    george.pendown()

    george.forward(40)
    george.left(90)
    george.forward(70)
    george.left(90)
    george.forward(40)
    george.left(90)
    george.forward(70)
```

```
    george.penup()
    george.left(90)
    george.forward(210)
    george.pendown()

wn.exitonclick()
```

16.Solution

```
import turtle, random

wn = turtle.Screen()
george = turtle.Turtle()
george.shape("turtle")
george.pensize(3)

for i in range(10):
    #Pick random x, y values and move the turtle to that position
    x = random.randrange(-200, 200)
    y = random.randrange(-200, 200)
    george.penup()
    george.goto(x, y)
    george.pendown()

    sides = random.randrange(5, 10)    #Pick random number of sides
    length = random.randrange(20, 50)  #Pick random length of sides

    #Draw the polygon
    for j in range(sides):
        george.forward(length)
        george.right(360 / sides)

wn.exitonclick()
```

Chapter 24

24.14 Review Questions: True/False

1. True
2. True
3. True
4. False
5. True
6. False
7. True
8. True
9. False
10. False
11. False
12. True
13. False
14. True
15. False
16. True
17. False
18. True
19. False
20. True
21. False
22. False
23. True
24. False
25. True
26. True
27. False
28. False
29. True
30. True
31. True
32. False

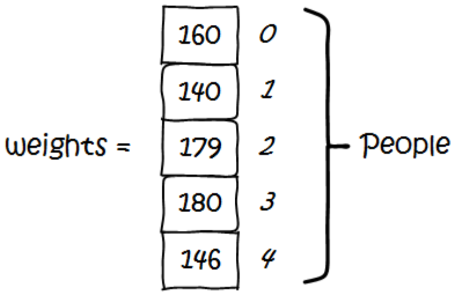
33. False
34. True
35. True
36. False
37. False
38. True
39. False
40. True
41. True
42. True
43. True
44. True
45. True
46. True
47. False

24.15 Review Questions: Multiple Choice

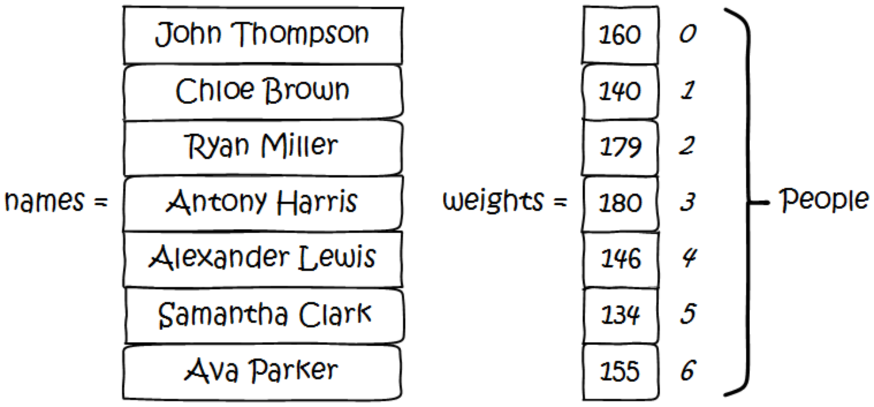
1. b
2. c
3. b
4. d
5. d
6. d
7. c
8. a
9. b
10. a
11. b
12. b
13. a
14. a
15. a
16. c
17. a
18. a
19. b
20. a

24.16 Review Exercises

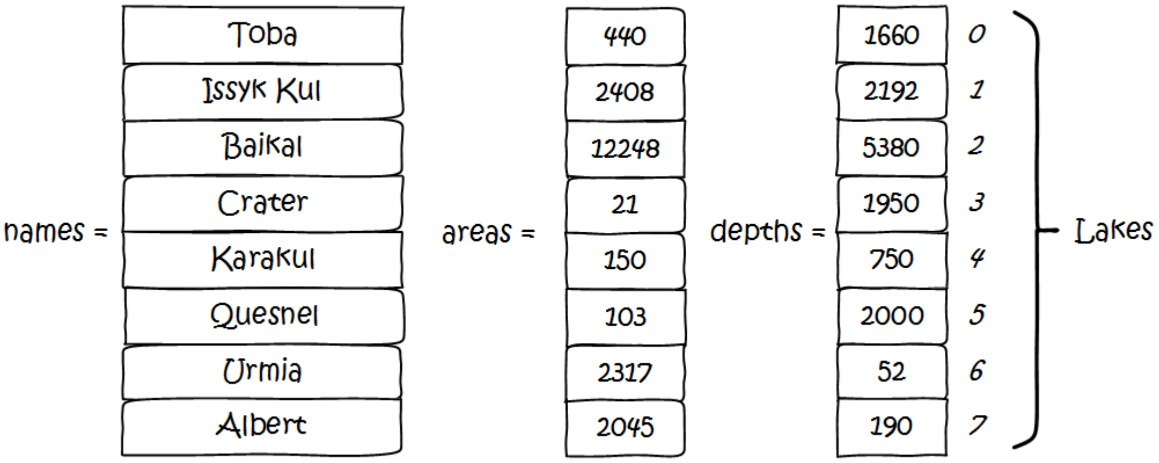
1. Solution



2. Solution

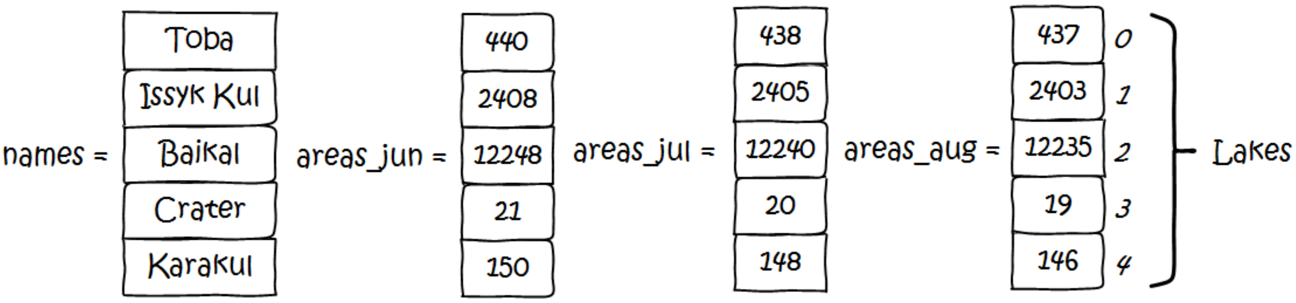


3. Solution



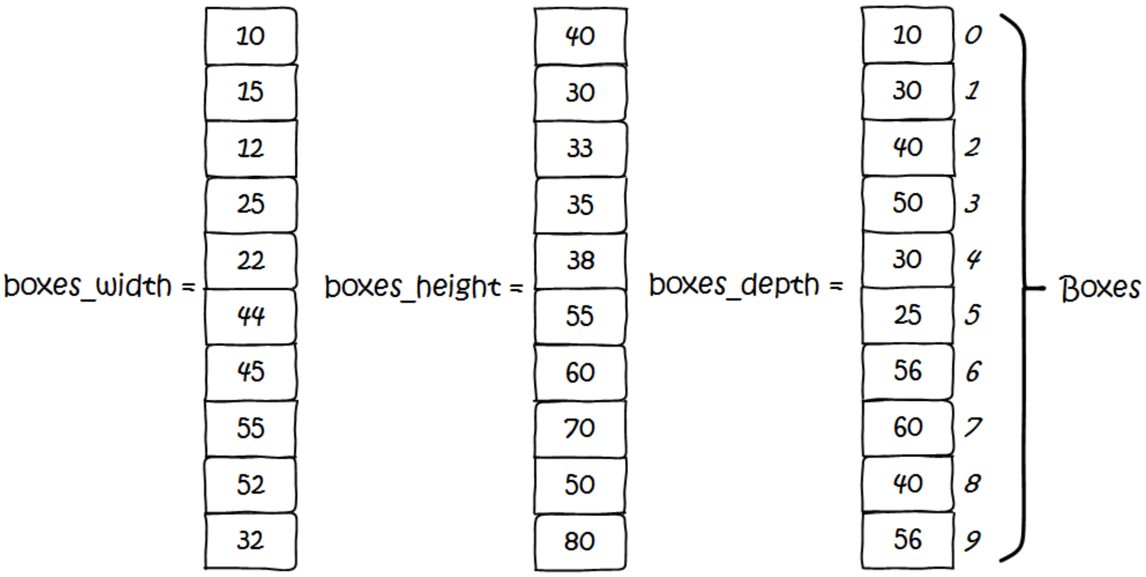
names =	areas =	depths =	
Toba	440	1660	0
Issyk Kul	2408	2192	1
Baikal	12248	5380	2
Crater	21	1950	3
Karakul	150	750	4
Quesnel	103	2000	5
Urmia	2317	52	6
Albert	2045	190	7

4. Solution



names =	areas_jun =	areas_jul =	areas_aug =	
Toba	440	438	437	0
Issyk Kul	2408	2405	2403	1
Baikal	12248	12240	12235	2
Crater	21	20	19	3
Karakul	150	148	146	4

5. Solution



boxes_width =	boxes_height =	boxes_depth =	
10	40	10	0
15	30	30	1
12	33	40	2
25	35	50	3
22	38	30	4
44	55	25	5
45	60	56	6
55	70	60	7
52	50	40	8
32	80	56	9

6. Solution

[16, 4, 1]

7. Solution

[4, 5, 11, 20, 10]

8. Solution

[18, 11, 46, 11, 11, 50]

9. Solution

[10, 22, 45, 67, 86, 19]

10.Solution

Navajo

Cherokee

Sioux

11.Solution

```
ELEMENTS = 100

a = [None] * ELEMENTS
for i in range(ELEMENTS):
    a[i] = float(input("Enter a number: "))

for i in range(ELEMENTS):
    print(a[i] ** 3)
```

12.Solution

```
ELEMENTS = 80

a = [None] * ELEMENTS
for i in range(ELEMENTS):
    a[i] = float(input("Enter a number: "))

for i in range(ELEMENTS):
    a[i] **= 2

for i in range(ELEMENTS - 1, -1, -1):
    print(a[i])
```

13.Solution

```
ELEMENTS = 50

a = []
for i in range(ELEMENTS):
    a.append(int(input("Enter an integer: ")))

for element in a:
    if element >= 10:
        print(element)
```

14.Solution

```
ELEMENTS = 30

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))
```

```
total = 0
for element in a:
    if element > 0:
        total += element

print(total)
```

15. Solution

```
ELEMENTS = 50

a = []
for i in range(ELEMENTS):
    a.append(int(input("Enter an integer: ")))

total = 0
for element in a:
    if element >= 10 and element <= 99:
        total += element

print(total)
```

16. Solution

```
ELEMENTS = 40

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

total_pos = 0
total_neg = 0
for element in a:
    if element > 0:
        total_pos += element
    elif element < 0:
        total_neg += element

print(total_pos, total_neg)
```

17. Solution

```
ELEMENTS = 20

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

total = 0
for element in a:
    total += element
```

```
print(total / ELEMENTS)
```

18.Solution

```
ELEMENTS = 50

a = []
for i in range(ELEMENTS):
    a.append(int(input("Enter an integer: ")))

for i in range(ELEMENTS):
    if a[i] < 20:
        print(i)
```

19.Solution

```
ELEMENTS = 60

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

for i in range(0, ELEMENTS, 2):
    print(a[i])
```

20.Solution

```
ELEMENTS = 20

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

total = 0
for i in range(0, ELEMENTS, 2):
    total += a[i]

print(total)
```

21.Solution

First approach

```
ELEMENTS = 100

a = [None] * ELEMENTS
for i in range(ELEMENTS):
    a[i] = i + 1
```

Second approach

```
ELEMENTS = 100

a = []
for i in range(ELEMENTS):
    a.append(i + 1)
```

22. Solution

```
a = []
for i in range(2, 202, 2):
    a.append(i)
```

23. Solution

```
n = int(input("Enter N: "))

a = []
for i in range(1, n + 1):
    a.append(i ** 2)

print(a)
```

24. Solution

```
ELEMENTS = 10

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

for i in range(ELEMENTS):
    if a[i] == int(a[i]):
        print(i)
```

25. Solution

```
ELEMENTS = 50

a = []
for i in range(ELEMENTS):
    a.append(float(input("Enter a number: ")))

count = 0
for i in range(ELEMENTS):
    if a[i] < 0:
        count += 1

print(count)
```

26. Solution

```
ELEMENTS = 50

words = []
for i in range(ELEMENTS):
    words.append(input("Enter a word: "))

for word in words:
    if len(word) >= 10:
        print(word)
```

27.Solution

```
ELEMENTS = 30

words = []
for i in range(ELEMENTS):
    words.append(input("Enter a word: "))

length_limits = (5, 10, 20)

for length_limit in length_limits:
    for word in words:
        if len(word) < length_limit:
            print(word)
```

28.Solution

```
ELEMENTS = 40

words = [None] * ELEMENTS
for i in range(ELEMENTS):
    words[i] = input("Enter a word: ")

for word in words:
    count = 0
    for letter in word:
        if letter == "w":
            count += 1
        if count == 2:
            print(word)
            break
```

Chapter 25

25.5 Review Questions: True/False

1. False
2. False
3. True
4. True

25.6 Review Exercises

1. Solution

```
ELEMENTS_OF_A = 50
ELEMENTS_OF_NEW = ELEMENTS_OF_A - 2

a = [None] * ELEMENTS_OF_A
for i in range(ELEMENTS_OF_A):
    a[i] = float(input("Enter a number: "))

new_arr = [None] * ELEMENTS_OF_NEW
for i in range(ELEMENTS_OF_NEW):
    new_arr[i] = (a[i] + a[i + 1] + a[i + 2]) / 3

print(new_arr)
```

2. Solution

First approach

```
ELEMENTS = 15

a = [None] * ELEMENTS
for i in range(ELEMENTS):
    a[i] = float(input("A - Enter a number: "))

b = [None] * ELEMENTS
for i in range(ELEMENTS):
    b[i] = float(input("B - Enter a number: "))

c = [None] * ELEMENTS
for i in range(ELEMENTS):
    c[i] = float(input("C - Enter a number: "))

new_arr = [None] * ELEMENTS
for i in range(ELEMENTS):
    minimum = a[i]
    if b[i] < minimum:
        minimum = b[i]
    if c[i] < minimum:
        minimum = c[i]
    new_arr[i] = minimum

print(new_arr)
```

Second approach

```
ELEMENTS = 15

a = []
for i in range(ELEMENTS):
    a.append(float(input("A - Enter a number: ")))

b = []
for i in range(ELEMENTS):
    b.append(float(input("B - Enter a number: ")))

c = []
for i in range(ELEMENTS):
    c.append((input("C - Enter a number: ")))
```

```
new_arr = []
for i in range(ELEMENTS):
    new_arr.append(min(a[i], b[i], c[i]))

print(new_arr)
```

3. Solution

```
MOUNTAINS = 30

names = [None] * MOUNTAINS
heights = [None] * MOUNTAINS
countries = [None] * MOUNTAINS
for i in range(MOUNTAINS):
    names[i] = input()
    heights[i] = float(input())
    countries[i] = input()

maximum = heights[0]
index_of_max = 0
minimum = heights[0]
index_of_min = 0
for i in range(1, MOUNTAINS):
    if heights[i] > maximum:
        maximum = heights[i]
        index_of_max = i
    if heights[i] < minimum:
        minimum = heights[i]
        index_of_min = i

print(heights[index_of_max], names[index_of_max], countries[index_of_max])
print(heights[index_of_min], names[index_of_min], countries[index_of_min])
```

4. Solution

```
CLASS1 = 20
CLASS2 = 25

print("Class A")
names1 = []
for i in range(CLASS1):
    names1.append(input("Enter name: "))

print("Class B")
names2 = []
for i in range(CLASS2):
    names2.append(input("Enter name: "))

needle = input("Enter a name to search: ")
```

```
found = False
for name in names1:
    if name == needle:
        found = True
        break

if found == True:
    print("Student found in class No 1")
else:
    found = False
    for name in names2:
        if name == needle:
            found = True
            break

    if found == True:
        print("Student found in class No 2")
    else:
        print("Student not found in either class")
```

5. *Solution*

```
usr = input("Enter username: ")
pwd = input("Enter password: ")

found = False
for i in range(100):
    if usernames[i] == usr:
        found = True
        break

if found == True:
    if usernames[i] == usr and passwords[i] == pwd:
        print("Login OK!")
    else:
        print("Login Failed!")
else:
    print("Login Failed!")
```

6. *Solution*

```
needle = input("Enter a value to search: ")

found = False
for i in range(1000):
    if SSNs[i] == needle:
        found = True
        print(SSNs[i], names[i])
        break

if found == False:
    for i in range(1000):
        if names[i] == needle:
            found = True
            print(SSNs[i], names[i])

if found == False:
    print("This value does not exist")
```

7. Solution

```
STUDENTS = 12

grades1 = []
grades2 = []
grades3 = []
for i in range(STUDENTS):
    grades1.append(int(input()))
    grades2.append(int(input()))
    grades3.append(int(input()))

found = False
for i in range(STUDENTS):
    if (grades1[i] + grades2[i] + grades3[i]) / 3 < 70:
        found = True
        break

if found == True:
    print("There is at least one student that has an average value below 70")
```

8. Solution

```
STUDENTS = 15

grades1 = []
grades2 = []
for i in range(STUDENTS):
    grades1.append(int(input()))
    grades2.append(int(input()))

for i in range(STUDENTS):
    print("Student No", (i + 1), ": ")

    average = (grades1[i] + grades2[i]) / 2

    if average < 60:
        print("E/F")
    elif average < 70:
        print("D")
    elif average < 80:
        print("C")
    elif average < 90:
        print("B")
    else:
        print("A")
```

9. Solution

```
PLAYERS = 15

points_match1 = []
points_match2 = []
points_match3 = []
points_match4 = []
for i in range(PLAYERS):
    points_match1.append(int(input()))
    points_match2.append(int(input()))
    points_match3.append(int(input()))
    points_match4.append(int(input()))

for i in range(PLAYERS):
    print("Player No", i + 1)
    print(points_match1[i] + points_match2[i] + points_match3[i] + points_match4[i])
```

10.Solution

```
HOURS = 24

t_city1 = []
t_city2 = []
t_city3 = []
for i in range(HOURS):
    t_city1.append(float(input()))
    t_city2.append(float(input()))
    t_city3.append(float(input()))

for i in range(HOURS):
    average = (t_city1[i] + t_city2[i] + t_city3[i]) / 3
    if average < 10:
        print("Hour:", (i + 1))
```

11.Solution

```
STUDENTS = 12

names = []
grd_lesson1 = []
grd_lesson2 = []
for i in range(STUDENTS):
    names.append(input())
    grd_lesson1.append(int(input()))
    grd_lesson2.append(int(input()))

#Create list average
average = []
for i in range(STUDENTS):
    average.append((grd_lesson1[i] + grd_lesson2[i]) / 2)

for i in range(STUDENTS):
    print(names[i], average[i])

for i in range(STUDENTS):
    if average[i] < 60:
        print(names[i])

for i in range(STUDENTS):
    if average[i] > 89:
        print(names[i], "Bravo!")
```

12.Solution

```
ARTISTS = 15

artist_names = []
song_titles = []
scoreA = []
scoreB = []
scoreC = []

for i in range(ARTISTS):
    artist_names.append(input("Name for artist No." + str(i + 1) + ": "))
    song_titles.append(input("Song title for artist: " + artist_names[i]))
    print("Score for artist:", artist_names[i])
    scoreA.append(int(input(" gotten from judge A: ")))
    scoreB.append(int(input(" gotten from judge B: ")))
    scoreC.append(int(input(" gotten from judge C: ")))

total = []
for i in range(ARTISTS):
    minimum = min(scoreA[i], scoreB[i], scoreC[i])
    total.append(scoreA[i] + scoreB[i] + scoreC[i] - minimum)

for i in range(ARTISTS):
    print(artist_names[i], song_titles[i], total[i])
```

13.Solution

```
CITIZENS = 20
```

```
answers1 = []
answers2 = []

prod_name1 = input("Enter Product Name 1: ")
for i in range(CITIZENS):
    answers1.append(input("Enter score for product " + prod_name1 + ": "))

prod_name2 = input("Enter Product Name 2:")
for i in range(CITIZENS):
    answers2.append(input("Enter score for product " + prod_name2 + ": "))

count_A = 0
for i in range(CITIZENS):
    if answers1[i] == "A":
        count_A += 1
print(prod_name1, count_A)

count_A = 0
for i in range(CITIZENS):
    if answers2[i] == "A":
        count_A += 1
print(prod_name2, count_A)
```

14.Solution

```
morseAlphabet = {
    "A": ".-",
    "B": "-...",
    "C": "-.-.",
    "D": "-..",
    "E": ".",
    "F": "..-.",
    "G": "--.",
    "H": "....",
    "I": "..",
    "J": ".---",
    "K": "-.-",
    "L": ".-..",
    "M": "--",
    "N": "-.",
    "O": "---",
    "P": ".--.",
    "Q": "--.-",
    "R": ".-.",
    "S": "...",
    "T": "-",
    "U": "..-",
    "V": "...-",
    "W": ".--",
    "X": "-..-",
    "Y": "-.--",
    "Z": "--..",
    " ": "/"
}

msg = input("Enter an English message: ")

for character in msg:
    print(morseAlphabet[character.upper()], end = " ")
```

15.Solution

```
import random

words = ["compiler", "interpreter", "error", "variable",
         "operator", "computer", "programmer", "algorithm"]

#Randomly choose a word
random_index = random.randrange(len(words))  #Alternativelly, these can be written as:
word = words[random_index]                   #word = words[random.randrange(len(words))]

wrong_guesses = 0

#Create list results
results = ["_"] * len(word)

while "_" in results and wrong_guesses < 6:
    #Display results
    for x in results:
        print(x, " ", end = "")

    letter = input("Enter a letter: ")

    if letter in word:
        #Replace corresponding underscores with letter in list results
        for i in range(len(word)):
            if letter == word[i]:
                results[i] = letter
    else:
        wrong_guesses += 1
        print("This letter does not exist in clue word!")

if wrong_guesses < 6:
    print("Congratulations, you found it!")
else:
    print("Game over!!!")
```

16.Solution

```
import random

words = ["compiler", "interpreter", "error", "variable",
         "operator", "computer", "programmer", "algorithm"]

wrong_guesses1 = wrong_guesses2 = 0

for player in range(1, 3):
    word = words[random.randrange(len(words))]   #Randomly choose a word

    wrong_guesses = 0

    results = ["_"] * len(word)
    while "_" in results and wrong_guesses < 6:
        #Display results
        for x in results:
            print(x, " ", end = "")

        letter = input("Player No: " + str(player) + " - Enter a letter: ")

        if letter in word:
            #Replace corresponding underscores with letter in list results
            for i in range(len(word)):
                if letter == word[i]:
                    results[i] = letter
        else:
            wrong_guesses += 1
            print("This letter does not exist in clue word!")

    print("Player No:", player, end = " - ")
    if wrong_guesses < 6:
        print("Congratulations, you found it!")
    else:
        print("Game over!!!")

    if player == 1:
        wrong_guesses1 = wrong_guesses
    else:
        wrong_guesses2 = wrong_guesses

#Display the winner
if wrong_guesses1 < wrong_guesses2:
    print("Winner: Player 1")
elif wrong_guesses2 < wrong_guesses1:
    print("Winner: Player 2")
else:
    print("It's a tie!")
```

Chapter 26

26.4 Review Questions: True/False

1. True
2. True
3. False
4. False
5. True
6. True
7. True
8. True
9. True
10. False
11. True

Chapter 27

27.11 Review Questions: True/False

1. True
2. True
3. False
4. True
5. True
6. True
7. False
8. True
9. True
10. True
11. False
12. True
13. True
14. True
15. True
16. True
17. False
18. False
19. False
20. False
21. True
22. True
23. True
24. False

27.12 Review Exercises

1. Solution

```
def find_max(a, b):
    if a > b:
        maximum = a
    else:
        maximum = b
    return maximum
```

2. Solution

It displays:

3 is positive

-7 is negative or zero

-9 is negative or zero

0 is negative or zero

4 is positive

3. Solution

```
def find_sum(a, b, c):
    return a + b + c
```

4. Solution

```
def find_avg(a, b, c, d):
    return (a + b + c + d) / 4
```

5. Solution

```
def maximum(a, b, c):
    m = a
    if b > m:
        m = b
    if c > m:
        m = c

    print(m)
```

6. Solution

```
def find_min(a, b):
    minimum = a
    if b < minimum:
        minimum = b
    return minimum

#Main code starts here
print("Enter four numbers: ")
x1 = float(input())
x2 = float(input())
x3 = float(input())
x4 = float(input())

#Display lowest value as follows (1st approach)
temp1 = find_min(x1, x2)
temp2 = find_min(x3, x4)
print(find_min(temp1, temp2))

#Display lowest value as follows (2nd approach)
print(find_min(find_min(x1, x2), find_min(x3, x4)))
```

7. Solution

```
def get_input():
    answer = input("Enter Yes or No: ")
    if answer.upper() == "YES":
        return True
    else:
        return False

def find_area(b, h):
    return b * h
```

```
#Main code starts here
answer = True
while answer == True:
    b = float(input("Enter the base of the parallelogram: "))
    h = float(input("Enter the height of the parallelogram: "))

    print("Area =", find_area(b, h))

    print("Would you like to repeat? ")
    answer = get_input()
```

Chapter 28

28.2 Review Exercises

1. Solution

```
def Kelvin_to_Fahrenheit(kelvin):
    return 1.8 * kelvin - 459.67

def Kelvin_to_Celsius(kelvin):
    return kelvin - 273.15

#Main code starts here
k = float(input("Enter a temperature in degrees Kelvin: "))
print("Fahrenheit:", Kelvin_to_Fahrenheit(k))
print("Celsius:", Kelvin_to_Celsius(k))
```

2. Solution

```
def num_of_days(month):
    if month in [4, 6, 9, 11]:
        days = 30
    elif month == 2:
        days = 28
    else:
        days = 31

    return days

#Main code starts here
x = int(input("Enter a month: "))
y = int(input("Enter a second month: "))

total = 0
for i in range(x, y + 1):
    total += num_of_days(i)

print(total)
```

3. Solution

```python
import random

def dice():
    return random.randrange(1, 7)

#Main code starts here
names = []
names.append(input("Player 1 enter name: "))
names.append(input("Player 2 enter name: "))

total = [0, 0]

for player in range(2):
    for i in range(10):
        print(names[player], ", hit the Enter key to roll the dice!")
        key = input()

        dice1 = dice()
        dice2 = dice()
        print(dice1, dice2)
        total[player] += dice1 + dice2

if total[0] == total[1]:
    print("Tie!")
elif total[0] > total[1]:
    print(names[0], "wins!")
else:
    print(names[1], "wins!")
```

4. Solution

```python
def bmi(w, h):
    b = w * 703 / h ** 2
    if b < 16:
        print("You must add weight.")
    elif b < 18.5:
        print("You should add some weight.")
    elif b < 25:
        print("Maintain your weight.")
    elif b < 30:
        print("You should lose some weight.")
    else:
        print("You must lose weight.")

#Main code starts here
weight = float(input("Enter your weight (in pounds): "))
age = int(input("Enter your age: "))
```

```
height = float(input("Enter your height (in inches): "))

if age < 18:
    print("I can't calculate your BMI. You must be adult!")
else:
    bmi(weight, height)
```

5. Solution

```
CARS = 40
GAS = 1
DIESEL = 2
HYBRID = 3

def get_choice():
    print("1. Gas")
    print("2. Diesel")
    print("3. Hybrid")
    choice = int(input("Enter type of the car: "))
    return choice

def get_days():
    days = int(input("Enter total number of rental days: "))
    return days

def get_charge(car_type, rental_days):
    if car_type == GAS:
        if rental_days <= 5:
            charge = rental_days * 24
        else:
            charge = rental_days * 22
    elif car_type == DIESEL:
        if rental_days <= 5:
            charge = rental_days * 28
        else:
            charge = rental_days * 25
    else:
        if rental_days <= 5:
            charge = rental_days * 30
        else:
            charge = rental_days * 28

    return charge

#Main code starts here
rented_car_types = [None] * CARS
rented_days = [None] * CARS
```

```python
for i in range(CARS):
    rented_car_types[i] = get_choice()
    rented_days[i] = get_days()

total = 0
for i in range(CARS):
    charge = get_charge(rented_car_types[i], rented_days[i])
    print("Amount to pay, car No", (i + 1), ":", charge)
    total += charge

count = 0
for i in range(CARS):
    if rented_car_types[i] == HYBRID:
        count += 1

print("Hybrids rented:", count)
print("Total profit:", total)
```

Chapter 29

29.9 Review Questions: True/False

1. False
2. True
3. True
4. False
5. False
6. False
7. False
8. True
9. True
10. False
11. True
12. True
13. True
14. False
15. True
16. False
17. False

29.10 Review Exercises

1. Solution

```python
class Trigonometry:
    def square_area(self, side):
        return side * side

    def rectangle_area(self, b, h):
        return b * h

    def triangle_area(self, b, h):
        return b * h / 2

#Main code starts here
tr = Trigonometry()

sqr_side = float(input("Enter square side: "))
```

```
rctngl_base = float(input("Enter rectangle base: "))
rctngl_height = float(input("Enter rectangle height: "))

trngl_base = float(input("Enter triangle base: "))
trngl_height = float(input("Enter triangle height: "))

print(tr.square_area(sqr_side))
print(tr.rectangle_area(rctngl_base, rctngl_height))
print(tr.triangle_area(trngl_base, trngl_height))
```

2. Solution

```
class Pet:
    #Define the constructor
    def __init__(self):
        self.kind = None  #Initial value for the instance field kind
        self.legs_number = None  #Initial value for the instance field legs_number

    def start_running(self):
        print("Pet is running")

    def stop_running(self):
        print("Pet stopped")

#Main code starts here
pet1 = Pet()
pet1.kind = "dog"
pet1.legs_number = 4

pet2 = Pet()
pet2.kind = "monkey"
pet2.legs_number = 2

pet1.start_running()
pet2.start_running()
pet1.stop_running()
```

3. Solution

```
class Pet:
    #Define the constructor
    def __init__(self, kind, legs_number):
        self.kind = kind  #Initial value for the property kind
        self.legs_number = legs_number  #Initial value for the property legs_number

    #Define the getter of the property kind
    def getKind(self):
        return self._kind

    #Define the setter of the property kind
    def setKind(self, value):
        if value != "":
            self._kind = value
        else:
            raise ValueError("Cannot be empty")

    #Define the property kind
    kind = property(getKind, setKind)

    #Define the getter of the property legs_number
    def getLegsNumber(self):
        return self._legs_number

    #Define the setter of the property legs_number
    def setLegsNumber(self, value):
        if value >= 0:
            self._legs_number = value
        else:
            raise ValueError("Cannot be negative")

    #Define the property legs_number
    legs_number = property(getLegsNumber, setLegsNumber)

    def start_running(self):
        print("Pet is running")

    def stop_running(self):
        print("Pet stopped")

#Main code starts here
pet1 = Pet("dog", 4)

pet1.start_running()
pet1.stop_running()

pet1.kind = ""  #This will throw an error
pet1.legs_number = -3  #This will throw an error
```

4. Solution

```
BOXES = 30

class Box:
    #Define the constructor
    def __init__(self, width, length, height):
        self.width = width    #Initial value for the instance field width
        self.length = length  #Initial value for the instance field length
        self.height = height  #Initial value for the instance field heigth
    def display_volume(self):
        print("Volume", self.width * self.length * self.height)

    def display_dimensions(self):
        print(self.width, "x", self.length, "x", self.height)

#Main code starts here
list_of_obj = [None] * BOXES  #Create a list

for i in range(BOXES):
    w = float(input("Enter width: "))
    l = float(input("Enter length: "))
    h = float(input("Enter height: "))

    #Add each new object to the list
    list_of_obj[i] = Box(w, l, h)

for i in range(BOXES):
    list_of_obj[i].display_dimensions()
    list_of_obj[i].display_volume()
```

5. Solution

```
BOXES = 30

class Box:
    #Define the constructor
    def __init__(self, width, length, height):
        self.width = width    #Initial value for the property width
        self.length = length  #Initial value for the property length
        self.height = height  #Initial value for the property height

    #Define the getter
    def getWidth(self):
        return self._width

    #Define the setter
    def setWidth(self, value):
        if value > 0:
            self._width = value
        else:
            raise ValueError("Cannot be negative or zero")

    #Define the property width
    width = property(getWidth, setWidth)

    #Define the getter
    def getLength(self):
        return self._length
    #Define the setter
    def setLength(self, value):
        if value > 0:
            self._length = value
        else:
            raise ValueError("Cannot be negative or zero")

    #Define the property length
    length = property(getLength, setLength)

    #Define the getter
    def getHeight(self):
        return self._height

    #Define the setter
    def setHeight(self, value):
        if value > 0:
            self._height = value
        else:
            raise ValueError("Cannot be negative or zero")

    #Define the property heigth
    height = property(getHeight, setHeight)

    def display_volume(self):
        print("Volume", self.width * self.length * self.height)

    def display_dimensions(self):
        print(self.width, "x", self.length, "x", self.height)

#Main code starts here
list_of_obj = [None] * BOXES  #Create a list

for i in range(BOXES):
    w = float(input("Enter width: "))
    l = float(input("Enter length: "))
    h = float(input("Enter height: "))

    #Add each new object to the list
    list_of_obj[i] = Box(w, l, h)

for i in range(BOXES):
    list_of_obj[i].display_dimensions()
    list_of_obj[i].display_volume()
```

6. Solution

```
class Cube:
    #Define the constructor
    def __init__(self, edge):
        self.edge = edge  #Initial value for the instance field edge

    def display_volume(self):
        print("Volume:", self.edge ** 3)

    def display_one_surface(self):
        print("One surface:", self.edge ** 2)

    def display_total_surface(self):
        print("Total surface:", 6 * self.edge ** 2)

#Main code starts here
edge = float(input("Enter edge length of a cube: "))

cube1 = Cube(edge)

cube1.display_volume()
cube1.display_one_surface()
cube1.display_total_surface()
```

7. Solution

```
class Cube:
    #Define the constructor
    def __init__(self, edge):
        self.edge = edge  #Initial value for the property edge

    #Define the getter
    def getEdge(self):
        return self._edge

    #Define the setter
    def setEdge(self, value):
        if value > 0:
            self._edge = value
        else:
            raise ValueError("Cannot be negative or zero")

    #Define the property edge
    edge = property(getEdge, setEdge)

    def display_volume(self):
        print("Volume:", self.edge ** 3)

    def display_one_surface(self):
        print("One surface:", self.edge ** 2)
    def display_total_surface(self):
        print("Total surface:", 6 * self.edge ** 2)

#Main code starts here
edge = float(input("Enter edge length of a cube: "))

cube1 = Cube(edge)

cube1.display_volume()
cube1.display_one_surface()
cube1.display_total_surface()
```

8. Solution

```
def display_menu():
    print("1. Enter radius")
    print("2. Display radius")
    print("3. Display diameter")
    print("4. Display area")
    print("5. Display perimeter")
    print("6. Exit")

class Circle:
    #Define the getter
    def getRadius(self):
        if self._radius != None:
            return self._radius
        else:
            raise ValueError("Radius is not set")

    #Define the setter
    def setRadius(self, value):
        if value > 0:
            self._radius = value
        else:
            raise ValueError("Cannot be negative or zero")

    #Define the property radius
    radius = property(getRadius, setRadius)

    def get_diameter(self):
        return 2 * self.radius

    def get_area(self):
        return 3.14 * self.radius ** 2

    def get_perimeter(self):
        return 2 * 3.14 * self.radius
```

```
#Main code starts here
circle1 = Circle()

display_menu()
choice = int(input("Enter a choice: "))
while choice != 6:
    if choice == 1:
        radius = float(input("Enter radius: "))
        circle1.radius = radius
    elif choice == 2:
        print("Radius:", circle1.radius)
    elif choice == 3:
        print("Diameter:", circle1.get_diameter())
    elif choice == 4:
        print("Area:", circle1.get_area())
    else:
        print("Perimeter:", circle1.get_perimeter())

    display_menu()
    choice = int(input("Enter a choice: "))
```

9. Solution

```
class Info:
    #Define the getter
    def getUserText(self):
        return self._user_text

    #Define the setter
    def setUserText(self, value):
        if value != "":
            self._user_text = value
        else:
            raise ValueError("Cannot be set to empty")

    #Define the property user_text
    user_text = property(getUserText, setUserText)

    def get_spaces_count(self):
        count = 0
        for char in self.user_text:
            if char == " ":
                count += 1
        return count

    def get_words_count(self):
        return self.get_spaces_count() + 1
    def get_vowels_count(self):
        count = 0
        for char in self.user_text.lower():
            if char in "aeiou":
                count += 1
        return count

    def get_letters_count(self):
        return len(self.user_text) - self.get_spaces_count()

#Main code starts here
inf = Info()

text = input("Enter a text: ")

inf.user_text = text

print("Text:", inf.user_text)
print("Spaces:", inf.get_spaces_count())
print("Words:", inf.get_words_count())
print("Vowels:", inf.get_vowels_count())
print("Total number of letters:", inf.get_letters_count())
```

10.Solution

```
def display_menu():
    print("1. Enter Encryption/Decryption key")
    print("2. Encrypt a message")
    print("3. Decrypt a message")
    print("4. Exit")

class EncryptDecrypt:
    alphabet = " abcdefghijklmnopqrstuvwxyz"  #Space is a valid character!

    #Define the getter
    def getEncrDecrKey(self):
        if self._encr_decr_key != None:
            return self._encr_decr_key
        else:
            raise ValueError("Key is not set")

    #Define the setter
    def setEncrDecrKey(self, value):
        if value in range(1, 27):
            self._encr_decr_key = value
        else:
            raise ValueError("Key must be between 1 and 26")

    #Define the property encr_decr_key
    encr_decr_key = property(getEncrDecrKey, setEncrDecrKey)

    def encrypt(self, message):
        return_value = ""
        for char in message:
            index = self.alphabet.find(char)
            new_index = index + self.encr_decr_key
            if new_index >= 27:
                new_index -= 27
            new_letter = self.alphabet[new_index]
            return_value += new_letter
        return return_value

    def decrypt(self, enc_message):
        return_value = ""
        for char in enc_message:
            index = self.alphabet.find(char)
            new_index = index - self.encr_decr_key
            if new_index < 0:
                new_index += 27
            new_letter = self.alphabet[new_index]
            return_value += new_letter
        return return_value

#Main code starts here
ed = EncryptDecrypt()

display_menu()
choice = int(input("Enter a choice: "))
while choice != 4:
    if choice == 1:
        ed.encr_decr_key = int(input("Enter encryption/decryption key: "))
    elif choice == 2:
        text = input("Enter message to encrypt: ")
        print("Encrypted message:", ed.encrypt(text))
    else:
        text = input("Enter message to decrypt: ")
        print("Decrypted message:", ed.decrypt(text))

    display_menu()
    choice = int(input("Enter a choice: "))
```

11.Solution

```
class Vehicle:
    #Define the constructor
    def __init__(self, number_of_wheels, color, length, width, height):
        self.number_of_wheels = number_of_wheels
```

```python
        self.color = color
        self.length = length
        self.width = width
        self.height = height

    def start_engine(self):
        print("The engine started")

    def stop_engine(self):
        print("The engine stopped")

class Car(Vehicle):
    #Define the constructor
    def __init__(self, number_of_wheels, color, length, width, height):
        super().__init__(number_of_wheels, color, length, width, height)
        self.boot_capacity = 0

    def turn_windshield_wipers_on(self):
        print("The windshield wipers have been turned on!")

class Motorcycle(Vehicle):
    #Define the constructor
    def __init__(self, number_of_wheels, color, length, width, height):
        super().__init__(number_of_wheels, color, length, width, height)
        self.has_luggage = False

    def do_a_wheelie(self):
        print("I am doing a wheelie!!!")

#Main code starts here
car1 = Car(4, "Red", 5, 2, 1.5)
car1.boot_capacity = 300
car1.start_engine()
car1.turn_windshield_wipers_on()
car1.stop_engine()

car2 = Car(4, "Green", 4.5, 2.2, 1.4)
car2.boot_capacity = 400
car2.start_engine()
car2.turn_windshield_wipers_on()
car2.stop_engine()

motorcycle1 = Motorcycle(2, "Blue", 2, 0.9, 1.3)
motorcycle1.has_luggage = True
motorcycle1.start_engine()
motorcycle1.do_a_wheelie()
motorcycle1.stop_engine()
```

12.Solution

```
class SchoolMember:
    #Define the constructor
    def __init__(self, name, age):
        self.setName(name)
        self.setAge(age)
        print("A school member was initialized")

    #Define the getter
    def getName(self):
        return self._name

    #Define the setter
    def setName(self, value):
        if value != "":
            self._name = value
        else:
            raise ValueError("Name cannot be empty")

    #Define the getter
    def getAge(self):
        return self._age

    #Define the setter
    def setAge(self, value):
        if value > 0:
            self._age = value
        else:
            raise ValueError("Age cannot be negative or zero")

class Teacher(SchoolMember):
    #Define the constructor
    def __init__(self, name, age, salary):
        #Call the constructor of the parent class SchoolMember
        super().__init__(name, age)

        self.setSalary(salary)

        #This is an additional statement for this constructor
        print("A teacher was initialized")

    #Define the getter
    def getSalary(self):
        return self._salary

    #Define the setter
    def setSalary(self, value):
        if value >= 0:
            self._salary = value
        else:
            raise ValueError("Salary cannot be negative")

    #This is an additional method for this class
    def display_values(self):
        print("Name:", self.getName())
        print("Age:", self.getAge())
        print("Salary:", self.getSalary())

class Student(SchoolMember):
    #Define the constructor
    def __init__(self, name, age, final_grade):
        #Call the constructor of the parent class SchoolMember
        super().__init__(name, age)

        self.setFinalGrade(final_grade)

        #This is an additional statement for this constructor
        print("A student was initialized")

    #Define the getter
    def getFinalGrade(self):
        return self._final_grade

    #Define the setter
    def setFinalGrade(self, value):
        if value != "":
            self._final_grade = value
        else:
            raise ValueError("Final grade cannot be empty")

    #This is an additional method for this class
    def display_values(self):
        print("Name:", self.getName())
        print("Age:", self.getAge())
        print("Final grade:", self.getFinalGrade())

#Main code starts here
teacher1 = Teacher("Mr. John Scott", 43, 35000)
teacher2 = Teacher("Mrs. Ann Carter", 45, 32000)

student1 = Student("Peter Nelson", 14, "A")
student2 = Student("Helen Morgan", 13, "B")
```

13.Solution

```
class SchoolMember:
    #Define the constructor
    def __init__(self, name, age):
        self.name = name
        self.age = age
        print("A school member was initialized")

    #Define the getter
    def getName(self):
        return self._name

    #Define the setter
    def setName(self, value):
        if value != "":
            self._name = value
        else:
            raise ValueError("Name cannot be empty")

    #Define the property name
    name = property(getName, setName)

    #Define the getter
    def getAge(self):
        return self._age

    #Define the setter
    def setAge(self, value):
        if value > 0:
            self._age = value
        else:
            raise ValueError("Age cannot be negative or zero")

    #Define the property age
    age = property(getAge, setAge)

class Teacher(SchoolMember):
    #Define the constructor
    def __init__(self, name, age, salary):
        #Call the constructor of the parent class SchoolMember
        super().__init__(name, age)

        self.salary = salary

        #This is an additional statement for this constructor
        print("A teacher was initialized")

    #Define the getter
    def getSalary(self):
        return self._salary

    #Define the setter
    def setSalary(self, value):
        if value >= 0:
            self._salary = value
        else:
            raise ValueError("Salary cannot be negative")

    #Define the property salary
    salary = property(getSalary, setSalary)

    #This is an additional method for this class
    def display_values(self):
        print("Name:", self.getName())
        print("Age:", self.getAge())
        print("Salary:", self.getSalary())

class Student(SchoolMember):
    #Define the constructor
    def __init__(self, name, age, final_grade):
        #Call the constructor of the parent class SchoolMember
        super().__init__(name, age)

        self.final_grade = final_grade

        #This is an additional statement for this constructor
        print("A student was initialized")

    #Define the getter
    def getFinalGrade(self):
        return self._final_grade

    #Define the setter
    def setFinalGrade(self, value):
        if value != "":
            self._final_grade = value
        else:
            raise ValueError("Final grade cannot be empty")

    #Define the property final_grade
    final_grade = property(getFinalGrade, setFinalGrade)

    #This is an additional method for this class
    def display_values(self):
        print("Name:", self.getName())
        print("Age:", self.getAge())
        print("Final grade:", self.getFinalGrade())

#Main code starts here
teacher1 = Teacher("Mr. John Scott", 43, 35000)
teacher2 = Teacher("Mrs. Ann Carter", 45, 32000)

student1 = Student("Peter Nelson", 14, "A")
student2 = Student("Helen Morgan", 13, "B")

teacher1.display_values()
teacher2.display_values()
student1.display_values()
student2.display_values()
```

Chapter 30

30.7 Review Questions: True/False

1. False
2. False
3. True
4. False
5. False
6. False
7. False
8. False
9. True
10. False
11. True
12. True
13. False
14. False
15. True
16. True
17. False
18. False
19. True
20. True
21. True

30.8 Review Exercises

1. Solution

```
PATH = "c:/temp/"

days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]

f = open(PATH + "days_of_week.txt", "w")
for d in days:
    f.write(d + "\n")
f.close()
```

2. Solution

```
PATH = "c:/temp/"

days = []

f = open(path + "days_of_week.txt", "r")
for line in f:
    days.append(line)
f.close()
```

```
for day in days[::-1]:
    print(day, end = "")
```

3. Solution

```
PATH = "c:/temp/"

f = open(PATH + "days_of_week.txt", "a")
f.write("*** End of File ***")
f.close()
```

4. Solution

```
PATH = "c:/temp/"

import random

f = open(PATH + "randoms.txt", "w")
for i in range(50):
    f.write(str(random.randrange(1, 101)) + "\n")
f.close()
```

5. Solution

```
PATH = "c:/temp/"

import random

for i in range(1, 11):
    f = open(PATH + "file" + str(i) + ".txt", "w")
    f.write(str(random.randrange(100, 1000)))
    f.close()
```

6. Solution

```
PATH = "c:/temp/"

f = open(PATH + "multiplication_table.txt", "w")

for i in range(1, 11):
    for j in range(1, 5):
        f.write(str(i) + " x " + str(j) + " = " + str(i * j) + "\n")

f.close()
```

7. Solution

```
PATH = "c:/temp/"

f = open(PATH + "a_file.txt", "r")

for line in f:
    print(len(line) - 1)  #Minus one due to \n at the end of the line
```

```
f.close()
```

8. Solution

```
PATH = "c:/temp/"

f = open(PATH + "a_file.txt", "r")

count_lines = 0
count_chars = 0
for line in f:
    count_lines += 1
    count_chars += len(line) - 1

f.close()

print("Total characters: ", count_chars)
print("Total Lines: ", count_lines)
```

9. Solution

First approach

```
PATH = "c:/temp/"

f = open(PATH + "a_file.txt", "r")

i = 1
for line in f:
    for character in line:
        if character in ",.!":
            print("There is a punctuation mark on line No", i)
            break
    i += 1

f.close()
```

Second approach

```
PATH = "c:/temp/"

f = open(PATH + "a_file.txt", "r")

i = 1
for line in f:
    if "," in line or "." in line or "!" in line:
        print("There is a punctuation mark on line No", i)

    i += 1

f.close()
```

Chapter 31

31.2 Review Exercises

1. Solution

```
PATH = "c:/temp/"

fin = open(PATH + "f_data31.2-1.txt")
values = fin.readline().split()
fin.close()

total = 0
count = 0
for value in values:
    number = int(value)
    if number > 50:
        total += number
        count += 1

if count > 0:
    print(total / count)
```

2. Solution

```
PATH = "c:/temp/"

fin = open(PATH + "f_data31.2-2.txt")
values = fin.readline().split(",")
fin.close()

total = 0
count = 0

for value in values:
    number = int(value)
    if 300 <= number <= 500:
        total += number
        count += 1

if count > 0:
    print(total / count)
```

3. Solution

```
PATH = "c:/temp/"

fin = open(PATH + "f_data31.2-3.txt")

#Read the first line
```

```
line = fin.readline()
b = line.split(",")

maximum = int(b[0])
minimum = int(b[0])
max_name = b[1]
min_name = b[1]

#Read the rest of the lines
for line in fin:
    b = line.split(",")
    grade = int(b[0])

    if grade > maximum:
        maximum = grade
        max_name = b[1]

    if grade < minimum:
        minimum = grade
        min_name = b[1]

fin.close()

print(max_name, end = "")
print(min_name, end = "")
```

4. Solution

First approach

```
filename1 = input("Enter filename No 1: ")

if filename1[-4:] != ".txt":
    print("Wrong filename")
else:
    filename2 = input("Enter filename No 2: ")
    if filename2[-4:] != ".txt":
        print("Wrong filename")
    else:
        fin = open(filename2)
        content = fin.read()
        fin.close()

        fin = open(filename1)
        content += fin.read()  #Concatenation
        fin.close()

        fout = open("final.txt", "w")
        fout.write(content)
        fout.close()
```

Second approach

```
filename1 = input("Enter filename No 1: ")

if filename1[-4:] != ".txt":
    print("Wrong filename")
else:
    filename2 = input("Enter filename No 2: ")
    if filename2[-4:] != ".txt":
        print("Wrong filename")
    else:
        fin1 = open(filename1)
        fin2 = open(filename2)
        fout = open("final.txt", "w")

        fout.write(fin2.read() + fin1.read())

        fout.close()
        fin2.close()
        fin1.close()
```

5. *Solution*

```
PATH = "c:/temp/"

fin = open(PATH + "f_data31.2-5.txt")

numbers = []
for line in fin:
    numbers.append(float(line))

fin.close()

numbers.sort()

fout = open(PATH + "f_data31.2-5.txt", "a")
fout.write("********* Sorted numbers *************\n")
for number in numbers:
    fout.write(str(number) + "\n")

fout.close()
```

6. *Solution*

```
PATH = "c:/temp/"

fin = open(PATH + "f_data31.2-6.txt", "r")

cities = []
temperatures = []
onCityLine = True
```

```
for line in fin:
    if onCityLine:
        cities.append(line)
    else:
        temperatures.append(float(line))

    onCityLine = not onCityLine  #True becomes False, and False becomes True

fin.close()

N = len(temperatures)

total = 0
for i in range(N):
    total += temperatures[i]

average = total / N
print(average)

maximum = max(temperatures)
print("Highest temperature:", maximum)
for i in range(N):
    if temperatures[i] == maximum:
        print(cities[i])
```

7. Solution

```
PATH = "c:/temp/"
x = " ABCDEFGHIJKLMNOPQRSTUVWXYZ"  #The space character remains as is
y = " JKWCTAMEDXSLFBYUNGRZOIQVHP"

initial_message = input("Enter a message to encrypt: ").upper()

encrypted_message = ""
for letter in initial_message:
    #Search for letter in variable x
    for i in range(len(x)):
        if letter == x[i]:
            #Create encrypted message using letters from variable y
            encrypted_message += y[i]
            break

fout = open(PATH + "encrypted.txt", "w")
fout.write(encrypted_message)
fout.close()
```

8. Solution

```
PATH = "c:/temp/"
x = " ABCDEFGHIJKLMNOPQRSTUVWXYZ"  #The space character remains as is
```

```
y = " JKWCTAMEDXSLFBYUNGRZOIQVHP"

fin = open(PATH + "encrypted.txt")
encrypted_message = fin.readline()
fin.close()

initial_message = ""
for letter in encrypted_message:
    #Search for letter in variable y
    for i in range(len(y)):
        if letter == y[i]:
            #Create decrypted message using letters from variable x
            initial_message += x[i]
            break

fout = open(PATH + "decrypted.txt", "w")
fout.write(initial_message)
fout.close()
```

9. Solution

First approach

```
def copy(source, destination):
    fin = open(source, "r")
    x = fin.read()
    fin.close()

    fout = open(destination, "w")
    fout.write(x)
    fout.close()
```

Second approach

```
def copy(source, destination):
    fin = open(source, "r")
    fout = open(destination, "w")

    fout.write(fin.read())

    fin.close()
    fout.close()
```

10.Solution

```
import math

PATH = "c:/temp/"

class Triangle:
    #Define the constructor
    def __init__(self):
        fin = open(PATH + "f_data31.2-10.txt", "r")
        self._sideA = float(fin.readline())
        self._sideB = float(fin.readline())
        self._sideC = float(fin.readline())
        fin.close()

    def can_be_triangle(self):
        if self._sideA > 0 and self._sideB > 0 and self._sideC > 0 and   \
           self._sideA + self._sideB > self._sideC and   \
           self._sideB + self._sideC > self._sideA and   \
           self._sideC + self._sideA > self._sideB:
            return True
        else:
            return False

    def display_lengths(self):
        print("Side A:", self._sideA)
        print("Side B:", self._sideB)
        print("Side C:", self._sideC)
        if self.can_be_triangle():
            print("Can be lengths of the three sides of a triangle!")
        else:
            print("Cannot be lengths of the three sides of a triangle!")

    def display_area(self):
        if self.can_be_triangle():
            s = (self._sideA + self._sideB + self._sideC) / 2
            area = math.sqrt(s * (s - self._sideA) * (s - self._sideB) * (s - self._sideC))
            print("Area:", area)

    def display_perimeter(self):
        if self.can_be_triangle():
            perimeter = self._sideA + self._sideB + self._sideC
            print("Perimeter:", perimeter)

tr = Triangle()

tr.display_lengths()
tr.display_area()
tr.display_perimeter()
```