# C# for Tweens and Teens

Learn Computational and Algorithmic Thinking

Revised Second Edition

# The Answers

By
Aristides S. Bouras

C# for Tweens and Teens – The Answers

Revised Second Edition

Copyright © by Aristides S. Bouras

https://www.bouraspage.com

RCode: 220225

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, mechanical or electronic, including photocopying, recording, or by any information storage and retrieval system, without written permission from the authors.

## Warning and Disclaimer

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

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 ..... 5
Chapter 1 ..... 6
1.7 Review Questions: True/False ..... 6
1.8 Review Questions: Multiple Choice ..... 6
Chapter 3 ..... 6
3.11 Review Questions: True/False ..... 6
3.12 Review Questions: Multiple Choice ..... 6
Chapter 4 ..... 6
4.5 Review Questions: True/False ..... 6
4.6 Review Questions: Multiple Choice ..... 7
4.7 Review Exercises ..... 7
Chapter 5 ..... 7
5.4 Review Questions: True/False ..... 7
5.5 Review Questions: Multiple Choice ..... 7
Chapter 6 ..... 8
6.6 Review Questions: True/False ..... 8
6.7 Review Questions: Multiple Choice ..... 8
6.8 Review Exercises ..... 8
Chapter 9 ..... 8
9.2 Review Exercises ..... 8
Chapter 10 ..... 11
10.3 Review Questions: True/False ..... 11
10.4 Review Exercises ..... 11
Chapter 11 ..... 11
11.4 Review Questions: True/False ..... 12
11.5 Review Questions: Multiple Choice ..... 12
11.6 Review Exercises ..... 12
Chapter 12 ..... 13
12.10 Review Questions: True/False ..... 13
12.11 Review Questions: Multiple Choice ..... 14
12.12 Review Exercises ..... 14
Chapter 13 ..... 15
13.2 Review Questions: True/False ..... 15
13.3 Review Questions: Multiple Choice ..... 15
13.4 Review Exercises ..... 15
Chapter 14 ..... 19
14.2 Review Questions: True/False ..... 19
14.3 Review Questions: Multiple Choice ..... 19
14.4 Review Exercises ..... 20
Chapter 15 ..... 21
15.2 Review Questions: True/False ..... 22
15.3 Review Exercises ..... 22
Chapter 16 ..... 31
16.2 Review Questions: True/False ..... 32

16.3 Review Exercises ..... 32
Chapter 17 ..... 34
17.3 Review Questions: True/False ..... 35
Chapter 18 ..... 35
18.2 Review Questions: True/False ..... 35
18.3 Review Questions: Multiple Choice ..... 36
18.4 Review Exercises ..... 36
Chapter 19 ..... 38
19.2 Review Questions: True/False ..... 39
19.3 Review Questions: Multiple Choice ..... 39
19.4 Review Exercises ..... 39
Chapter 20 ..... 41
20.2 Review Questions: True/False ..... 41
20.3 Review Questions: Multiple Choice ..... 42
20.4 Review Exercises ..... 42
Chapter 21 ..... 43
21.7 Review Questions: True/False ..... 44
21.8 Review Questions: Multiple Choice ..... 44
21.9 Review Exercises ..... 44
Chapter 22 ..... 47
22.2 Review Exercises ..... 47
Chapter 23 ..... 54
23.12 Review Exercises ..... 54
Chapter 24 ..... 60
24.14 Review Questions: True/False ..... 62
24.15 Review Questions: Multiple Choice ..... 63
24.16 Review Exercises ..... 63
Chapter 25 ..... 71
25.5 Review Questions: True/False ..... 72
25.6 Review Exercises ..... 72
Chapter 26 ..... 81
26.4 Review Questions: True/False ..... 84
Chapter 27 ..... 85
27.11 Review Questions: True/False ..... 85
27.12 Review Exercises ..... 85
Chapter 28 ..... 86
28.2 Review Exercises ..... 87
Chapter 29 ..... 90
29.8 Review Questions: True/False ..... 92
29.9 Review Exercises ..... 92
Chapter 30 ..... 105
30.9 Review Questions: True/False ..... 112
30.10 Review Exercises ..... 112
Chapter 31 ..... 114
31.2 Review Exercises ..... 114

# 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

## 4.8 Review Questions: True/False

1. false
2. true
3. false
4. false
5. true
6. false
7. true
8. false
9. true
10. false
11. true
12. true
13. true
14. true
15. false

## 4.9 Review Questions: Multiple Choice

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

## 4.10 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 | `string name = "Mark";` |
| My address | String | `string address = "254 Lookout Rd. Wilson, NY 27893";` |
| The average daily temperature | Float | `double average = 70.3;` |
| A telephone number | String | `string phone_number = "1-891-764-2410";` |
| My Social Security Number (SSN) | String | `string ssn = "123-45-6789";` |
| The speed of a car | Float | `double speed = 90.5;` |
| The number of children in a family | Integer | `int 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

## 6.7 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
20. true
21. false

## 6.8 Review Questions: Multiple Choice

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

## 6.9 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 8

## 8.2 Review Exercises

### *1. Solution*

```
double b, h, area;

Console.Write("Enter base: ");
b = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter height: ");
h = Convert.ToDouble(Console.ReadLine());

area = b * h / 2;
```

```
Console.WriteLine(area);
```

### 2. Solution

```
double f, k;

Console.Write("Enter temperature in Fahrenheit: ");
f = Convert.ToDouble(Console.ReadLine());

k = (f + 459.67) / 1.8;

Console.WriteLine(k);
```

### 3. Solution

```
double angle1, angle2, angle3;

Console.Write("Enter 1st angle: ");
angle1 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 2nd angle: ");
angle2 = Convert.ToDouble(Console.ReadLine());

angle3 = 180 - angle1 - angle2;

Console.WriteLine(angle3);
```

### 4. Solution

```
double average, g1, g2, g3, g4;

Console.Write("Enter 1st grade: ");
g1 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter 2nd grade: ");
g2 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter 3rd grade: ");
g3 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter 4th grade: ");
g4 = Convert.ToInt32(Console.ReadLine());

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

Console.WriteLine(average);
```

### 5. Solution

```
const double PI = 3.14159;

double r, perimeter;

Console.Write("Enter radius: ");
r = Convert.ToDouble(Console.ReadLine());

perimeter = 2 * PI * r;
```

```
Console.WriteLine(perimeter);
```

### 6. Solution

```
const double PI = 3.14159;
double d, radius, volume;

Console.Write("Enter diameter (in meters): ");
d = Convert.ToDouble(Console.ReadLine());

radius = d / 2;

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

Console.WriteLine(volume);
```

### 7. Solution

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

### 8. Solution

```
const double PI = 3.14159;
double d, radius, perimeter, area, volume;

Console.Write("Enter diameter (in meters): ");
d = Convert.ToDouble(Console.ReadLine());

radius = d / 2;
perimeter = 2 * PI * radius;
area = PI * radius * radius;

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

Console.WriteLine(radius + " " + perimeter + " " + area + " " + volume);
```

### 9. Solution

```
int w, h;
double bmi;

Console.Write("Enter weight in pounds: ");
w = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter height in inches: ");
h = Convert.ToInt32(Console.ReadLine());

bmi = w * 703.0 / (h * h);

Console.WriteLine(bmi);
```

### 10.Solution

```
int d, m, days_passed, days_left;

Console.Write("Enter current month: ");
```

```
m = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter current day: ");
d = Convert.ToInt32(Console.ReadLine());

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

Console.WriteLine(days_left);
```

### 11. Solution

```
string first_name, middle_name, last_name, title;

Console.Write("First name: ");
first_name = Console.ReadLine();
Console.Write("Middle name: ");
middle_name = Console.ReadLine();
Console.Write("Last name: ");
last_name = Console.ReadLine();
Console.Write("Title: ");
title = Console.ReadLine();

Console.WriteLine(title + " " + first_name + " " + middle_name + " " + last_name);
Console.WriteLine(first_name + " " + middle_name + " " + last_name);
Console.WriteLine(last_name + ", " + first_name);
Console.WriteLine(last_name + ", " + first_name + " " + middle_name);
Console.WriteLine(last_name + ", " + first_name + " " + middle_name + ", " + title);
Console.WriteLine(first_name + " " + last_name);
```

# Chapter 9

## 9.3 Review Questions: True/False

1. true
2. false
3. false
4. true
5. false
6. true
7. false
8. false
9. true
10. true
11. false

## 9.4 Review Exercises

1. 2
2. i – 2.5, ii – 2.2
3. i – 4, ii – 9
4. i – 12, ii – 8.5
5. i – 5, ii – 4

### 6. Solution

```
double a, b, hypotenuse;
```

```
Console.Write("Enter right angle side A of a right-angled triangle: ");
a = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter right angle side B of a right-angled triangle: ");
b = Convert.ToDouble(Console.ReadLine());

hypotenuse = Math.Sqrt(Math.Pow(a, 2) + Math.Pow(b, 2));

Console.WriteLine(hypotenuse);
```

# Chapter 10

## 10.4 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

## 10.5 Review Questions: Multiple Choice

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

## 10.6 Review Exercises

### *1. Solution*

```
string alphabet;
Random rnd = new();

alphabet = "abcdefghijklmnopqrstuvwxyz";

Console.Write(alphabet[rnd.Next(0, 26)].ToString().ToUpper());
Console.Write(alphabet[rnd.Next(0, 26)]);
Console.Write(alphabet[rnd.Next(0, 26)]);
Console.Write(alphabet[rnd.Next(0, 26)]);
```

### *2. Solution*

```
string name, x, secret_password;
Random rnd = new();

Console.Write("Enter name: ");
name = Console.ReadLine();

x = name.ToLower().Replace(" ", "");    //Convert to lower case and remove spaces

secret_password = "" + x[rnd.Next(0, x.Length)] +
                       x[rnd.Next(0, x.Length)] +
```

```
                        x[rnd.Next(0, x.Length)] +
                        rnd.Next(1000, 10000);

Console.WriteLine(secret_password);
```

### 3. Solution

**First approach**

```
int number, reversed_number;
string s_number, digit1, digit2, digit3;

Console.Write("Enter a three-digit integer: ");
number = Convert.ToInt32(Console.ReadLine());

s_number = "" + number;   //s_number is of type string

digit1 = "" + s_number[0];
digit2 = "" + s_number[1];
digit3 = "" + s_number[2];

reversed_number = 100 * Convert.ToInt32(digit3) +
                  10 * Convert.ToInt32(digit2) +
                  Convert.ToInt32(digit1);

Console.WriteLine(reversed_number);
```

**Second approach**

```
int number, reversed_number;
string s_number;

Console.Write("Enter a three-digit integer: ");
number = Convert.ToInt32(Console.ReadLine());

s_number = "" + number;   //s_number is of type string
reversed_number = Convert.ToInt32("" + s_number[2] + s_number[1] + s_number[0]);

Console.WriteLine(reversed_number);
```

# Chapter 11

## 11.9 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

## 11.10 Review Questions: Multiple Choice

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

## 11.11 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 \|\| BE2* | *BE1 && BE2* | *!(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 \|\| c > b && c > 1 | a > 3 && c > b \|\| c > 1 |
|---|---|---|---|---|
| 4 | –6 | 2 | *true* | *true* |
| –3 | 2 | –4 | *false* | *false* |

### 5. Solution

| Expression | Value |
|---|---|
| `Math.Pow(x + y, 3)` | *8.0* |
| `(x + y) / (Math.Pow(x, 2) - 14)` | *1.0* |
| `(x - 1) == y + 5` | *true* |
| `x > 2 && y == 1` | *false* |
| `x == 1 \|\| !(flag == false)` | *true* |

### 6. Solution

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

b. `age >= 6 && age <= 9 || age == 11`

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

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

e. `age >= 6 && age <= 12 && age != 8`

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

# Chapter 12

## 12.2 Review Questions: True/False

1. false
2. false
3. true
4. false

## 12.3 Review Questions: Multiple Choice

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

## 12.4 Review Exercises

### 1. Solution

```
double x, y;

x = Convert.ToDouble(Console.ReadLine());

y = -5;
if (x * y / 2 > 20)
    y *= 2;
    x += 4 * Math.Pow(x, 2);
}
Console.WriteLine(x + " " + y);
```

### 2. Solution

i. 9 12

ii. 2 2

### 3. Solution

```
double x;

Console.Write("Enter a number: ");
x = Convert.ToDouble(Console.ReadLine());

if (x > 0) {
    Console.WriteLine("Positive");
}
```

### 4. Solution

```
double x, y;

Console.Write("Enter a number: ");
x = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter a second number");
y = Convert.ToDouble(Console.ReadLine());

if (x > 0 && y > 0) {
    Console.WriteLine("Positives");
}
```

### 5. Solution

```
int x;

Console.Write("Enter your age: ");
x = Convert.ToInt32(Console.ReadLine());

if (x > 14) {
    Console.WriteLine("You can drive a car in Kansas (USA)");
}
```

### 6. Solution

```
string s;

Console.Write("Enter a string: ");
s = Console.ReadLine();

if (s == s.ToUpper()) {
    Console.WriteLine("Uppercase");
}
```

### 7. Solution

```
string s;

Console.Write("Enter a string: ");
s = Console.ReadLine();

if (s.Length > 20) {
    Console.WriteLine("Many characters");
}
```

### 8. Solution

```
double n1, n2, n3;

Console.Write("Enter 1st number: ");
n1 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 2nd number: ");
n2 = Convert.ToDouble(Console.ReadLine());
```

```
Console.Write("Enter 3rd number: ");
n3 = Convert.ToDouble(Console.ReadLine());

if (n1 < 0 || n2 < 0 || n3 < 0) {
    Console.WriteLine("Among the given numbers, there is a negative one!");
}
```

## 9. Solution

```
double t1, t2, t3, average;

Console.Write("Enter 1st temperature: ");
t1 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 2nd temperature: ");
t2 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 3rd temperature: ");
t3 = Convert.ToDouble(Console.ReadLine());

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

if (average > 60) {
    Console.WriteLine("Heat Wave");
}
```

## 10.Solution

```
double w1, w2, w3, w4, maximum;

Console.Write("Enter the weight of the 1st person: ");
w1 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter the weight of the 2nd person: ");
w2 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter the weight of the 3rd person: ");
w3 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter the weight of the 4th person: ");
w4 = Convert.ToDouble(Console.ReadLine());

maximum = w1;

if (w2 > maximum) {
    maximum = w2;
}

if (w3 > maximum) {
    maximum = w3;
}

if (w4 > maximum) {
    maximum = w4;
}
```

```
Console.WriteLine(maximum);
```

### 11.Solution

```
string n1, n2, n3, n4, m_name;
int a1, a2, a3, a4, minimum;

Console.Write("Enter the age of the 1st person: ");
a1 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the name of the 1st person: ");
n1 = Console.ReadLine();

Console.Write("Enter the age of the 2nd person: ");
a2 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the name of the 2nd person: ");
n2 = Console.ReadLine();

Console.Write("Enter the age of the 3rd person: ");
a3 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the name of the 3rd person: ");
n3 = Console.ReadLine();

Console.Write("Enter the age of the 4th person: ");
a4 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the name of the 4th person: ");
n4 = Console.ReadLine();

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;
}

Console.WriteLine("The youngest person is " + m_name);
```

### 12.Solution

```
int a1, a2, a3, minimum, maximum, middle;

Console.Write("Enter the age of the 1st person: ");
a1 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the age of the 2nd person: ");
a2 = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter the age of the 3rd person: ");
a3 = Convert.ToInt32(Console.ReadLine());

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;

Console.WriteLine(middle);
```

# Chapter 13

## 13.2 Review Questions: True/False

1. false
2. true
3. false
4. false

## 13.3 Review Questions: Multiple Choice

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

## 13.4 Review Exercises

### 1. Solution

i. 1 ii. 5

### 2. Solution

i. 7.0 18.0 ii. 0.5 3.5

### 3. Solution

```
double num;

Console.Write("Enter a number: ");
num = Convert.ToDouble(Console.ReadLine());

if (num > 100) {
    Console.WriteLine("Given number is greater than 100");
}
else {
    Console.WriteLine("Given number is less than or equal to 100");
}
```

### 4. Solution

```
double num;

Console.Write("Enter a number: ");
num = Convert.ToDouble(Console.ReadLine());

if (num >= 0 && num <= 100) {
    Console.WriteLine("Given number is between 0 and 100");
}
else {
    Console.WriteLine("Given number is not between 0 and 100");
}
```

### 5. Solution

```
int num;

num = Convert.ToInt32(Console.ReadLine());

if (num >= 1000 && num <= 9999) {
    Console.WriteLine(num + " is a four-digit integer");
}
else {
    Console.WriteLine(num + " is not a four-digit integer");
}
```

### 6. Solution

```
double num1, num2;
```

```
Console.Write("Enter first number: ");
num1 = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter second number: ");
num2 = Convert.ToDouble(Console.ReadLine());

if (num1 < num2) {
    Console.WriteLine(num1);
}
else {
    Console.WriteLine(num2);
}
```

## 7. Solution

```
string name1, name2;
int goals1, goals2;

Console.Write("Enter team name 1: ");
name1 = Console.ReadLine();
Console.Write("Enter team name 2: ");
name2 = Console.ReadLine();

Console.Write("Enter goals " + name1 + " scored: ");
goals1 = Int32.Parse(Console.ReadLine());
Console.Write("Enter goals " + name2 + " scored: ");
goals2 = Int32.Parse(Console.ReadLine());

if (goals1 > goals2) {
    Console.WriteLine("Winner: " + name1);
}
else {
    Console.WriteLine("Winner: " + name2);
}
```

## 8. Solution

```
double a, b, c, average;

Console.Write("Enter 1st jump in meters: ");
a = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 2nd jump in meters: ");
b = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter 3rd jump in meters: ");
c = Convert.ToDouble(Console.ReadLine());

average = (a + b + c) / 3;

if (average >= 8) {
    Console.WriteLine("Qualified");
```

```
}
else {
    Console.WriteLine("Disqualified");
}
```

# Chapter 14

## 14.2 Review Questions: True/False

1. true
2. false
3. false
4. false
5. false
6. true
7. true

## 14.3 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*

```
string name1, name2;
int goals1, goals2;

Console.Write("Enter team name 1: ");
name1 = Console.ReadLine();
Console.Write("Enter team name 2: ");
name2 = Console.ReadLine();

Console.Write("Enter goals " + name1 + " scored: ");
goals1 = Int32.Parse(Console.ReadLine());
Console.Write("Enter goals " + name2 + " scored: ");
goals2 = Int32.Parse(Console.ReadLine());

if (goals1 > goals2) {
    Console.WriteLine("Winner: " + name1);
}
else if (goals2 > goals1) {
    Console.WriteLine("Winner: " + name2);
}
else {
    Console.WriteLine("It's a tie!");
}
```

## 4. Solution

### First approach

```
int a, n;

Console.Write("Enter an integer between -9999 and 9999: ");
a = Convert.ToInt32(Console.ReadLine());

if (a >= -9999 && a <= -1000 || a >= 1000 && a <= 9999) {
    n = 4;
}
else if (a >= -999 && a <= -100 || a >= 100 && a <= 999) {
    n = 3;
}
else if (a >= -99 && a <= -10 || a >= 10 && a <= 99) {
    n = 2;
}
else {
    n = 1;
}

Console.Write("You entered a " + n + "-digit integer");
```

### Second approach

```
int a, n;

Console.Write("Enter an integer between -9999 and 9999: ");
a = Convert.ToInt32(Console.ReadLine());

//If variable a is negative, make it positive
if (a < 0) {
    a = (-1) * a;
}

if (a >= 1000 && a <= 9999) {
    n = 4;
}
else if (a >= 100 && a <= 999) {
    n = 3;
}
else if (a >= 10 && a <= 99) {
    n = 2;
}
else {
    n = 1;
}

Console.Write("You entered a " + n + "-digit integer");
```

## 5. Solution

### First approach

```
int a;

Console.Write("Enter an integer between -9999 and 9999: ");
a = Convert.ToInt32(Console.ReadLine());

if (a >= -9999 && a <= -1000 || a >= 1000 && a <= 9999) {
    Console.WriteLine("You entered a 4-digit integer");
}
else if (a >= -999 && a <= -100 || a >= 100 && a <= 999) {
    Console.WriteLine("You entered a 3-digit integer");
}
else if (a >= -99 && a <= -10 || a >= 10 && a <= 99) {
    Console.WriteLine("You entered a 2-digit integer");
}
else if (a >= -9 && a <= -1 || a >= 1 && a <= 9) {
    Console.WriteLine("You entered a 1-digit integer");
}
else {
    Console.WriteLine("Error: Invalid value!");
}
```

### Second approach

```
int a;

Console.Write("Enter an integer between -9999 and 9999: ");
a = Convert.ToInt32(Console.ReadLine());

//If variable a is negative, make it positive
if (a < 0) {
    a = (-1) * a;
}

if (a >= 1000 && a <= 9999) {
    Console.WriteLine("You entered a 4-digit integer");
}
else if (a >= 100 && a <= 999) {
    Console.WriteLine("You entered a 3-digit integer");
}
else if (a >= 10 && a <= 99) {
    Console.WriteLine("You entered a 2-digit integer");
}
else if (a >= 1 && a <= 9) {
    Console.WriteLine("You entered a 1-digit integer");
}
else {
```

```
    Console.WriteLine("Error: Invalid value!");
}
```

### 6. Solution

```
int m;

Console.Write("Enter the number of a month between 1 and 12: ");
m = Convert.ToInt32(Console.ReadLine());

if (m <= 2 || m == 12) {
    Console.WriteLine("Winter");
}
else if (m <= 5) {
    Console.WriteLine("Spring");
}
else if (m <= 8) {
    Console.WriteLine("Summer");
}
else {
    Console.WriteLine("Fall (Autumn)");
}
```

### 7. Solution

```
int m;

Console.Write("Enter the number of a month between 1 and 12: ");
m = Convert.ToInt32(Console.ReadLine());

if (m < 1 || m > 12) {
    Console.WriteLine("Error: Invalid value!");
}
else if (m <= 2 || m == 12) {
    Console.WriteLine("Winter");
}
else if (m <= 5) {
    Console.WriteLine("Spring");
}
else if (m <= 8) {
    Console.WriteLine("Summer");
}
else {
    Console.WriteLine("Fall (Autumn)");
}
```

### 8. Solution

```
string name;

Console.Write("Enter the name of a month: ");
```

```
name = Console.ReadLine().ToUpper();

if (name == "JANUARY") {
    Console.WriteLine(1);
}
else if (name == "FEBRUARY") {
    Console.WriteLine(2);
}
else if (name == "MARCH") {
    Console.WriteLine(3);
}
else if (name == "APRIL") {
    Console.WriteLine(4);
}
else if (name == "MAY") {
    Console.WriteLine(5);
}
else if (name == "JUNE") {
    Console.WriteLine(6);
}
else if (name == "JULY") {
    Console.WriteLine(7);
}
else if (name == "AUGUST") {
    Console.WriteLine(8);
}
else if (name == "SEPTEMBER") {
    Console.WriteLine(9);
}
else if (name == "OCTOBER") {
    Console.WriteLine(10);
}
else if (name == "NOVEMBER") {
    Console.WriteLine(11);
}
else if (name == "DECEMBER") {
    Console.WriteLine(12);
}
else {
    Console.WriteLine("Error");
}
```

## 9. Solution

```
string letter;

Console.Write("Enter a letter between A and F: ");
```

```
letter = Console.ReadLine();

if (letter == "A") {
    Console.WriteLine("90 - 100");
}
else if (letter == "B") {
    Console.WriteLine("80 - 89");
}
else if (letter == "C") {
    Console.WriteLine("70 - 79");
}
else if (letter == "D") {
    Console.WriteLine("60 - 69");
}
else {
    Console.WriteLine("0 - 59");
}
```

## 10.Solution

```
string roman;

Console.Write("Enter a Roman numeral between I and X: ");
roman = Console.ReadLine();

if (roman == "I") {
    Console.WriteLine(1);
}
else if (roman == "II") {
    Console.WriteLine(2);
}
else if (roman == "III") {
    Console.WriteLine(3);
}
else if (roman == "IV") {
    Console.WriteLine(4);
}
else if (roman == "V") {
    Console.WriteLine(5);
}
else if (roman == "VI") {
    Console.WriteLine(6);
}
else if (roman == "VII") {
    Console.WriteLine(7);
}
else if (roman == "VIII") {
```

```
    Console.WriteLine(8);
}
else if (roman == "IX") {
    Console.WriteLine(9);
}
else if (roman == "X") {
    Console.WriteLine(10);
}
else {
    Console.WriteLine("Error");
}
```

### 11.Solution

```
int total;

Console.Write("Enter the total number of CDs purchased in a month: ");
total = Convert.ToInt32(Console.ReadLine());

if (total == 1) {
    Console.WriteLine("You are awarded 3 points");
}
else if (total == 2) {
    Console.WriteLine("You are awarded 10 points");
}
else if (total == 3) {
    Console.WriteLine("You are awarded 20 points");
}
else {
    Console.WriteLine("You are awarded 45 points");
}
```

### 12.Solution

```
string num;

Console.Write("Enter a number (0 - 3) in words");
num = Console.ReadLine();

if (num == "zero") {
    Console.WriteLine(0);
}
else if (num == "one") {
    Console.WriteLine(1);
}
else if (num == "two") {
    Console.WriteLine(2);
}
else if (num == "three") {
```

```
    Console.WriteLine(3);
}
else {
    Console.WriteLine("I don't know this number!");
}
```

### 13.Solution

```
int b;

Console.Write("Enter Beaufort number: ");
b = Convert.ToInt32(Console.ReadLine());

if (b == 0) {
    Console.WriteLine("Calm");
}
else if (b == 1) {
    Console.WriteLine("Light Air");
}
else if (b == 2) {
    Console.WriteLine("Light breeze");
}
else if (b == 3) {
    Console.WriteLine("Gentle breeze");
}
else if (b == 4) {
    Console.WriteLine("Moderate breeze");
}
else if (b == 5) {
    Console.WriteLine("Fresh breeze");
}
else if (b == 6) {
    Console.WriteLine("Strong breeze");
}
else if (b == 7) {
    Console.WriteLine("Moderate gale");
}
else if (b == 8) {
    Console.WriteLine("Gale");
}
else if (b == 9) {
    Console.WriteLine("Strong gale");
}
else if (b == 10) {
    Console.WriteLine("Storm");
}
else if (b == 11) {
```

```
    Console.WriteLine("Violent storm");
}
else if (b == 12) {
    Console.WriteLine("Hurricane force");
}
else {
    Console.WriteLine("Invalid Beaufort number!");
}
```

### 14.Solution

```
double wind;

Console.Write("Enter wind speed (in miles/hour): ");
wind = Convert.ToDouble(Console.ReadLine());

if (wind < 0) {
    Console.WriteLine("Entered value is negative");
}
else if (wind < 1) {
    Console.Write("Beaufort: 0\nCalm");
}
else if (wind < 4) {
    Console.Write("Beaufort: 1\nLight air");
}
else if (wind < 8) {
    Console.Write("Beaufort: 2\nLight breeze");
}
else if (wind < 13) {
    Console.Write("Beaufort: 3\nGentle breeze");
}
else if (wind < 18) {
    Console.Write("Beaufort: 4\nModerate breeze");
}
else if (wind < 25) {
    Console.Write("Beaufort: 5\nFresh breeze");
}
else if (wind < 31) {
    Console.Write("Beaufort: 6\nStrong breeze");
}
else if (wind < 39) {
    Console.Write("Beaufort: 7\nModerate gale");
}
else if (wind < 47) {
    Console.Write("Beaufort: 8\nGale");
}
else if (wind < 55) {
```

```
    Console.Write("Beaufort: 9\nStrong gale");
}
else if (wind < 64) {
    Console.Write("Beaufort: 10\nStorm");
}
else if (wind < 74) {
    Console.Write("Beaufort: 11\nViolent storm");
}
else {
    Console.Write("Beaufort: 12\nHurricane force");
}
```

### 15.Solution

```
int choice;
double kelvin, fahrenheit, celsius;

Console.WriteLine("1. Convert Kelvin to Fahrenheit");
Console.WriteLine("2. Convert Fahrenheit to Kelvin");
Console.WriteLine("3. Convert Fahrenheit to Celsius");
Console.WriteLine("4. Convert Celsius to Fahrenheit");

Console.Write("Enter a choice: ");
choice = Convert.ToInt32(Console.ReadLine());

if (choice == 1) {
    Console.Write("Enter a temperature in degrees Kelvin: ");
    kelvin = Convert.ToDouble(Console.ReadLine());
    fahrenheit = 1.8 * kelvin - 459.67;
    Console.Write(kelvin + " degrees Kelvin = " + fahrenheit + " degrees Fahrenheit");
}
else if (choice == 2) {
    Console.Write("Enter a temperature in degrees Fahrenheit: ");
    fahrenheit = Convert.ToDouble(Console.ReadLine());
    kelvin = (fahrenheit + 459.67) / 1.8;
    Console.Write(fahrenheit + " degrees Fahrenheit = " + kelvin + " degrees Kelvin");
}
else if (choice == 3) {
    Console.Write("Enter a temperature in degrees Fahrenheit: ");
    fahrenheit = Convert.ToDouble(Console.ReadLine());
    celsius = 5.0 / 9.0 * (fahrenheit - 32);
    Console.Write(fahrenheit + " degrees Fahrenheit = " + celsius + " degrees Celsius");
}
else if (choice == 4) {
    Console.Write("Enter a temperature in degrees Celsius: ");
    celsius = Convert.ToDouble(Console.ReadLine());
    fahrenheit = 9.0 / 5.0 * celsius + 32;
```

```
    Console.Write(celsius + " degrees Celsius = " + " degrees Fahrenheit");
}
else {
    Console.WriteLine("Invalid choice!");
}
```

# Chapter 15

## 15.2 Review Questions: True/False

1. true
2. true
3. false

## 15.3 Review Exercises

### 1. Solution

i. 25 6

ii. 10 9

iii. 50 2

### 2. Solution

**First approach**

```
int age;

Console.Write("Enter your age: ");
age = Convert.ToInt32(Console.ReadLine());

if (age < 0) {
    Console.WriteLine("Error: Invalid age!");
}
else {
    if (age < 16) {
        Console.WriteLine("You cannot drive either a small scooter or a car");
    }
    else {
        if (age < 18) {
            Console.WriteLine("You can drive a small scooter");
        }
        else {
            Console.WriteLine("You can drive a car and a small scooter");
        }
    }
}
```

**Second approach**

```
int age;

Console.Write("Enter your age: ");
age = Convert.ToInt32(Console.ReadLine());
```

```
if (age < 0) {
    Console.WriteLine("Error: Invalid age!");
}
else {
    if (age < 16) {
        Console.WriteLine("You cannot drive either a small scooter or a car");
    }
    else if (age < 18) {
        Console.WriteLine("You can drive a small scooter");
    }
    else {
        Console.WriteLine("You can drive a car and a small scooter");
    }
}
```

### Third approach

```
int age;

Console.Write("Enter your age: ");
age = Convert.ToInt32(Console.ReadLine());

if (age < 0) {
    Console.WriteLine("Error: Invalid age!");
}
else if (age < 16) {
    Console.WriteLine("You cannot drive either a small scooter or a car");
}
else if (age < 18) {
    Console.WriteLine("You can drive a small scooter");
}
else {
    Console.WriteLine("You can drive a car and a small scooter");
}
```

## 3. Solution

### First approach

```
double t, w;

Console.Write("Enter temperature (in Fahrenheit): ");
t = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter wind speed (in miles/hour): ");
w = Convert.ToDouble(Console.ReadLine());

if (t > 75) {
    if (w > 12) {
        Console.WriteLine("The day is hot and windy");
    }
```

```
    else {
        Console.WriteLine("The day is hot and not windy");
    }
}
else {
    if (w > 12) {
        Console.WriteLine("The day is cold and windy");
    }
    else {
        Console.WriteLine("The day is cold and not windy");
    }
}
```

Second approach

```
double t, w;
string message1, message2;

Console.Write("Enter temperature (in Fahrenheit): ");
t = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter wind speed (in miles/hour): ");
w = Convert.ToDouble(Console.ReadLine());

if (t > 75) {
    message1 = "hot";
}
else {
    message1 = "cold";
}

if (w > 12) {
    message2 = "windy";
}
else {
    message2 = "not windy";
}

Console.WriteLine("The day is " + message1 + " and " + message2);
```

## 4. Solution

```
int a, w, h;
double bmi;

Console.Write("Enter age: ");
a = Convert.ToInt32(Console.ReadLine());
if (a < 18) {
    Console.WriteLine("Invalid age");
}
else {
```

```
    Console.Write("Enter weight in pounds: ");
    w = Convert.ToInt32(Console.ReadLine());
    Console.Write("Enter height in inches: ");
    h = Convert.ToInt32(Console.ReadLine());

    bmi = w * 703.0 / Math.Pow(h, 2);

    if (bmi < 15) {
        Console.WriteLine("Very severely underweight");
    }
    else if (bmi < 16) {
        Console.WriteLine("Severely underweight");
    }
    else if (bmi < 18.5) {
        Console.WriteLine("Underweight");
    }
    else if (bmi < 25) {
        Console.WriteLine("Normal");
    }
    else if (bmi < 30) {
        Console.WriteLine("Overweight");
    }
    else if (bmi < 35) {
        Console.WriteLine("Severely overweight");
    }
    else {
        Console.WriteLine("Very severely overweight");
    }
}
```

# Chapter 16

## 16.3 Review Questions: True/False

1. true
2. true
3. false
4. true

# Chapter 17

## 17.3 Review Questions: True/False

1. true
2. false
3. false
4. false
5. false
6. false
7. false
8. false
9. true
10. true
11. true

## 17.4 Review Questions: Multiple Choice

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

## 17.5 Review Exercises

### 1. Solution

```
double i = 30;
while (i > 5) {
  Console.WriteLine(i);
  i /= 2;
}
Console.WriteLine("The end");
```

### 2. Solution

```
int i = 3;
do {
    i--;
} while (i > 0);
Console.Write("The end");
```

### 3. Solution

Four

### 4. Solution

Zero

### 5. Solution

It displays

2

14

6

and performs three iterations

### 6. Solution

i. –1
ii. 9
iii. 0.5
iv. –7
v. A value between 17 and 32
vi. 1.4

## 7. Solution

i. –1
ii. 18
iii. 0.5
iv. –20
v. 128
vi. 11.25

## 8. Solution

i. 4
ii. –2
iii. 2
iv. 10

## 9. Solution

```
double x, total;
int i;

total = 0;

i = 1;
while (i <= 20) {
    Console.Write("Enter a number: ");
    x = Convert.ToDouble(Console.ReadLine());
    total += x;
    i += 1;
}
Console.WriteLine(total / 20);
```

## 10.Solution

```
int n, i;
double p, x;

Console.Write("Enter N: ");
n = Convert.ToInt32(Console.ReadLine());

p = 1;
i = 1;
while (i <= n) {
    Console.Write("Enter a number: ");
    x = Convert.ToDouble(Console.ReadLine());
    if (x > 0) {
        p *= x;
    }
    i += 1;
```

```
}
Console.WriteLine(p);
```

### 11.Solution

```
int i, x, total;

total = 0;

i = 1;
while (i <= 10) {
    Console.Write("Enter an integer: ");
    x = Convert.ToInt32(Console.ReadLine());
    if (x >= 100 && x <= 200) {
        total += x;
    }
    i += 1;
}
Console.WriteLine(total);
```

### 12.Solution

```
int i, x, total;

total = 0;
i = 1;
while (i <= 20) {
    Console.Write("Enter an integer: ");
    x = Convert.ToInt32(Console.ReadLine());
    if (x >= 100 && x <= 999) {
        total += x;
    }
    i += 1;
}
Console.WriteLine(total);
```

### 13.Solution

```
double x, p;

p = 1;
Console.Write("Enter a number: ");
x = Convert.ToDouble(Console.ReadLine());
while (x != 0) {
    p *= x;
    Console.Write("Enter a number: ");
    x = Convert.ToDouble(Console.ReadLine());
}
Console.WriteLine(p);
```

# Chapter 18

## 18.3 Review Questions: True/False

1. true
2. true
3. false
4. false
5. true
6. true
7. false
8. false
9. true

## 18.4 Review Questions: Multiple Choice

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

## 18.5 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. Any value greater than or equal to 2 and less than 2.5

iii. −7 (or −6)

iv. −1

### 4. Solution

```
int i;
double x, p, total;

p = 1;
total = 0;
for (i = 1; i <= 20; i++) {
    Console.Write("Enter a number: ");
    x = Convert.ToDouble(Console.ReadLine());
    p *= x;
    total += x;
```

```
}
Console.WriteLine(p + " " + total / 20);
```

### 5. Solution

```
int n, i, count, x;

Console.Write("Enter N: ");
n = Convert.ToInt32(Console.ReadLine());

count = 0;
for (i = 1; i <= n; i++) {
    Console.Write("Enter an integer: ");
    x = Convert.ToInt32(Console.ReadLine());
    if (x > 0) {
        count += 1;
    }
}

if (count > 0) {
    Console.WriteLine(count);
}
else {
    Console.WriteLine("You entered no positive integers");
}
```

### 6. Solution

```
int i, start, finish;

Console.Write("Enter value for start: ");
start = Convert.ToInt32(Console.ReadLine());
Console.Write("Enter value for finish: ");
finish = Convert.ToInt32(Console.ReadLine());

for (i = start; i <= finish; i++) {
    Console.WriteLine(i);
}
```

### 7. Solution

```
int exp, i;
double p, b;

Console.Write("Enter a value for base: ");
b = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter an integer for exponent: ");
exp = Convert.ToInt32(Console.ReadLine());

p = 1;
for (i = 1; i <= exp; i++) {
```

```
    p *= b;
}

Console.WriteLine(p);
```

### 8. Solution

**First approach**

```
string msg;
int count, words;

Console.Write("Enter a message: ");
msg = Console.ReadLine();

count = 0;
foreach (var ch in msg) {
    if (ch.ToString() == " ") {  //Equivallent to: ch == ' '
        count += 1;
    }
}
words = count + 1;

Console.WriteLine("The message entered contains " + words + " words");
```

**Second approach**

```
string msg;
int i, count, words;

Console.Write("Enter a message: ");
msg = Console.ReadLine();

count = 0;
for (i = 0; i <= msg.Length - 1; i++) {
    if (msg[i].ToString() == " ") {  //Equivallent to: msg[i] == ' '
        count += 1;
    }
}
words = count + 1;

Console.WriteLine("The message entered contains " + words + " words");
```

# Chapter 19

## 19.2 Review Questions: True/False

1. true
2. false
3. true
4. false
5. true
6. true

## 19.3 Review Questions: Multiple Choice

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

## 19.4 Review Exercises

### 1. Solution

i. 10

ii. A value greater than or equal to 4.5 and less than 5.0

iii. −7 (or −8)

iv. 138 (or 139)

### 2. Solution

```
int hour, minutes;

for (hour = 0; hour <= 23; hour++) {
    for (minutes = 0; minutes <= 59; minutes++) {
        Console.WriteLine(hour + "\t" + minutes);
    }
}
```

### 3. Solution

```
int i, j;

for (i = 5; i >= 1; i -= 1) {
    for (j = 1; j <= i; j++) {
        Console.Write(i + " ");
    }
    Console.WriteLine();
}
```

### 4. Solution

```
int i, j;

for (i = 0; i <= 5; i++) {
    for (j = 0; j <= i; j++) {
        Console.Write(j + " ");
    }
    Console.WriteLine();
}
```

### 5. Solution

**First approach (The amateur way!!!)**

```
Console.WriteLine("*  *  *  *  *  *  *  *  *  *");
Console.WriteLine("*  *  *  *  *  *  *  *  *  *");
```

```
Console.WriteLine("*  *  *  *  *  *  *  *  *  *");
Console.WriteLine("*  *  *  *  *  *  *  *  *  *");
```

**Second approach**

```
int i, j;

for (i = 1; i <= 4; i++) {
    for (j = 1; j <= 10; j++) {
        Console.Write("*  ");
    }
    Console.WriteLine();
}
```

## 6. Solution

```
int y, i, j;

Console.Write("Enter an integer between 3 and 20: ");
y = Convert.ToInt32(Console.ReadLine());

for (i = 1; i <= y; i++) {
    for (j = 1; j <= y; j++) {
        Console.Write("*  ");
    }
    Console.WriteLine();
}
```

## 7. Solution

```
int i, j;

for (i = 1; i <= 5; i++) {
    for (j = 1; j <= i; j++) {
        Console.Write("*  ");
    }
    Console.WriteLine();
}
```

## 8. Solution

```
int i, j;

for (i = 1; i <= 5; i++) {
    for (j = 1; j <= i; j++) {
        Console.Write("*  ");
    }
    Console.WriteLine();
}

for (i = 4; i >= 1; i -= 1) {
    for (j = 1; j <= i; j++) {
        Console.Write("*  ");
```

```
    }
    Console.WriteLine();
}
```

# Chapter 20

## 20.7 Review Questions: True/False

1. false
2. false
3. false
4. false
5. true
6. true
7. false
8. false

## 20.8 Review Questions: Multiple Choice

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

## 20.9 Review Exercises

### 1. Solution

```
count_names = 0;
count_not_johns = 0;
name = "";
Console.Write("Enter a name: ");
name = Console.ReadLine();
while (name != "STOP") {
    Console.Write("Enter a name: ");
    name = Console.ReadLine();
    count_names++;
    if (name != "John") {
        count_not_johns++;
    }
    Console.Write("Enter a name: ");
    name = Console.ReadLine();
}
Console.WriteLine("Total names entered: " + count_names);
Console.WriteLine("Names other than John entered: " + count_not_johns);
```

### 2. Solution

```
string text;
bool found;

Console.Write("Enter a text: ");
text = Console.ReadLine();

found = false;
foreach (var character in text) {
```

```
    if (character.ToString() == " ") {  //Equivallently: character == ' '
        found = true;
        break;
    }
}

if (found == false) {
    Console.WriteLine("One Single Word");
}
else {
    Console.WriteLine("Complete Sentence");
}
```

## 3. Solution

### First approach

```
string sentence;

bool found;
Console.Write("Enter a sentence: ");
sentence = Console.ReadLine();

found = false;
foreach (var character in sentence) {
    if ("0123456789".IndexOf(character) > -1) {
        found = true;
        break;
    }
}

if (found == true) {
    Console.WriteLine("The sentence contains a number");
}
```

### Second approach

```
string sentence;

bool found;
Console.Write("Enter a sentence: ");
sentence = Console.ReadLine();

found = false;
foreach (var digit in "0123456789") {
    if (sentence.IndexOf(digit) > -1) {
        found = true;
        break;
    }
}
```

```
if (found == true) {
    Console.WriteLine("The sentence contains a number");
}
```

### 4. Solution

```
Console.WriteLine("Printing all integers from 1 to 100");
i = 1;
while (i < 101) {
    Console.WriteLine(i);
    i++;
}
```

### 5. Solution

```
Console.WriteLine("Printing odd integers from 1 to 99");
i = 1;
while (!(i > 100)) {
    Console.WriteLine(i);
    i += 2;
}
```

### 6. Solution

```
int i, j;

for (i = 1; i <= 4; i++) {
    for (j = 1; j <= 4; j++) {
        Console.WriteLine(i + " x " + j + " = " + (i * j));
    }
}
```

### 7. Solution

```
int i, j;

Console.Write("\t|\t");
for (i = 1; i <= 12; i++) {
    Console.Write(i + "\t");
}
Console.WriteLine();

for (i = 1; i <= 12; i++) {
    Console.Write("---------");
}
Console.WriteLine();

for (i = 1; i <= 12; i++) {
    Console.Write(i + "\t|\t");
    for (j = 1; j <= 12; j++) {
        Console.Write(i * j + "\t");
    }
```

```
    Console.WriteLine();
}
```

### 8. Solution

```
int i, j, n;

Console.Write("Enter an integer: ");
n = Convert.ToInt32(Console.ReadLine());

Console.Write("\t|\t");
for (i = 1; i <= n; i++) {
    Console.Write(i + "\t");
}
Console.WriteLine();

for (i = 1; i <= n; i++) {
    Console.Write("---------");
}
Console.WriteLine();

for (i = 1; i <= n; i++) {
    Console.Write(i + "\t|\t");
    for (j = 1; j <= n; j++) {
        Console.Write(i * j + "\t");
    }
    Console.WriteLine();
}
```

# Chapter 21

## 21.2 Review Exercises

### 1. Solution

```
int i, total;

total = 0;
for (i = 1; i <= 99; i += 2) {
    total += i;
}

Console.WriteLine(total);
```

### 2. Solution

```
int n, total, i;

Console.Write("Enter N: ");
n = Convert.ToInt32(Console.ReadLine());
```

```
total = 0;
for (i = 2; i <= 2 * n; i += 2) {
    total += i;
}

Console.WriteLine(total);
```

### 3. Solution

```
int count_pos, count_neg, total_pos, total_neg, i, x;

count_pos = 0;
count_neg = 0;
total_pos = 0;
total_neg = 0;

for (i = 1; i <= 50; i++) {
    Console.Write("Enter an integer: ");
    x = Convert.ToInt32(Console.ReadLine());
    if (x > 0) {
        count_pos += 1;
        total_pos += x;
    }
    else if (x < 0) {
        count_neg += 1;
        total_neg += x;
    }
}

if (count_pos > 0) {
    Console.WriteLine(total_pos / (double)count_pos);
}

if (count_neg > 0) {
    Console.WriteLine(total_neg / (double)count_neg);
}
```

### 4. Solution

```
int n, i, grade, total, count;

Console.Write("Enter total number of students: ");
n = Convert.ToInt32(Console.ReadLine());

total = 0;
count = 0;
for (i = 1; i <= n; i++) {
    Console.Write("Enter grade: ");
    grade = Convert.ToInt32(Console.ReadLine());
```

```
    if (grade >= 90 && grade <= 100) {
        total += grade;
        count += 1;
    }
}

if (count > 0) {
    Console.WriteLine(total / (double)count);
}
else {
    Console.WriteLine("There are no students that got an A");
}
```

### 5. Solution

```
int count;
double total, x;

total = 0;
count = 0;
do {
    Console.Write("Enter a number: ");
    x = Convert.ToDouble(Console.ReadLine());
    if (x == 0) {
        count += 1;
    }
    total += x;
} while (total <= 3000);

Console.WriteLine(count);
```

### 6. Solution

```
string answer;
double r, area;

do {
    Console.Write("Enter the length of a radius of a circle`: ");
    r = Convert.ToDouble(Console.ReadLine());

    area = 3.141 * Math.Pow(r, 2);
    Console.WriteLine("The area is: " + area);

    Console.Write("Would you like to repeat? ");
    answer = Console.ReadLine();
} while (answer.ToUpper() == "YES");
```

### 7. Solution

```
long x;
```

```
x = 1;
while (x <= 1073741824) {
    Console.WriteLine(x);
    x *= 2;
}
```

## 8. Solution

```
int i;

for (i = 1; i <= 100; i++) {
    Console.WriteLine(-i + "\n " + i);
}
```

## 9. Solution

### First approach

```
int i;
double offset, value;

value = 0;
for (i = 0; i <= 7; i++) {
    offset = Math.Pow(10, i);
    value += offset;
    Console.WriteLine(value);
}
```

### Second approach

```
string value;
int i;

value = "1";
for (i = 0; i <= 7; i++) {
    Console.WriteLine(value);
    value += "1";
}
```

## 10.Solution

```
double t, maximum, total;
int i;

Console.Write("Enter temperature for day 1: ");
t = Convert.ToDouble(Console.ReadLine());
maximum = t;
total = t;
for (i = 2; i <= 31; i++) {
    Console.Write("Enter temperature for day " + i + ": ");
    t = Convert.ToDouble(Console.ReadLine());

    total += t;
    if (t > maximum) {
```

```
        maximum = t;
    }
}
Console.WriteLine(total / 31 + " " + maximum);
```

## 11.Solution

```
double level, maximum, minimum;
int hour, min_hour, max_hour, i;

Console.Write("Enter level: ");
level = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter hour: ");
hour = Convert.ToInt32(Console.ReadLine());

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

for (i = 2; i <= 24; i++) {
    Console.Write("Enter level: ");
    level = Convert.ToDouble(Console.ReadLine());
    Console.Write("Enter hour: ");
    hour = Convert.ToInt32(Console.ReadLine());

    if (level > maximum) {
        maximum = level;
        max_hour = hour;
    }

    if (level < minimum) {
        minimum = level;
        min_hour = hour;
    }
}
Console.WriteLine(maximum + " " + max_hour + " " + minimum + " " + min_hour);
```

## 12.Solution

```
int attempts, attempts_1st_player  = 0, attempts_2nd_player = 0;
int guess, i, secret_number;

Random rnd = new();

for (i = 1; i <= 2; i++) {
    secret_number = rnd.Next(1, 101);

    attempts = 1;
    Console.Write("Enter a guess: ");
```

```
    guess = Convert.ToInt32(Console.ReadLine());
    while (guess != secret_number) {
        if (guess > secret_number) {
            Console.Write("Your guess is bigger than my secret number. Try again.");
        }
        else {
            Console.Write("Your guess is smaller than my secret number. Try again.");
        }
        attempts++;
        Console.Write("Enter a guess: ");
        guess = Convert.ToInt32(Console.ReadLine());
    }

    Console.WriteLine("You found it!");
    Console.WriteLine("Attempts: " + attempts);

    if (i == 0) {
        attempts_1st_player = attempts;
    }
    else {
        attempts_2nd_player = attempts;
    }
}

if (attempts_1st_player < attempts_2nd_player) {
    Console.WriteLine("First Player Wins!");
}
else if (attempts_1st_player > attempts_2nd_player) {
    Console.WriteLine("Second Player Wins!");
}
else {
    Console.WriteLine("It's a draw");
}
```

## 13.Solution

```
string gender;
int n, i, grade, total, total_a, count_a, total_b, count_b;
int total_a_boys, count_a_boys, count_cdef_girls;

Console.Write("Enter total number of students: ");
n = Convert.ToInt32(Console.ReadLine());

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 = 1; i <= n; i++) {
    Console.Write("Enter grade for student No " + i + ": ");
    grade = Convert.ToInt32(Console.ReadLine());

    Console.Write("Enter gender for student No " + i + ": ");
    gender = Console.ReadLine();

    if (grade >= 90 && grade <= 100) {
        total_a += grade;
        count_a += 1;
        if (gender == "M") {
            total_a_boys += grade;
            count_a_boys += 1;
        }
    }
    else if (grade >= 80 && grade <= 89) {
        total_b += grade;
        count_b += 1;
    }
    else {
        if (gender == "F") {
            count_cdef_girls += 1;
        }
    }
    total += grade;
}

if (count_a > 0) {
    Console.Write("Average value of those who got an 'A': ");
    Console.WriteLine(total_a / (double)count_a);
}

if (count_b > 0) {
    Console.Write("Average value of those who got a 'B': ");
    Console.WriteLine(total_b / (double)count_b);
}

if (count_a_boys > 0) {
    Console.Write("Average value of boys who got an 'A': ");
    Console.WriteLine(total_a_boys / (double)count_a_boys);
}

Console.Write("Total number of girls that got less than 'B': ");
```

```
Console.WriteLine(count_cdef_girls);

Console.Write("Average grade of the whole class: ");
Console.WriteLine(total / (double)n);
```

### *14.Solution*

```
string answer;
double amount, discount;

do {
    Console.Write("Enter amount: ");
    amount = Convert.ToDouble(Console.ReadLine());

    if (amount < 20) {
        discount = 0;
    }
    else if (amount < 50) {
        discount = 3;
    }
    else if (amount < 100) {
        discount = 5;
    }
    else {
        discount = 10;
    }

    Console.WriteLine("Discount: " + discount + "%");

    Console.Write("Would you like to repeat? ");
    answer = Console.ReadLine();
} while (answer.ToUpper() == "YES");
```

# Chapter 22

## 22.12 Review Exercises

### *1. Solution*

```
private void Form1_Shown(object sender, EventArgs e) {
    Turtle.Forward(150);
    Turtle.RotateTo(-130);
    Turtle.Forward(50);

    Turtle.X = 0;
    Turtle.Y = 150;

    Turtle.RotateTo(130);
```

```
    Turtle.Forward(50);
}
```

## 2. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    Turtle.Rotate(20);
    Turtle.Forward(100);
    Turtle.Rotate(90 - 20);
    Turtle.Forward(200);
    Turtle.Rotate(90 + 20);
    Turtle.Forward(100);
    Turtle.Rotate(90 - 20);
    Turtle.Forward(200);
}
```

## 3. Solution

### First approach

```
private void Form1_Shown(object sender, EventArgs e) {
    Turtle.Rotate(30);
    Turtle.Forward(100);
    Turtle.Rotate(120);
    Turtle.Forward(100);
    Turtle.Rotate(60);
    Turtle.Forward(100);
    Turtle.Rotate(120);
    Turtle.Forward(100);
}
```

### Second approach

```
private void Form1_Shown(object sender, EventArgs e) {
    Turtle.RotateTo(30);
    Turtle.Forward(100);
    Turtle.RotateTo(150);
    Turtle.Forward(100);
    Turtle.RotateTo(210);
    Turtle.Forward(100);
    Turtle.RotateTo(330);
    Turtle.Forward(100);
}
```

## 4. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    Turtle.Rotate(45);
    Turtle.Forward(141);
    Turtle.Rotate(45);
    Turtle.Forward(100);
    Turtle.Rotate(45);
```

```
    Turtle.Forward(141);
    Turtle.Rotate(45 + 90);
    Turtle.Forward(300);
}
```

### 5. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int k, i;

    Turtle.PenSize = 2;
    for (k = 1; k <= 2; k++) {
        for (i = 1; i <= 4; i++) {
            Turtle.Forward(100);
            Turtle.Rotate(90);
        }
        Turtle.X -= 100;
    }
}
```

### 6. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int m, n, i;

    for (m = 1; m <= 2; m++) {
        for (n = 1; n <= 2; n++) {
            for (i = 1; i <= 4; i++) {
                Turtle.Forward(100);
                Turtle.Rotate(90);
            }
            Turtle.X -= 150;
        }
        Turtle.X = 0;
        Turtle.Y -= 150;
    }
}
```

### 7. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int size, length, height;

    size = Convert.ToInt32(Interaction.InputBox("Enter pen size:"));
    length = Convert.ToInt32(Interaction.InputBox("Enter length:"));
    height = Convert.ToInt32(Interaction.InputBox("Enter heigth:"));

    Turtle.PenSize = size;
    Turtle.Forward(height);
    Turtle.Rotate(90);
```

```
    Turtle.Forward(length);
    Turtle.Rotate(90);
    Turtle.Forward(height);
    Turtle.Rotate(90);
    Turtle.Forward(length);
}
```

## 8. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int length = Convert.ToInt32(Interaction.InputBox("Enter length of the side:"));

    Turtle.RotateTo(90);

    Turtle.Forward(length);
    Turtle.Rotate(-120);
    Turtle.Forward(length);
    Turtle.Rotate(-120);
    Turtle.Forward(length);
    Turtle.Rotate(-120);
    Turtle.Forward(length);
}
```

## 9. Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int i;

    for (i = 1; i <= 12; i++) {
        Turtle.Forward(100);
        Turtle.X = 0;
        Turtle.Y = 0;
        Turtle.Rotate(30);
    }
}
```

## 10.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int k, i;

    Turtle.PenSize = 2;
    Turtle.RotateTo(90);
    for (k = 1; k <= 180; k += 60) {
        for (i = 1; i <= 5; i++) {
            Turtle.Forward(150 + k);
            Turtle.Rotate((float)180 / 5 * 4);
        }
        Turtle.X -= 30;
        Turtle.Y += 10;
```

```
    }
}
```

### 11.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int k, i;
    Turtle.PenSize = 2;
    for (k = 1; k <= 3; k++) {
        for (i = 1; i <= 4; i++) {
            Turtle.Forward(100);
            Turtle.Rotate(90);
        }
        Turtle.Rotate(-30);
    }
}
```

### 12.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int k, i;
    Turtle.PenSize = 2;
    for (k = 1; k <= 12; k++) {
        for (i = 1; i <= 4; i++) {
            Turtle.Forward(100);
            Turtle.Rotate(90);
        }
        Turtle.Rotate(-30);
    }
}
```

### 13.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int k, i;
    Turtle.PenSize = 2;
    for (k = 1; k <= 8; k++) {
        for (i = 1; i <= 4; i++) {
            Turtle.Forward(100);
            Turtle.Rotate(90);
        }
        Turtle.Rotate(-45);
    }
}
```

### 14.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int i, k;

    Turtle.PenSize = 1;
```

```
Turtle.PenColor = Color.Blue;

//Draw a blue rectangle
Turtle.Forward(100);
Turtle.Rotate(90);
Turtle.Forward(200);
Turtle.Rotate(90);
Turtle.Forward(100);
Turtle.Rotate(90);
Turtle.Forward(200);

//Move the turtle to the top left corner of the rectangle
Turtle.Rotate(90);
Turtle.PenUp();
Turtle.Forward(100);
Turtle.PenDown();

//Draw the red roof
Turtle.RotateTo(45);
Turtle.PenColor = Color.Red;
Turtle.Forward(141);
Turtle.Rotate(90);
Turtle.Forward(141);

//Draw the windows
Turtle.PenColor = Color.Brown;
Turtle.X -= 180;
Turtle.Y = 50;

for (k = 0; k <= 1; k++) {
    Turtle.RotateTo(0);

    for (i = 0; i <= 3; i++) {
        Turtle.Forward(40);
        Turtle.Rotate(90);
    }

    Turtle.RotateTo(0);
    Turtle.X += 20;
    Turtle.Forward(40);

    Turtle.RotateTo(90);
    Turtle.X -= 20;
    Turtle.Y = 70;
    Turtle.Forward(40);

    Turtle.X += 80;
```

```
        Turtle.Y -= 20;
    }

    //Draw the door
    Turtle.X -= 180;
    Turtle.Y = 0;
    Turtle.RotateTo(0);

    Turtle.Forward(70);
    Turtle.Rotate(90);
    Turtle.Forward(40);
    Turtle.Rotate(90);
    Turtle.Forward(70);
    Turtle.Rotate(90);
    Turtle.Forward(40);
}
```

### 15.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int m;
    //Move to poll position
    Turtle.X = -300;

    for (m = 1; m <= 3; m++) {
        //Here goes the code of the previous exersise

        Turtle.RotateTo(0);
        Turtle.X += 200;
    }
}
```

The final program becomes

```
private void Form1_Shown(object sender, EventArgs e) {
    int m;
    //Move to poll position
    Turtle.X = -300;

    for (m = 1; m <= 3; m++) {
        int i, k;

        Turtle.PenSize = 1;
        Turtle.PenColor = Color.Blue;

        //Draw a blue rectangle
        Turtle.Forward(100);
        Turtle.Rotate(90);
        Turtle.Forward(200);
        Turtle.Rotate(90);
```

```
Turtle.Forward(100);
Turtle.Rotate(90);
Turtle.Forward(200);

//Move the turtle to the top left corner of the rectangle
Turtle.Rotate(90);
Turtle.PenUp();
Turtle.Forward(100);
Turtle.PenDown();

//Draw the red roof
Turtle.RotateTo(45);
Turtle.PenColor = Color.Red;
Turtle.Forward(141);
Turtle.Rotate(90);
Turtle.Forward(141);

//Draw the windows
Turtle.PenColor = Color.Brown;
Turtle.X -= 180;
Turtle.Y = 50;

for (k = 0; k <= 1; k++) {
    Turtle.RotateTo(0);

    for (i = 0; i <= 3; i++) {
        Turtle.Forward(40);
        Turtle.Rotate(90);
    }

    Turtle.RotateTo(0);
    Turtle.X += 20;
    Turtle.Forward(40);

    Turtle.RotateTo(90);
    Turtle.X -= 20;
    Turtle.Y = 70;
    Turtle.Forward(40);

    Turtle.X += 80;
    Turtle.Y -= 20;
}

//Draw the door
Turtle.X -= 180;
Turtle.Y = 0;
Turtle.RotateTo(0);
```

```
        Turtle.Forward(70);
        Turtle.Rotate(90);
        Turtle.Forward(40);
        Turtle.Rotate(90);
        Turtle.Forward(70);
        Turtle.Rotate(90);
        Turtle.Forward(40);

        Turtle.RotateTo(0);
        Turtle.X += 200;
    }
}
```

### 16.Solution

```
private void Form1_Shown(object sender, EventArgs e) {
    int i, j, sides, length;
    Random rnd = new();

    Turtle.PenSize = 2;

    for (i = 1; i <= 10; i++) {
        //Pick random x, y values and move the turtle to that position
        Turtle.X = rnd.Next(-200, 201);
        Turtle.Y = rnd.Next(-200, 201);

        sides = rnd.Next(5, 10);     //Pick random number of sides
        length = rnd.Next(20, 50);   //Pick random length of sides

        //Draw the polygon
        for (j = 1; j <= sides; j++) {
            Turtle.Forward(length);
            Turtle.Rotate((float)360 / sides);
        }
    }
}
```

# Chapter 23

## 23.14 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

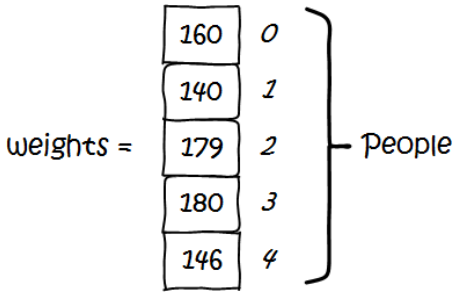
17. false
18. true
19. false
20. true
21. false
22. false
23. true
24. false
25. true
26. true
27. false
28. false
29. true
30. true
31. true
32. false
33. false
34. true
35. true
36. false
37. false
38. true
39. false
40. true
41. true
42. true
43. true
44. true
45. true
46. true
47. false

## 23.15 Review Questions: Multiple Choice

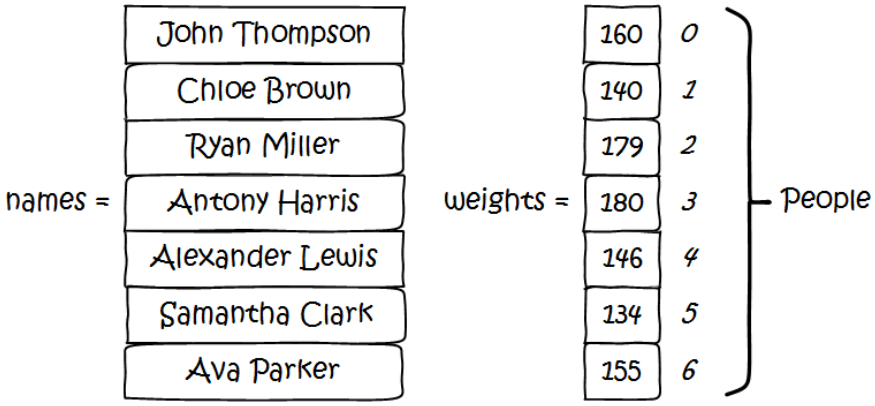
1. b
2. c
3. b
4. d
5. d
6. d
7. c
8. a
9. b
10. a
11. b
12. b
13. a
14. a
15. a
16. c
17. a
18. a
19. b
20. a

## 23.16 Review Exercises

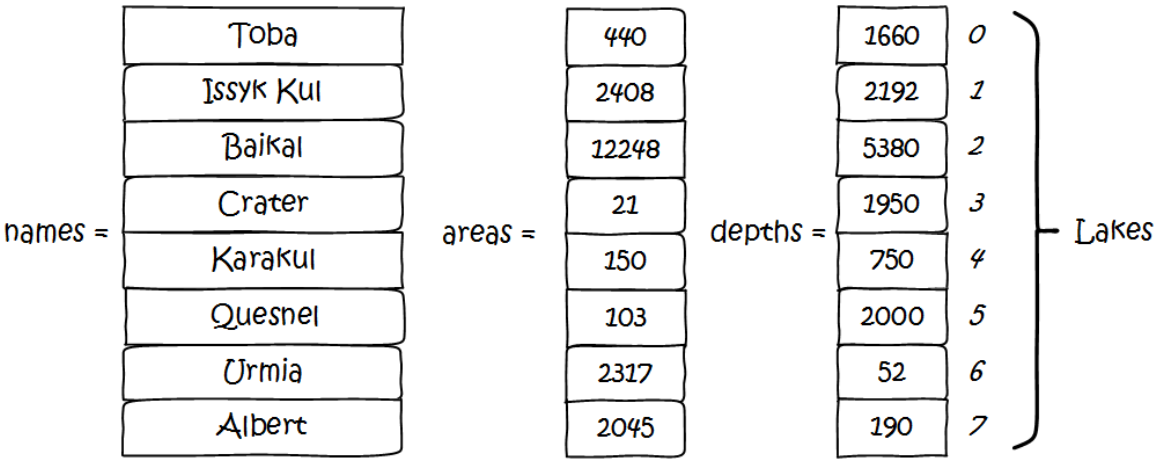
### 1. Solution



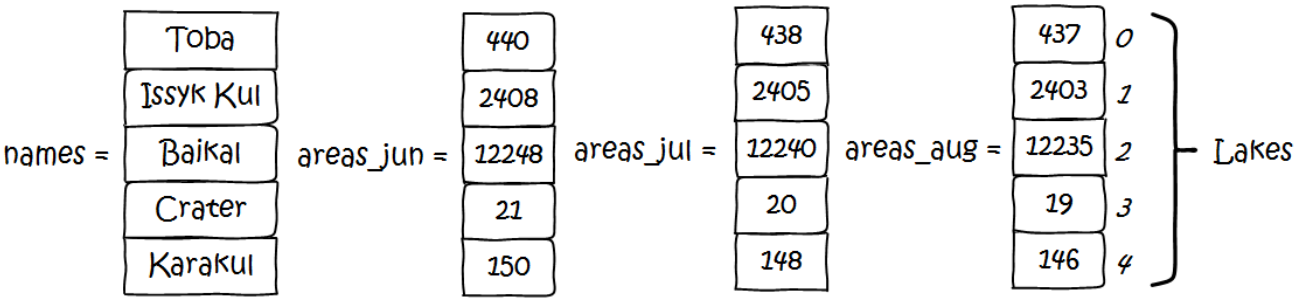

### 2. Solution



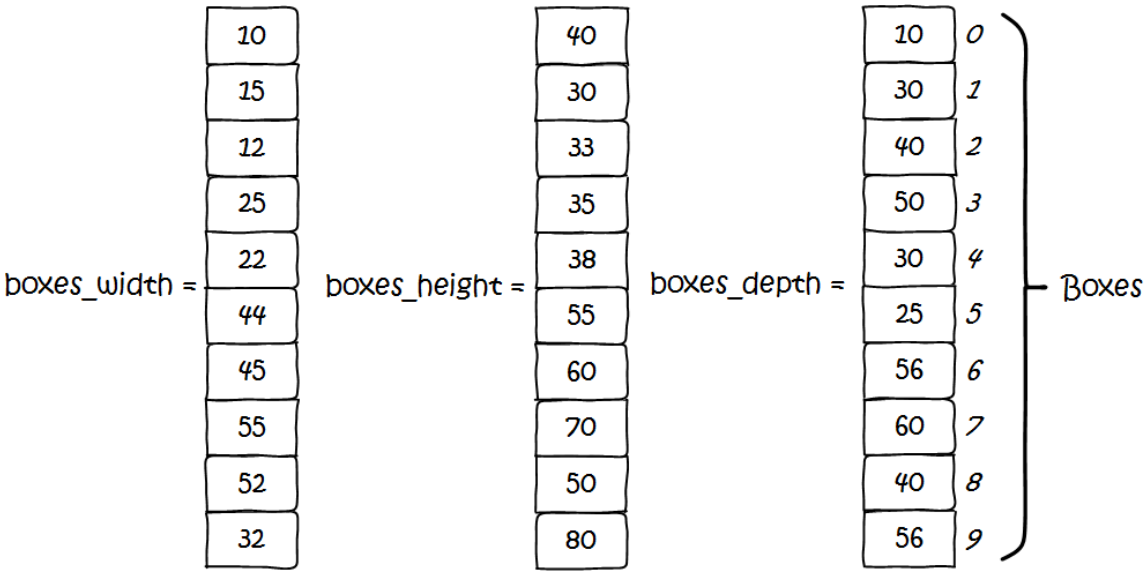
## 3. Solution




## 4. Solution




## 5. Solution




## 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

```
const int ELEMENTS = 100;

int i;
double[] a = new double[ELEMENTS];

for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.WriteLine(Math.Pow(a[i], 3));
}
```

## 12.Solution

```
const int ELEMENTS = 80;

int i;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

for (i = 0; i <= ELEMENTS - 1; i++) {
    a[i] = Math.Pow(a[i], 2);
}

for (i = ELEMENTS - 1; i >= 0; i -= 1) {
    Console.WriteLine(a[i]);
}
```

## 13.Solution

```
const int ELEMENTS = 50;

int i;

int[] a = new int[ELEMENTS];
```

```
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter an integer: ");
    a[i] = Convert.ToInt32(Console.ReadLine());
}

foreach (var element in a) {
    if (element >= 10) {
        Console.WriteLine(element);
    }
}
```

### 14.Solution

```
const int ELEMENTS = 30;
int i;
double total;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

total = 0;
foreach (var element in a) {
    if (element > 0) {
        total += element;
    }
}
Console.WriteLine(total);
```

### 15.Solution

```
const int ELEMENTS = 50;

int i, total;

int[] a = new int[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter an integer: ");
    a[i] = Convert.ToInt32(Console.ReadLine());
}

total = 0;
foreach (var element in a) {
    if (element >= 10 && element <= 99) {
        total += element;
    }
}
```

```
Console.WriteLine(total);
```

### 16.Solution

```
const int ELEMENTS = 40;

int i;
double total_pos, total_neg;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

total_pos = 0;
total_neg = 0;
foreach (var element in a) {
    if (element > 0) {
        total_pos += element;
    }
    else if (element < 0) {
        total_neg += element;
    }
}

Console.WriteLine(total_pos + " " + total_neg);
```

### 17.Solution

```
const int ELEMENTS = 20;

int i;
double total;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

total = 0;
foreach (var element in a) {
    total += element;
}

Console.WriteLine(total / ELEMENTS);
```

### 18.Solution

```
const int ELEMENTS = 50;
```

```
int i;

int[] a = new int[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter an integer: ");
    a[i] = Convert.ToInt32(Console.ReadLine());
}

for (i = 0; i <= ELEMENTS - 1; i++) {
    if (a[i] < 20) {
        Console.WriteLine(i);
    }
}
```

### 19.Solution

```
const int ELEMENTS = 60;

int i;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

for (i = 0; i <= ELEMENTS - 1; i += 2) {
    Console.WriteLine(a[i]);
}
```

### 20.Solution

```
const int ELEMENTS = 20;

int i;
double total;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

total = 0;
for (i = 0; i <= ELEMENTS - 1; i += 2) {
    total += a[i];
}

Console.WriteLine(total);
```

## 21.Solution

### First approach

```
int i;
int[] a = new int[100];
for (i = 0; i <= 99; i++) {
    a[i] = i + 1;
}
```

### Second approach

```
int i;
int[] a = new int[100];
for (i = 1; i <= 100; i++) {
    a[i - 1] = i;
}
```

## 22.Solution

### First approach

```
int i;
int[] a = new int[100];
for (i = 0; i <= 99; i++) {
    a[i] = 2 * (i + 1);
}
```

### Second approach

```
int i;
int[] a = new int[100];
for (i = 1; i <= 100; i++) {
    a[i - 1] = 2 * i;
}
```

## 23.Solution

```
int i, n;

Console.Write("Enter N: ");
n = Convert.ToInt32(Console.ReadLine());

double[] a = new double[n];
for (i = 1; i <= n; i++) {
    a[i - 1] = Math.Pow(i, 2);
}

foreach (var element in a) {
    Console.WriteLine(element);
}
```

## 24.Solution

```
const int ELEMENTS = 10;

int i;
```

```
double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

for (i = 0; i <= ELEMENTS - 1; i++) {
    if (a[i] == (int)a[i]) {
        Console.WriteLine(i);
    }
}
```

### 25.Solution

```
const int ELEMENTS = 50;

int i, count;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

count = 0;
for (i = 0; i <= ELEMENTS - 1; i++) {
    if (a[i] < 0) {
        count += 1;
    }
}

Console.WriteLine(count);
```

### 26.Solution

```
const int ELEMENTS = 50;

int i;

string[] words = new string[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a word: ");
    words[i] = Console.ReadLine();
}

foreach (var word in words) {
    if (word.Length >= 10) {
        Console.WriteLine(word);
    }
```

```
}
```

### 27.Solution

```
const int ELEMENTS = 30;

int i;

string[] words = new string[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a word: ");
    words[i] = Console.ReadLine();
}

int[] length_limits = { 5, 10, 20 };

foreach (var length_limit in length_limits) {
    foreach (var word in words) {
        if (word.Length < length_limit) {
            Console.WriteLine(word);
        }
    }
}
```

### 28.Solution

```
const int ELEMENTS = 40;

int i, count;

string[] words = new string[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("Enter a word: ");
    words[i] = Console.ReadLine();
}

foreach (var word in words) {
    count = 0;
    foreach (var letter in word) {
        if (letter.ToString() == "w") {   //Equivallent to: letter == 'w'
            count += 1;
        }

        if (count == 2) {
            Console.WriteLine(word);
            break;
        }
    }
}
```

# Chapter 24

## 24.5 Review Questions: True/False

1. false
2. false
3. true
4. true

## 24.6 Review Exercises

### 1. Solution

```
const int ELEMENTS_OF_A = 50;
const int ELEMENTS_OF_NEW = ELEMENTS_OF_A - 2;

int i;

double[] a = new double[ELEMENTS_OF_A];
for (i = 0; i <= ELEMENTS_OF_A - 1; i++) {
    Console.Write("Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

double[] new_arr = new double[ELEMENTS_OF_NEW];
for (i = 0; i <= ELEMENTS_OF_NEW - 1; i++) {
    new_arr[i] = (a[i] + a[i + 1] + a[i + 2]) / 3;
}

for (i = 0; i <= ELEMENTS_OF_NEW - 1; i++) {
    Console.WriteLine(new_arr[i]);
}
```

### 2. Solution

```
const int ELEMENTS = 15;

int i;
double minimum;

double[] a = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("A - Enter a number: ");
    a[i] = Convert.ToDouble(Console.ReadLine());
}

double[] b = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("B - Enter a number: ");
    b[i] = Convert.ToDouble(Console.ReadLine());
}
```

```
double[] c = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.Write("C - Enter a number: ");
    c[i] = Convert.ToDouble(Console.ReadLine());
}

double[] new_arr = new double[ELEMENTS];
for (i = 0; i <= ELEMENTS - 1; i++) {
    minimum = a[i];
    if (b[i] < minimum) {
        minimum = b[i];
    }
    if (c[i] < minimum) {
        minimum = c[i];
    }
    new_arr[i] = minimum;
}

for (i = 0; i <= ELEMENTS - 1; i++) {
    Console.WriteLine(new_arr[i]);
}
```

## 3. Solution

```
const int MOUNTAINS = 30;

int i, index_of_max, index_of_min;
double maximum, minimum;

string[] names = new string[MOUNTAINS];
double[] heights = new double[MOUNTAINS];
string[] countries = new string[MOUNTAINS];
for (i = 0; i <= MOUNTAINS - 1; i++) {
    names[i] = Console.ReadLine();
    heights[i] = Convert.ToDouble(Console.ReadLine());
    countries[i] = Console.ReadLine();
}

maximum = heights[0];
index_of_max = 0;
minimum = heights[0];
index_of_min = 0;
for (i = 1; i <= MOUNTAINS - 1; i++) {
    if (heights[i] > maximum) {
        maximum = heights[i];
        index_of_max = i;
    }
    if (heights[i] < minimum) {
```

```
        minimum = heights[i];
        index_of_min = i;
    }
}

Console.WriteLine(heights[index_of_max] + ", " + names[index_of_max] + ", " +
                  countries[index_of_max]);
Console.WriteLine(heights[index_of_min] + ", " + names[index_of_min] + ", " +
                  countries[index_of_min]);
```

## 4. Solution

```
const int CLASS1 = 20;
const int CLASS2 = 25;

int i;
string needle;
bool found;

Console.WriteLine("Class A");
string[] names1 = new string[CLASS1];
for (i = 0; i <= CLASS1 - 1; i++) {
    Console.Write("Enter name: ");
    names1[i] = Console.ReadLine();
}

Console.WriteLine("Class B");
string[] names2 = new string[CLASS2];
for (i = 0; i <= CLASS2 - 1; i++) {
    Console.Write("Enter name: ");
    names2[i] = Console.ReadLine();
}

Console.Write("Enter a name to search: ");
needle = Console.ReadLine();

found = false;
foreach (var name in names1) {
    if (name == needle) {
        found = true;
        break;
    }
}

if (found == true) {
    Console.WriteLine("Student found in class No 1");
}
else {
```

```
    found = false;
    foreach (var name in names2) {
        if (name == needle) {
            found = true;
            break;
        }
    }

    if (found == true) {
        Console.WriteLine("Student found in class No 2");
    }
    else {
        Console.WriteLine("Student not found in either class");
    }
}
```

## 5. Solution

```
Console.Write("Enter username: ");
usr = Console.ReadLine();
Console.Write("Enter password: ");
pwd = Console.ReadLine();

found = false;
for (i = 0; i <= 99; i++) {
    if (usernames[i] == usr) {
        found = true;
        break;
    }
}

if (found == true) {
    if (usernames[i] == usr && passwords[i] == pwd) {
        Console.WriteLine("Login OK!");
    }
    else {
        Console.WriteLine("Login Failed!");
    }
}
else {
    Console.WriteLine("Login Failed!");
}
```

## 6. Solution

```
Console.Write("Enter a value to search: ");
needle = Console.ReadLine();

found = false;
```

```
for (i = 0; i <= 999; i++) {
    if (SSNs[i] == needle) {
        found = true;
        Console.WriteLine(SSNs[i] + " " + names[i]);
        break;
    }
}

if (found == false) {
    for (i = 0; i <= 999; i++) {
        if (names[i] == needle) {
            found = true;
            Console.WriteLine(SSNs[i] + " " + names[i]);
        }
    }
}

if (found == false) {
    Console.WriteLine("This value does not exist");
}
```

## 7. Solution

```
const int STUDENTS = 12;

int i;
bool found;

int[] grades1 = new int[STUDENTS];
int[] grades2 = new int[STUDENTS];
int[] grades3 = new int[STUDENTS];
for (i = 0; i <= STUDENTS - 1; i++) {
    grades1[i] = Convert.ToInt32(Console.ReadLine());
    grades2[i] = Convert.ToInt32(Console.ReadLine());
    grades3[i] = Convert.ToInt32(Console.ReadLine());
}

found = false;
for (i = 0; i <= STUDENTS - 1; i++) {
    if ((grades1[i] + grades2[i] + grades3[i]) / 3.0 < 70) {
        found = true;
        break;
    }
}

if (found == true) {
    Console.Write("There is at least one student has an average value below 70");
}
```

## 8. Solution

```
const int STUDENTS = 15;

int i;
double average;

int[] grades1 = new int[STUDENTS];
int[] grades2 = new int[STUDENTS];
for (i = 0; i <= STUDENTS - 1; i++) {
    grades1[i] = Convert.ToInt32(Console.ReadLine());
    grades2[i] = Convert.ToInt32(Console.ReadLine());
}

for (i = 0; i <= STUDENTS - 1; i++) {
    Console.WriteLine("Student No " + (i + 1) + ": ");

    average = (grades1[i] + grades2[i]) / 2.0;

    if (average < 60) {
        Console.WriteLine("E/F");
    }
    else if (average < 70) {
        Console.WriteLine("D");
    }
    else if (average < 80) {
        Console.WriteLine("C");
    }
    else if (average < 90) {
        Console.WriteLine("B");
    }
    else {
        Console.WriteLine("A");
    }
}
```

## 9. Solution

```
const int PLAYERS = 15;

int i, total;

int[] points_match1 = new int[PLAYERS];
int[] points_match2 = new int[PLAYERS];
int[] points_match3 = new int[PLAYERS];
int[] points_match4 = new int[PLAYERS];
for (i = 0; i <= PLAYERS - 1; i++) {
    points_match1[i] = Convert.ToInt32(Console.ReadLine());
    points_match2[i] = Convert.ToInt32(Console.ReadLine());
```

```
    points_match3[i] = Convert.ToInt32(Console.ReadLine());
    points_match4[i] = Convert.ToInt32(Console.ReadLine());
}

for (i = 0; i <= PLAYERS - 1; i++) {
    Console.WriteLine("Player No " + (i + 1));
    total = points_match1[i] + points_match2[i] + points_match3[i] + points_match4[i];
    Console.WriteLine(total);
}
```

## 10.Solution

```
const int HOURS = 24;

int i;
double average;

double[] t_city1 = new double[HOURS];
double[] t_city2 = new double[HOURS];
double[] t_city3 = new double[HOURS];
for (i = 0; i <= HOURS - 1; i++) {
    t_city1[i] = Convert.ToDouble(Console.ReadLine());
    t_city2[i] = Convert.ToDouble(Console.ReadLine());
    t_city3[i] = Convert.ToDouble(Console.ReadLine());
}

for (i = 0; i <= HOURS - 1; i++) {
    average = (t_city1[i] + t_city2[i] + t_city3[i]) / 3;
    if (average < 10) {
        Console.WriteLine("Hour: " + (i + 1));
    }
}
```

## 11.Solution

```
const int STUDENTS = 12;

int i;

string[] names = new string[STUDENTS];
int[] grd_lesson1 = new int[STUDENTS];
int[] grd_lesson2 = new int[STUDENTS];
for (i = 0; i <= STUDENTS - 1; i++) {
    names[i] = Console.ReadLine();
    grd_lesson1[i] = Convert.ToInt32(Console.ReadLine());
    grd_lesson2[i] = Convert.ToInt32(Console.ReadLine());
}

//Create array average
double[] average = new double[STUDENTS];
```

```
for (i = 0; i <= STUDENTS - 1; i++) {
    average[i] = (grd_lesson1[i] + grd_lesson2[i]) / 2.0;
}

for (i = 0; i <= STUDENTS - 1; i++) {
    Console.WriteLine(names[i] + " " + average[i]);
}

for (i = 0; i <= STUDENTS - 1; i++) {
    if (average[i] < 60) {
        Console.WriteLine(names[i]);
    }
}

for (i = 0; i <= STUDENTS - 1; i++) {
    if (average[i] > 89) {
        Console.WriteLine(names[i] + " Bravo!");
    }
}
```

## 12.Solution

```
const int ARTISTS = 15;

int i, minimum;

string[] artist_names = new string[ARTISTS];
string[] song_titles = new string[ARTISTS];
int[] scoreA = new int[ARTISTS];
int[] scoreB = new int[ARTISTS];
int[] scoreC = new int[ARTISTS];

for (i = 0; i <= ARTISTS - 1; i++) {
    Console.Write("Name for artist No " + (i + 1) + ": ");
    artist_names[i] = Console.ReadLine();

    Console.Write("Song title for artist " + artist_names[i] + ": ");
    song_titles[i] = Console.ReadLine();

    Console.WriteLine("Score for artist " + artist_names[i]);

    Console.Write(" gotten from judge A: ");
    scoreA[i] = Convert.ToInt32(Console.ReadLine());

    Console.Write(" gotten from judge B: ");
    scoreB[i] = Convert.ToInt32(Console.ReadLine());

    Console.Write(" gotten from judge C: ");
    scoreC[i] = Convert.ToInt32(Console.ReadLine());
```

```
}

int[] total = new int[ARTISTS];
for (i = 0; i <= ARTISTS - 1; i++) {
    minimum = scoreA[i];
    if (scoreB[i] < minimum) {
        minimum = scoreB[i];
    }
    if (scoreC[i] < minimum) {
        minimum = scoreC[i];
    }

    total[i] = scoreA[i] + scoreB[i] + scoreC[i] - minimum;
}

for (i = 0; i <= ARTISTS - 1; i++) {
    Console.WriteLine(artist_names[i] + " " + song_titles[i] + " " + total[i]);
}
```

### 13.Solution

```
const int CITIZENS = 20;

string prod_name1, prod_name2;
int i, count_A;

string[] answers1 = new string[CITIZENS];
string[] answers2 = new string[CITIZENS];

Console.Write("Enter Product Name 1: ");
prod_name1 = Console.ReadLine();
for (i = 0; i <= CITIZENS - 1; i++) {
    Console.Write("Enter score for product " + prod_name1 + ": ");
    answers1[i] = Console.ReadLine();
}

Console.Write("Enter Product Name 2: ");
prod_name2 = Console.ReadLine();
for (i = 0; i <= CITIZENS - 1; i++) {
    Console.Write("Enter score for product " + prod_name2 + ": ");
    answers2[i] = Console.ReadLine();
}

count_A = 0;
for (i = 0; i <= CITIZENS - 1; i++) {
    if (answers1[i] == "A") {
        count_A += 1;
    }
```

```
}
Console.WriteLine(prod_name1 + " " + count_A);

count_A = 0;
for (i = 0; i <= CITIZENS - 1; i++) {
    if (answers2[i] == "A") {
        count_A += 1;
    }
}
Console.WriteLine(prod_name2 + " " + count_A);
```

### 14.Solution

```
string msg;

Dictionary<string, string> morseAlphabet = new() {
    { "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", "--.." },
    { " ", "/" }
};

Console.Write("Enter an English message: ");
msg = Console.ReadLine();
```

```
foreach (var character in msg) {
    Console.Write(morseAlphabet[character.ToString().ToUpper()] + " ");
}
```

### 15.Solution

```
string word, letter;
int i, random_index, wrong_guesses;

Random rnd = new();

string[] words = {"compiler", "interpreter", "error", "variable",
                  "operator", "computer", "programmer", "algorithm"};

//Randomly choose a word
random_index = rnd.Next(words.Length);  //Alternativelly, these can be written as:
word = words[random_index];             //word = words[rnd.Next(words.Length)]

wrong_guesses = 0;

//Create array results
string[] results = new string[word.Length];
for (i = 0; i < results.Length; i++) {
    results[i] = "_";
}

//The method Contains() returns true when the array results contains the underscore
//character within its elements.
while (results.Contains("_") && wrong_guesses < 6) {
    //Display results
    foreach (var x in results) {
        Console.Write(x + "  ");
    }

    Console.Write("Enter a letter: ");
    letter = Console.ReadLine();

    if (word.IndexOf(letter) > -1) {
        //Replace corresponding underscores with letter in array results
        for (i = 0; i < word.Length; i++) {
            if (letter == word[i].ToString()) {
                results[i] = letter;
            }
        }
    }
    else {
        wrong_guesses += 1;
        Console.WriteLine("This letter does not exist in clue word!");
```

```
    }
}

if (wrong_guesses < 6) {
    Console.WriteLine("Congratulations, you found it!");
}
else {
    Console.WriteLine("Game over!!!");
}
```

### 16.Solution

```
string word, letter;
int i, player, wrong_guesses;

Random rnd = new();

string[] words = {"compiler", "interpreter", "error", "variable",
                  "operator", "computer", "programmer", "algorithm"};

int wrong_guesses1 = 0, wrong_guesses2 = 0;

for (player = 1; player <= 2; player++) {
    word = words[rnd.Next(words.Length)];     //Randomly choose a word

    wrong_guesses = 0;

    //Create array results
    string[] results = new string[word.Length];
    for (i = 0; i < results.Length; i++) {
        results[i] = "_";
    }

    while (results.Contains("_") && wrong_guesses < 6) {
        //Display results
        foreach (var x in results) {
            Console.Write(x + "  ");
        }

        Console.Write("Player No: " + player + " - Enter a letter: ");
        letter = Console.ReadLine();

        if (word.IndexOf(letter) > -1) {
            //Replace corresponding underscores with letter in array results
            for (i = 0; i < word.Length; i++) {
                if (letter == word[i].ToString()) {
                    results[i] = letter;
                }
            }
```

```
        }
        else {
            wrong_guesses += 1;
            Console.WriteLine("This letter does not exist in clue word!");
        }
    }

    Console.Write("Player No:" + player + " - ");
    if (wrong_guesses < 6) {
        Console.WriteLine("Congratulations, you found it!");
    }
    else {
        Console.WriteLine("Game over!!!");
    }

    if (player == 1) {
        wrong_guesses1 = wrong_guesses;
    }
    else {
        wrong_guesses2 = wrong_guesses;

    }
}

//Display the winner
if (wrong_guesses1 < wrong_guesses2) {
    Console.WriteLine("Winner: Player 1");
}
else if (wrong_guesses2 < wrong_guesses1) {
    Console.WriteLine("Winner: Player 2");
}
else {
    Console.WriteLine("It's a tie!");
}
```

# Chapter 25

## 25.4 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 26

## 26.11 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

## 26.12 Review Exercises

### 1. Solution

```
int find_max(int a, int b) {
    int maximum;
    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

```
double find_sum(double a, double b, double c) {
    return a + b + c;
}
```

### 4. Solution

```
double find_avg(double a, double b, double c, double d) {
    return (a + b + c + d) / 4;
}
```

## 5. Solution

```
double maximum(double a, double b, double c) {
    double m;

    m = a;
    if (b > m) {
        m = b;
    }
    if (c > m) {
        m = c;
    }
    Console.WriteLine(m);
}
```

## 6. Solution

```
double find_min(double a, double b) {
    double minimum;
    minimum = a;
    if (b < minimum) {
        minimum = b;
    }
    return minimum;
}

//Main code starts here
double x1, x2, x3, x4, temp1, temp2;

Console.WriteLine("Enter four numbers: ");
x1 = Convert.ToDouble(Console.ReadLine());
x2 = Convert.ToDouble(Console.ReadLine());
x3 = Convert.ToDouble(Console.ReadLine());
x4 = Convert.ToDouble(Console.ReadLine());

//Display lowest value as follows (1st approach)
temp1 = find_min(x1, x2);
temp2 = find_min(x3, x4);
Console.WriteLine(find_min(temp1, temp2));

//Display lowest value as follows (2nd approach)
Console.WriteLine(find_min(find_min(x1, x2), find_min(x3, x4)));
```

## 7. Solution

```
bool get_input() {
    string answer;
    Console.Write("Enter Yes or No: ");
    answer = Console.ReadLine();
    if (answer.ToUpper() == "YES") {
```

```
        return true;
    }
    else {
        return false;
    }
}

double find_area(double b, double h) {
    return b * h;
}

//Main code starts here
double b, h;
bool answer;

do {
    Console.Write("Enter the base of the parallelogram: ");
    b = Convert.ToDouble(Console.ReadLine());

    Console.Write("Enter the height of the parallelogram: ");
    h = Convert.ToDouble(Console.ReadLine());

    Console.WriteLine("Area = " + find_area(b, h));

    Console.WriteLine("Would you like to repeat? ");
    answer = get_input();    //Or you can write...
} while (answer == true);    //while(get_input());
```

# Chapter 27

## 27.2 Review Exercises

### *1. Solution*

```
double Kelvin_to_Fahrenheit(double kelvin) {
    return 1.8 * kelvin - 459.67;
}

double Kelvin_to_Celsius(double kelvin) {
    return kelvin - 273.15;
}

//Main code starts here
double k;

Console.Write("Enter a temperature in degrees Kelvin: ");
k = Convert.ToDouble(Console.ReadLine());
```

```
Console.WriteLine("Fahrenheit: " + Kelvin_to_Fahrenheit(k));
Console.Write("Celsius: " + Kelvin_to_Celsius(k));
```

## 2. Solution

```
int num_of_days(int month) {
    int days;

    if (month == 4 || month == 6 || month == 9 || month == 11) {
        days = 30;
    }
    else if (month == 2) {
        days = 28;
    }
    else {
        days = 31;
    }
    return days;
}

//Main code starts here
int i, x, y, total;

Console.Write("Enter a month: ");
x = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter a second month: ");
y = Convert.ToInt32(Console.ReadLine());

total = 0;
for (i = x; i <= y; i++) {
    total += num_of_days(i);
}

Console.WriteLine(total);
```

## 3. Solution

```
int dice() {
    Random rnd = new();
    return rnd.Next(1, 7);
}

//Main code starts here
int i, dice1, dice2, player;

string[] names = new string[2];
Console.Write("Player 1 enter name: ");
names[0] = Console.ReadLine();
Console.Write("Player 2 enter name: ");
```

```
names[1] = Console.ReadLine();

int[] total = new int[2];

for (player = 0; player <= 1; player++) {
    total[player] = 0;
    for (i = 1; i <= 10; i++) {
        Console.Write(names[player]);
        Console.WriteLine(", hit the Enter key to roll the dice!");
        Console.ReadLine();

        dice1 = dice();
        dice2 = dice();
        Console.WriteLine(dice1 + " " + dice2);
        total[player] += dice1 + dice2;
    }
}

if (total[0] == total[1]) {
    Console.WriteLine("Tie!");
}
else if (total[0] > total[1]) {
    Console.WriteLine(names[0] + " wins!");
}
else {
    Console.WriteLine(names[1] + " wins!");
}
```

## 4. Solution

```
void bmi(double w, double h) {
    double b;

    b = w * 703 / Math.Pow(h, 2);
    if (b < 16) {
        Console.WriteLine("You must add weight.");
    }
    else if (b < 18.5) {
        Console.WriteLine("You should add some weight.");
    }
    else if (b < 25) {
        Console.WriteLine("Maintain your weight.");
    }
    else if (b < 30) {
        Console.WriteLine("You should lose some weight.");
    }
    else {
        Console.WriteLine("You must lose weight.");
```

```
    }
}

//Main code starts here
double height, weight;
int age;

Console.Write("Enter your weight (in pounds): ");
weight = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Enter your age: ");
age = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter your height (in inches): ");
height = Convert.ToDouble(Console.ReadLine());

if (age < 18) {
    Console.WriteLine("I can't calculate your BMI. You must be adult!");
}
else {
    bmi(weight, height);
}
```

## 5. Solution

```
const int CARS = 40;
const int GAS = 1;
const int DIESEL = 2;
const int HYBRID = 3;

int get_choice() {
    int choice;

    Console.WriteLine("1. Gas");
    Console.WriteLine("2. Diesel");
    Console.WriteLine("3. Hybrid");
    Console.Write("Enter type of the car: ");
    choice = Convert.ToInt32(Console.ReadLine());
    return choice;
}

int get_days() {
    int days;

    Console.Write("Enter total number of rental days: ");
    days = Convert.ToInt32(Console.ReadLine());
    return days;
}
```

```
double get_charge(int car_type, int rental_days) {
    double charge;

    if (car_type == GAS) {
        if (rental_days <= 5) {
            charge = rental_days * 24;
        }
        else {
            charge = rental_days * 22;
        }
    }
    else if (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
int i, count;
double charge, total;

int[] rented_car_types = new int[CARS];
int[] rented_days = new int[CARS];

for (i = 0; i <= CARS - 1; i++) {
    rented_car_types[i] = get_choice();
    rented_days[i] = get_days();
}

total = 0;
for (i = 0; i <= CARS - 1; i++) {
    charge = get_charge(rented_car_types[i], rented_days[i]);
    Console.WriteLine("Amount to pay, car No " + (i + 1) + ": " + charge);
```

```
    total += charge;
}

count = 0;
for (i = 0; i <= CARS - 1; i++) {
    if (rented_car_types[i] == HYBRID) {
        count += 1;
    }
}
Console.WriteLine("Hybrids rented: " + count);
Console.WriteLine("Total profit: " + total);
```

# Chapter 28

## 28.8 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

## 28.9 Review Exercises

### *1. Solution*

```
double sqr_side, rctngl_base, rctngl_height, trngl_base, trngl_height;

Trigonometry tr = new();

Console.Write("Enter square side: ");
sqr_side = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter rectangle base: ");
rctngl_base = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter rectangle height: ");
rctngl_height = Convert.ToDouble(Console.ReadLine());

Console.Write("Enter triangle base: ");
trngl_base = Convert.ToDouble(Console.ReadLine());
Console.Write("Enter triangle height: ");
trngl_height = Convert.ToDouble(Console.ReadLine());

Console.WriteLine(tr.square_area(sqr_side));
Console.WriteLine(tr.rectangle_area(rctngl_base, rctngl_height));
Console.WriteLine(tr.triangle_area(trngl_base, trngl_height));
```

```
class Trigonometry {
    public double square_area(double side) {
        return side * side;
    }

    public double rectangle_area(double b, double h) {
        return b * h;
    }

    public double triangle_area(double b, double h) {
        return b * h / 2;
    }
}
```

## 2. Solution

```
Pet pet1 = new();
pet1.kind = "dog";
pet1.legs_number = 4;

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

pet1.start_running();
pet2.start_running();
pet1.stop_running();

class Pet {
    public string kind;
    public int legs_number;

    public void start_running() {
        Console.WriteLine("Pet is running");
    }

    public void stop_running() {
        Console.WriteLine("Pet stopped");
    }
}
```

## 3. Solution

```
Pet pet1 = new("dog", 4);

pet1.start_running();
pet1.stop_running();

pet1.Kind = "";         //This will throw an error
pet1.Legs_number = -3;  //This will throw an error
```

```
class Pet {
    private string _kind;
    private int _legs_number;

    //Define the constructor
    public Pet(string k, int l) {
        this.Kind = k;         //Initial value for the property Kind
        this.Legs_number = l;  //Initial value for the property Legs_number
    }

    //Defines a public property
    public string Kind {
        //Define the getter
        get {
            return this._kind;
        }
        //Define the setter
        set {
            if (value != "") {
                this._kind = value;
            }
            else {
                throw new Exception("Cannot be empty");
            }
        }
    }

    //Defines a public property
    public int Legs_number {
        //Define the getter
        get {
            return this._legs_number;
        }
        //Define the setter
        set {
            if (value >= 0) {
                this._legs_number = value;
            }
            else {
                throw new Exception("Cannot be negative");
            }
        }
    }

    public void start_running() {
```

```
        Console.WriteLine("Pet is running");
    }

    public void stop_running() {
        Console.WriteLine("Pet stopped");
    }
}
```

### 4. Solution

```
const int BOXES = 30;

int i;
double w, l, h;

Box[] array_of_obj = new Box[BOXES];   //Create an array

for (i = 0; i <= BOXES - 1; i++) {
    Console.Write("Enter width: ");
    w = Convert.ToDouble(Console.ReadLine());
    Console.Write("Enter length: ");
    l = Convert.ToDouble(Console.ReadLine());
    Console.Write("Enter height: ");
    h = Convert.ToDouble(Console.ReadLine());

    //Add each new object to the array
    array_of_obj[i] = new(w, l, h);
}

for (i = 0; i <= BOXES - 1; i++) {
    array_of_obj[i].display_dimensions();
    array_of_obj[i].display_volume();
}

class Box {
    private double _width, _length, _height;

    //Define the constructor
    public Box(double width, double length, double height) {
        //Initialize fields
        this._width = width;
        this._length = length;
        this._height = height;
    }

    public void display_volume() {
        Console.WriteLine("Volume: " + (this._width * this._length * this._height));
    }
```

```
    public void display_dimensions() {
        Console.WriteLine(this._width + " x " + this._length + " x " + this._height);
    }
}
```

### 5. Solution

```
const int BOXES = 30;

int i;
double w, l, h;

Box[] array_of_obj = new Box[BOXES];   //Create an array

for (i = 0; i <= BOXES - 1; i++) {
    Console.Write("Enter width: ");
    w = Convert.ToDouble(Console.ReadLine());
    Console.Write("Enter length: ");
    l = Convert.ToDouble(Console.ReadLine());
    Console.Write("Enter height: ");
    h = Convert.ToDouble(Console.ReadLine());

    //Add each new object to the array
    array_of_obj[i] = new(w, l, h);
}

for (i = 0; i <= BOXES - 1; i++) {
    array_of_obj[i].display_dimensions();
    array_of_obj[i].display_volume();
}

class Box {
    private double _width, _length, _height;

    //Define the constructor
    public Box(double width, double length, double height) {
        //Initialize properties
        this.Width = width;
        this.Length = length;
        this.Height = height;
    }

    //Define public property Width
    public double Width {
        //Define the getter
        get {
            return this._width;
        }
```

```
        //Define the setter
        set {
            if (value > 0) {
                this._width = value;
            }
            else {
                throw new Exception("Cannot be negative or zero");
            }
        }
    }

    //Define public property Length
    public double Length {
        //Define the getter
        get {
            return this._length;
        }
        //Define the setter
        set {
            if (value > 0) {
                this._length = value;
            }
            else {
                throw new Exception("Cannot be negative or zero");
            }
        }
    }

    //Define public property Height
    public double Height {
        //Define the getter
        get {
            return this._height;
        }
        //Define the setter
        set {
            if (value > 0) {
                this._height = value;
            }
            else {
                throw new Exception("Cannot be negative or zero");
            }
        }
    }

    public void display_volume() {
```

```
        Console.WriteLine("Volume: " + (this.Width * this.Length * this.Height));
    }

    public void display_dimensions() {
        Console.WriteLine(this.Width + " x " + this.Length + " x " + this.Height);
    }
}
```

## 6. Solution

```
double edge;

Console.Write("Enter edge length of a cube: ");
edge = Convert.ToDouble(Console.ReadLine());

Cube cube1 = new(edge);

cube1.display_volume();
cube1.display_one_surface();
cube1.display_total_surface();

class Cube {
    private double _edge;

    //Define the constructor
    public Cube(double edge) {
        this._edge = edge;   //Initialize field
    }

    public void display_volume() {
        Console.WriteLine("Volume: " + Math.Pow(this._edge, 3));
    }

    public void display_one_surface() {
        Console.WriteLine("One surface: " + Math.Pow(this._edge, 2));
    }

    public void display_total_surface() {
        Console.WriteLine("Total surface: " + 6 * Math.Pow(this._edge, 2));
    }
}
```

## 7. Solution

```
double edge;

Console.Write("Enter edge length of a cube: ");
edge = Convert.ToDouble(Console.ReadLine());

Cube cube1 = new(edge);
```

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

class Cube {
    private double _edge;

    //Define the constructor
    public Cube(double edge) {
        this.Edge = edge;   //Initialize property
    }

    //Define public property Edge
    public double Edge {
        //Define the getter
        get {
            return this._edge;
        }
        //Define the setter
        set {
            if (value > 0) {
                this._edge = value;
            }
            else {
                throw new Exception("Cannot be negative or zero");
            }
        }
    }

    public void display_volume() {
        Console.WriteLine("Volume: " + Math.Pow(this.Edge, 3));
    }

    public void display_one_surface() {
        Console.WriteLine("One surface: " + Math.Pow(this.Edge, 2));
    }

    public void display_total_surface() {
        Console.WriteLine("Total surface: " + (6 * Math.Pow(this.Edge, 2)));
    }
}
```

## 8. Solution

```
void display_menu() {
    Console.WriteLine("1. Enter radius");
    Console.WriteLine("2. Display radius");
    Console.WriteLine("3. Display diameter");
```

```
    Console.WriteLine("4. Display area");
    Console.WriteLine("5. Display perimeter");
    Console.WriteLine("6. Exit");
}

int choice;
double radius;

Circle circle1 = new();

display_menu();
Console.Write("Enter a choice: ");
choice = Convert.ToInt32(Console.ReadLine());
while (choice != 6) {
    if (choice == 1) {
        Console.Write("Enter radius: ");
        radius = Convert.ToDouble(Console.ReadLine());
        circle1.Radius = radius;
    }
    else if (choice == 2) {
        Console.WriteLine("Radius: " + circle1.Radius);
    }
    else if (choice == 3) {
        Console.WriteLine("Diameter: " + circle1.get_diameter());
    }
    else if (choice == 4) {
        Console.WriteLine("Area: " + circle1.get_area());
    }
    else if (choice == 5) {
        Console.WriteLine("Perimeter: " + circle1.get_perimeter());
    }

    display_menu();
    Console.Write("Enter a choice: ");
    choice = Convert.ToInt32(Console.ReadLine());
}

class Circle {
    private double _radius = -1;

    //Define public property Radius
    public double Radius {
        //Define the getter
        get {
            if (this._radius > 0) {
                return this._radius;
```

```
            }
            else {
                throw new Exception("Radius is not set");
            }
        }
        //Define the setter
        set {
            if (value > 0) {
                this._radius = value;
            }
            else {
                throw new Exception("Cannot be negative or zero");
            }
        }
    }

    public double get_diameter() {
        return 2 * this.Radius;
    }

    public double get_area() {
        return 3.14 * Math.Pow(this.Radius, 2);
    }

    public double get_perimeter() {
        return 2 * 3.14 * this.Radius;
    }
}
```

## 9. Solution

```
Info inf = new();

Console.Write("Enter a text: ");
inf.User_text = Console.ReadLine();

Console.WriteLine("Text: " + inf.User_text);
Console.WriteLine("Spaces: " + inf.get_spaces_count());
Console.WriteLine("Words: " + inf.get_words_count());
Console.WriteLine("Vowels: " + inf.get_vowels_count());
Console.WriteLine("Total number of letters: " + inf.get_letters_count());

class Info {
    private string _user_text;

    //Define public property User_text
    public string User_text {
        //Define the getter
```

```
        get {
            return this._user_text;
        }
        //Define the setter
        set {
            if (value != "") {
                this._user_text = value;
            }
            else {
                throw new Exception("Cannot be set to empty");
            }
        }
    }

    public int get_spaces_count() {
        int count = 0;
        foreach (var character in this.User_text) {
            if (character.ToString() == " ") {
                count += 1;
            }
        }
        return count;
    }

    public int get_words_count() {
        return this.get_spaces_count() + 1;
    }

    public int get_vowels_count() {
        int count = 0;
        foreach (var character in this.User_text.ToLower()) {
            if ("aeiou".IndexOf(character) > -1) {
                count += 1;
            }
        }
        return count;
    }

    public int get_letters_count() {
        return this.User_text.Length - this.get_spaces_count();
    }
}
```

## 10.Solution

```
void display_menu() {
    Console.WriteLine("1. Encryption/Decryption key");
```

```
    Console.WriteLine("2. Encrypt a message");
    Console.WriteLine("3. Decrypt a message");
    Console.WriteLine("4. Exit");
}

string text;
int choice;

EncryptDecrypt ed = new();

display_menu();
Console.Write("Enter a choice: ");
choice = Convert.ToInt32(Console.ReadLine());
while (choice != 4) {
    if (choice == 1) {
        Console.Write("Enter encryption/decryption key: ");
        ed.Encr_decr_key = Convert.ToInt32(Console.ReadLine());
    }
    else if (choice == 2) {
        Console.Write("Enter message to encrypt: ");
        text = Console.ReadLine();
        Console.WriteLine("Encrypted message: " + ed.encrypt(text));
    }
    else if (choice == 3) {
        Console.Write("Enter message to decrypt: ");
        text = Console.ReadLine();
        Console.WriteLine("Decrypted message: " + ed.decrypt(text));
    }

    display_menu();
    Console.Write("Enter a choice: ");
    choice = Convert.ToInt32(Console.ReadLine());
}

class EncryptDecrypt {
    private const string alphabet = " abcdefghijklmnopqrstuvwxyz";  //Space is a valid
                                                                    //character!
    private int _encr_decr_key = -1;

    //Define public property Encr_decr_key
    public int Encr_decr_key {
        //Define the getter
        get {
            if (_encr_decr_key != -1) {
                return this._encr_decr_key;
            }
```

```
            else {
                throw new Exception("Key is not set");
            }
        }
        //Define the setter
        set {
            if (value >= 1 && value <= 26) {
                this._encr_decr_key = value;
            }
            else {
                throw new Exception("Must be between 1 and 26");
            }
        }
    }

    public string encrypt(string message) {
        string new_letter, return_value = "";
        int index, new_index;

        foreach (var character in message) {
            index = alphabet.IndexOf(character);
            new_index = index + this.Encr_decr_key;
            if (new_index >= 27) {
                new_index -= 27;
            }
            new_letter = alphabet[new_index].ToString();
            return_value += new_letter;
        }
        return return_value;
    }

    public string decrypt(string enc_message) {
        string new_letter, return_value = "";
        int index, new_index;

        foreach (var character in enc_message) {
            index = alphabet.IndexOf(character);
            new_index = index - this.Encr_decr_key;
            if (new_index < 0) {
                new_index += 27;
            }
            new_letter = alphabet[new_index].ToString();
            return_value += new_letter;
        }
        return return_value;
    }
```

```
}
```

### 11.Solution

```
Car car1 = new(4, "Red", 5, 2, 1.5);
car1.boot_capacity = 300;
car1.start_engine();
car1.turn_windshield_wipers_on();
car1.stop_engine();

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

Motorcycle motorcycle1 = new(2, "Blue", 2, 0.9, 1.3);
motorcycle1.has_luggage = true;
motorcycle1.start_engine();
motorcycle1.do_a_wheelie();
motorcycle1.stop_engine();

class Vehicle {
    public int number_of_wheels;
    public string color;
    public double length, width, height;

    //Define the constructor
    public Vehicle(int number_of_wheels, string color,
                   double length, double width, double height) {
        this.number_of_wheels = number_of_wheels;
        this.color = color;
        this.length = length;
        this.width = width;
        this.height = height;
    }

    public void start_engine() {
        Console.WriteLine("The engine started");
    }

    public void stop_engine() {
        Console.WriteLine("The engine stopped");
    }
}

class Car : Vehicle {
    public int boot_capacity;
```

```
    //Define the constructor
    public Car(int number_of_wheels, string color,
               double length, double width, double height)
               : base (number_of_wheels, color, length, width, height) {
        this.boot_capacity = 0;
    }

    public void turn_windshield_wipers_on() {
        Console.WriteLine("The windshield wipers have been turned on!");
    }
}

class Motorcycle : Vehicle {
    public bool has_luggage;

    //Define the constructor
    public Motorcycle(int number_of_wheels, string color,
                      double length, double width, double height)
                      : base(number_of_wheels, color, length, width, height) {
        this.has_luggage = false;
    }

    public void do_a_wheelie() {
        Console.WriteLine("I am doing a wheelie!!!");
    }
}
```

### 12.Solution

```
Teacher teacher1 = new("Mr. John Scott", 43, 35000);
Teacher teacher2 = new("Mrs. Ann Carter", 5, 32000);

Student student1 = new("Mark Nelson", 14, "A");
Student student2 = new("Mary Morgan", 13, "B");

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

class SchoolMember {
    private string _name;
    private int _age;

    //Define the constructor
    public SchoolMember(string name, int age) {
        this.setName(name);
        this.setAge(age);
```

```
        Console.WriteLine("A school member was initialized");
    }

    //Define the getter
    public string getName() {
        return this._name;
    }

    //Define the setter
    public void setName(string value) {
        if (value != "") {
            this._name = value;
        }
        else {
            throw new Exception("Name cannot be empty");
        }
    }

    //Define the getter
    public int getAge() {
        return this._age;
    }

    //Define the setter
    public void setAge(int value) {
        if (value > 0) {
            this._age = value;
        }
        else {
            throw new Exception("Age cannot be negative or zero");
        }
    }
}

class Teacher : SchoolMember {
    private double _salary;  //This is an additional field for this class

    //Define the constructor
    public Teacher(string name, int age, double salary) : base(name, age) {
        this.setSalary(salary);

        //This is an additional statement for this constructor
        Console.WriteLine("A teacher was initialized");
    }

    //Define the getter
    public double getSalary() {
```

```
        return this._salary;
    }

    //Define the setter
    public void setSalary(double value) {
        if (value >= 0) {
            this._salary = value;
        }
        else {
            throw new Exception("Salary cannot be negative");
        }
    }

    //This is an additional method for this class
    public void display_values() {
        Console.WriteLine("Name: " + this.getName());
        Console.WriteLine("Age: " + this.getAge());
        Console.WriteLine("Salary: " + this.getSalary());
    }
}

class Student : SchoolMember {
    private string _final_grade;  //This is an additional field for this class

    //Define the constructor
    public Student(string name, int age, string final_grade) : base(name, age) {
        this.setFinalGrade(final_grade);

        //This is an additional statement for this constructor
        Console.WriteLine("A student was initialized");
    }

    //Define the getter
    public string getFinalGrade() {
        return this._final_grade;
    }

    //Define the setter
    public void setFinalGrade(string value) {
        if (value != "") {
            this._final_grade = value;
        }
        else {
            throw new Exception("Final grade cannot be empty");
        }
    }
```

```
    //This is an additional method for this class
    public void display_values() {
        Console.WriteLine("Name: " + this.getName());
        Console.WriteLine("Age: " + this.getAge());
        Console.WriteLine("Final grade: " + this.getFinalGrade());
    }
}
```

### 13.Solution

```
Teacher teacher1 = new("Mr. John Scott", 43, 35000);
Teacher teacher2 = new("Mrs. Ann Carter", 5, 32000);

Student student1 = new("Mark Nelson", 14, "A");
Student student2 = new("Mary Morgan", 13, "B");

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

class SchoolMember {
    private string _name;
    private int _age;

    //Define the constructor
    public SchoolMember(string name, int age) {
        this.Name = name;
        this.Age = age;
        Console.WriteLine("A school member was initialized");
    }

    //Define public property Name
    public string Name {
        //Define the getter
        get {
            return this._name;
        }

        //Define the setter
        set {
            if (value != "") {
                this._name = value;
            }
            else {
                throw new Exception("Name cannot be empty");
            }
        }
```

```
    }

    //Define public property Age
    public int Age {
        //Define the getter
        get {
            return this._age;
        }

        //Define the setter
        set {
            if (value > 0) {
                this._age = value;
            }
            else {
                throw new Exception("Age cannot be negative or zero");
            }
        }
    }
}

class Teacher : SchoolMember {
    private double _salary;  //This is an additional field for class Teacher

    //Define the constructor
    public Teacher(string name, int age, double salary) : base(name, age) {
        this.Salary = salary;
        Console.WriteLine("A teacher was initialized");
    }

    //Define public property Salary
    public double Salary {
        //Define the getter
        get {
            return this._salary;
        }

        //Define the setter
        set {
            if (value >= 0) {
                this._salary = value;
            }
            else {
                throw new Exception("Salary cannot be negative");
            }
        }
```

```
    }

    //This is an additional method for this class
    public void display_values() {
        Console.WriteLine("Name: " + this.Name);
        Console.WriteLine("Age: " + this.Age);
        Console.WriteLine("Salary: " + this.Salary);
    }

}

class Student : SchoolMember {
    private string _final_grade;  //This is an additional field for class Student

    //Define the constructor
    public Student(string name, int age, string final_grade) : base(name, age) {
        this.FinalGrade = final_grade;
        Console.WriteLine("A student was initialized");
    }

    //Define public property Grades
    public string FinalGrade {
        //Define the getter
        get {
            return this._final_grade;
        }

        //Define the setter
        set {
            if (value != "") {
                this._final_grade = value;
            }
            else {
                throw new Exception("Final grade cannot be empty");
            }
        }
    }

    //This is an additional method for this class
    public void display_values() {
        Console.WriteLine("Name: " + this.Name);
        Console.WriteLine("Age: " + this.Age);
        Console.WriteLine("Final Grade: " + this.FinalGrade);
    }
}
```

# Chapter 29

## 29.7 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

## 29.8 Review Exercises

### 1. Solution

```
const string PATH = "c:/temp/";

string[] days = {"Sunday", "Monday", "Tuesday", "Wednesday",
                 "Thursday", "Friday", "Saturday"};

StreamWriter f = File.CreateText(PATH + "days_of_week.txt");
foreach (var d in days) {
    f.WriteLine(d);
}
f.Close();
```

### 2. Solution

```
const string PATH = "c:/temp/";

int i;
string[] days = new string[7];

StreamReader f = File.OpenText(PATH + "days_of_week.txt");
for (i = 0; i <= 6; i++) {
    days[i] = f.ReadLine();
}
f.Close();

for (i = 6; i >= 0; i--) {
    Console.WriteLine(days[i]);
}
```

### 3. Solution

```
const string PATH = "c:/temp/";

StreamWriter f = File.AppendText(PATH + "days_of_week.txt");
f.WriteLine("*** End of File ***");
```

```
f.Close();
```

### 4. Solution

```
const string PATH = "c:/temp/";

int i;
Random rnd = new();

StreamWriter f = File.CreateText(PATH + "randoms.txt");
for (i = 0; i <= 49; i++) {
    f.WriteLine(rnd.Next(1, 101));
}
f.Close();
```

### 5. Solution

```
const string PATH = "c:/temp/";

int i;
StreamWriter f;
Random rnd = new();

for (i = 1; i <= 10; i++) {
    f = File.CreateText(PATH + "file" + i + ".txt");
    f.Write(rnd.Next(100, 10000));
    f.Close();
}
```

### 6. Solution

```
const string PATH = "c:/temp/";

int i, j;
StreamWriter f = File.CreateText(PATH + "multiplication_table.txt");

for (i = 1; i <= 10; i++) {
    for (j = 1; j <= 4; j++) {
        f.WriteLine(i + " x " + j + " = " + (i * j));
    }
}
f.Close();
```

### 7. Solution

```
const string PATH = "c:/temp/";

StreamReader f = File.OpenText(PATH + "a_file.txt");

while (!f.EndOfStream) {
    Console.WriteLine(f.ReadLine().Length);
}
f.Close();
```

### 8. Solution

**First approach**

```
const string PATH = "c:/temp/";

int i;
string line;
StreamReader f = File.OpenText(PATH + "a_file.txt");

i = 1;
while (!f.EndOfStream) {
    line = f.ReadLine();
    foreach (var character in line) {
        if (",.!".IndexOf(character) > -1) {
            Console.WriteLine("There is a punctuation mark on line No " + i);
            break;
        }
    }
    i++;
}

f.Close();
```

**Second approach**

```
const string PATH = "c:/temp/";

int i;
string line;
StreamReader f = File.OpenText(PATH + "a_file.txt");

i = 1;
while (!f.EndOfStream) {
    line = f.ReadLine();
    if (line.IndexOf(",") > -1 || line.IndexOf(".") > -1 || line.IndexOf("!") > -1) {
        Console.WriteLine("There is a punctuation mark on line No " + i);
    }
    i++;
}

f.Close();
```

# Chapter 30

## 30.2 Review Exercises

### 1. Solution

```
const string PATH = "c:/temp/";
```

```
int total, count, number;

StreamReader fin = File.OpenText(PATH + "f_data30.2-1.txt");
string[] values = fin.ReadLine().Split();
fin.Close();

total = 0;
count = 0;
foreach (var value in values ) {
    number = Convert.ToInt32(value);
    if (number > 50) {
        total += number;
        count += 1;
    }
}

if (count > 0) {
    Console.WriteLine(total / count);
}
```

## 2. Solution

```
const string PATH = "c:/temp/";

int total, count, number;

StreamReader fin = File.OpenText(PATH + "f_data30.2-2.txt");
string[] values = fin.ReadLine().Split(",");
fin.Close();

total = 0;
count = 0;
foreach (var value in values ) {
    number = Convert.ToInt32(value);
    if (number >= 300 && number <= 500) {
        total += number;
        count += 1;
    }
}

if (count > 0) {
    Console.WriteLine(total / count);
}
```

## 3. Solution

```
const string PATH = "c:/temp/";

string max_name, min_name;
int maximum, minimum, grade;
```

```
StreamReader fin = File.OpenText(PATH + "f_data30.2-3.txt");

//Read the first line
string[] b = fin.ReadLine().Split(",");

maximum = Convert.ToInt32(b[0]);
minimum = maximum;
max_name = b[1];
min_name = b[1];

//Read the rest of the lines
while (!fin.EndOfStream) {
    b = fin.ReadLine().Split(",");
    grade = Convert.ToInt32(b[0]);

    if (grade > maximum) {
        maximum = grade;
        max_name = b[1];
    }
    if (grade < minimum) {
        minimum = grade;
        min_name = b[1];
    }
}

fin.Close();

Console.WriteLine(max_name);
Console.WriteLine(min_name);
```

## 4. Solution

### First approach

```
string filename1, filename2, content;
StreamReader fin;
StreamWriter fout;

Console.Write("Enter filename No 1: ");
filename1 = Console.ReadLine();

if (filename1.Substring(filename1.Length - 4) != ".txt") {
    Console.WriteLine("Wrong filename");
}
else {
    Console.Write("Enter filename No 2: ");
    filename2 = Console.ReadLine();
    if (filename2.Substring(filename1.Length - 4) != ".txt") {
        Console.WriteLine("Wrong filename");
```

```
    }
    else {
        fin = File.OpenText(filename2);
        content = fin.ReadToEnd();
        fin.Close();

        fin = File.OpenText(filename1);
        content += fin.ReadToEnd();    //Concatenation
        fin.Close();

        fout = File.CreateText("final.txt");
        fout.Write(content);
        fout.Close();
    }
}
```

### Second approach

```
string filename1, filename2;

Console.Write("Enter filename No 1: ");
filename1 = Console.ReadLine();

if (filename1.Substring(filename1.Length - 4) != ".txt") {
    Console.WriteLine("Wrong filename");
}
else {
    Console.Write("Enter filename No 2: ");
    filename2 = Console.ReadLine();
    if (filename2.Substring(filename1.Length - 4) != ".txt") {
        Console.WriteLine("Wrong filename");
    }
    else {
        StreamReader fin1 = File.OpenText(filename1);
        StreamReader fin2 = File.OpenText(filename2);
        StreamWriter fout = File.CreateText("final.txt");

        fout.Write(fin2.ReadToEnd() + fin1.ReadToEnd());

        fout.Close();
        fin2.Close();
        fin1.Close();
    }
}
```

## 5. Solution

### First approach – For 15 numbers

```
const string PATH = "c:/temp/";
const int ELEMENTS = 15;
```

```
int i;
double[] numbers = new double[ELEMENTS];

StreamReader fin = File.OpenText(PATH + "f_data30.2-5.txt");
for (i = 0; i < ELEMENTS; i++) {
    numbers[i] = Convert.ToDouble(fin.ReadLine());
}
fin.Close();

Array.Sort(numbers);

StreamWriter fout = File.AppendText(PATH + "f_data30.2-5.txt");
fout.WriteLine("\n********* Sorted numbers *************");
foreach (var number in numbers) {
    fout.WriteLine(number);
}
fout.Close();
```

**First approach – For any number of numbers**

```
const string PATH = "c:/temp/";

int i;

//Create array s_numbers of type string
StreamReader fin = File.OpenText(PATH + "f_data30.2-5.txt");
string[] s_numbers = fin.ReadToEnd().Split("\n");
fin.Close();

//Create array numbers of type double
double[] numbers = new double[s_numbers.Length];
for (i = 0; i < s_numbers.Length; i++) {
    numbers[i] = Convert.ToDouble(s_numbers[i]);
}

Array.Sort(numbers);

StreamWriter fout = File.AppendText(PATH + "f_data30.2-5.txt");
fout.WriteLine("\n********* Sorted numbers *************");
foreach (var number in numbers) {
    fout.WriteLine(number);
}
fout.Close();
```

## 6. Solution

```
const string PATH = "c:/temp/";

int i, j;
```

```
bool onCityLine;
double total, average, maximum;

StreamReader fin = File.OpenText(PATH + "f_data30.2-6.txt");
string[] s_values = fin.ReadToEnd().Split("\n");
fin.Close();

string[] cities = new string[s_values.Length / 2];
double[] temperatures = new double[s_values.Length / 2];

//Split array s_values into two arrays (cities and temperatures)
i = 0;
j = 0;
onCityLine = true;
foreach (var s_value in s_values) {
    if (onCityLine) {
        cities[i++] = s_value;
    }
    else {
        temperatures[j++] = Convert.ToDouble(s_value);
    }
    onCityLine = !onCityLine;  //true becomes false, and false becomes true
}

total = 0;
for (i = 0; i < temperatures.Length; i++) {
    total += temperatures[i];
}

average = total / temperatures.Length;
Console.WriteLine(average);

maximum = temperatures.Max();
Console.WriteLine("Highest temperature: " + maximum);
for (i = 0; i < temperatures.Length; i++) {
    if (temperatures[i] == maximum) {
        Console.WriteLine(cities[i]);
    }
}
```

## 7. Solution

```
const string PATH = "c:/temp/";

const string x = " ABCDEFGHIJKLMNOPQRSTUVWXYZ";   //The space character remains as is
const string y = " JKWCTAMEDXSLFBYUNGRZOIQVHP";

string initial_message, encrypted_message;
```

```
int i;

Console.Write("Enter a message to encrypt: ");
initial_message = Console.ReadLine().ToUpper();

encrypted_message = "";
foreach (var letter in initial_message) {
    //Search for letter in const x
    for (i = 0; i < x.Length; i++) {
        if (letter == x[i]) {
            //Create encrypted message using letters from const y
            encrypted_message += y[i];
            break;
        }
    }
}

StreamWriter fout = File.CreateText(PATH + "encrypted.txt");
fout.Write(encrypted_message);
fout.Close();
```

## 8. Solution

```
const string PATH = "c:/temp/";

const string x = " ABCDEFGHIJKLMNOPQRSTUVWXYZ";   //The space character remains as is
const string y = " JKWCTAMEDXSLFBYUNGRZOIQVHP";

string initial_message, encrypted_message;
int i;

StreamReader fin = File.OpenText(PATH + "encrypted.txt");
encrypted_message = fin.ReadLine();
fin.Close();

initial_message = "";
foreach (var letter in encrypted_message) {
    //Search for letter in const y
    for (i = 0; i <= y.Length - 1; i++) {
        if (letter == y[i]) {
            //Create decrypted message using letters from const x
            initial_message += x[i];
            break;
        }
    }
}

StreamWriter fout = File.CreateText(PATH + "decrypted.txt");
```

```
fout.Write(initial_message);
fout.Close();
```

## 9. Solution

### First approach

```
void copy(string source, string destination) {
    StreamReader fin = File.OpenText(source);
    string x = fin.ReadToEnd();
    fin.Close();

    StreamWriter fout = File.CreateText(destination);
    fout.Write(x);
    fout.Close();
}
```

### Second approach

```
void copy(string source, string destination) {
    StreamReader fin = File.OpenText(source);
    StreamWriter fout = File.CreateText(destination);

    fout.Write(fin.ReadToEnd());

    fin.Close();
    fout.Close();
}
```

## 10.Solution

```
Triangle tr = new();

tr.display_lengths();
tr.display_area();
tr.display_perimeter();

class Triangle {
    const string PATH = "c:/temp/";

    private double _sideA, _sideB, _sideC;

    //Define the constructor
    public Triangle() {
        StreamReader fin = File.OpenText(PATH + "f_data30.2-10.txt");
        this._sideA = Convert.ToDouble(fin.ReadLine());
        this._sideB = Convert.ToDouble(fin.ReadLine());
        this._sideC = Convert.ToDouble(fin.ReadLine());
        fin.Close();
    }

    public bool can_be_triangle() {
        if (this._sideA > 0 && this._sideB > 0 && this._sideC > 0 &&
```

```
            this._sideA + this._sideB > this._sideC &&
            this._sideB + this._sideC > this._sideA &&
            this._sideC + this._sideA > this._sideB) {
            return true;
        }
        else {
            return false;
        }
    }

    public void display_lengths() {
        Console.WriteLine("Side A: " + this._sideA);
        Console.WriteLine("Side B: " + this._sideB);
        Console.WriteLine("Side C: " + this._sideC);
        if (this.can_be_triangle()) {
            Console.WriteLine("Can be lengths of the three sides of a triangle!");
        }
        else {
            Console.WriteLine("Cannot be lengths of the three sides of a triangle!");
        }
    }

    public void display_area() {
        double s, area;

        if (this.can_be_triangle()) {
            s = (this._sideA + this._sideB + this._sideC) / 2;
            area = Math.Sqrt(s * (s - this._sideA) * (s - this._sideB) * (s -
this._sideC));
            Console.WriteLine("Area: " + area);
        }
    }

    public void display_perimeter() {
        double perimeter;

        if (this.can_be_triangle()) {
            perimeter = this._sideA + this._sideB + this._sideC;
            Console.WriteLine("Perimeter: " + perimeter);
        }
    }
}
```