Chapter 01, 02, 04
Special edition

Exercise solutions of
Java TM How to Program
Early Objects
TENTH edition
Paul Deitel • Harvey Deitel

BLANK PAGE
Yet an another sample XD

1.1 Fill in the blanks in each of the following statements:

a) Computers process data under the control of sets of instructions called **programs**.
b) The key logical units of the computer are the **input unit**, **output unit**, **memory unit**, **central processing unit**, **arithmetic and logic unit** and **secondary storage unit**.
c) The three types of languages discussed in the chapter are **machine languages**, **assembly languages** and **high-level languages.**
d) The programs that translate high-level language programs into machine language are called **compilers**.
e) **Android** is an operating system for mobile devices based on the Linux kernel and Java.
f) **Release candidate** software is generally feature complete, (supposedly) bug free and ready for use by the community.
g) The Wii Remote, as well as many smartphones, use **accelerometer** which allows the device to respond to motion.

1.2 Fill in the blanks in each of the following sentences about the Java environment:

a) The **java** command from the JDK executes a Java application.
b) The **javac** command from the JDK compiles a Java program.
c) A Java source code file must end with the **.java** file extension.
d) When a Java program is compiled, the file produced by the compiler ends with the **.class** file extension.
e) The file produced by the Java compiler contains **bytecodes** that are executed by the Java
Virtual Machine.

1.3 in the blanks in each of the following statements (based on Section 1.5):

a) Objects enable the design practice of **information hiding** although they may know how to communicate with one another across well-defined interfaces, they normally are not allowed
to know how other objects are implemented.
b) Java programmers concentrate on creating **classes**, which contain fields and the set of
methods that manipulate those fields and provide services to clients.

c) The process of analyzing and designing a system from an object-oriented point of view
is called **object oriented analysis and design (OOAD)**.
d) A new class of objects can be created conveniently by **inheritance** the new class (called the subclass) starts with the characteristics of an existing class (called the superclass),
possibly customizing them and adding unique characteristics of its own.
e) **The Unified Modeling language (UML)** is a graphical language that allows people who design software systems to use an industry-standard notation to represent them.
f) The size, shape, color and weight of an object are considered **attributes** of the object's class.

1.4 Fill in the blanks in each of the following statements:

a) The logical unit that receives information from outside the computer for use by the computer is the **input unit**.
b) The process of instructing the computer to solve a problem is called **programming**.
c) **High level language** is a type of computer language that uses English like abbreviations for machine-language instructions.
d) **Output unit** is a logical unit that sends information which has already been processed by
the computer to various devices so that it may be used outside the computer.
e) **Memory** and **storage** are logical units of the computer that retain information.
f) is a logical unit of the computer that performs calculations.
g) **CPU** is a logical unit of the computer that makes logical decisions.
h) **High level language** languages are most convenient to the programmer for writing programs quickly and easily.
i) The only language a computer can directly understand is that computer's **Machine language**.
j) **CPU** is a logical unit of the computer that coordinates the activities of all the other
logical units.

1.5 Fill in the blanks in each of the following statements:

a) **Java** is a platform independent programming language that was built with the objective of allowing programs to be written once and then run on a large variety of electronic devices without modification.
b) **Standard**, **Enterprise** and **Micro** are the names of the three editions of Java that can be used to build different kinds of applications.
c) **Bandwith** is the information-carrying capacity of communication lines, and has rapidly increased over the years and become more affordable. Its availability is a cornerstone for building applications that are significantly connected.
d) An **assembler** is a translator that can convert early assembly-language programs to machine language with reasonable efficiency.

1.6 Fill in the blanks in each of the following statements:

a) Java programs normally go through five phases **editing**, **compilation**, **loading**, **execution**, and **testing**.
b) A(n) **IDE** provides many tools that support the software development process, such as editors for writing and editing programs, debuggers for locating logic errors in programs, and many other features.
c) The command java invokes the **JVM**, which executes Java programs.
d) A(n) **emulator** is a software application that simulates a computer, but hides the underlying operating system and hardware from the programs that interact with it.
e) The **class loader** takes the .class files containing the program's bytecodes and transfers
them to primary memory.
f) The **bytecode verifier** examines bytecodes to ensure that they're valid.

1.7 Explain what a just-in-time (JIT) compiler of Java does.

Ans: A just-in-time (JIT) compiler of Java is a component of the Java virtual machine (JVM) that dynamically compiles bytecode into native machine code at runtime, just before the code is executed. This allows the Java code to run faster because it is translated into a form that can be more directly executed by the CPU.

When Java code is executed, it is initially interpreted by the JVM. This interpretation process can be relatively slow, because the JVM must repeatedly analyze and execute the same code each time the program is run. To improve performance, the JIT compiler optimizes commonly executed code by translating it into machine code and storing the compiled code in memory.

The next time the code is executed, the JVM can directly execute the optimized machine code, which is typically much faster than interpreting the Java bytecode.

1.8 One of the world’s most common objects is a wrist watch. Discuss how each of the following terms and concepts applies to the notion of a watch: object, attributes, behaviors, class, inheritance (consider, for example, an alarm clock), modeling, messages, encapsulation, interface and information hiding.

Object: A watch is an object that can be described as a timepiece that is worn on the wrist to keep track of time. It can be physical, digital or a combination of both.
Attributes: A watch has various attributes such as its material, color, shape, size, design, and the type of information it provides, such as time, date, alarms, and other features.

Behaviors: A watch has certain behaviors, such as displaying the time accurately, having the ability to set alarms, tracking time in different time zones, and other features that allow users to interact with it.
Class: A watch belongs to the class of timepieces, which include other objects that are used to measure or display time, such as clocks, timers, sundials and hourglasses.
Inheritance: Inheritance is the concept in object-oriented programming where a class can derive properties and attributes from another class. For example, an alarm clock is a type of watch that can inherit properties from the watch class.
Modeling: Modeling is the process of representing real-world objects or systems using object-oriented programming languages or other modeling languages. A watch can be modeled using these languages to create software simulations that replicate its behaviors and interactions with users.
Messages: In object-oriented programming, objects communicate with each other through messages. For example, if a user wants to set an alarm on a watch, they would send a message to the watch object informing it to set the alarm.
Encapsulation: Encapsulation is a mechanism in object-oriented programming that protects the internal workings of an object from external access. This ensures that the object's state and behavior are kept private and can only be accessed through defined interfaces. A watch can implement encapsulation by hiding its internal workings from users and only exposing certain methods and properties.

Interface: An interface is a set of methods and properties that define how an object can be used by other objects. A watch's interface could include methods for displaying the time, setting alarms and changing time zones, and properties such as the watch's physical size and color.
Information hiding: Information hiding is the practice of limiting access to an object's internal attributes and functionality to ensure that it is used correctly and efficiently. A watch can implement information hiding by only exposing certain attributes and methods to users, and hiding the details of how it functions internally.

2.1 Fill in the blanks in each of the following statements:

a) A(n) { left brace begins the body of every method, and a(n) right brace ends the body of every method.
b) You can use the if statement to make decisions.
c) // begins an end-of-line comment.
d) Space, newlines and tabs are called white space.
e) Keywords are reserved for use by Java.
f) Java applications begin execution at method main .
g) Methods System.out.print, System.out.println and System.out.printf display information in a command window.

2.2 State whether each of the following is true or false. If false, explain why.:

a) Comments cause the computer to print the text after the // on the screen when the program executes.
Ans: False. Comments are avoided by the compiler.
b) All variables must be given a type when they're declared.
Ans: True.
c) Java considers the variables number and NuMbEr to be identical.
Ans: False. Variables are case-sensitive.
d) The remainder operator (%) can be used only with integer operands.
Ans: False. It also works with non-integer.
e) The arithmetic operators *, /, %, + and - all have the same level of precedence.
Ans: False. Some of them has higher priority.

2.3 Write statements to accomplish each of the following tasks:

a) Declare variables c, thisIsAVariable, q76354 and number to be of type int.

```java
int c, thisIsAVariable, q76354, number;
```

b) Prompt the user to enter an integer.

```java
System.out.print("Enter an integer: ");
```

c) Input an integer and assign the result to int variable value. Assume Scanner variable input can be used to read a value from the keyboard.

```java
value = input.nextInt();
```

d) Print "This is a Java program" on one line in the command window. Use method System.out.println.

```
System.out.println("This is a Java program");
```

e) Print "This is a Java program" on two lines in the command window. The first line should end with Java. Use method System.out.printf and two %s format specifiers.

```
System.out.printf("%s%n%s%n", "This is a Java", "program");
```

f) If the variable number is not equal to 7, display "The variable number is not equal to 7" .

```
    if (number != 7)
System.out.println("The variable number is not equal to 7");
```

2.4 Identify and correct the errors in each of the following statements:

a) if (c < 7);
System.out.println("c is less than 7");
Ans: There will be no semicolon after the condition of if.

b) if (c => 7)
System.out.println("c is equal to or greater than 7");
Ans: The condition has an invalid relational operation.

2.5 Write declarations, statements or comments that accomplish each of the following tasks:

a) State that a program will calculate the product of three integers.

```
// A program to calculate the product of three integer.
```

b) Create a Scanner called input that reads values from the standard input.

```
Scanner input = new Scanner(System.in);
```

c) Declare the variables x, y , z and result to be of type int.

```
int x, y, z, result;
```

d) Prompt the user to enter the first integer.

```
System.out.print("Enter first integer: ");
```

e) Read the first integer from the user and store it in the variable x.

```
x = input.nextInt();
```

f) Prompt the user to enter the second integer.

```java
System.out.print("Enter second integer: ");
```

g) Read the second integer from the user and store it in the variable y.

```java
y = input.nextInt();
```

h) Prompt the user to enter the third integer.

```java
System.out.print("Enter third integer: ");
```

i) Read the third integer from the user and store it in the variable z.

```java
z = input.nextInt();
```

j) Compute the product of the three integers contained in variables x, y and z, and assign the result to the variable result.

```java
result = x * y * z;
```

k) Use System.out.printf to display the message "Product is" followed by the value of the variable result.

```java
System.out.printf("Product is %d%n", result);
```

2.6 Using the statements you wrote in Exercise 2.5, write a complete program that calculates and prints the product of three integers.

```java
import java.util.Scanner;

/**
 * Main
 */
public class Main {
    public static void main(String[] args) {
        int x, y, z, result;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter first integer: ");
        x = input.nextInt();
        System.out.print("Enter second integer: ");
        y = input.nextInt();
        System.out.print("Enter third integer: ");
        z = input.nextInt();
        input.close();

        result = x * y * z;
        System.out.printf("Product is %d%n", result);
    }
}
```

2.7 Fill in the blanks in each of the following statements:

a) Comments are used to document a program and improve its readability.
b) A decision can be made in a Java program with a(n) if.
c) Calculations are normally performed by mathematical statements.
d) The arithmetic operators with the same precedence as multiplication are division and modulo operator.
e) When parentheses in an arithmetic expression are nested, the innermost set of parentheses is evaluated first.
f) A location in the computer's memory that may contain different values at various times throughout the execution of a program is called a(n).variable.

2.8 Write Java statements that accomplish each of the following tasks:

a) Display the message "Enter an integer: ", leaving the cursor on the same line.

```
System.out.print("Enter an integer: ");
```

b) Assign the product of variables b and c to variable a.

```
a = b * c;
```

c) Use a comment to state that a program performs a sample payroll calculation.

```
// This is a program to perform a sample payroll calculation
```

2.9 State whether each of the following is true or false. If false, explain why.

a) Addition is executed first in the following expression: a * b / (c + d) * 5.
Ans: True.
b) The following are all valid variable names: AccountValue, $value, value_in_$, account_no_1234, US$, her_sales_in_$, his_$checking_account, X!, _$_, a@b, and _name.
Ans: False. Because they can't contain special characters like @ or !.
c) In 2 + 3 + 5 / 4, addition has the highest precedence.
Ans: False. The division has more priority than addition.

d) The following are all invalid variable names: name@email.com, 87 , x%, 99er, and 2_.
Ans: True.

2.10 Assuming that x = 5 and y = 1, what does each of the following statements display?

a) System.out.printf("x = %d%n", x + 5);

```
Ans: x = 10
```

b) System.out.printf("Value of %d * %d is %d\n", x, y, (x * y));

```
Ans: Value of 5 * 1 is 5
```

c) System.out.printf("x is %d and y is %d", x, y);

```
Ans: x is 5 and y is 1
```

d) System.out.printf("%d is not equal to %d\n", (x + y), (x * y));

```
Ans: 6 is not equal to 5
```

2.11 Which of the following Java statements contain variables whose values are modified?

a) p = i + j + k + 7;
Ans: Modified.
b) System.out.println("variables whose values are modified");
Ans: Not modified.
c) System.out.println("a = 5");
Ans: Not modified.
d) value = input.nextInt();
Ans: Modified.

2.12 Given that y = ax2 + 5x + 2, which of the following are correct Java statements for this equation?

a) y = a * x * x + 5 * x + 2;
Ans: Correct.
b) y = a * x * x + (5 * x) + 2;
Ans: Correct.
c) y = a * x * x + 5 * (x + 2);
Ans: Not correct.
d) y = a * (x * x) + 5 * x + 2;
Ans: Correct.
e) y = a * x * (x + 5 * x) + 2;
Ans: Not correct.
f) y = a * (x * x + 5 * x + 2);
Ans: Not correct.

2.13 State the order of evaluation of the operators in each of the following Java statements, and show the value of x after each statement is performed:

a) x = 7 + 3 * 6 / 2 – 1;

Ans: First there will be multiplication and then division. Later other arithmetic. So the result will be 15.

b) x = 2 % 2 + 2 * 2 - 2 / 2;

Ans: Here modulo operation, multiplication and division will occur first then addition, so the result will be 3.

c) x = (3 * 9 * (3 + (9 * 3 / (3))));

Ans: First innermost operation 9 * 3 / 3 will be executed and later on addition and multiplication. So the result will be, 324.

2.14 Write an application that displays the numbers 1 to 4 on the same line, with each pair of adjacent numbers separated by one space. Use the following techniques:

a) Use one System.out.println statement.

b) Use four System.out.print statements.

c) Use one System.out.printf statement.

```java
/**
 * Main
 */
public class Main {
    public static void main(String[] args) {
        System.out.println(1 + " " + 2 + " " + 3 + " " + 4);

        System.out.print(1 + " ");
        System.out.print(2 + " ");
        System.out.print(3 + " ");
        System.out.print(4 + " ");

        System.out.printf("%n%d %d %d %d%n", 1, 2, 3, 4);
    }
}
```

2.15 (Arithmetic) Write an application that asks the user to enter two integers, obtains them from the user and prints the square of each, the sum of their squares, and the difference of the squares (first number squared minus the second number squared). Use the techniques shown in Fig. 2.7.

```java
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter first integer: ");
        int first = input.nextInt();
        System.out.print("Enter second integer: ");
        int second = input.nextInt();
        input.close();

         System.out.printf("First: %d%nSecond: %d%n", first, second);
         System.out.printf("First squared: %d%nSecond squared: %d%n", first * first, second * second);
         System.out.printf("Sum of squares: %d%nDifference of squares: %d%n", first * first + second * second, first * first - second * second);
     }
}
```

2.16 (Comparing Integers) Write an application that asks the user to enter one integer, obtains it from the user and displays whether the number and its square are greater than, equal to, not equal to, or less than the number 100. Use the techniques shown in Fig. 2.15.

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int number = input.nextInt();
        input.close();

        int square = number * number;
         if (number > 100) {
            System.out.printf("%d's square is greater than 100%n", number);
         }
         else if (number < 100) {
            System.out.printf("%d's square is less than 100%n", number);
         }
         else if (number == 100) {
            System.out.printf("%d's square is equal to 100%n", number);
         }
     }
}
```

2.17 (Arithmetic, Smallest and Largest) Write an application that inputs three integers from the user and displays the sum, average, product, smallest and largest of the numbers. Use the techniques shown in Fig. 2.15. [Note: The calculation of the average in this exercise should result in an integer representation of the average. So, if the sum of the values is 7, the average should be 2, not 2.3333....]

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int num1, num2, num3, sum, product, smallest, largest;
        double average;

        System.out.print("Enter first integer: ");
         num1 = input.nextInt();

         System.out.print("Enter second integer: ");
         num2 = input.nextInt();

         System.out.print("Enter third integer: ");
         num3 = input.nextInt();
         input.close();

         sum = num1 + num2 + num3;
         product = num1 * num2 * num3;
         average = sum / 3.0;

         smallest = num1;
         if (num2 < smallest) {
             smallest = num2;
         }
         if (num3 < smallest) {
             smallest = num3;
         }

         largest = num1;
         if (num2 > largest) {
             largest = num2;
         }
         if (num3 > largest) {
             largest = num3;
        }

        System.out.printf("Sum is %d%n", sum);
        System.out.printf("Average is %.2f%n", average);
        System.out.printf("Product is %d%n", product);
        System.out.printf("Smallest is %d%n", smallest);
        System.out.printf("Largest is %d%n", largest);
    }
}
```

2.18 (Displaying Shapes with Asterisks) Write an application that displays a box, an oval, an arrow and a diamond using asterisks (*):

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        System.out.println("*********   ***     *        *    ");
        System.out.println("*       *  *   *   ***      * *   ");
        System.out.println("*       * *     * *****    *   *  ");
        System.out.println("*       * *     *   *     *     * ");
        System.out.println("*       * *     *   *    *       *");
        System.out.println("*       * *     *   *     *     * ");
        System.out.println("*       * *     *   *      *   *  ");
        System.out.println("*       *  *   *    *       * *   ");
        System.out.println("*********   ***     *        *    ");
    }
}
```

2.19 What does the following code print?

```
> System.out.printf("*%n**%n***%n****%n*****%n");
```

Ans:

```
*
**
***
****
*****
```

2.20 What does the following code print?

```
System.out.println("*");
System.out.println("***");
System.out.println("*****");
System.out.println("****");
System.out.println("**");
```

Ans:

```
*
***
*****
****
**
```

2.21 What does the following code print?

```
System.out.print("*");
System.out.print("***");
System.out.print("*****");
System.out.print("****");
System.out.println("**");
```

Ans:

```
***************
```

2.22 What does the following code print?

```
System.out.print("*");
System.out.println("***");
System.out.println("*****");
System.out.print("****");
System.out.println("**");
```

Ans:

```
****
*****
******
```

2.23 What does the following code print?

```
System.out.printf("%s%n%s%n%s%n", "*", "***", "*****");
```

Ans:

```
*
***
*****
```

2.24 (Largest and Smallest Integers) Write an application that reads five integers and determines and prints the largest and smallest integers in the group. Use only the programming techniques you learned in this chapter.

```java
import java.util.Scanner;
```

```java

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int number1;
        int number2;
        int number3;
        int number4;
        int number5;

        number1 = input.nextInt();
        number2 = input.nextInt();
        number3 = input.nextInt();
        number4 = input.nextInt();
        number5 = input.nextInt();

        input.close();

        if (number1 > number2 && number1 > number3 && number1 > number4 && number1 > number5) {
            System.out.printf("%d is the largest number%n", number1);
        } else if (number2 > number1 && number2 > number3 && number2 > number4 && number2 > number5) {
            System.out.printf("%d is the largest number%n", number2);
        } else if (number3 > number1 && number3 > number2 && number3 > number4 && number3 > number5) {
            System.out.printf("%d is the largest number%n", number3);
        } else if (number4 > number1 && number4 > number2 && number4 > number3 && number4 > number5) {
            System.out.printf("%d is the largest number%n", number4);
        } else if (number5 > number1 && number5 > number2 && number5 > number3 && number5 > number4) {
            System.out.printf("%d is the largest number%n", number5);
        }

        if (number1 < number2 && number1 < number3 && number1 < number4 && number1 < number5) {
            System.out.printf("%d is the smallest number%n", number1);
        } else if (number2 < number1 && number2 < number3 && number2 < number4 && number2 < number5) {
            System.out.printf("%d is the smallest number%n", number2);
        } else if (number3 < number1 && number3 < number2 && number3 < number4 && number3 < number5) {
            System.out.printf("%d is the smallest number%n", number3);
        } else if (number4 < number1 && number4 < number2 && number4 < number3 && number4 < number5) {
            System.out.printf("%d is the smallest number%n", number4);
        } else if (number5 < number1 && number5 < number2 && number5 < number3 && number5 < number4) {
            System.out.printf("%d is the smallest number%n", number5);
        }
    }
}
```

2.25 (Divisible by 3) Write an application that reads an integer and determines and prints whether it's divisible by 3 or not.

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter an integer: ");
        int number = input.nextInt();
```

```
      input.close();

       if (number % 3 == 0) {
          System.out.printf("%d is divisible by 3%n", number);
      } else {
          System.out.printf("%d is not divisible by 3%n", number);
      }
   }
}
```

2.26 (Multiples) Write an application that reads two integers, determines whether the first number tripled is a multiple of the second number doubled, and prints the result.

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      int num1, num2;
      System.out.print("Enter first number: ");
      num1 = input.nextInt();
      System.out.print("Enter second number: ");
       num2 = input.nextInt();
       input.close();

       if (num1 * 3 % (num2 * 2) == 0) {
          System.out.printf("%d is a multiple of %d%n", num1 * 3, num2 * 2);
      } else {
          System.out.printf("%d is not a multiple of %d%n", num1 * 3, num2 * 2);
      }
   }
}
```

2.27 (Checkerboard Pattern of Asterisks) Write an application that displays a checkerboard pattern:

```
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      Scanner input = new Scanner(System.in);
      System.out.println("Enter the size of the checkerboard: ");
```

```
        int size = input.nextInt();
        input.close();

        for (int i = 0; i < size; i++) {
            if (i % 2 == 0) {
                System.out.print(" ");
            }
            for (int j = 0; j < size; j++) {
                System.out.print("* ");
            }
            System.out.println();
        }
    }
}
```

2.28 (Diameter, Circumference and Area of a Circle)

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the radius of a circle: ");
        int radius = input.nextInt();
        System.out.printf("Diameter: %d%nCircumference: %f%nArea: %f%n",
2 * radius, 2 * Math.PI * radius,
                Math.PI * radius * radius);
        input.close();
    }
}
```

2.29 (Integer Value of a Character)

```
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a character: ");
        char character = input.next().charAt(0);
        input.close();

        System.out.printf("The character %c has the value %d%n", character,
((int) character));
```

```java
    }
}
```

2.30 (Separating the Digits in an Integer)

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.print("Enter a five-digit integer: ");
        int number = input.nextInt();
        input.close();

         int digit1 = number / 10000;
         int digit2 = (number % 10000) / 1000;
         int digit3 = (number % 1000) / 100;
         int digit4 = (number % 100) / 10;
         int digit5 = number % 10;

         System.out.printf("%d   %d   %d   %d   %d%n", digit1, digit2, digit3, digit4, digit5);
    }
}
```

2.31 (Table of Squares and Cubes)

```java
public class Main {
    public static void main(String[] args) {
        System.out.println("number    square    cube");
        for (int i =0; i <= 10; i++)
        {
            System.out.println(i + "        " + i * i + "        " + i * i * i);
        }
    }
}
```

2.32 (Negative, Positive and Zero Values)

```java
import java.util.Scanner;

public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        int negativeCount = 0;
        int positiveCount = 0;
        int zeroCount = 0;

        for (int i = 0; i < 5; i++) {
            System.out.print("Enter a number: ");
            int num = input.nextInt();

            if (num < 0) {
                negativeCount++;
            } else if (num > 0) {
                positiveCount++;
            } else {
                zeroCount++;
            }
        }
        input.close();

        System.out.println("Negative numbers: " + negativeCount);
        System.out.println("Positive numbers: " + positiveCount);
        System.out.println("Zero numbers: " + zeroCount);
    }
}
```

4.1 Fill in the blanks in each of the following statements:

a) All programs can be written in terms of three types of control structures: **sequence**, **selection** and **repetition.**
b) The **if … else** statement is used to execute one action when a condition is true and another when that condition is false.
c) Repeating a set of instructions a specific number of times is called **counter controlled or definite** repetition.
d) When it's not known in advance how many times a set of statements will be repeated, a(n) **sentinel** value can be used to terminate the repetition.
e) The **sequence** structure is built into Java; by default, statements execute in the order they appear.
f) Instance variables of types char, byte, short, int, long, float and double are all given the value **zero** by default.
g) Java is a(n) **strongly typed** language; it requires all variables to have a type.
h) If the increment operator is **prefixed** to a variable, first the variable is incremented by 1, then its new value is used in the expression.

4.2 State whether each of the following is true or false. If false, explain why.:

a) An algorithm is a procedure for solving a problem in terms of the actions to execute and the order in which they execute.

Ans: True

b) A set of statements contained within a pair of parentheses is called a block.

Ans: False. A set of statements withing braces is called a block.

c) A selection statement specifies that an action is to be repeated while some condition remains true.

Ans: False. A repetition statement specify that an action is to be repeated while some condition remain true.

d) A nested control statement appears in the body of another control statement.

Ans: True.

e) Java provides the arithmetic compound assignment operators +=, -=, *=, /= and %= for abbreviating assignment expressions.

Ans: True.

f) The primitive types (boolean, char, byte, short, int, long, float and double) are portable across only Windows platforms.

Ans: False. As they are cross-platform, they can support almost all platforms.

g) Specifying the order in which statements execute in a program is called program control.

Ans: True.

h) The unary cast operator (double) creates a temporary integer copy of its operand.

Ans: False. The unary cast operator (double) creates a temporary floating point copy of its operand.

I) Instance variables of type boolean are given the value true by default.

Ans: False. Instance variables of type boolean are given the value false by default.

j) Pseudo-code helps you think out a program before attempting to write it in a programming language.

Ans: True.

4.3 Write four different Java statements that each add 1 to integer variable x.

```
X = X + 1;

X++;

++X;

X+=1;
```

4.4 Write Java statements to accomplish each of the following tasks:

a) Use one statement to assign the sum of x and y to z, then increment x by 1.

```
z = x++ + y;
```

b) Test whether variable count is greater than 10. If it is, print "Count is greater than 10".

```
if ( count > 10 )
     System.out.println(“Count is greater than 10”);
```

c) Use one statement to decrement the variable x by 1, then subtract it from variable total and store the result in variable total.

```
total = total - --x;
```

d) Calculate the remainder after q is divided by divisor , and assign the result to q. Write this statement in two different ways.

```
Q = Q % divisor;
Q %= divisor;
```

4.5 Write Java statements to accomplish each of the following tasks:

a) Declare variables sum of type int and initialize it to 0.

```
Int sum = 0;
```

b) Declare variables x of type int and initialize it to 1.

```
int x = 1;
```

c) Add variable x to variable sum, and assign the result to variable sum.

```
sum = sum + x;
```

d) Print "The sum is: ", followed by the value of variable sum.

```
System.out.println(“The sum is: ” + sum);
```

4.6 Combine the statements that you wrote in Exercise 4.5 into a Java application that calculates and prints the sum of the integers from 1 to 10. Use a while statement to loop through the calculation and increment statements. The loop should terminate when the value of x becomes 11.

```
public class Main {
   public static void main(String[] args) {
      int sum = 0;
      int x = 1;
      while ( x <= 10 )
      {
         sum += x++;
      }
      System.out.println(sum);
   }
}
```

4.7 Determine the value of the variables in the statement product *= x++; after the calculation is performed. Assume that all variables are type int and initially have the value 5.

Ans: The value of x will be 25.

4.8 Identify and correct the errors in each of the following sets of code:

```
a) while (c <= 5)
{
product *= c;
++c;
```

Ans: The closing bracket is missing.

```
b) if (gender == 1)
System.out.println("Woman");
```

```
else;
System.out.println("Man");
```

Ans: There's a semicolon after else, so it'll not work as expected.

4.9 What is wrong with the following while statement?

```
while (z >= 0)
      sum += z;
```

Ans: It'll cause an infinity loop. Cause the value of z will remain unchanged.

4.10 Compare and contrast the if single-selection statement and the while repetition statement. How are these two statements similar? How are they different?

Ans: The if statement and while repetition statement both works on a certain condition. If the condition is true then both of these works, or they avoids a lenght of statements.

On the other hand, if selection statement can execute some statements once but the while repetition can execute certain statements until the condition is met.

4.11 Explain what happens when a Java program attempts to divide one integer by another. What happens to the fractional part of the calculation? How can you avoid that outcome?

Ans: When a Java program attempts to divide one integer by another using the division operator ("/"), the result will be an integer division. The fractional part of the calculation is truncated, meaning it is discarded, and we will get the integer quotient as the result. For example, if you divide 7 by 2, the result will be 3, and the fractional part (0.5) is discarded.

To avoid integer division and preserve the fractional part, we can use casting to convert one or both of the integers to floating-point numbers. For example: (double) a / b or a / (double) b.

4.12 Describe the two ways in which control statements can be combined.

Ans: Control statement can be combined in two ways. Those are,

1. Sequential Combination: Control statements are executed in sequence, one after the other. This is the default behavior in Java. For example, statements in a method are executed one after the other, from top to bottom.

2. Nested Combination: In this process one control statement appears inside another. If the outer condition one is true and executed then inner one works.

4.13 What type of repetition would be appropriate for obtaining an input from the user until the user indicates there is no more input to provide? What type would be appropriate for calculating the factorial of 5? Briefly describe how each of these tasks could be performed.

Ans: For such type of repetition, while loop will be much more effective. For instance, the following code will take input from the user and will create a list until the user press 0.

```java
import java.util.LinkedList;
import java.util.Scanner;

public class Main {
   public static void main(String[] args) {
      LinkedList<Integer> list = new LinkedList<Integer>();
      Scanner sc = new Scanner(System.in);
      while (true)
      {
          int temp = sc.nextInt();
          if (temp == 0) break;
          list.add(temp);
       }
       sc.close();
       System.out.println(list.toString());
    }
}
```

For finding the factorial of five, while repetition will be much more effiecint and time saving. The code will be like,

```java
public class Main {
   public static void main(String[] args) {
      int fact = 5, result = 1;
      while (fact != 1)
         result *= fact--;
      System.out.println(result);
   }
}
```

4.14 If integers x and y are set to 7 and 3, what is the value of x after x = y++ and x = ++y?

Ans: 3 and 4

4.15 Identify and correct the errors in each of the following pieces of code. [Note: There may be more than one error in each piece of code.]

```java
a) if (age >= 65);
```

```
2 System.out.println("Age is greater than or equal to 65");
3 else
4 System.out.println("Age is less than 65)";
```

Ans: At line 1, there's a semicolon after the if condition. And in the fourth line, the semicolon will have to be put after the right parentheses.

b)

```
1 int x == 1, total == 0;
2 while (x <= 10)
3 {
4 total ++x;
5 System.out.println(x);
6 }
```

Ans: At line one, we have to assign a number to x and, so we have to use, x = 1; And on the fourth line, ++ is a unary operator, so we just have to use one operand.

c)

```
1 while (x <= 100)
2 total += x;
3 ++x;
```

Ans: After while statement block statement or braces are missing. In this code only the total += x; will be executed, which will cause an infinity loop. So we have to use braces to define block statement.

d)

```
1 while (y =! 0)
2 {
3 System.out.println (y);
```

Ans: Here the control statement would be y != 0 and there will be a right brace at the very end.

4.16

What does the following program print?

```
// Exercise 4.16: Mystery.java
public class Mystery
{
public static void main(String[] args)
{
int x = -2;
int total = 0;
while (x <= 10)
{
 int y = x + 2;
 x++;
```

```java
total += y;
System.out.printf("Y is: %d and total is %d\n", y, total);
}// end while
}// end main
} // end class Mystery
```

Ans: The output will be,

```
Y is: 0 and total is 0
Y is: 1 and total is 1
Y is: 2 and total is 3
Y is: 3 and total is 6
Y is: 4 and total is 10
Y is: 5 and total is 15
Y is: 6 and total is 21
Y is: 7 and total is 28
Y is: 8 and total is 36
Y is: 9 and total is 45
Y is: 10 and total is 55
Y is: 11 and total is 66
Y is: 12 and total is 78
```