Here are the answers to all the questions in the document "Java Assignment Chapter 4":

**Exercise 1 – Error Correction**

1. **i = 1; while (i <= 10); ++i; }**
   * **Error:** Semicolon after the while condition makes it an empty loop. Also, unmatched closing brace.
   * **Fix:**
2. **for (k = 0.1; k != 1.0; k += 0.1)**
   * **Error:** Floating-point precision can cause an infinite loop.
   * **Fix:** Use an integer loop or compare with a tolerance.
   * for (double k = 0.1; k < 1.0; k += 0.1)
   * System.out.println(k);
3. **Switch without break:**
4. switch (n) {
5. case 1:
6. System.out.println("The number is 1");
7. case 2:
8. System.out.println("The number is 2");
9. break;
10. default:
11. System.out.println("The number is not 1 or 2");
12. break;
13. }
    * **Error:** Missing break; after case 1: causes fall-through.
    * **Fix:**
    * switch (n) {
    * case 1:
    * System.out.println("The number is 1");
    * break;
    * case 2:
    * System.out.println("The number is 2");
    * break;
    * default:
    * System.out.println("The number is not 1 or 2");
    * break;
    * }
14. **n = 1; while (n < 10) System.out.println(n++);**
    * **Error:** Loop will not print 10. Loop condition should be n <= 10.
    * **Fix:**
    * int n = 1;
    * while (n <= 10)
    * System.out.println(n++);

**Exercise 2**

**4.5** Four basic elements of counter-controlled repetition:

1. Control variable initialization.
2. Loop-continuation condition.
3. Increment/decrement of the control variable.
4. Body of the loop.

**4.6** while vs for:

* while: Best when the number of iterations isn’t known.
* for: Best when iterations are predictable, since it consolidates initialization, condition, and increment in one line.

**4.7** Use do...while when the loop body must run at least once regardless of the condition. For example, a menu that should appear once before asking for a valid input.

**4.8** break vs continue:

* break: Exits the loop completely.
* continue: Skips the rest of the current iteration and continues with the next.

**4.9** Code error fixes:

a)

// Incorrect: For (i = 100, i >= 1, i++)

for (int i = 100; i >= 1; i--)

System.out.println(i);

b)

// Missing break statements

switch (value % 2) {

case 0:

System.out.println("Even integer");

break;

case 1:

System.out.println("Odd integer");

break;

}

c)

// Loop increments instead of decrements

for (int i = 19; i >= 1; i -= 2)

System.out.println(i);

d)

// "While" should be lowercase, and condition should be <=

int counter = 2;

do {

System.out.println(counter);

counter += 2;

} while (counter <= 100);

**4.10 What does the program do?**

It prints 10 lines, each containing 5 @ characters:

@@@@@

@@@@@

...

@@@@@

**4.11 Find the Smallest Value**

import java.util.Scanner;

public class SmallestValue {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

System.out.print("Enter number of values: ");

int count = input.nextInt();

int smallest = Integer.MAX\_VALUE;

for (int i = 0; i < count; i++) {

System.out.print("Enter value: ");

int value = input.nextInt();

if (value < smallest) smallest = value;

}

System.out.println("Smallest value: " + smallest);

}

}

**4.12 Product of Odd Integers (1 to 15)**

public class ProductOdd {

public static void main(String[] args) {

long product = 1;

for (int i = 1; i <= 15; i += 2)

product \*= i;

System.out.println("Product: " + product);

}

}

**4.13 Factorials (1 to 20)**

public class Factorials {

public static void main(String[] args) {

System.out.printf("%-10s%-20s%n", "n", "n!");

for (int i = 1; i <= 20; i++) {

long factorial = 1;

for (int j = 1; j <= i; j++)

factorial \*= j;

System.out.printf("%-10d%-20d%n", i, factorial);

}

}

}

* **Problem with 100!:** It exceeds the range of long, leading to overflow.

**4.14 Modified Compound-Interest**

public class CompoundInterest {

public static void main(String[] args) {

double principal = 1000.0;

for (int rate = 5; rate <= 10; rate++) {

System.out.printf("Interest Rate: %d%%%n", rate);

for (int year = 1; year <= 10; year++) {

double amount = principal \* Math.pow(1 + rate / 100.0, year);

System.out.printf("Year %2d: %.2f%n", year, amount);

}

System.out.println();

}

}

}

**4.30 Global Warming Quiz App**

* Design 5 MCQs (balanced viewpoint).
* Use Scanner for input, use a counter for correct answers.
* Show result with feedback.

Let me know if you want the full code here.

**4.31 FairTax Program**

import java.util.Scanner;

public class FairTax {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

String[] categories = {"Housing", "Food", "Clothing", "Transportation", "Education", "Healthcare", "Vacations"};

double totalExpenses = 0;

for (String category : categories) {

System.out.print("Enter expense for " + category + ": ");

totalExpenses += input.nextDouble();

}

double fairTax = totalExpenses \* 0.23;

System.out.printf("Estimated FairTax: $%.2f%n", fairTax);

}

}

**4.32 Facebook User Base Growth**

public class FacebookGrowth {

public static void main(String[] args) {

double users = 1\_000\_000\_000;

int monthsTo1\_5B = 0, monthsTo2B = 0;

// 1.5B

double current = users;

while (current < 1\_500\_000\_000) {

current \*= 1.04;

monthsTo1\_5B++;

}

// 2B

current = users;

while (current < 2\_000\_000\_000) {

current \*= 1.04;

monthsTo2B++;

}

System.out.println("Months to reach 1.5 billion: " + monthsTo1\_5B);

System.out.println("Months to reach 2 billion: " + monthsTo2B);

}

}