Snippet 1:

```
public class Main {
public void main(String[] args) {
System.out.println("Hello, World!"); }
 }
```

What error do you get when running this code?

Error: Could not find or load main class Main
Caused by: java.lang.ClassNotFoundException: Main

Explain the Error: static word is missing and the file name should be Main.java because the class is declared in public

Fix the Error:
```
public class Main {
public static void main(String[] args) {
System.out.println("Hello, World!"); }
 }
```

Snippet 2:
```
public class Main {
    static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```
What happens when you compile and run this code?

Error: Could not find or load main class Main
Caused by: java.lang.ClassNotFoundException: Main

Explain the Error: public word is missing and the file name should be Main.java because the class is declared in public

Snippet 3:

```
public class Main {
    public static int main(String[] args) {
        System.out.println("Hello, World!");
        return 0;
    }


}
```

What error do you encounter? Why is void used in the main method?

Error:Main method must return a value of type void in class Main, please define the main method as:
public static void main(String[] args)

Explain the Error:void is use to return value.Void Method will not return any value.

Snippet 4:

```
public class Main {
    public static void main() {
        System.out.println("Hello, World!");
    }
}
```

What happens when you compile and run this code? Why is String[] args needed?

Error: Main method must return a value of type void in class Main, please define the main method as:
public static void main(String[] args)

Explain: string args is array of parameters of type string. So public static void main(String[] args) is correct

Snippet 5:

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

Can you have multiple main methods? What do you observe?

Error: No Error.

Explain: Yes we have can more than one main methods in java, but JVM will always call String[] argument main() method

Snippet 6:

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

What error occurs? Why must variables be declared?

Error: cannot find symbol

```
        int x = y + 10;
                ^
  symbol:   variable y
  location: class Main
```

Variable is must be declared because to specify the variable name and its data type.and store the value

Snippet 7:

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

What compilation error do you see? Why does Java enforce type safety?

Error: incompatible types: String cannot be converted to int

```
        int x = "Hello";
```

Compiler checks the types and gives error if wrong data type is assigned.

Snippet 8:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, World!"
    }
}
```

What syntax errors are present? How do they affect compilation?

Error: ')' expected

```
        System.out.println("Hello, World!"
                                          ^
```

Closing bracket is use to complete the syntax.
And compiler expects semicolon at end of every line.

Snippet 9:

```
public class Main {
   public static void main(String[] args) {
      int class = 10;
      System.out.println(class);
   }
}
```

What error occurs? Why can't reserved keywords be used as identifiers?

```
Error :Main.java:61: error: not a statement
      int class = 10;
      ^
Main.java:61: error: ';' expected
      int class = 10;
        ^
Main.java:61: error: <identifier> expected
      int class = 10;
            ^
Main.java:62: error: illegal start of expression
      System.out.println(class);
                    ^
Main.java:62: error: <identifier> expected
      System.out.println(class);
```

Reserved keywords are use for special uses.keywords are pre defined.

//

Snippet 10:

```
public class Main {
   public void display() {
      System.out.println("No parameters");
   }
   public void display(int num) {
      System.out.println("With parameter: " + num);
   }
   public static void main(String[] args) {
      display();
      display(5);
   }
}
```

What happens when you compile and run this code? Is method overloading allowed?

//

Snippet 11:

```
public class Main {
    public static void main(String[] args) {
        int[] arr = {1, 2, 3};
        System.out.println(arr[5]);
    }
}
```

What runtime exception do you encounter? Why does it occur?

Error: class, interface, enum, or record expected

```
}
```

Snippet 12:

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

What happens when you run this code? How can you avoid infinite loops?

Error: code goes into infinite loop

Correction: adding a break in while loop

Snippet 13:

```
public class Main {
    public static void main(String[] args) {
        String str = null;
        System.out.println(str.length());
    }
}
```

What exception is thrown? Why does it occur?

Error:Exception in thread "main" java.lang.NullPointerException: Cannot invoke "String.length()" because "<local1>" is null
        at Main.main(Main.java:109)

Explain:null denote absence of object to clear the code we have to add a string

Snippet 14:

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

What compilation error occurs? Why does Java enforce data type constraints?

Error: in.java
Main.java:115: error: incompatible types: String cannot be converted to double
        double num = "Hello";

Explain: Data types are of different sizes and values that can be stored in the variable that is use as per requirements and use cases. Data constrains ensures that data is stored in predefined rules.

Snippet 15:

```
public class Main {
    public static void main(String[] args) {
        int num1 = 10;
        double num2 = 5.5;
        int result = num1 + num2;
        System.out.println(result);
    }
}
```

What error occurs when compiling this code? How should you handle different data types in operations?

error: incompatible types: possible lossy conversion from double to int
        int result = num1 + num2;

Explain: this error comes when we try to store date in a datatype which has less size in this case we are trying to put double value in result which is int

Correct Code: public class Main {
```
    public static void main(String[] args) {
        int num1 = 10;
        double num2 = 5.5;
        double result = num1 + num2;
        System.out.println(result);
    }
}
```

Snippet 16:
```
public class Main {
    public static void main(String[] args) {
        int num = 10;
        double result = num / 4;
        System.out.println(result);
    }
}
```
What is the result of this operation? Is the output what you expected?

Correct code: public class Main {
```
    public static void main(String[] args) {
        int num = 10;
        double result = (double)num / 4;
        System.out.println(result);
    }
}
```
There are no errors. Here the 0.5 gets cancelled because num is in int. We have to convert that int to double

Snippet 17:

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        int b = 5;
        int result = a ** b;
        System.out.println(result);
    }
}
```

What compilation error occurs? Why is the ** operator not valid in Java?

error: illegal start of expression

```
        int result = a ** b;
```

Explain: should use the Math.pow() method to calculate the power of a number

Snippet 18:

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        int b = 5;
        int result = a + b * 2;
        System.out.println(result);
    }
}
```

What is the output of this code? How does operator precedence affect the result?

Output is 20. mutiplication is executed first and then adds to the result .

Snippet 19:

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        int b = 0;
        int result = a / b;
        System.out.println(result);
    }
}
```

What runtime exception is thrown? Why does division by zero cause an issue in Java?

```
Error: Exception in thread "main" java.lang.ArithmeticException: / by zero
        at Main.main(Main.java:162)
```

Explain: in java division with zero is not allowed. But it is possible in float and double

Snippet 20:

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

What syntax error occurs? How does the missing semicolon affect compilation?

Compiler excepts ‘ ; ’ at end of the line.compiler reads it as a end of line.

Snippet 21:

```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    // Missing closing brace here
}
```

What does the compiler say about mismatched braces?

```
 error: reached end of file while parsing
}
 ^
```

Snippet 22:

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

What syntax error occurs? Can a method be declared inside another method?

```
Main.java:176: error: illegal start of expression
        static void displayMessage() {
        ^
Main.java:180: error: class, interface, enum, or record expected
}
^
2 errors
```

Snippet 23:

```
public class Confusion {
    public static void main(String[] args) {
        int value = 2;
        switch(value) {
            case 1:
                System.out.println("Value is 1");
            case 2:
                System.out.println("Value is 2");
            case 3:
                System.out.println("Value is 3");
            default:
                System.out.println("Default case");
        }
    }
}
```

Error to Investigate: Why does the default case print after "Value is 2"? How can you prevent the program from executing the default case?

Use break; to close the loop if the condition is true and tell compiler to avoid further evaluation

Snippet 24:

```
public class MissingBreakCase {
    public static void main(String[] args) {
        int level = 1;
        switch(level) {
            case 1:
                System.out.println("Level 1");
            case 2:
                System.out.println("Level 2");
            case 3:
                System.out.println("Level 3");
            default:
                System.out.println("Unknown level");
        }
    }
}
```

Error to Investigate: When level is 1, why does it print "Level 1", "Level 2", "Level 3", and "Unknown level"? What is the role of the break statement in this situation?

Break is use to terminate the loop if condition is true

Snippet 25:

```
public class Switch {
    public static void main(String[] args) {
        double score = 85.0;
        switch(score) {
            case 100:
                System.out.println("Perfect score!");
                break;
            case 85:
                System.out.println("Great job!");
                break;
            default:
                System.out.println("Keep trying!");
        }
    }
}
```

Error to Investigate: Why does this code not compile? What does the error tell you about the types allowed in switch expressions? How can you modify the code to make it work?

Float and double are not allowed in switch statements .to avoid this error change double score = 85.0; to int score = 85;

Snippet 26:

```
public class Switch {
    public static void main(String[] args) {
        int number = 5;
        switch(number) {
            case 5:
                System.out.println("Number is 5");
```

```
            break;
        case 5:
            System.out.println("This is another case 5");
            break;
        default:
            System.out.println("This is the default case");
    }
  }
}
```

Error to Investigate: Why does the compiler complain about duplicate case labels? What happens when you have two identical case labels in the same switch block?

Error: error: duplicate case label

```
        case 5:
```

If some case is duplicated second case will never get executed.

Section 2: Java Programming with Conditional Statements

Question 1: Grade Classification

Write a program to classify student grades based on the following criteria:

If the score is greater than or equal to 90, print "A"

If the score is between 80 and 89, print "B"

If the score is between 70 and 79, print "C"

If the score is between 60 and 69, print "D"

If the score is less than 60, print "F"

Solution:

```
class Grades{
	public static void main (String args[]) {

		int score=45;

		if(score >= 90){
			System.out.println("Grade A");
		}
		else if(score>= 80 && score>89){
			System.out.println("Grade B");
		}
		else if(score>= 70 && score>79){
			System.out.println("Grade C");
		}
		else if(score>= 60 && score>69){
			System.out.println("Grade C");
		}
		else if(score<60){
			System.out.println("Grade F");
		}
		else{
			System.out.println("not valid");
		}
	}
}
```

Question 2: Days of the Week

Write a program that uses a nested switch statement to print out the day of the week based on an
integer input (1 for Monday, 2 for Tuesday, etc.). Additionally, within each day,
print whether it is a weekday or weekend.

Solution:

```
class week{
        public static void main (String args[]) {
                int day = 7;
switch (day) {
  case 1:
    System.out.println("Monday");
    break;
  case 2:
    System.out.println("Tuesday");
    break;
  case 3:
    System.out.println("Wednesday");
    break;
  case 4:
    System.out.println("Thursday");
    break;
  case 5:
    System.out.println("Friday");
    break;
  case 6:
    System.out.println("Saturday");
    break;
  case 7:
    System.out.println("Sunday");
        System.out.println("week end");
    break;
}
        }
}
```

Question 3: Calculator

Write a program that acts as a simple calculator. It should accept two numbers and an operator (+, -, *, /) as input. Use a switch statement to perform the appropriate operation. Use nested if else to check if division by zero is attempted and display an error message.

Solution:

```
import java.util.Scanner;

class calculator {
  public static void main(String args[]) {

    char operator;
    Double number1, number2, result;

    Scanner input = new Scanner(System.in);

    System.out.println("Choose an operator: +, -, *,/");
    operator = input.next().charAt(0);

    //user to enter numbers
    System.out.println("Enter first number");
    number1 = input.nextDouble();

    System.out.println("Enter second number");
    number2 = input.nextDouble();

    switch (operator) {

      //  addition
      case '+':
        result = number1 + number2;
        System.out.println(number1 + " + " + number2 + " = " + result);
        break;

      //  subtraction
      case '-':
        result = number1 - number2;
```

```
        System.out.println(number1 + " - " + number2 + " = " + result);
        break;

      //  multiplication
      case '*':
        result = number1 * number2;
        System.out.println(number1 + " * " + number2 + " = " + result);
        break;

      // division
      case '/':
        if (number2 == 0) {
            System.out.println("Error: Division by zero is not allowed.");

         } else {
               result = number1 / number2;
         }
        break;

      default:
        System.out.println("Invalid operator!");
        break;
    }
  }
}
```

Question 4: Discount Calculation
Write a program to calculate the discount based on the total purchase amount. Use the following criteria:
If the total purchase is greater than or equal to Rs.1000, apply a 20% discount.
If the total purchase is between Rs.500 and Rs.999, apply a 10% discount.
If the total purchase is less than Rs.500, apply a 5% discount.
Additionally, if the user has a membership card, increase the discount by 5%.

Solution:

```
import java.util.Scanner;

class DiscountCalculator {

    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // total purchase amount
        System.out.print("Enter the total purchase amount (Rs): ");
        double totalAmount = scanner.nextDouble();

        // membership status
        System.out.print("Do you have a membership card? (yes/no): ");
        String membership = scanner.next();

        // Calculate base discount
        double discountPercentage;
        if (totalAmount >= 1000) {
            discountPercentage = 20.0;
        } else if (totalAmount >= 500) {
            discountPercentage = 10.0;
        } else {
            discountPercentage = 5.0;
        }

        // Adjust discount if the user has a membership card
        if (membership.equalsIgnoreCase("yes")) {
            discountPercentage += 5.0;
        }

        // Calculate discount
        double discountAmount = (discountPercentage / 100) * totalAmount;
        double finalAmount = totalAmount - discountAmount;
```

```
        // Display the discount and final amount
        System.out.println("Discount applied: " + discountPercentage + "%");
        System.out.println("Discount amount: Rs " + discountAmount);
        System.out.println("Final amount to be paid: Rs " + finalAmount);

    }
}
```

Question 5: Student Pass/Fail Status with Nested Switch Write a program that determines whether a student passes or fails based on their grades in three subjects. If the student scores more than 40 in all subjects, they pass. If the student fails in one or more subjects, print the number of subjects they failed in.

```
class PassFail {
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);

        // Input grades
        System.out.print("Enter grade for Subject 1: ");
        int subject1 = scanner.nextInt();

        System.out.print("Enter grade for Subject 2: ");
        int subject2 = scanner.nextInt();

        System.out.print("Enter grade for Subject 3: ");
        int subject3 = scanner.nextInt();

        int failedSub = 0;

        // Check subject 1
        switch (subject1 > 40 ? 1 : 0) {
            case 0:
                failedSub++;
                // Check subject 2
                switch (subject2 > 40 ? 1 : 0) {
                    case 0:
                        failedSub++;
                        // Check subject 3
                        switch (subject3 > 40 ? 1 : 0) {
                            case 0:
                                failedSub++;
                                System.out.println("Failed in " + failedSub + " subjects.");
```

```
                break;
            case 1:
                System.out.println("Failed in " + failedSub + " subjects.");
                break;
        }
        break;
    case 1:
        // Check subject 3
        switch (subject3 > 40 ? 1 : 0) {
            case 0:
                failedSub++;
                System.out.println("Failed in " + failedSub + " subjects.");
                break;
            case 1:
                System.out.println("Failed in " + failedSub + " subjects.");
                break;
        }
        break;
}
break;
case 1:
    // Check subject 2
    switch (subject2 > 40 ? 1 : 0) {
        case 0:
            failedSub++;
            // Check subject 3
            switch (subject3 > 40 ? 1 : 0) {
                case 0:
                    failedSub++;
                    System.out.println("Failed in " + failedSub + " subjects.");
                    break;
                case 1:
                    System.out.println("Failed in " + failedSub + " subjects.");
                    break;
            }
            break;
        case 1:
            // Check subject 3
            switch (subject3 > 40 ? 1 : 0) {
                case 0:
                    failedSub++;
                    System.out.println("Failed in " + failedSub + " subjects.");
                    break;
                case 1:
```

```
                    System.out.println("Passed in all subjects.");
                    break;
                }
                break;
            }
            break;
        }
    }
}
```