// Snippet 1:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

// What error do you get when running this code?

// Ans: Main method is not static in class Main.

// Snippet 3:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, World!");

}

}

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

// Ans: main method must return void instead of int , because it the entry point of java program.

// Snippet 4:

public class Main {

public static void main(String args[]) {

System.out.println("Hello, World!");

}

}

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

// Ans: Incorrect main method signature,"String args[]" was missing.

// 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?

// 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?

// Ans: can't find symbol, y is not defined.

// Snippet 7:

public class Main {

public static void main(String[] args) {

int x = "Hello";

System.out.println(x);

}

}

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

// Ans: Incompatible types, string can't be converted to int.

// Snippet 8:

public class Main {

public static void main(String[] args) {

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

}

}

// ´éÀ What syntax errors are present? How do they affect compilation?

// Snippet 9:

public class Main {

public static void main(String[] args) {

int class = 10;

System.out.println(class);

}

}

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

// Ans: identifier expected, illegal start of expression

// Snippet 10:

public class Main {

public static void display() {

System.out.println("No parameters");

}

public static void display(int num) {

System.out.println("With parameter: " + num);

}

public static void main(String[] args) {

display();

display(5);

}

}

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

// Ans: non-static method display(int) cannot be referenced from a static context display(5);

// Snippet 11:

public class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 3};

System.out.println(arr[5]);

}

}

// ´éÀ What runtime exception do you encounter? Why does it occur?

// Ans: Index 5 out of bounds for length 3, because array has three ele.

// Snippet 12:

public class Main {

public static void main(String[] args) {

while (true) {

System.out.println("Infinite Loop");

}

}

}

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

// Ans: infinite loop , because in while loop condition is always true.

// Snippet 13:

public class Main {

public static void main(String[] args) {

String str = null;

System.out.println(str.length());

}

}

// ´éÀ What exception is thrown? Why does it occur?

// Ans: java.lang.NullPointerException, string str is set to null.

// Snippet 14:

public class Main {

public static void main(String[] args) {

double num = "Hello";

System.out.println(num);

}

}

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

// Ans: incompatible types: String cannot be converted to double.

// Snippet 15:

public class Main {

public static void main(String[] args) {

int num1 = 10;

double num2 = 5.5;

int result = num1 + (int) num2;

System.out.println(result);

}

}

// ´éÀ What error occurs when compiling this code? How should you handle different data types

// in operations?

// Ans: incompatible types: possible lossy conversion from double to int

// Snippet 16:

public class Main {

public static void main(String[] args) {

int num = 10;

double result = num /(double) 4;

System.out.println(result);

}

}

// ´éÀ What is the result of this operation? Is the output what you expected?

// Ans: 2.0, No

// 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);

}

}

// ´éÀ What compilation error occurs? Why is the \*\* operator not valid in Java?

// Ans: illegal start of expression, it is derefrencing operator used in c++ , however java dosen't have pointers.

// 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);

}

}

// ´éÀ What is the output of this code? How does operator precedence affect 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);

}

}

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

// Ans: In Java, division by zero with integers (int or long) is not allowed ,

// Java cannot perform this operation and throws an ArithmeticException at runtime. to avoid this we can put condition check value 0 at condition.

// Snippet 20:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, World")

}

}

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

// Ans: ';' represents termination of the statement.

// Snippet 21:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, World!");

// Missing closing brace here

}

// ´éÀ What does the compiler say about mismatched braces?

// Snippet 22:

public class Main {

public static void main(String[] args) {

static void displayMessage() {

System.out.println("Message");

}

}

}

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

// Ans: in java,methods cannot be defined inside other methods. Methods must be defined directly inside a class.

// Snippet 23:

public class Main {

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");

}

}

}

// ´éÀ Error to Investigate: Why does the default case print after "Value is 2"? How can you prevent

// the program from executing the default case?

// Ans: by putting break statement after each case.

// Snippet 24:

public class Main {

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");

}

}

}

// ´éÀ 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?

// Snippet 25:

public class Main {

public static void main(String[] args) {

double score = 85.0;

switch((int) score) {

case 100:

System.out.println("Perfect score!");

break;

case 85:

System.out.println("Great job!");

break;

default:

System.out.println("Keep trying!");

}

}

}

// ´éÀ 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?

// Ans: switch statement does not support float or double . so, we can type cast score to int.

// Snippet 26:

public class Main {

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");

}

}

}

// ´éÀ 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?

// Ans: compilation error: duplicate case label