Snippet 1

```
public class Snippet_1 {
 public void main(String[] args) {
 System.out.println("Hello, World!");
 }
}
 /*
```

In this class main method is not static so we should it static.

Error: Main method is not static in class Snippet_1, please define the main method as:

```
   public static void main(String[] args)
 */
```

Snippet 2

```
public class Snippet_2{
 static void main(String[] args) {
 System.out.println("Hello, World!");
 }
}
/*
```

At the time of compilation , program was successfully compiled but in run time we have put public class .

Error: Main method not found in class Snippet_2, please define the main method as:

```
   public static void main(String[] args)
```

or a JavaFX application class must extend
javafx.application.Application

*/

Snippet 3

```
public class Snippet_3 {
 public static int main(String[] args) {
 System.out.println("Hello, World!");
 return 0;
 }
 /*
        The `void` is a reserved type used to specify that a method does not return any data type.
        error: reached end of file while parsing
        }
  ^
 }
 */
```

Snippet 4

```
public class Snippet_4 {
 public static void main() {
 System.out.println("Hello, World!");
 }
```

```
}
/*
```

At the time compilation program was successfully complied but in a program program is not in correct format .There is not correct syntax of main class.

Error: Main method not found in class Snippet_4, please define the main method as:

public static void main(String[] args)

or a JavaFX application class must extend javafx.application.Application

String.args[] is important because it is the part of the syntax in java.

```
/*
```

Snippet 5

```
public class Snippet_5 {
 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");
 }
}
/*
```

It is successfully compiled as well as executed and gives ouput which is "Main method with String[]args"

*/

Snippet 6

```
public class Snippet_6 {
 public static void main(String[] args) {
 int x = y + 10;
 System.out.println(x);
 }
}
```

/*

It gives error, because declaration is importnat to a variable. Variable must be declare. because it shows that the value is integer type or string.

error: cannot find symbol

int x = y + 10;

*/

Snippet 7

```
public class Snippet_7 {
 public static void main(String[] args) {
 int x = "Hello";
 System.out.println(x);
```

```
 }
}
/*          We have to do a type casting for a string value.
             Error: incompatible types: String cannot be converted to
int
            int x = "Hello";
*/
```

Snippet 8

```
public class Snippet_8 {
 public static void main(String[] args) {
 System.out.println("Hello, World!"
 }
}
/*    It is not in a correct syntax .
```

Each program having a appropriate syntax. In this program Bracket is missing .

```
            error: ')' expected
             System.out.println("Hello, World!"
*/
```

Snippet 9

```
//Snippet 9:
public class Snippet_9 {
 public static void main(String[] args) {
 int class = 10;
 System.out.println(class);
 }
}
/*
```

Using reserved keyword (class) as a variable name it gives following types of errors can be occurred because confuse other programmers or make it difficult for the programming language to function properly

```
Snippet_9.java:4: error: not a statement
 int class = 10;
 ^
Snippet_9.java:4: error: ';' expected
 int class = 10;
        ^
Snippet_9.java:4: error: <identifier> expected
 int class = 10;
                ^
Snippet_9.java:5: error: illegal start of expression
 System.out.println(class);
                                ^
Snippet_9.java:5: error: <identifier> expected
```

```
        System.out.println(class);
*/
```

Snippet 10

```
//Snippet 10:
public class Snippet_10 {
 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);
 }
}
/*
```

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

If we want to call a non static method into static method then we have to create a instance variable of that class.

And Method overloading is allowed You can have any number of main methods in a class by method overloading.

Error: non-static method display() cannot be referenced from a static context

display();

^

Snippet_10.java:11: error: non-static method display(int) cannot be referenced from a static context

display(5);

*/

Snippet 11

```
public class Snippet_11 {
 public static void main(String[] args) {
 int[] arr = {1, 2, 3};
 System.out.println(arr[5]);
 }
}
```

/*

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

It occur when an out-of-range index is detected by an array object. Because there is array range is 3 and we are passing value 5 to the array. So we get a exception while executing of program.

Exception in thread "main"
java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 3

at Snippet_11.main(Snippet_11.java:4)

*/

Snippet 12

```
public class Snippet_12 {
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 loop

Ans: Its goes to the infinte loop.

So we can write as a

```
boolean b = true;
while(b){
        System.out.println("Inside while");
        b=false;
```

*/

Snippet 13

```
public class Snippet_13 {
public static void main(String[] args) {
```

```
 String str = null;
 System.out.println(str.length());
 }
}
```

/*At the time of run it gives exception:

We initialized a str to the null. So when length method is call that time lenth method find of the string length.

So that time it gives null pointer exception.

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

at Snippet_13.main(Snippet_13.java:4)

*/

Snippet 14

```
public class Snippet_14 {
 public static void main(String[] args) {
double num = "Hello"; System.out.println(num);
 }
 }
```

/*In this program type casting is required for string value. We did not pass string value to integer.

error: incompatible types: String cannot be converted to double

double num = "Hello"; System.out.println(num);

*/

Snippet 15

```
public class Snippet_15 {
 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?
We can handle by Type casting.
error: incompatible types: possible lossy conversion from double to int
 int result = num1 + num2;
*/
```

Snippet 16

```
public class Snippet_16 {
 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?

Ans: number divided by 4, so output will be 2.0.

*/

Snippet 17

```
public class Snippet_17 {
 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?

ANS: ** this operator is not a valid arithamatic operator.

error: illegal start of expression

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

*/

Snippet 18:

```
public class Snippet_18 {
 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?

ANS: Output is 20.

10+5*2

10+10

20

*/

Snippet 19

```
public class Snippet_19 {
 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?

Any number divided by zero is undefined. So it gives an exception.

ANS: Exception in thread "main" java.lang.ArithmeticException: / by zero

at Snippet_19.main(Snippet_19.java:5)

*/

Snippet 20

```
public class Snippet_20 {
 public static void main(String[] args) {
 System.out.println("Hello, World")
 }
}
```

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

ANS: error: ';' expected

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

It is a syntax error.

*/

Snippet 21

```
public class Snippet_21 {
 public static void main(String[] args) {
 System.out.println("Hello, World!");
 // Missing closing brace here
}
 /* What does the compiler say about mismatched braces?
```

ANS: A variable defined within the braces that mark the start and end of a method has *method scope*.

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

Snippet 22:

```
public class Snippet_22 {
 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?

We can pass the result of a method call as an argument to another method.

error: illegal start of expression

static void displayMessage() {

Snippet_22.java:7: error: class, interface, enum, or record expected

}

*/

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?

Ans: Value is 2

Value is 3

Default case

Using break statement we can prevent the program from executing the default case.

*/

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?
        Ans:  because there is no break statement to break the loop.
*/
```

Snippet 25

```
public class Snippet_25 {
 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!");
 }
 }
}
/*
```

ANS: we can't use float/double type value in switch-case(due to imprecise calculation).

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?

```
Snippet_25.java:4: error: patterns in switch statements are a preview feature and are disabled by default.
                switch(score) {
                       ^
  (use --enable-preview to enable patterns in switch statements)
Snippet_25.java:5: error: constant label of type int is not compatible with switch selector type double
                case 100:
                     ^
Snippet_25.java:8: error: constant label of type int is not compatible with switch selector type double
                case 85:
*/
```

Snippet 26

```
public class Snippet_26 {
 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");
 }
 }
}
```

/* Q. 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: It gives an error which is duplicate case.

error: duplicate case label

case 5:

^

*/