Snippet 1

Original Code:

public class Main {

public void main(String[] args) {

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

}

}

Error: Main method not found.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: The main method must be public static void to be recognized as the entry point of the program.

Snippet 2

Original Code:

public class Main {

static void main(String[] args) {

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

}

}

Error: Main method not found.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: The main method must be public static void to be recognized as the entry point of the program.

Snippet 3

Original Code:

public class Main {

public static int main(String[] args) {

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

return 0;

}

}

Error: Main method must return void.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: The main method must have a void return type as it is the entry point of the Java application.

Snippet 4

Original Code:

public class Main {

public static void main() {

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

}

}

Error: Main method not found.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: The main method must accept a String[] argument to be recognized as the entry point.

Snippet 5

Original Code:

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");

}

}

Output:

Main method with String[] args

Explanation: Java does not call the overloaded main method. Only the public static void main(String[] args) method is executed.

Snippet 6

Original Code:

public class Main {

public static void main(String[] args) {

int x = y + 10;

System.out.println(x);

}

}

Error: Cannot find symbol.

Corrected Code:

public class Main {

public static void main(String[] args) {

int y = 5;

int x = y + 10;

System.out.println(x);

}

}

Explanation: Variables need to be declared before use.

Snippet 7

Original Code:

public class Main {

public static void main(String[] args) {

int x = "Hello";

System.out.println(x);

}

}

Error: Incompatible types.

Corrected Code:

public class Main {

public static void main(String[] args) {

String x = "Hello";

System.out.println(x);

}

}

Explanation: Variable types must match the assigned value.

Snippet 8

Original Code:

public class Main {

public static void main(String[] args) {

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

}

}

Error: Reached end of file while parsing.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: A missing closing parenthesis causes a syntax error.

Snippet 9

Original Code:

public class Main {

public static void main(String[] args) {

int class = 10;

System.out.println(class);

}

}

Error: Not a statement.

Corrected Code:

public class Main {

public static void main(String[] args) {

int myClass = 10;

System.out.println(myClass);

}

}

Explanation: Reserved keywords cannot be used as identifiers.

Snippet 10

Original Code:

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);

}

}

Error: Non-static method cannot be referenced from a static context.

Corrected Code:

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) {

Main obj = new Main();

obj.display();

obj.display(5);

}

}

Explanation: Non-static methods must be called on an instance of the class.

Snippet 11

Original Code:

public class Main {

public static void main(String[] args) {

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

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

}

}

Error: ArrayIndexOutOfBoundsException.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

System.out.println(arr[2]); // Access valid index

}

}

Explanation: Array indices must be within bounds.

Snippet 12

Original Code:

public class Main {

public static void main(String[] args) {

while (true) {

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

}

}

}

Behavior: Runs indefinitely.

Correction: Add a termination condition or interrupt manually.

Snippet 13

Original Code:

public class Main {

public static void main(String[] args) {

String str = null;

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

}

}

Error: NullPointerException.

Corrected Code:

public class Main {

public static void main(String[] args) {

String str = "Hello";

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

}

}

Explanation: Accessing methods on a null reference causes NullPointerException.

Snippet 14

Original Code:

public class Main {

public static void main(String[] args) {

double num = "Hello";

System.out.println(num);

}

}

Error: Incompatible types.

Corrected Code:

public class Main {

public static void main(String[] args) {

String num = "Hello";

System.out.println(num);

}

}

Explanation: Type mismatch between variable and assigned value.

Snippet 15

Original Code:

public class Main {

public static void main(String[] args) {

int num1 = 10;

double num2 = 5.5;

int result = num1 + num2;

System.out.println(result);

}

}

Error: Possible lossy conversion.

Corrected 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);

}

}

Explanation: Type conversion issues when mixing int and double.

Snippet 16

Original Code:

public class Main {

public static void main(String[] args) {

int num = 10;

double result = num / 4;

System.out.println(result);

}

}

Output:

2.5

Explanation: Integer division result is promoted to double when assigned to a double variable.

Snippet 17

Original Code:

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 5;

int result = a \*\* b;

System.out.println(result);

}

}

Error: Cannot find symbol.

Corrected Code:

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 5;

int result = (int) Math.pow(a, b);

System.out.println(result);

}

}

Explanation: \*\* is not a valid operator in Java; use Math.pow for exponentiation.

Snippet 18

Original Code:

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 5;

int result = a + b \* 2;

System.out.println(result);

}

}

Output:

20

Explanation: Multiplication has higher precedence than addition.

Snippet 19

Original Code:

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 0;

int result = a / b;

System.out.println(result);

}

}

Error: ArithmeticException: / by zero.

Corrected Code:

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 1; // Change b to a non-zero value

int result = a / b;

System.out.println(result);

}

}

Explanation: Division by zero is not allowed in Java.

Snippet 20

Original Code:

public class Main {

public static void main(String[] args) {

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

}

}

Error: Reached end of file while parsing.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Explanation: Missing semicolon at the end of the statement.

Snippet 21

Original Code:

public class Main {

public static void main(String[] args) {

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

// Missing closing brace here

}

}

Error: Reached end of file while parsing.

Corrected Code:

public class Main {

public static void main(String[] args) {

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

}

}

Snippet 22

Original Code:

public class Main {

public static void main(String[] args) {

static void displayMessage() {

System.out.println("Message");

}

}

}

Error: Illegal modifier for the local class.

Corrected Code:

public class Main {

public static void displayMessage() {

System.out.println("Message");

}

public static void main(String[] args) {

displayMessage();

}

}

Snippet 23

Original Code:

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");

}

}

}

Output:

Value is 2

Value is 3

Default case

Explanation: Missing break statements cause fall-through.

Corrected Code:

public class Confusion {

public static void main(String[] args) {

int value = 2;

switch(value) {

case 1:

System.out.println("Value is 1");

break;

case 2:

System.out.println("Value is 2");

break;

case 3:

System.out.println("Value is 3");

break;

default:

System.out.println("Default case");

}

}

}

Snippet 24

Original Code:

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");

}

}

}

Output:

Level 1

Level 2

Level 3

Unknown level

Explanation: Missing break statements cause fall-through.

Corrected Code:

public class MissingBreakCase {

public static void main(String[] args) {

int level = 1;

switch(level) {

case 1:

System.out.println("Level 1");

break;

case 2:

System.out.println("Level 2");

break;

case 3:

System.out.println("Level 3");

break;

default:

System.out.println("Unknown level");

}

}

}

Snippet 25

Original Code:

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: Type mismatch in switch expression.

Corrected Code:

public class Switch {

public static void main(String[] args) {

int score = 85; // Use int instead of double

switch(score) {

case 100:

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

break;

case 85:

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

break;

default:

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

}

}

}

Snippet 26

Original Code:

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: Duplicate case labels.

Corrected Code:

public class Switch {

public static void main(String[] args) {

int number = 5;

switch(number) {

case 5:

System.out.println("Number is 5");

break;

// Removed duplicate case

default:

System.out.println("This is the default case");

}

}

}