Shreyash\_Banayat\_Juhu\_Lab\_Assignment

Snippet 1:

Error: The main method missing the static keyword. The compiler search for the main method and starts executing from their. And there can be only one static method in a java program.

public class Main {

public static void main(String[] args) {

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

}

}

Snippet 2:

Error: The code gets compiled but does not run because the main method does not have any access specifier so it will automatically take ‘private’ as default access specifier. The JVM will not be able to find the private method.

public class Main {

public static void main(String[] args) {

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

}

}

Snippet 3:

Error: The code compile successfully but error occur while running.

‘Main method must return a value of type void in Main class’.

The main method should return a void value because as it terminates program ends and if we return any value at the end JVM will not able to anything with it.

public class Main {

public static void main(String[] args) {

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

}

}

Snippet 4:

Error: The code compiles perfectly but throes error while running.

‘String args[]’ is an array of type String. It is used to give command line arguments to a java program.

public class Main {

public static void main(String args[]) {

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

}

}

Snippet 5:

Error: A java program cannot have multiple ‘Overloaded’ main method in a program .

The Overloaded main method did not run.

Snippet 6:

Error: The compiler could not identify the symbol ‘y’ as it is not declared.

We have to declare the variable to let the compiler know that it is an identifier and what type of identifier it is so it can handle it accordingly.

public class Main {

public static void main(String[] args) {

int y=0;

int x = y + 10;

System.out.println(x);

}

}

Snippet 7:

Error: Incompatible types : String cannot be converted to int.

Java enforces type safety because every type has its own different size, methods and react differently on operators.

public class Main {

public static void main(String[] args) {

int x = 10;

System.out.println(x);

}

}

Snippet 8:

Error: A Right parenthesis and semicolon are missing from the snippet.

A semicolon is important to insure that an expression or statement is ended.

Parenthesis are used to confine an expression or statement.

public class Main {

public static void main(String[] args) {

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

}

}

Snippet 9:

Error: Illegal start of statement, Not an statement, Identifier expected

Reserved keywords cannot be used as they are already assigned a work within a language redeclaring or reassigning it may cause issues.

public class Main {

public static void main(String[] args) {

int num= 10;

System.out.println(num);

}

}

Snippet 10:

Error: When we compile and run the it throws an error that “non-static method cannot be referenced in static context”

Yes method overloading is allowed here.

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);

}

}

Snippet 11:

Error: We get ArrayIndexOutOfBound Exception after running the snoippet.

We get this exception because the array is of size 3 and the element asked is of 5th index .

public class Main {

public static void main(String[] args) {

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

System.out.println(arr[0]);

}

}

Snippet 12:

Error: After running the snippet we get stuck in an infinite loop that prints “Infinite loop”.

To avoid stuck in an Infinite loop we should always add a stopping condition.

public class Main {

public static void main(String[] args) {

boolean flag=true;

while (flag) {

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

flag=false;

}

}

}

Snippet 13:

Error: NullPointerException is thrown.

This exception is thrown because string variable is initialized to a null value.

public class Main {

public static void main(String[] args) {

String str = "Juhu";

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

}

}

Snippet 14:

Error: incompatible types: String cannot be converted to double.

Java enforces type safety because every type has its own different size, methods and react differently on operators.

public class Main {

public static void main(String[] args) {

double num = 24.14;

System.out.println(num);

}

}

Snippet 15:

Error: Incompatible types: possible lossy conversion from double to int.

While handling different datatypes we can ‘type cast’ one datatype into another one.

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);

}

}

Snippet 16:

The result of the snippet is 2.0.

The expected result was 2.5. To get expected result we have to type-cast ‘num’

public class Main {

public static void main(String[] args) {

int num = 10;

double result =(double) num / 4;

System.out.println(result);

}

}

Snippet 17:

Error: illegal start of expression

int result = a \*\* b;

The error is mostly because of ‘\*\*’ are used for pointer and java does not have pointers.

Snippet 18:

The output of the code is 20. The given equation is ‘a+b\*2’ according to “Order of Precede” multiplication will occur first and then addition.

Snippet 19:

Error: The runtime exception thrown is ‘ArithmeticException’ .

If we try to divide any number by zero program will throw this exception because it results in infinity. (see Baeldung)

Snippet 20:

Error: Expected ‘:’ at the end.

Semicolon ‘;’ are used to terminate a statement. If it is missing after a statement compiler will continue thinking that it is the same line this could mix two completely different statements and will cause an error or it could also cause an logical error.

Snippet 21:

Error: ‘Reached end of file while parsing’

Opening and ending curly braces are important as they define the block. Every opening curly brace should have an ending curly brace.

Snippet 22:

Error: ‘illegal start of expression’

In Java we cannot define method within a method.

Snippet 23:

The default value print after value 2 is because there is no break statement use to stop the process after value is matched. To stop printing the default value add break statements after each case.

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 :

It prints all the values because of no break statements after the cases. To stop printing all the output put break statement after each case .

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:

Java does noy support double as case value .

Snippet 26:

If you have duplicate case labels in a block then the compiler will not under which label to print in case it match the condition.