**Snippet1:**

class Main {

public void main(String[] args) {

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

}

}

**Ans**: Main method is not static in class

class Main is public, should be declared in a file named Main.java.

**Correct program:**

class Main {

public static void main(String[] args) {

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

}

}

**Snippet2:**

class Main {

static void main (String[] args) {

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

} }

**Ans**: when we compile the code error cause Because of the Public which will declare in class

Run time error cause because due to not declare public in main the the error was cause as-> Main method not found in class Main,

**Correct program**:

class Main {

public static void main (String[] args) {

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

}

}

**Snippet3:**

class Main {

public static int main (String [] args) {

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

return 0;

} }

**Ans**: Error cause Because of the Public which will declare in class

Main method must return a value of type void in class Main,

Main method doesn’t return anything its return type is void.

**Correct program:**

class Main {

public static void main(String[] args) {

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

return 0;

}

}

**Snippet4:**

public class Main {

public static void main() {

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

} }

**Error**: compile error due to class is declare as public

Runtime error due to string[]args not declare in main method.

**Ans:** String args is used to pass command line argument to a java. It allows external data to be passed into the program when it start.

**Corrrect program:**

class Main {

public static void main(String[]args) {

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

}

}

**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");

} }

Ans: java allows method overloading which means you can have multiple method with same name nut different parameter

**Snippet 6:**

public class Main {

public static void main(String[] args) {

int x = y + 10;

System.out.println(x);

} }

**Error**:variable y should be not declare and initialized.

Variable must be declare for assigning a value for performing specific operation.

**Correct program:**

class Main {

public static void main(String[] args) {

int x;

int y=6;

x = y + 10;

System.out.println(x);

}

}

**Snippet 7:**

public class Main {

public static void main(String[] args) {

int x = "Hello";

System.out.println(x);

} }

**Error:** we can pass a variable as string but we can pass a datatype as int that’s why this program causes a error we correct that error using declare x variable using a String instead of int.

Java checks type safety because ensure that any object a method may try to manipulate is of proper type.

**Correct program**:

class Main {

public static void main(String[] args) {

String x = "Hello";

System.out.println(x);

}

}

**Snippet 8:**

public class Main {

public static void main(String[] args) {

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

} }

**Error**:brackets are missing after world!”.

**Snippet 9:**

public class Main {

public static void main(String[] args) {

int class = 10;

System.out.println(class);

} }

**Error**: in this code identifier expected class is predefined keyword in java.keywords are reserved wordsin java so cannot be used as identifier.

**Correct program:**

class Main {

public static void main(String[] args) {

int a = 10;

System.out.println(a);

}}

**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);

} }

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

display();

in this program we will not create any instance or object for class 1st we need to create and then call it using new keyword.method overloading allows you to define a multiple methods with the same name but different parameter.

**Correct program:**

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 a = new Main();

a.display();

a.display(5);

}

}

**Snippet 11:**

public class Main {

public static void main(String[] args) {

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

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

} }

Error: Index 5 out of bounds for length 3 at Main.main(assi.java:5)

It compile successfully but at the time running it have error due to we pass a lenth of array as 0 to 3 and print the index number was 5.

**Correct code:**

class Main {

public static void main(String[] args) {

int[] arr = {1, 2, 3,7,9,8};

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

}

}

**Snippet 12:**

public class Main {

public static void main(String[] args) {

while (true) {

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

} } }

Just we want to put if condition for exit the loop after some iteration.use break statement to exit the loop

**Correct program:**

class Main {

public static void main(String[] args) {

int i=0

while (true) {

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

i++;

if(i>=5){

break;

}

}

}

}

**Snippet 13:**

public class Main {

public static void main(String[] args) {

String str = null;

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

} }

**Error:** Cannot invoke "String.length()" because "<local1>" is null at Main.main(assi.java:5)

Because string declare null character its not showing any null null means a nothing in under the stringfor correct this we will put string instead of null keyword to show tring leght.

**Correct code**:

class Main {

public static void main(String[] args) {

String str = "Trupti";

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

}

}

**Snippet 14:**

public class Main {

public static void main(String[] args) {

double num = "Hello";

System.out.println(num);

} }

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

String will never accept any double value. Double is interger format datatype which should never accept any string value.

Correct code:

class Main {

public static void main(String[] args) {

String num = "Hello";

System.out.println(num);

}

}

**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); } }

**Error:** compatible types: possible lossy conversion from double to int

Error cousing because we declare result variable as int and num2 as double.double cannot directly assign to the int that’s why we using typecasting.

**Correct code:**

class Main {

public static void main(String[] args) {

int num1 = 10;

double num2 = 5.5;

int result =(int) (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); } }

**Ans:**This code will successfully compiled and run

With 2.0 output

**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); } }

**Error:** illegal start of expression

The \*\* is not valid expression for java for multiplication we usw single \* and for power we use math.pow() method

**Correct code:**

class Main {

public static void main(String[] args) {

int a = 10;

int b = 5;

int result = a \*b;

System.out.println(result);

}

}

**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); } }

Output:20

In operator precedence it will 1st perform brackets operation 2nd multiplication

**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); } }

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

at Main.main(assi.java:6)

, when you attempt to divide a number by zero, it results in a **runtime exception** known as **ArithmeticException.** This exception occurs because dividing any number by zero is undefined in mathematics.

**Snippet 20:**

public class Main {

public static void main(String[] args) {

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

} }

**Error**: ‘;’ Expected.

: semicolon is used for terminated the every staenment. Missing the semicolon is meaning that we caanot terminate the statement after completing it.

**Coorect program:**

public class Main {

public static void main(String[] args) {

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

} }

**Snippet 21:**

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.

public class Main {

public static void main(String[] args) {

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

}

}

**Snippet 22:**

public class Main {

public static void main(String[] args) {

static void displayMessage() {

System.out.println("Message"); } } }

**Error**: illegal start of expression static void displayMessage() {

^assi.java:8: 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"); } } }

**Ans:** The issue in your code lies in the absence of break statements within the switch cases.

To prevent this error add break statement after every case statement

**Correct code**:

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:**

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:** Exception in thread "main" java.lang.ArithmeticException: / by zero at Main.main(assi.java:6)

**Ans**:If we see we not put break statement after every case satement that’s why when we use level 1 only it will show all levels also with default statement. Without Breck statement execution will continuously accepting cases which will put in code.

**Correct code:**

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:**

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:** error: constant label of type int is not compatible with switch selector type double

case 100:

ans: the switch statemenet in java does not work with double program have use double variable in switch statement. We modify the code using typecasting expressions.

Correct code: class Switch {

public static void main(String[] args) {

double score = 85.0;

int scoree = (int)(score);

switch(scoree) {

case 100:

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

break;

case 85:

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

break;

default:

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

}

}

}

**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:** ssi.java:8: error: duplicate case label

case 5:

program have same case block it not acceptable by java

The program enters the switch block based on the value of the expression

It evaluates the first case 5 label and executes the associated code block, printing

**Correct code**

class Switch {

public static void main(String[] args) {

int number = 5;

switch(number) {

case 5:

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

break;

case 6:

System.out.println("This is another case 5");

break;

default:

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

}

}

}