**Assingnment 2: 21/08/2024**

**Code Snippet 1:**

public class Main {

public void main(String[] args) {

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

}

}

**Error** = Main method is not Static

**Corrected code**

public class Main {

public static void main(String[] args) {

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

}

}

**Snippet 2:**

public class Main {

static void main(String[] args) {

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

}

}

**Error:** Main method is not found in class main (public is missing)

**Corrected code**

public class Main {

public static void main(String[] args) {

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

}

}

**Snippet 3:**

public class Main {

public static int main(String[] args) {

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

return 0;

}

}

**Error:** main method must return a value of type void in class main

Void is a return type and it says it wont return any value as here int is used it must

return something.

**Corrected code**

public class Main {

public static void main(String[] args) {

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

}

}

**Snippet 4:**

public class Main {

public static void main() {

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

}

}

**Error:** main meth0d is not found

String [] args : String is class in java and String [] args is used to pass command line arguments in java program

**Corrected code**

public 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");

}

}

**Can you have multiple main methods? What do you observe?**

* Yes we can have the multiple main methods in java program usind method overloading but with different parameters list like other static methos
* Here in above snippet only standard main methos is executed if we want to execute other method we have call it explicitly in main method with different parameter list

**Corrected code**

public class Main {

public static void main(String[] args) {

System.out.println("Main method with String[] args");

main(new int[]{1, 2, 3});

}

public static void main(int[] args) {

System.out.println("Overloaded main method with int[] args");

}

}

**Snippet 6:**

public class Main {

public static void main(String[] args) {

int x = y + 10; System.out.println(x);

}

}

**Error:** compilation error ( variable is not decleared )

**Why must variables be declared?**

* Java is statically typed language that means data type of the variable must be deaclear
* This is also the reason why java is not fully object oriented language cause data types are not objects

**Corrected code:**

public class Main {

public static void main(String[] args) {

int y = 10;

int 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:** compilation erroe( incompatible type)

* X is a variable of type integer an hello the value of x is type of string data type must be same

**Why does java enforce type safety**

-java prevents programs from accessing memory in inappropriate ways by controlling the memory access of each object.

**Corrected code:**

public 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:** Syntax error( missing parenthesis)

**How syntax error affect compilation ?**

**-** sytax error prevents compiling successfully when compiler encounter syntax error it can not interpret code correctly

**Corrected code:**

public class Main {

public static void main(String[] args) {

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

}

}

**Snippet 9:**

public class Main {

public static void main(String[] args) {

int class = 10;

System.out.println(class);

} }

**Error:**  not a statement (compilation error);

* Here we can not use class as a variable there are rules to follow for variables we cant use keywords as vriables class is keyword

**Why can't reserved keywords be used as identifiers?**

* Keywords are predefines reserve words has a special meaning for the syntas and structure we should not use it because it creates ambiguity , to maintain reliability and consistency and avoiding error.

**Corrected code:**

public class Main {

public static void main(String[] args) {

int num = 10;

System.out.println(num);

} }

**Code 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 methos display() can not be referenced from a static context

* Here main method is static it can not access non sattic method directly we have to create an instance object for display methods in main metod.

**Is method overloading allowed?**

method overloading is allowed in Java Method overloading allows a class to have multiple methods with the same name but different parameters.

**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);

} }

**Code Snippet 11:**

public class Main {

public static void main(String[] args) {

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

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

} }

**Error :** Run time error ( array out of bound Exception)

* Here we have created an array of size 3 and while printing we are trying to print 5th (6) element which is not available we only 3 elements.

**Corrected code :**

public class Main {

public static void main(String[] args) {

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

System.out.println(arr[2]);

} }

**Code Snippet 12:**

public class Main {

public static void main(String[] args) {

while (true) {

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

} } }

**Error:**  when we run this code it goes in infinite loop

**How can we avoid infinte loop ?**

* We can avoid infinite loop by changing the condition by which code goes in infinite loop.

**Example code :**

public class Main {

public static void main(String[] args) {

int num = 0;

while (num < 10) {

System.out.println("Loop iteration: " + num);

num++;

}

}

}

**Code Snippet 13:**

public class Main {

public static void main(String[] args) {

String str = null;

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

} }

**Exception :** java.lang NullPointerException.

**Why does it ouccur?**

-This exception occurs because code is trying to call the length() method on a null reference. results in a NullPointerException.

**Corrected code :**

public class Main {

public static void main(String[] args) {

String str = "null";

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

}

}

**Code Snippet 14:**

public class Main {

public static void main(String[] args) {

double num = "Hello";

System.out.println(num); } }

**Error**: compilation error (Incompatible types)

* Here num variable is of type double and hello is a string data type should be same

**Corrected code:**

public class Main {

public static void main(String[] args) {

String num = "Hello";

System.out.println(num);

} }

**Code 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:** incompatible datatypes

* Num1 and num2 are of different data types

**How should you handle different data types in operations?**

* If we have different data types we should use type casting to change the data type
* As in below corrected code.

**Corrected code:**

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);

}

}

**Code snippet 16:**

public class Main {

public static void main(String[] args) {

int num = 10;

double result = num / 4;

System.out.println(result);

} }

**Result:** 2.0

-result was as expected as in java integer divide by integer pruduces integer and fractional part is discarded

**Code 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

-\*\* is not valid in java because in java for exponential we have a function.

**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);

}

}

**O/p: 20.**

**How operator precedence affects**

Java determines the order in which operators are evaluated in an expression. Operators with higher precedence are evaluated before operators with lower precedence