**Assignment 2**

Snippet 1:

public class Main {

public void main(String[] args) {

System.Out.Println("Hello, World!"); } }

Q. What error do you get when running this code?

Ans: No error occurs. O/P- Hello, World!

Snippet 2:

public class Main {

static void main(String[] args) {

System.Out.Println("Hello, World!"); } }

Q. What happens when you compile and run this code?

Ans: o/p - Hello, World!

Snippet 3:

public class Main {

public static int main(String[] args) {

System.Out.Println("Hello, World!");

return 0; } }

Q. What error do you encounter? Why is void used in the main method?

Ans: Error - please define the main method as: public static void main(String[] args)

The main() method doesn't return anything, its return type is void. main() method terminates, the Java program terminates too. If main method is not void, we will get an error.

Snippet 4:

public class Main {

public static void main() {

System.Out.Println("Hello, World!"); } }

Q. What happens when you compile and run this code? Why is String[] args needed?

Ans: please define the main method as: public static void main(String[] args)

String args in Java is used to pass command-line arguments to a Java program. It allows external data to be passed into the program when it starts, making it more versatile and adaptable to various inputs.

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"); } }

Q. Can you have multiple main methods? What do you observe?

Ans. o/p- Main method with String[] args

Snippet 6:

public class Main {

public static void main(String[] args) {

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

Q. What error occurs? Why must variables be declared?

Ans: Error- y cannot be resolved to a variable

to store the required data in the memory location in the form of variables so that we can use them in our program to perform any operation or task.

Snippet 7:

public class Main {

public static void main(String[] args) {

int x = "Hello"; System.out.println(x); } }

Q. What compilation error do you see? Why does Java enforce type safety?

Ans: Error- Type mismatch: cannot convert from String to int

Snippet 8:

public class Main {

public static void main(String[] args) {

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

}

}

Q. What syntax errors are present? How do they affect compilation?

Ans:-Error-closing bracket error

Snippet 9:

public class Main

{ public static void main(String[] args)

{

int class = 10;

System.out.println(class); }

}

• What error occurs? Why can't reserved keywords be used as identifiers?

Ans:

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); } }

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

Ans: Cannot make a static reference to the non-static method display() from the type Main

Snippet 11:

public class Main {

public static void main (String[] args)

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

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

• What runtime exception do you encounter? Why does it occur

Ans: we give array size 3 and try to print 5 which is give array out of bound exception.

Snippet 12:

public class Main {

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 loops?

Ans: if we run this code it will goes to infinite loop to stop this use break statement

Snippet 13:

public class Main {

public static void main(String[] args)

{

String str = null;

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

• What exception is thrown? Why does it occur?

Ans: it will be throw nullpointer exception we initialize str is null that’s why str length cannot be calculated

Snippet 14:

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

double num = "Hello";

System.out.println(num); } }

• What compilation error occurs? Why does Java enforce data type constraints?

Ans: compilation error will be Type mismatch we cannot convert from String to double

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); } }

• What error occurs when compiling this code? How should you handle different data types in operations

Ans: we cannot convert double to int code gives type mismatch error to handle different data type operation used typecasting

Snippet 16:

public class Main { 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: result is 2.0 no expected output is 2.5

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); } }

• What compilation error occurs? Why is the \*\* operator not valid in Java?

Ans: it gives Syntax error on token \*\* not operator in java

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); } }

• What is the output of this code? How does operator precedence affect the result?

Ans: o/p is 20 yes operator precedence affect the result firstly solve \* then +

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); } }

• What runtime exception is thrown? Why does division by zero cause an issue in Java

Ans: it gives arithematic exception for integer there is no bit pattern that can be used to store the result

Snippet 20:

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

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

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

Ans: it gives syntax error to complete BlockStatements semicolon is compulsory.

Snippet 21:

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

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

// Missing closing brace here }

• What does the compiler say about mismatched braces

Ans: compiler gives syntax error insert "}" to complete ClassBody

Snippet 22:

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

static void displayMessage()

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

• What syntax error occurs? Can a method be declared inside another metho

Ans: Syntax error on token "void", record expected

void is an invalid type for the variable displayMessage

Syntax error, insert "Identifier (" to complete MethodHeaderName

Syntax error, insert ")" to complete MethodDeclaration

Syntax error, insert ";" to complete MethodDeclaration

Syntax error, insert "}" to complete RecordBody

Syntax error, insert ")" to complete Expression

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: default case print after value 2 beacause there is no break statement

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: they print because of break statement if we cannot used break after case then it will be continuously execute another cases

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 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?

Ans: switch statement do not support double in java code to make it work convert double to int then code successfully compile

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 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: duplicate case labels not allowed because compiler get confused which one is take first or to perform operations or which block of code to execute based on the provided expression

**Section 2: Java Programming with Conditional Statements**