Day2 : Assignment2

Snippet 1

public class Main {

public void main(String[] args) {

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

}

}

• What error do you get when running this code

**Error**: Main method is not static in class Main, please define the main method as:

public static void main(String[] args)

**Explaination**:  The main method must be static so that JVM can call it without creating the object of the class to start execution.

**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 not found in class, please define the main method as:

public static void main(String[] args)

or a JavaFX application class must extend javafx.application.Application

**Explaination:** If the main method is not public then it’s access is restricted.

main method must be public because Java needs to access it when execution get started.

**Corrected code** :

public class Main {

public static void main(String[] args) {

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

}

}

Snippet3

public class Main {

public static int main(String[] args) {

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

}

}

**Error** : Main method must return a value of type void in class Snippet, please

define the main method as:

public static void main(String[] args)

**Explaination** : Main method must return a value of type void because *main()* method doesn’t return anything, its return type is void. Method is declared as int in snippet but java does not recognize while starting execution because JVM searches for a signature public static void main(String[] args) method to start the program.

**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: The main method **must** accept a String[] args parameter.

Why is String[] args needed?

Ans:  The main() method also accepts some data from the user. It accepts a group of strings, which is called a string array. It is used to hold the command line arguments in the form of string values. agrs[] is the array name, and it is of String type.

**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");

}

}

O/P: Main method with String[] args

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

Yes we can have multiple main method with different number of parameter or arguments (Overloading concept) but JVM always calls the original main() method. It does not call the overloaded main() method.

Snippet 6:

public class Snippet {

public static void main(String[] args) {

int x = y + 10;

System.out.println(x);

}

}

What error occurs? Why must variables be declared?

error: cannot find symbol

symbol: variable y

why: Variable Declaration is a statement that provides the variable name and its type, allowing the program to allocate memory for storing values.

**Corrected code**

import java.util.Scanner; // Import the Scanner class

public class Main {

public static void main(String[] args) {

Scanner inp = new Scanner(System.in); // Create a Scanner object

System.out.println("Enter number");

int y = inp.nextInt();

int x = y + 10;

System.out.println(x);

}

}

**Snippet7**

public class Main{

public static void main(String[] args) {

int x = "Hello";

System.out.println(x);

}

}

**error**: incompatible types: String cannot be converted to int

int x = "Hello";

**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!"

}

}

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

: error: ')' expected

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

^

**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);

}

}

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

error: <identifier> expected

int class = 10;

^

Keywords and reserved words in Java, such as int, class, or static, cannot be used as identifiers. These words have predefined meanings in the Java programming language.

**Corrected code**

public class Main {

public static void main(String[] args) {

int num= 10;

System.out.println(num);

}

}

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?

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

display();

^

Snippet.java:72: error: non-static method display(int) cannot be referenced from a static context

display(5);

^

Expl: we cannot call the display() method **without an object** as it is a **non-static** method. To call the non-static methods in Main (which is static) method we need to create an object of Main class(in given code)

Yes method overloading is allowed

**Corrected code :**

public class Snippet {

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) {

Snippet obj = new Snippet(); // Create an object of Snippet

obj.display(); // Call the method using the object

obj.display(5); // Call the overloaded method

}

}

**Snippet 11**

public class Main {

public static void main(String[] args) {

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

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

}

}

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of bounds for length 3

The array arr is declared as {1, 2, 3}, meaning it has indices 0, 1, and 2. It tries to access arr[5], which is out of bounds since the valid indices are only 0, 1, and 2.

**Corrected code :**

public class Main {

public static void main(String[] args) {

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

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

}

}

Snippet 12

public class Snippet {

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?**

It will run an infinite loop because of the while (true) statement.

**Corrected Code**

public class Snippet {

public static void main(String[] args) {

do {

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

} while(false);

}

}

The do-while loop executes the loop body at least once then condition is checked . Since the condition is false, the loop will not iterate again after the first

Snippet 13

public class Main {

public static void main(String[] args) {

String str = null;

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

}

}

Exception in thread "main" java.lang.NullPointerException:

The variable str initialized to null so str.length() tries to access the length of a null reference.

Since str does not point to any valid String object, this causes a NullPointerException.

**Corrected Code:**

public class Main {

public static void main(String[] args) {

String str = null;

if (str != null) {

System.out.println( " The length of string : " + str.length());

} else {

System.out.println("String is null, cannot access length.");

}

}

}

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 types: possible lossy conversion from double to int

actually num1 + num2 = result will be double and here we are trying to store double to int (larger number into smaller number ) which causes type mismatch

**Corrected code** : we can store smaller number into larger (int to double or double to double)

public class Main {

public static void main(String[] args) {

int num1 = 10;

double num2 = 5.5;

double result = num1 + num2;

System.out.println(result);

}

}

o/p: 15.5

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?

o/p: 2.0

**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?**

compile time error: illegal start of expression

int result = a \*\* b;

\*\* is exponentiation operator and it is invalid syntax in Java as Java does not have a built-in exponentiation operator.

**Corrected code:**

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 5;

double result = Math.pow(a, b); // 10^5

System.out.println(result);

}

}

O/p: 100000.0

**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?

Your Java program will compile and run correctly.

o/p : 20

Explanation Operator Precedence in Java

Multiplication (\*) has a higher precedence than addition (+).

So, b \* 2 is evaluated first, then added to a.

--------------------------------------------------------------------------------------------------------------------------

**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

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

In Java, division by zero using integers (int or long) is not allowed and it causes ArithmeticException.

**Corrected code**

public class Main {

public static void main(String[] args) {

int a = 10;

int b = 0;

if (b > 0) {

int result = a / b;

System.out.println(result);

} else {

System.out.println("Throw Error: Division by zero is not allowed.");

}

}

}

-------------------------------------------------------------------------------------------------------------------------

**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?

error: ';' expected

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

It will not compile

**Corrected code**

public class Main {

public static void main(String[] args) {

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

}

}

o/p: Hello, World

--------------------------------------------------------------------------------------------------------------------------

**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?

error: reached end of file while parsing

}

^

Explaination: Every Java class and method must have properly matched opening ({) and closing (}) braces.

**Corrected code :**

public class Main {

public static void main(String[] args) {

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

// Missing closing brace here

}

}

--------------------------------------------------------------------------------------------------------------------------

**Snippet 22:**

public class Main {

public static void main(String[] args) {

static void displayMessage() {

System.out.println("Message");

}

}

}

What syntax error occurs?

error: illegal start of expression

static void displayMessage() {

^

Can a method be declared inside another method?

--> No

In Java, you cannot define a method inside another method.

In the displayMessage() method is declared inside main(), which is not valid.

**Corrected code :**

public class Main {

// Define the method outside main()

static void displayMessage() {

System.out.println("Message");

}

public static void main(String[] args) {

displayMessage(); // Call the method inside main()

}

}

--------------------------------------------------------------------------------------

**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"?

The program matches case 2 and prints "Value is 2".Since there's no break, it continues executing case 3 and prints "Value is 3".

Again, there's no break, so it runs the default case too, printing "Default case".

How can you prevent the program from executing the default case

by using Break statment

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");

}

}

}

O/p : Value is 2

Section 2

----------------------------------------------------------------------------------------------------------------------------

Question 1: Grade Classification

Write a program to classify student grades based on the following criteria:

 If the score is greater than or equal to 90, print "A"

 If the score is between 80 and 89, print "B"

 If the score is between 70 and 79, print "C"

 If the score is between 60 and 69, print "D"

 If the score is less than 60, print "F"

import java.util.Scanner;

public class GradeClassifier {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the student's score: ");

int score = scanner.nextInt();

if (score >= 90) {

System.out.println("Grade: A");

} else if (score >= 80) {

System.out.println("Grade: B");

} else if (score >= 70) {

System.out.println("Grade: C");

} else if (score >= 60) {

System.out.println("Grade: D");

} else {

System.out.println("Grade: F");

}

scanner.close();

}

}

O/p

Enter the student's score: 70

Grade: C

-----------------------------------------------------------------------------------------------------

Question 2: Days of the Week

Write a program that uses a nested switch statement to print out the day of the week based on an integer input (1 for Monday, 2 for Tuesday, etc.). Additionally, within each day, print whether it is a weekday or weekend.

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number from 1 to 7 for the day of the week: ");

int day = scanner.nextInt();

// Outer switch to determine the day

switch (day) {

case 1:

System.out.print("Monday ");

break;

case 2:

System.out.print("Tuesday ");

break;

case 3:

System.out.print("Wednesday ");

break;

case 4:

System.out.print("Thursday ");

break;

case 5:

System.out.print("Friday ");

break;

case 6:

System.out.print("Saturday ");

break;

case 7:

System.out.print("Sunday" );

break;

default:

System.out.println("Invalid input! Please enter a number between 1 to 7.");

scanner.close();

return;

}

switch (day) {

case 1, 2, 3, 4, 5:

System.out.println("Weekday");

break;

case 6, 7:

System.out.println("Weekend");

break;

}

scanner.close();

}

}

O/P : Enter a number from 1 to 7 for the day of the week: 5

Friday Weekday

-----------------------------------------------------------------------------------------------------------------------

Question 3: Calculator

Write a program that acts as a simple calculator. It should accept two numbers and an operator

(+, -, \*, /) as input. Use a switch statement to perform the appropriate operation. Use nested ifelse to check if division by zero is attempted and display an error message.

import java.util.Scanner;

public class SimpleCalculator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter 1st number: ");

double num1 = scanner.nextDouble();

System.out.print("Enter an operator (+, -, \*, /): ");

char operator = scanner.next().charAt(0);

System.out.print("Enter 2nd number: ");

double num2 = scanner.nextDouble();

double res;

switch (operator) {

case '+':

res = num1 + num2;

System.out.println("Res: " + res);

break;

case '-':

res = num1 - num2;

System.out.println("Res: " + res);

break;

case '\*':

res = num1 \* num2;

System.out.println("Res: " + res);

break;

case '/':

if (num2 == 0) {

System.out.println("Error: Division by zero is not allowed.");

} else {

res = num1 / num2;

System.out.println("Res: " + res);

}

break;

default:

System.out.println("Error: Invalid operator.");

}

scanner.close();

}

}

o/p:

Enter 1st number: 8

Enter an operator (+, -, \*, /): /

Enter 2nd number: 0

ERROR!

Error: Division by zero is not allowed.

----------------------------------------------------------------------------------------------------------------------------------

Question 4: Discount Calculation

Write a program to calculate the discount based on the total purchase amount. Use the following

criteria:

 If the total purchase is greater than or equal to Rs.1000, apply a 20% discount.

 If the total purchase is between Rs.500 and Rs.999, apply a 10% discount.

 If the total purchase is less than Rs.500, apply a 5% discount.

import java.util.Scanner;

public class DiscountCalculator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter total purchase amount: Rs.");

double totalPurchase = scanner.nextDouble();

double discount;

if (totalPurchase >= 1000) {

discount = totalPurchase \* 0.20;

} else if (totalPurchase >= 500) {

discount = totalPurchase \* 0.10;

} else {

discount = totalPurchase \* 0.05;

}

double finalAmount = totalPurchase - discount;

System.out.println("Discount applied: Rs." + discount);

System.out.println("Final amount to be paid: Rs." + finalAmount);

scanner.close();

}

}

o/p :

Enter total purchase amount: Rs.800

Discount applied: Rs.80.0

Final amount to be paid: Rs.720.0

-------------------------------------------------------------------------------------------------------------