CDAC Mumbai
Lab Assignment

Section 1: Error-Driven Learning in Java

Objective: This assignment focuses on understanding and fixing common errors encountered in Java programming. By analyzing and correcting the provided code snippets, you will develop a deeper understanding of Java's syntax, data types, and control structures.

Instructions:

1. **Identify the Errors:** Review each code snippet to identify the errors or issues present.
2. **Explain the Error:** Write a brief explanation of the error and its cause.
3. **Fix the Error:** Modify the code to correct the errors. Ensure that the code compiles and runs as expected.
4. **Submit Your Work:** Provide the corrected code along with explanations for each snippet.

Snippet 1:

```
public class Main {
   public void main(String[] args) {
      System.out.println("Hello, World!");
   }
}
```

- **What error do you get when running this code?**

ANS:

1)Static word is missing .
2) static keyword allows jvm calls it without creating instance.
3)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!");
   }
}
```

- **What happens when you compile and run this code?**

ANS:

1)"Public" Access modifier is missing
2)for jvm entrypoint it needs access modifier.for entry it need public protected
3) 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;
      }
   }
```

- **What error do you encounter? Why is void used in the main method?**

Ans:

1)replace int with void.
2) main method have return type of void. Even we cant use print method in int data type it only returns
3) 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!");
   }
}
```

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

ANS

1)String [] args missing.

2)it is a arrays of strings.this line allows us to pass an number of values of the data type to the The parameter.

3) 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?**

ANS:

1) **No Error in code.**
2) **We can use same name of method .but you need to have different parameter .that is what we called method overloading.**

Snippet 6:

```
public class Main {
   public static void main(String[] args) {
      int x = y + 10;
      System.out.println(x);
   }
}
```

- **What error occurs? Why must variables be declared?**

ANS:

1) "y" variable is not declared.
2) Because of declaration compiler knows what type of data it is.and it allocates the memory
3)Corrected code

```
public class Main {
  public static void main(String[] args)
    {
   int  y = 5;    //Declared
   int x = y + 10;
  System.out.println(x);
    }
  }
```

a

Snippet 7:

```
public class Main {
   public static void main(String[] args) {
      int x = "Hello";
      System.out.println(x);
   }
}
```

- **What compilation error do you see? Why does Java enforce type safety?**

ANS:

1) "Hello" is string ."
2)We defined int data type to string.variable "x" defined as int data type.
3)We should defined variable "x" to String data type.
4)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?**

ANS:

1) **Mising ‘)’ parenthesis and ‘;’ semicolon.**
2) **We cants compile code successfully.becuse of syntax error.cause javac check before converting to binary type.**
3) **Correct code is:**

```
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?**

Ans:

1) **We used keyword as variable.**
2) **If we use keyword as variable it will create confusion for javac for compiling.**

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?**

1) **It will create an error.compilation error**
2) **Cause after create method in main method we have not created instance(object) of methods.**
3) **If we create an instance the method overloading will occur. As the parameters are different.**
4) **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 disp = new Main();
      disp.display();
      disp.display(5);
  }
 }
```

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:

1)first we have initialized arrays as length of 3.

2)we gave the command to access array index at 5. Then compiler gives you error of "arraysindexoutof bound".

3)for fetching we have to to give command that within the the range.

4)corrected code.

```
public class Main {
   public static void main(String[] args)
      { int[] arr = {1, 2, 3};
      System.out.println(arr[2]);
   }
}
```

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:

1) **It will print ”Infinte loop ” at infinite times.**
2) **We have to put base condition in while loop.**
3) **We put break condition after after printing once.**
4) **corrected code**.

```
public class Main {
  public static void main(String[]
    args) { while (true) {
       System.out.println("Infinite  Loop");
       break;
    }
  }
```

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:

1)string is initialized to the null value. That means there is nothing present in str variable.
2)after calling length() function so there is no actual value in it.
3)corrected code.

```
public class Main {
   public static void main(String[] args) {
      String str = “Sanket kavanekar”;
      System.out.println(str.length());   //16
   }
}
```

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:
1)We have assigned String to the double data type.
2)incompatible type error happens.
3)corrected code:

```
public 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);
   }
}
```

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

1)Incompatible error occurs.
2) we can't convert double into int.it is typecasting error.for conversion flow goes like int→double
3)we have to change data type of result into double.
4)corrected code is:

```
public class Main {
   public static void main(String[] args)
      { int num1 = 10;
      double num2 = 5.5;
      Double result = 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);
   }
}
```

- **What is the result of this operation? Is the output what you expected?**

Ans:

1)the result is 2.

2) I expected 2.5 as the data type is double but the division worked as int.

3) if we have to get double type then one of them should be of double type.

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:

1)Compile error will occur.

2)javac does not recognize ** as exponentian operator.

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:

1)output will be 20.

2) in java operator precedence is "%, /,*,+,-''. So multiplication comes first.if we don't have operator precedence then the result may differ.result will be 30.

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:
  1)Arithmetic exception will occur.
  2)when you try to divide some number by zero it is undefined. So in java also javac will show you error.

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:

1)Syntax error will occur.
2) ‘‘;” indicates end of statement.
3) if it is missing then javac is unable to recognize is statement ended or still continues.

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:

1)Compile error will occur.
2)javac needs closing bracket for opening bracket for recognition that bracket should match.

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 method?**

Ans:

1)Compilation error occurs.
2)in Java you cannot add another method into main method but we can add add in class or outside main class.

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:

1) because we have not give break statement inside that.

2) even if we got the case 2 .it will print "value of 2'' And Also default case.so break statement is mandatory in switch statement to break a loops.

3)Corrected code

```
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:

```
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:

1) because we have not give break statement inside that.

2) even if we got the case 2 .it will print "Level 1", "Level 2", "Level 3", And Also default case.so break statement is mandatory in switch statement to break a loop.

3)Corrected code:

```
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:

```
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:

1)incompatible error will occur.

2)java don't support double and float.it supports int byte short string enum for switch values.

3)corrected code:

```
public class Switch {
   public static void main(String[] args) {
      int  score = 85;
      switch(score) {
         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 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:
1)switch case is deppends on each case. It should be unique.
2)it can’t process because javac will get confused which case to implement.
3)for copiling switch statement needs to be unique.

Section 2: Java Programming with Conditional Statements

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"

Ans:
Code:

```
import java.util.*;
class Question1{
    public static void main(String[] args){
    int score;
    Scanner sc=new Scanner(System.in);
    score=sc.nextInt();

    if (score >= 90){
        System.out.println("A");
}
    else if(score>80 ){
        System.out.println("B");
}
    else if(score>70){
        System.out.println("C");
}
    else if(score>60){
        System.out.println("D");
}
    else{
        System.out.println("F");
}
}



}
```

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

Ans:

```
import java.util.Scanner;
public class Question2 {
    public static void main(String[] args) {
        Scanner st = new Scanner(System.in);
System.out.print("enter day : ");
int day =st.nextInt();
switch (day) {
    case 1:
       System.out.println("Monday");
          switch (1) {
               case 1:
            System.out.println(" Weekday.");
              break;
              }
              break;
    case 2:
       System.out.println("Tuesday");
            switch (1) {
              case 1:
                System.out.println(" Weekday.");
              break;
              }
              break;
     case 3:
              System.out.println("Wednesday");
              switch (1) {
                 case 1:
                    System.out.println("It's a Weekday.");
                    break;
              }
              break;
     case 4:
              System.out.println("Thursday");
              switch (1) {
                 case 1:
                    System.out.println("It's a Weekday.");
                    break;
              }
              break;
     case 5:
            System.out.println("Friday");
              switch (1) {
                 case 1:
                    System.out.println("It's a Weekday.");
```

```
                    break;
                }
                break;
    case 6:
            System.out.println("Saturday");
                switch (2) {
                    case 2:
                        System.out.println(" Weekend.");
                        break;
                }
                break;
    case 7:
         System.out.println("Sunday");
              switch (2) {
                case 2:
                        System.out.println("Weekend.");
                        break;
                }
                break;
    default:
        System.out.println("Not valid");
       }

   }
}
```

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 if-else to check if division by zero is attempted and display an error message.
Ans:

```
import java.util.Scanner;
public class Question3 {
   public static void main(String[] args) {
       Scanner st = new Scanner(System.in);
int a=st.nextInt();
char operation=st.next().charAt(0);
int b=st.nextInt();
switch (operation){
   case '+':
       int addition=(a+b);
       System.out.println("(a+b): " + addition);
       break;
   case '-':
       int subtraction=(a-b);
       System.out.println("(a-b): " + subtraction);
       break;
   case '*':
```

```
        int multiplication=(a*b);
        System.out.println(" a*b:" + multiplication);
        break;
    case '/':
        int division=(a/b);
        System.out.println("a/b: " + division);
        break;

    default:
        System.out.println("invalid operator");
}
}
}
```

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.

Ans:

```
import java.util.Scanner;
public class Question4 {
    public static void main(String[] args) {
        Scanner st = new Scanner(System.in);

int Total=st.nextInt();
double discount;
if(Total>=1000){
    System.out.println(("Total purchase amount:"+(Total-(discount=0.20*Total))));
}
else if(Total>=500){
    System.out.println(("Total purchase amount:"+(Total-(discount=0.10*Total))));
}
else {
    System.out.println(("Total purchase amount:"+(Total-(discount=0.05*Total))));
}
}
}
```

Additionally, if the user has a membership card, increase the discount by 5%.

Question 5: Student Pass/Fail Status with Nested Switch

Write a program that determines whether a student passes or fails based on their grades in three subjects. If the student scores more than 40 in all subjects, they pass. If the student fails in one or more subjects, print the number of subjects they failed in.
Ans:

```
import java.util.Scanner;
public class Question5 {
    public static void main(String[] args) {
        Scanner st = new Scanner(System.in);
System.out.print("enter physics marks: ");
int physics=st.nextInt();
System.out.print("enter chem marks: ");
int chem=st.nextInt();
System.out.print("enter maths marks: ");
int maths=st.nextInt();

int failno=0;
if(physics<=40){
    failno++;
}
if(chem<=40){
    failno++;
}
if(maths<=40){
    failno++;
}

switch(failno){

    case 0:
        System.out.println("passed.");
        break;
    default:
        System.out.println("fail");

}
}

}
```

-