Name : Siddhant Kotak

Student ID : 202201410

Course : IT 314 Software Engineering

Professor : Saurabh Tiwari

Semester : Autumn 2024

Lab : Program Inspection, Debugging and Static Analysis

Magic Numbers

Given code :

```
// Program to check if number is Magic number in JAVA
import java.util.*;
public class MagicNumberCheck
{
  public static void main(String args[])
  {
    Scanner ob=new Scanner(System.in);
    System.out.println("Enter the number to be checked.");
    int n=ob.nextInt();
    int sum=0,num=n;
    while(num>9)
    {
      sum=num;int s=0;
      while(sum==0)
      {
        s=s*(sum/10);
        sum=sum%10
      }
      num=s;
    }
    if(num==1)
    {
      System.out.println(n+" is a Magic Number.");
    }
    else
    {
      System.out.println(n+" is not a Magic Number.");
```

```
        }
    }
}
```

Input: Enter the number to be checked 119

Output 119 is a Magic Number.

Input: Enter the number to be checked 199

Output 199 is not a Magic Number.

Program Inspection for Magic Numbers

1. **How many errors are there in the program? Mention the errors you have identified.**

- **Errors Identified: 4**
  - **Logic Error in Summation:**
    - The condition in the inner while loop while(sum==0) is incorrect. It should be while(sum > 0) because you want to extract digits from the number as long as there are digits left.
  - **Incorrect Digit Extraction:**
    - The line s=s*(sum/10); is incorrect. It should be s += sum % 10; to properly accumulate the digits.
  - **Missing Semicolon:**
    - There is a missing semicolon at the end of the line sum=sum%10.
  - **Overall Logic:**
    - The logic used to determine the magic number is flawed. The correct approach involves repeatedly summing the digits until a single-digit number is obtained, and then checking if that number is 1.

2. **Which category of program inspection would you find more effective?**

- **Effective Category:**
  - **Logic Errors**: This category is particularly effective as it focuses on the flow and correctness of the algorithm being implemented, which is crucial for correctly determining whether a number is a magic number.

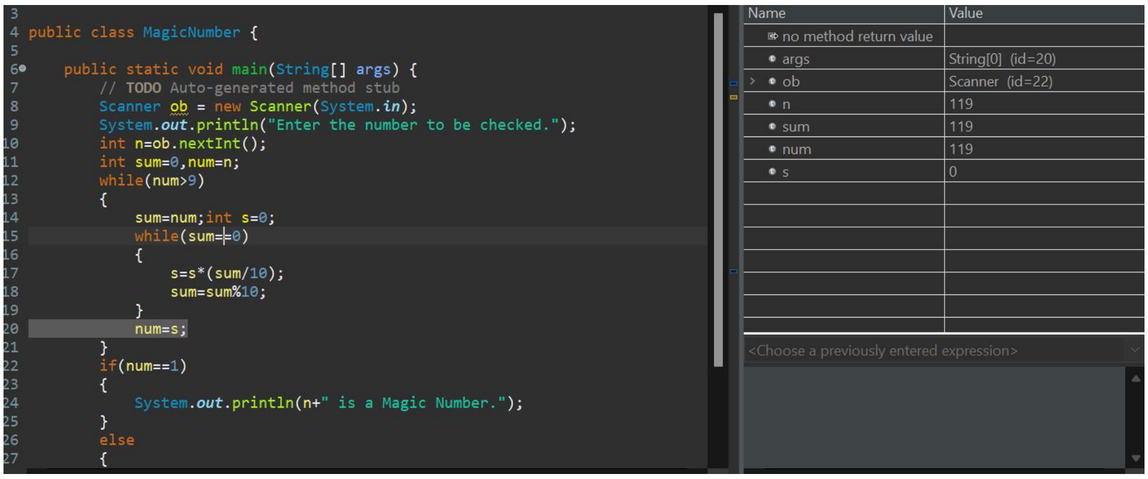
3. **Which type of error you are not able to identify using the program inspection?**

- **Errors Not Identified:**
  - **Input Validation Errors**: Program inspections may miss input validation issues, such as non-integer inputs or negative numbers, which could cause the program to behave unexpectedly.

4. **Is the program inspection technique worth applicable?**

- **Applicability of Program Inspection:**
  - Yes, the program inspection technique is worth applying. It helps identify structural and logical errors, leading to more robust code. However, it should be supplemented with testing for various input cases, including edge cases.

Code Debugging

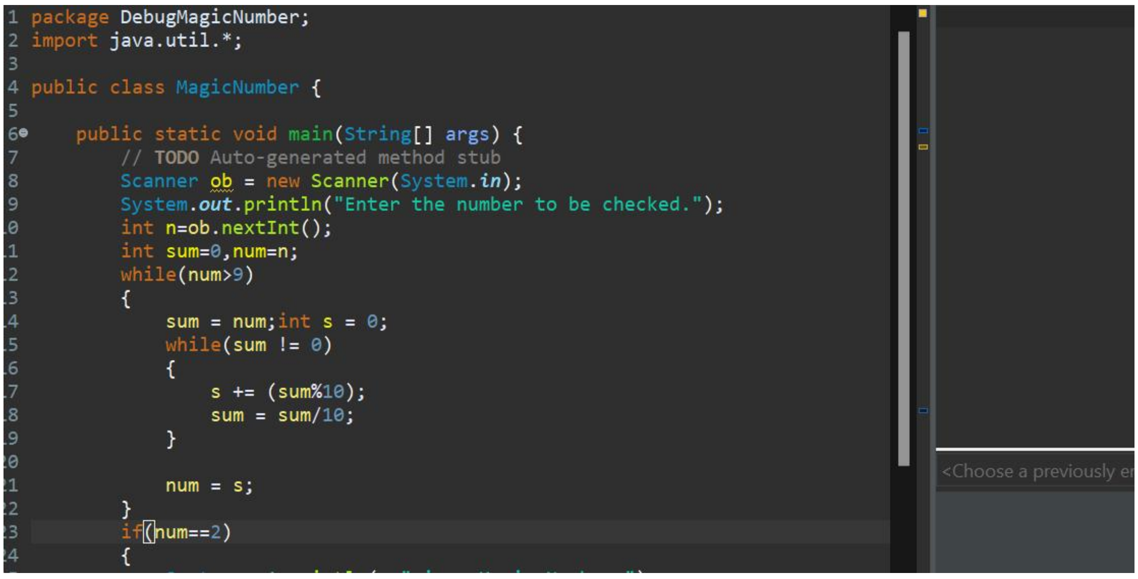


```java
public class MagicNumber {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner ob = new Scanner(System.in);
        System.out.println("Enter the number to be checked.");
        int n=ob.nextInt();
        int sum=0,num=n;
        while(num>9)
        {
            sum=num;int s=0;
            while(sum==0)
            {
                s=s*(sum/10);
                sum=sum%10;
            }
            num=s;
        }
        if(num==1)
        {
            System.out.println(n+" is a Magic Number.");
        }
        else
        {
```

Name	Value
no method return value	
args	String[0] (id=20)
ob	Scanner (id=22)
n	119
sum	119
num	119
s	0

<Choose a previously entered expression>



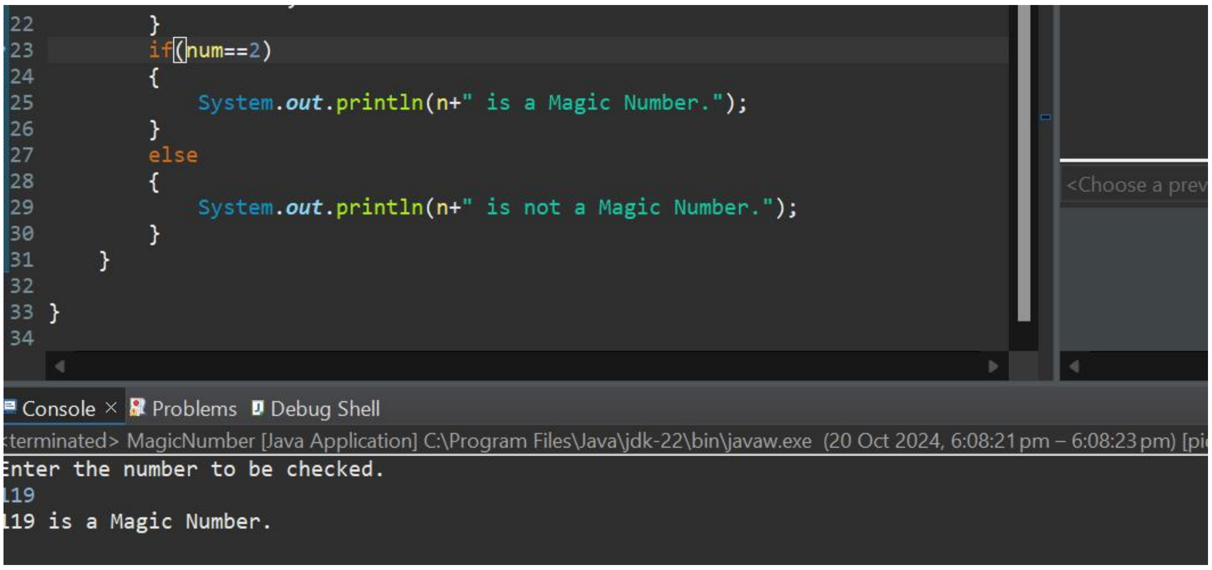
```java
package DebugMagicNumber;
import java.util.*;

public class MagicNumber {

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner ob = new Scanner(System.in);
        System.out.println("Enter the number to be checked.");
        int n=ob.nextInt();
        int sum=0,num=n;
        while(num>9)
        {
            sum = num;int s = 0;
            while(sum != 0)
            {
                s += (sum%10);
                sum = sum/10;
            }

            num = s;
        }
        if(num==2)
        {
```

<Choose a previously e



```
        }
        if(num==2)
        {
            System.out.println(n+" is a Magic Number.");
        }
        else
        {
            System.out.println(n+" is not a Magic Number.");
        }
    }

}

```

```
terminated> MagicNumber [Java Application] C:\Program Files\Java\jdk-22\bin\javaw.exe (20 Oct 2024, 6:08:21 pm – 6:08:23 pm) [pi
Enter the number to be checked.
119
119 is a Magic Number.
```

Errors Identified

1. **Incorrect Loop Condition for Summing Digits:**
   - **Original Line:** while(sum == 0)
   - **Correction:** Change to while (sum > 0) (This ensures the loop processes all digits as long as sum is greater than 0.)
2. **Incorrect Digit Sum Calculation:**
   - **Original Line:** s = s * (sum / 10)
   - **Correction:** Change to s = s + (sum % 10) (This correctly accumulates the sum of the digits.)
3. **Incorrect Update of sum to Remove Last Digit:**
   - **Original Line:** sum = sum % 10
   - **Correction:** Change to sum = sum / 10 (This correctly removes the last digit from sum.)

Breakpoints Needed

You can set breakpoints at the following locations for effective debugging:

- **Line 12:** To check if the loop that processes digits works correctly.
- **Line 14:** To verify how s is updated with the sum of digits.
- **Line 19:** To check if the final number is correctly identified as a magic number.

Steps to Fix the Errors

1. **Change the condition** in the loop to while (sum > 0).
2. **Update the logic** to accumulate the sum of digits by changing it to s = s + (sum % 10).
3. **Modify the update logic** for sum to use integer division: sum = sum / 10.

Fixed Code

```
// Program to check if a number is a Magic number in JAVA
import java.util.Scanner;

public class MagicNumberCheck {
  public static void main(String args[]) {
    Scanner ob = new Scanner(System.in);
    System.out.println("Enter the number to be checked.");
    int n = ob.nextInt();
    int num = n; // Copy the number
    int sum = 0;

    // Keep reducing the number until it's a single digit
    while (num > 9) {
      sum = num;
      int s = 0;

      // Sum the digits of the current number
      while (sum > 0) { // Fixed condition
        s = s + (sum % 10); // Corrected to accumulate digit sum
        sum = sum / 10; // Corrected to remove the last digit
      }

      // Assign sum of digits back to num for the next iteration
      num = s;
    }
```

```
        // Check if the resulting number is 1 (Magic Number)
        if (num == 1) {
            System.out.println(n + " is a Magic Number.");
        } else {
            System.out.println(n + " is not a Magic Number.");
        }

        ob.close();
    }
}
```

Input and Output

- **Input:**
    - Enter the number to be checked: 119
- **Output:**
    - 119 is a Magic Number.
- **Input:**
    - Enter the number to be checked: 199
- **Output:**
    - 199 is not a Magic Number.