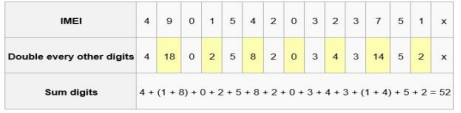
**NUMBER PROGRAMS**

PROGRAM 1

Write a program in Java to input the first 14 digits of an IMEI number and find the check (last) digit of it. The IMEI (15 decimal digits: 14 digits plus a check digit) includes information on the origin, model, and serial number of the device. The check digit (x) is obtained by computing the sum of digits then computing 9 times that value modulo 10.



In algorithm form:

1.Compute the sum of the digits (52 in this case).

2.Multiply the sum by 9 (9\*52 = 468).

3.Divide the result by 10 and note down the remainder (468 % 10)

4.The last digit, 8, is the check digit.

1

Algorithm

Step 1: Begin.

Step 2: Scanner object declared.

Step 3: First 14 digits of IMEI accepted from the user as n.

Step 4: The accepted values are converted from long to string(s).

Step 5: Length (1) of the string is calculated using length ().

Step 6: Initialized sum and d with 0.

Step 7: The following steps are executed if length is 14.

Step 7.1: A for loop is executed till value of i reaches 1.

Step 7.1.1: Digits (d) are extracted.

Step 7.1.2: If the digit is in even position, then it is multiplied by 2.

Step 7.1.3: The value of d is then passed to sumDig function for calculating sum.

Step 7.1.4: n value is divided by 10.

Step 7.1.5: Loop block is closed.

Step 8: Sum value is multiplied by 9 and then its remainder is obtained.

Step 9: The check digit of the IMEI is found and is displayed.

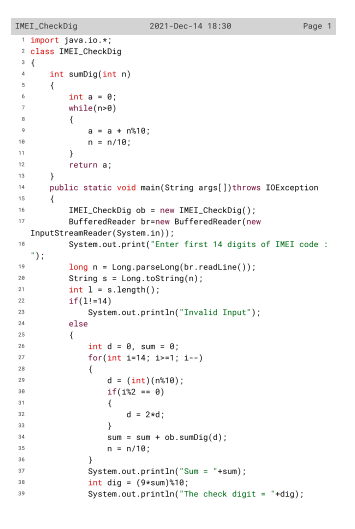
Step 10: If length of the string is not 14, then "Invalid input" is displayed.

Step 11: Main () block is closed.

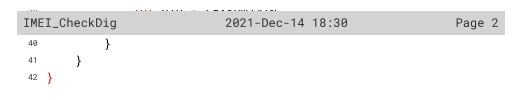
Step 12: Stop.

2

Source Code



3



4

Variable Description

|  |  |  |
| --- | --- | --- |
| VARIABLE | DATA TYPE | DESCRIPTION |
| n | long | To store the IMEI number |
| s | String | To Store String value of n |
| l | int | To store length of the string |
| d | int | To store remainder value |
| sum | int | To store the sum value |
| i | int | To act as control variable |
| dig | int | To store the check digit value |

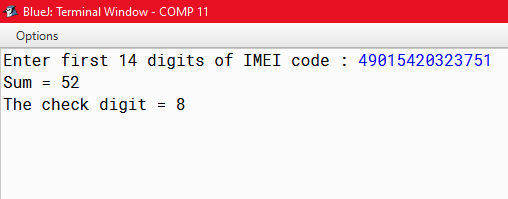
5

Function description

|  |  |  |
| --- | --- | --- |
| Function | Data type | Description |
| sumDig() | int | To calculate sum of the digit extracted |
| main() | void | To accept IMEI number and display the  check digit |

6

OUTPUT



7

PROGRAM 2

Write a program in java to accept a number and check whether it is a valid IMEI number or not. The IMEI is validated in three steps: 1. Starting from the right, double every other digit (e.g., 7 becomes 14). 2. Sum the digits (e.g., 14 → 1 + 4). 3. Check if the sum is divisible by 10. For Example: If input is IMEI = 490154203237518



Since, 60 is divisible by 10, hence the given IMEI number is Valid.

8

Algorithm

Step 1: Begin.

Step 2: Scanner object declared.

Step 3: First 14 digits of IMEI accepted from the user as ‘no’.

Step 4: A while loop is initiated with the condition that ‘no’ must be greater than 0.

Step 4.1: a variable c and r is taken such that if c is an even number r will be multiplied by 2.

Step 4.2: 2nd while loop is initiated with the condition that r must be greater than 9.

Step 4.2.1: A variable x is declared and initialized with r value.

Step 4.3: 3rd while loop is initiated with condition x>0.

Step 4.3.1: sum is calculated using sumofdigit() and stored in s

Step 4.2.2: r is re-initialized with s

Step 4.1.1: a variable ‘sum’ is used to store sum value with r

Step 4.1.2: If c was not an even then sum variable is used to store sum value with r

Step 4.1.3: 1st while loop block is closed with no being divided by 10 as final statement.

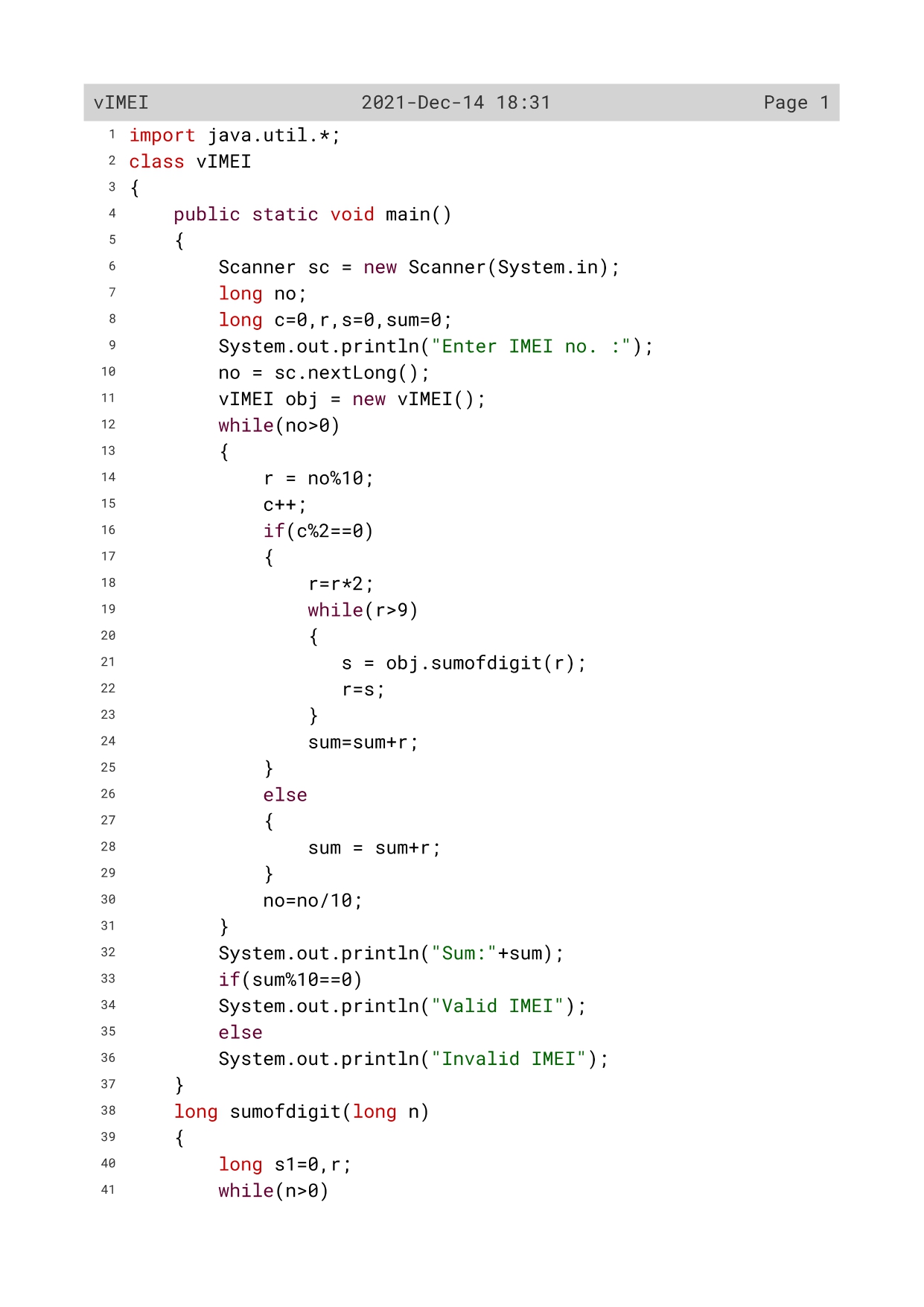
Step 5: If the value of remainder of sum equals to 0 then system prints ‘Valid IMEI’ .

Step 6: Else the system displays ’Invalid IMEI’.

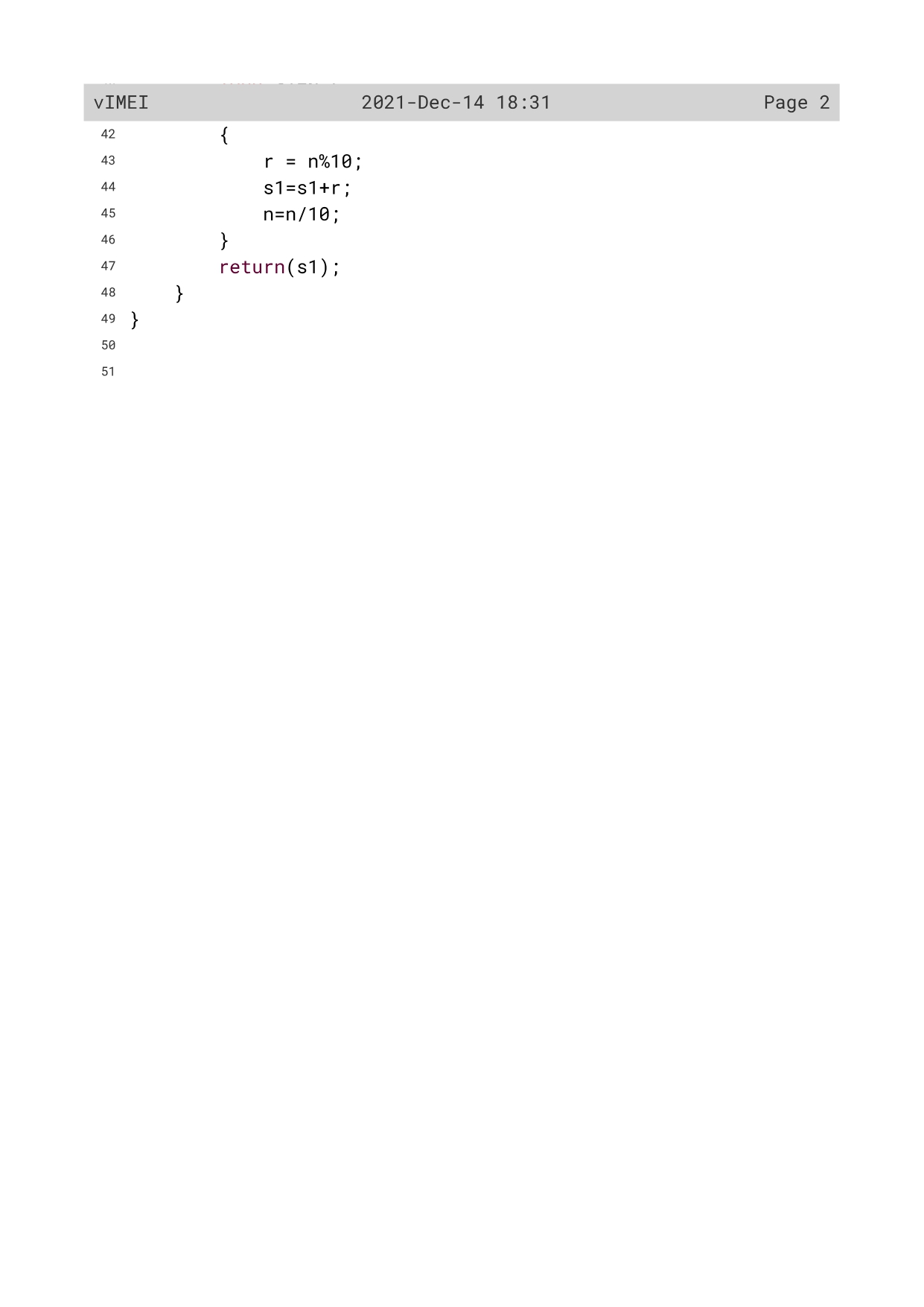
Step 7: Main () block is closed

Step 8: Stop.

9

Source Code 

10



11

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable | Data type | Description |
| no | long | To accept and store IMEI number |
| c | long | To act as counter variable |
| r | long | To store remainder value |
| s | long | To store 1st sum value |
| sum | long | To calculate and store the main sum value |
| s1 | long | To calculate sum value inside sumofdigit() |
| n | long | To act as parameters for sumofdigit() |

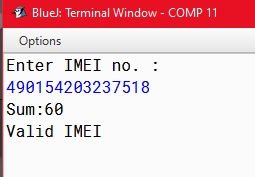
12

Function Description

|  |  |  |
| --- | --- | --- |
| Function | Data type | Description |
| sumofdigit() | long | To calculate sum of the digits |
| main() | void | To check whether given IMEI number is valid or not |

13

OUTPUT



14

PROGRAM 3

A Circular Prime is a prime number that remains prime under cyclic shifts of its digits. When the leftmost digit is removed and replaced at the end of the remaining string of digits, the generated number is still prime. The process is repeated until the original number is reached again. A number is said to be prime if it has only two factors I and itself. Example:

131

311

113

Hence, 131 is a circular prime.

15

Algorithm

Step 1: Start.

Step 2: An object ob of class CircularPrime is defined.

Step 3: Scanner class variable is initialized.

Step 4: A number is taken from the user.

Step 5: The loop control variable of the do while loop is initialized.

Step 6: The flag variable is initialized with 0.

Step 7: The do while loop begins where the following tasks are performed.

Step 7.1: It prints the value of loop control variable.

Step 7.2: If the loop control variable is prime then the flag variable turns 1 and the control comes out of the loop body.

Step 7.3: Then the number is sent to rotate function to rotate itself.

Step 7.4: If the rotated number is same as the user input then the loop terminates.

Step 8: If flag is 1 the number is not circular prime.

Step 9: If the flag value is 0 the number is circular prime.

Step 10: In the isPrime function the following task are performed:

Step 10.1: The counter variable is initialized to 0.

Step 10.2: The for loop starts where the counter variable increases if the number variable is divisible by any number.

Step 10.3: If the counter variable is more than 2 the function returns false else it returns true.

Step 11: In the rotate function the following task are performed.

Step 11.1: The number is first converted to string.

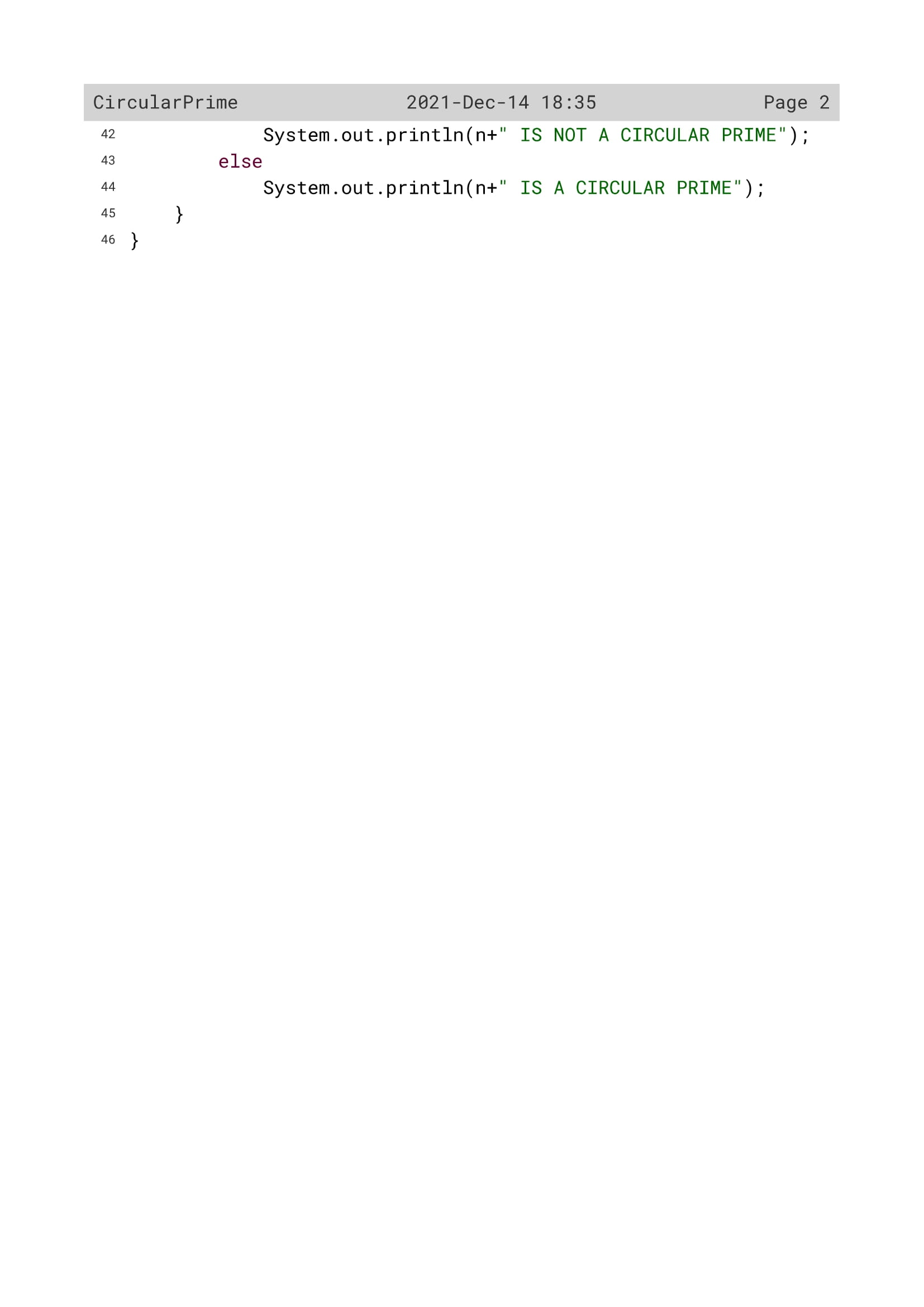
Step 11.2: After rotating the number.i.e.bringing the last digit at front the string is again coverted into number and returned to the calling module.

Step 12: Stop.

16

Source Code

17



18

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Description |
| c | int | Flag variable |
| i | int | Loop variable |
| n | int | To enter a number by the user |
| s | String | To convert the number from int to string data type |
| flag | int | Flag variable |
| p | String |  |
| a | Int | To convert the number from string to int data type |

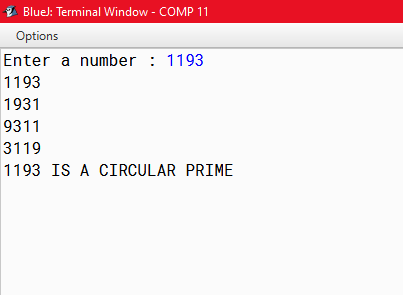
19

Function Description

|  |  |  |
| --- | --- | --- |
| Function name | Data type | description |
| isPrime() | Boolean | To check whether the given number is prime or not |
| rotate() | Int | Rotating the number to check whether all combinations are prime or not |
| main() | void | To check whether the given number is a circular prime or not |

20

OUTPUT



21

PROGRAM 4

Write a program to generate first n Catalan numbers using formula: (2𝑛)! /(𝑛!)^ 2 (𝑛 + 1) Catalan numbers are the numbers having the form of the above given formula where (n > 0). First few such numbers are 1, 2, 5, 14, 42,....

22

Algorithm

Step 1:Start.

Step 2:Scanner class object is defined.

Step 3:A number is taken from the user.

Step 4: The number is send to the function catalan.

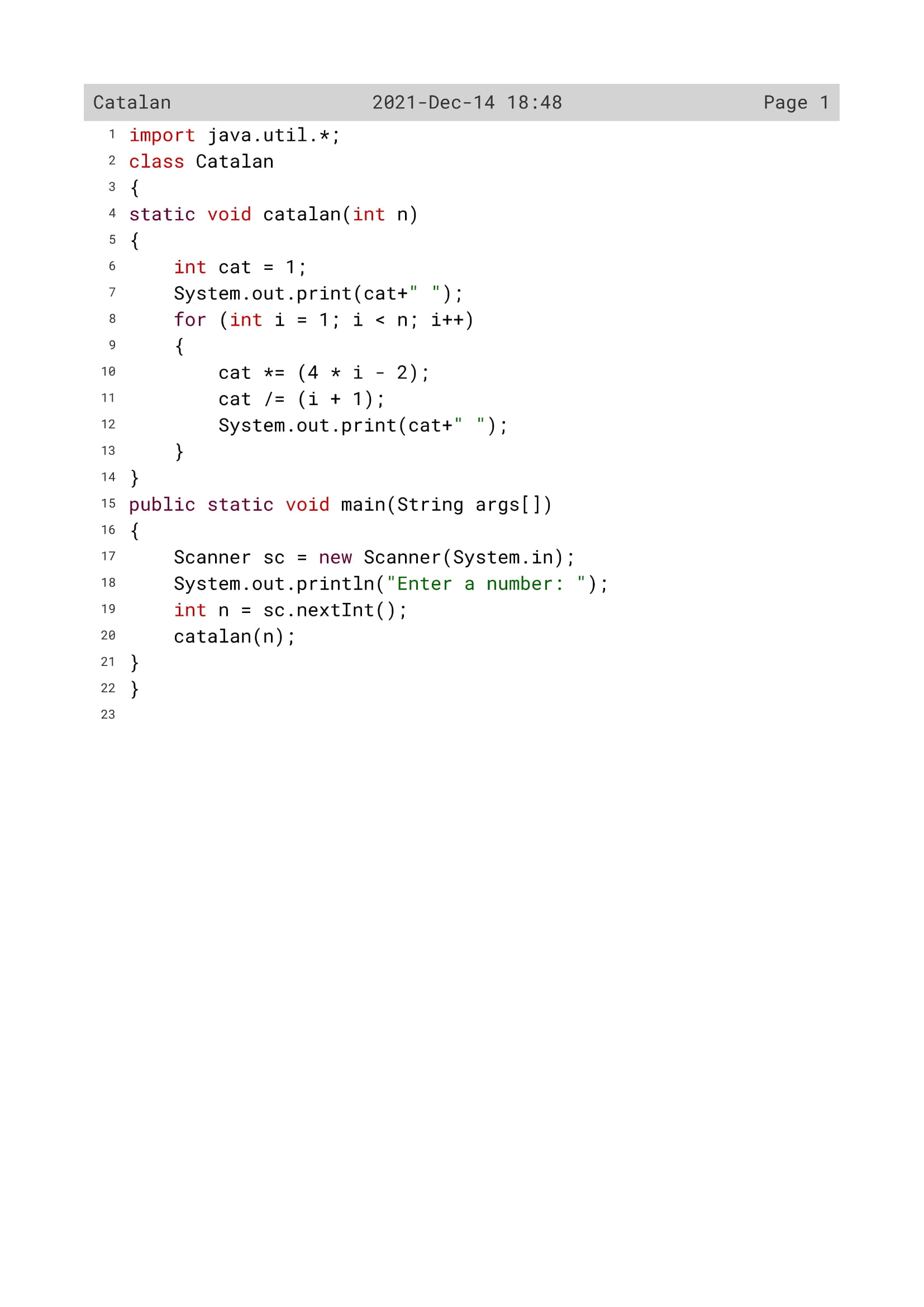
Step 5:In the function catalan the following steps are performed.

Step 5.1:The cat variable is initialized to 1.

Step 5.2:A for loop starts where some operations are performed on the variable cat and the value of it is printed.

Step 6:Stop.

23

Source Code

24

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Description |
| n | int | To enter the number by the user |
| i | int | Loop variable |
| cat | int | To find the catalan number |

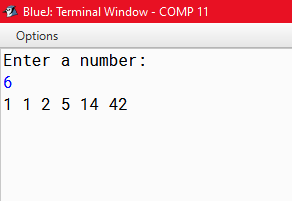
25

Function Description

|  |  |  |
| --- | --- | --- |
| Function name | Data type | description |
| catalan() | void | To calculate the catalan number |
| main() | void | To generate first n catalan numbers |

26

OUTPUT



27

PROGRAM 5

Write a Program in Java to input a number and check whether it is a Fascinating Number or not.. Fascinating Numbers : Some numbers of 3 digits or more exhibit a very interesting property. The property is such that, when the number is multiplied by 2 and 3, and both these products are concatenated with the original number, all digits from 1 to 9 are present exactly once, regardless of the number of zeroes.

Let’s understand the concept of Fascinating Number through the following example:

Consider the number 192,

192 x 1 = 192

192 x 2 = 384

192 x 3 = 576

Concatenating the results : 192384576

It could be observed that ‘192384576’ consists of all digits from 1 to 9 exactly once. Hence, it could be concluded that 192 is a Fascinating Number. Some examples of fascinating Numbers are : 192, 219, 273, 327, 1902, 1920, 2019 etc.

28

Algorithm

Step 1: Start.

Step 2:Scanner class variable is declared.

Step 3:A number is taken from the user.

Step 4:The number is multiplied with 1 2 3 and stored in separate variables n1 n2 n3 respectively.

Step 5:n1 n2 n3 are converted into string separately and stored in s1 s2 and s3.

Step 6:s1 s2 s3 are then concatenated in a single string s.

Step 7:The string s is converted into number and stored in variable num.

Step 8:The num variable is printed.

Step 9:The flag variable and counter variable is declared.

Step 10:A for loop begins and the following tasks are performed.

Step 10.1:A variable is initialized with num,and the flag and counter variable are initialized with 0.

Step 10.2:An inner while loop starts which counts the frequency of the digits.

Step 10.3:If the frequency of any digit is more than 1 the flag variable turns 0.

Step 11:If flag is 0 it is a fascinating number else its not a fascinating number.

Step 12:Stop.

29

Source Code

30



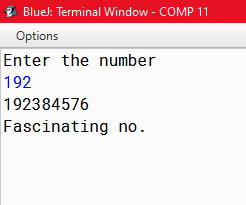
31

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Description |
| no | Int | To enter the number |
| n1 | Int | To multiply the number by 1 |
| n2 | Int | To multiply the number by 2 |
| n3 | int | To multiply the number by 3 |
| s1 | String | To convert variable n1 from int to string data type |
| s2 | String | To convert variable n1 from int to string data type |
| s3 | String | To convert variable n1 from int to string data type |
| s | String | To contatenate variable s1,s2 and s3 |
| num | Int | To convert variable s from string to int data type |
| flag | Int | Loop variable |
| r | Int | To find the remainder |
| c | Int | Flag variable |
| i | Int | Loop variable |

32

OUTPUT



33

PROGRAM 6

Write a Program in Java to input a number and check whether it is a Bouncy Number or not.

Increasing Number : Working from left-to-right if no digit is exceeded by the digit to its left it is called an increasing number; for example, 22344.

Decreasing Number : Similarly if no digit is exceeded by the digit to its right it is called a decreasing number; for example, 774410.

Bouncy Number : We shall call a positive integer that is neither increasing nor decreasing a “bouncy” number; for example, 155349.

Clearly there cannot be any bouncy numbers below 100.

Sample input: Enter a number: 324165

Sample output: 324165 is a Bouncy number

34

Algorithm

Step 1: Start.

Step 2: Scanner class object is declared.

Step 3: An object ob of the class is defined.

Step 4: A number is taken from the user.

Step 5: The number is passed to the function.

Step 6: In the function isBouncy the number is send to two other functions to check whether it is increasing decreasing or bouncy and the desired result is printed out.

Step 7: In the isIncreasing function the following steps are performed.

Step 7.1: The number is converted to string.

Step 7.2: A flag variable is initialized to 0.

Step 7.3: A for loop starts where the digits of the number are extracted.It is checked whether each digit is greater than its succeeding digit.If it is greater then the flag value turns 1.

Step 7.4: If the flag value is 1 the function returns false to the calling module else it returns true.

Step 8: In the isDecreasing function the following steps are performed.

Step 8.1: The number is converted to string.

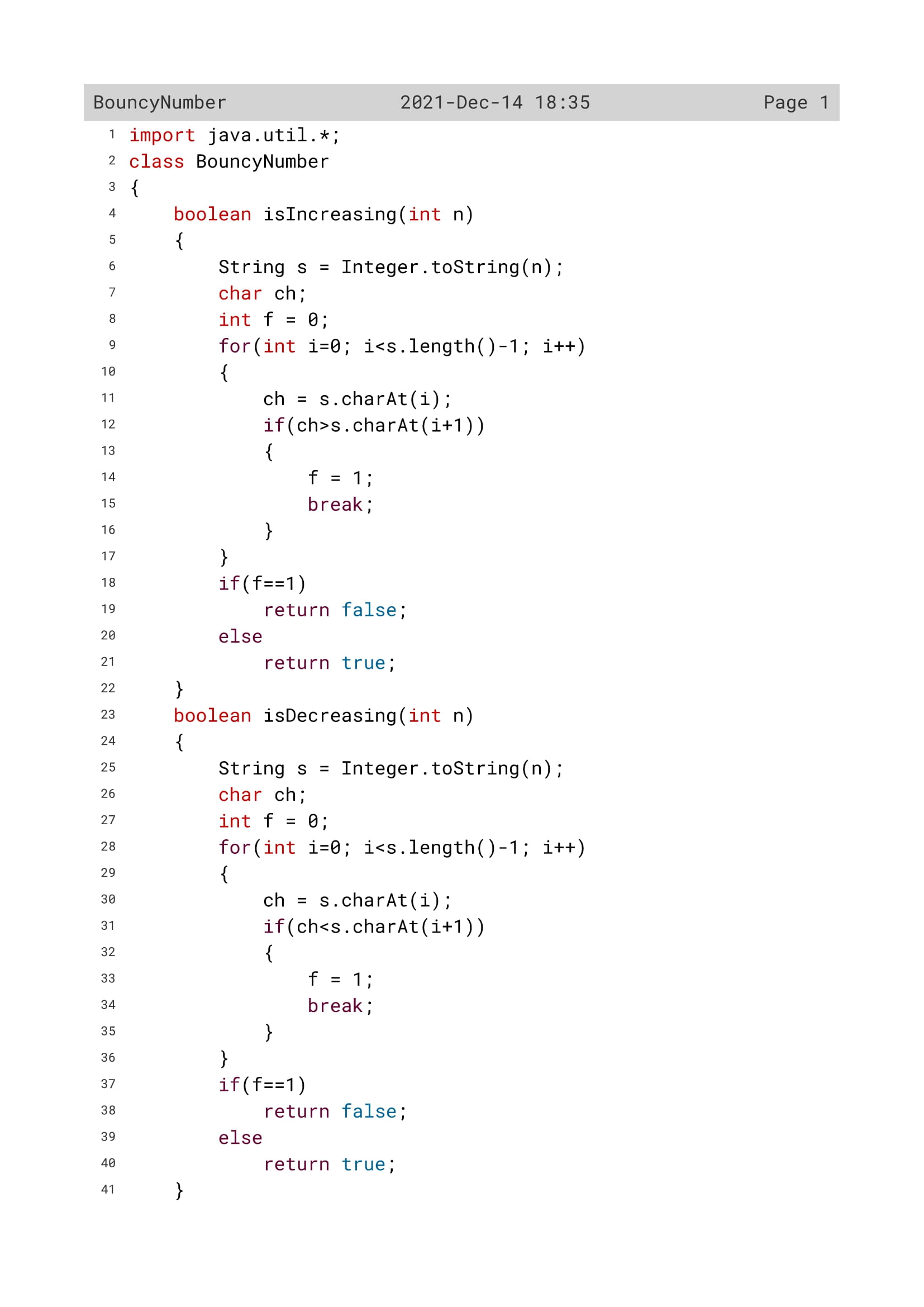
Step 8.2: A flag variable is initialized to 0.

Step 8.3: A for loop starts where the digits of the number are extracted.It is checked whether each digit is smaller than its succeeding digit.If it is smaller then the flag value turns 1.

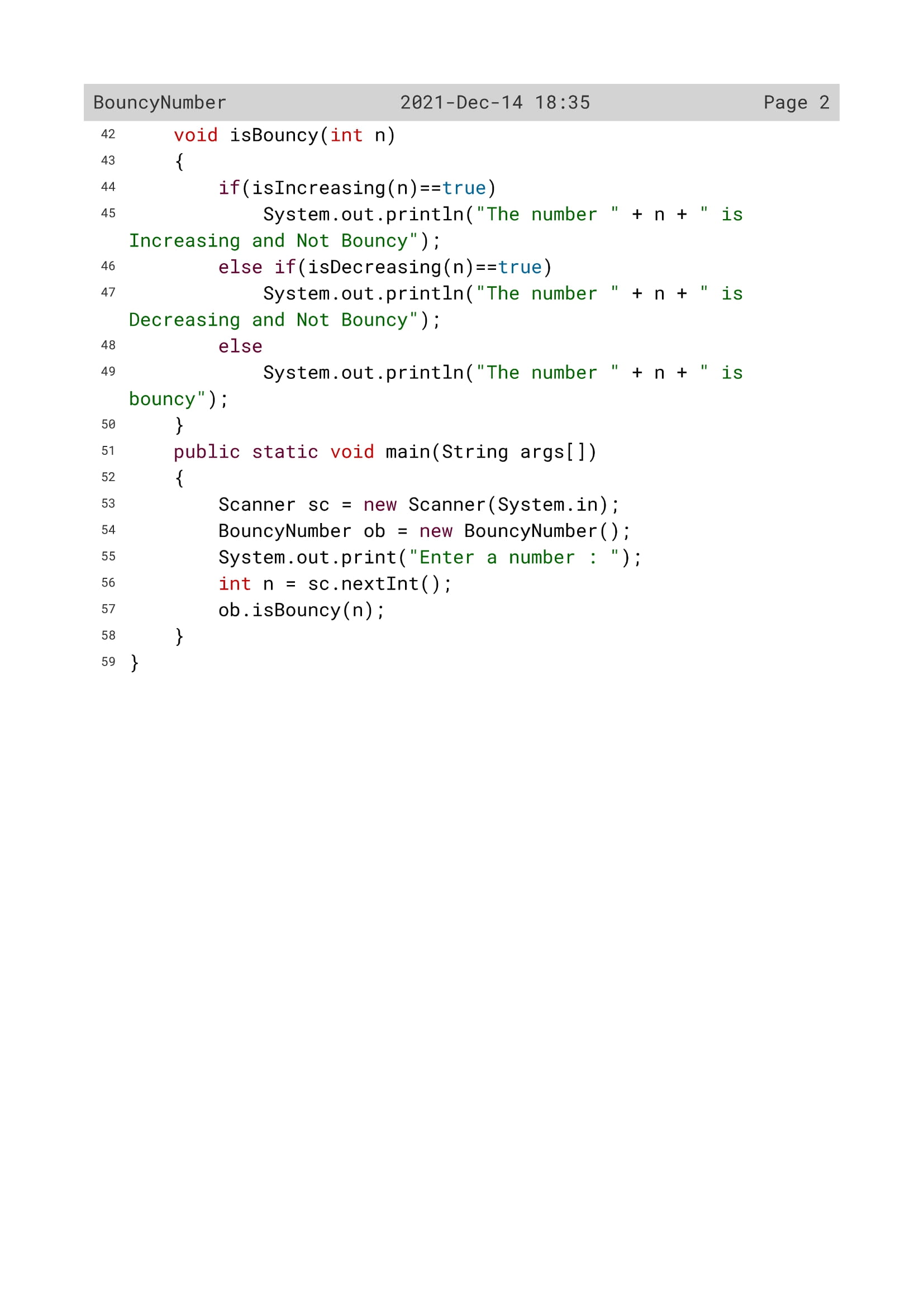
Step 8.4: If the flag value is 1 the function returns false to the calling module else it returns true.

Step 9:Stop.

35

Source code

36



37

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Description |
| s | String | To convert the number from int to string data type |
| ch | char | To extract the numbers |
| f | Int | Flag variable |
| i | Int | Loop variable |
| n | Int | To find the number |

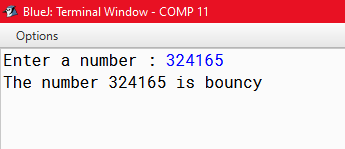
38

Function Description

|  |  |  |
| --- | --- | --- |
| Function name | Data type | description |
| isIncreasing() | boolean | To check whether the number is increasing or not |
| isDecreasing() | boolean | To check whether the number is decreasing or not |

39

OUTPUT



40

PROGRAM 7

Write a Program in Java to input a number and check whether it is an Evil Number or not.

Evil Number : An Evil number is a positive whole number which has even number of 1’s in its binary equivalent. Example: Binary equivalent of 9 is 1001, which contains even number of 1’s.

A few evil numbers are 3, 5, 6, 9….

Sample input:

Enter a number: 12

Sample output:

Binary equivalent of 12 is 1100

12 is an evil number

41

Algorithm

Step 1: Start.

Step 2: Scanner class variable is declared.

Step 3: A number is taken from the user.

Step 4: The counter variable is initialized to 0;

Step 5: A string variable is initialized with null.

Step 6: A while loop starts and the following functions are performed.

Step 6.1: The number is divided by 2 and the remainder is stored in a variable.

Step 6.2: In a string variable the remainders are concatenated.

Step 6.3: If the remainder is 1 the counter variable increases by 1.

Step 6.4: The number is divided by 2 and the loop continues till the number becomes 0.

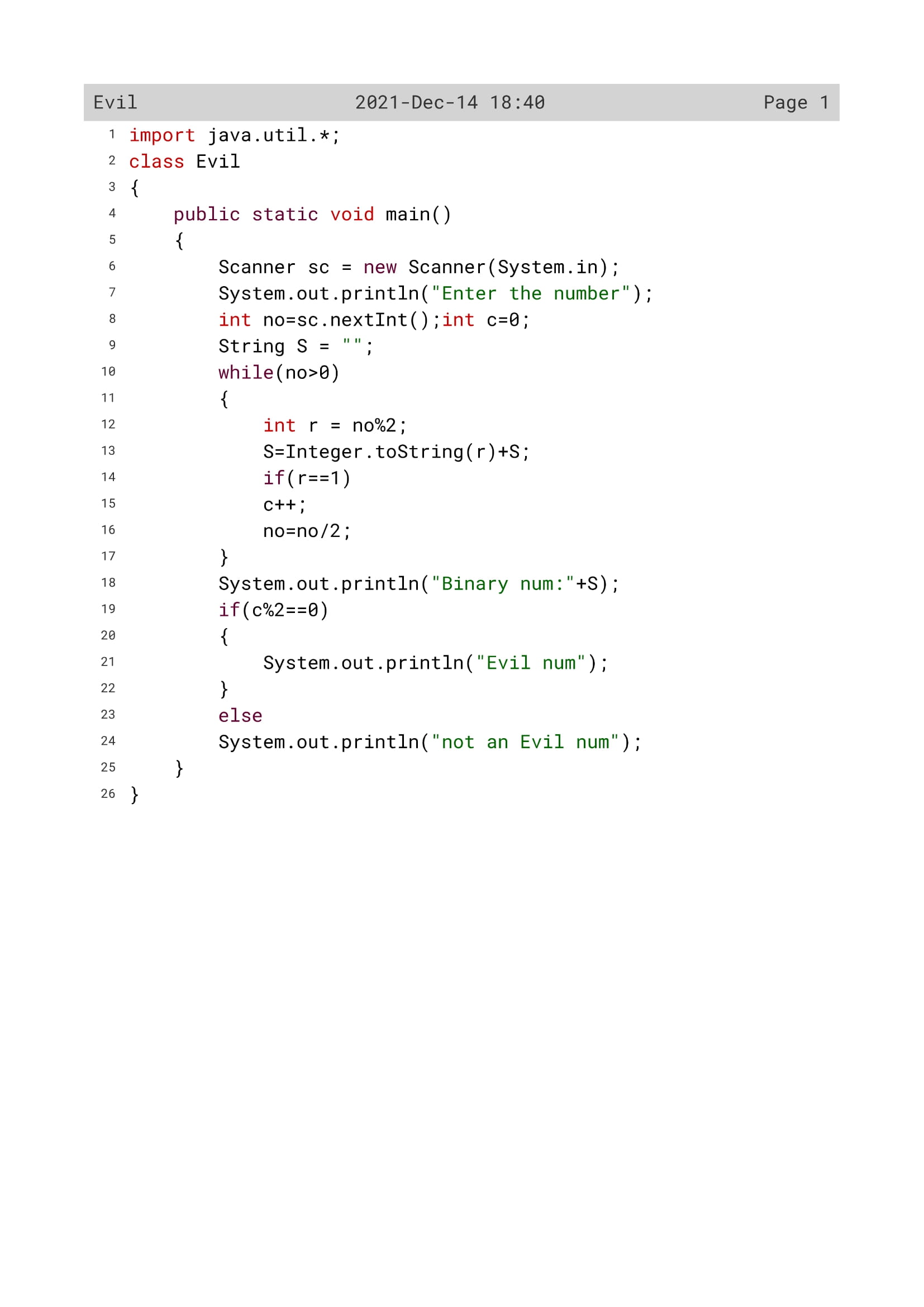
Step 7: The binary sum is printed.

Step 8: If the counter is even the number is evil else the number is not evil.

Step 9: Stop

42

Source Code



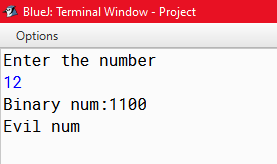
43

Variable Description

|  |  |  |
| --- | --- | --- |
| Variable name | Data type | Description |
| no | Int | To enter the number |
| s | String | To convert the number from int to string data type |
| r | int | To find the remainder |
| c | int | Flag variable |

44

OUTPUT



45