《Java》课程实验报告

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 专业名称 | 计算机科学与技术 | 年级 | 2023 | 班级 | 01 |
| 学生姓名 | 颜丙超 | 学号 | 202311000415 | 指导教师 | 梁成 |
| 实验题目 | 斐波那契数列和回文数 | | | 提交时间 | 10.17 |

1. 实验目的和要求

实验一：深入理解斐波那契（Fibonacci）数列的定义及其计算方法。

掌握递归和非递归两种编程方法在实现斐波那契数列计算中的应用。

通过编写程序，提升编程实践和问题解决能力。

程序应能够接收用户输入的一个大于等于3的整数N，并返回其对应的斐波那契数值。

程序的输出格式需严格按照给定的提示进行，包括输入提示和结果输出。

程序应包含两个样例运行的输出，且输出格式需与给定的样例完全一致，以确保程序能够正确通过平台测试。

实验二：

**理解回文数的概念：通过本次实验，深入理解回文数的定义，即一个数字或字符串从前往后读和从后往前读是相同的。**

**掌握回文数的判断方法：学习并实践如何编写程序来判断一个数字是否为回文数。**

**实现特定条件下的回文数求和：编写程序，能够找出所有满足特定条件的5位和6位回文数，并计算它们的和，以验证这些数字之和是否等于给定的值。**

**提升编程技能：通过本次实验，锻炼编程思维，提高编程实践能力，学会如何运用所学知识解决实际问题。**

1. 实验环境

硬件设施：计算机一台 软件设施：eclpise

三、实验内容及实施

**实验1**

根据斐波那契数列的定义

编写递归函数。在函数内部，添加适当的条件判断来处理边界情况（如N=3时直接返回2）。测试递归函数，确保它能够正确计算斐波那契数。

非递归函数实现：编写非递归函数，使用循环结构来计算斐波那契数。初始化两个变量来存储前两项的值，并通过循环迭代来计算第N项。测试非递归函数，确保它的计算结果与递归函数一致。用户交互程序实现.编写主程序，使用输入函数提示用户输入一个整数N。对用户输入进行验证，确保N大于等于3。调用递归函数和非递归函数，分别计算斐波那契数。按照给定的输出格式打印计算结果。

**实验二**

回文数判断：编写一个函数或方法，用于判断一个给定的数字是否为回文数。回文数是指从左到右和从右到左读起来都一样的数。

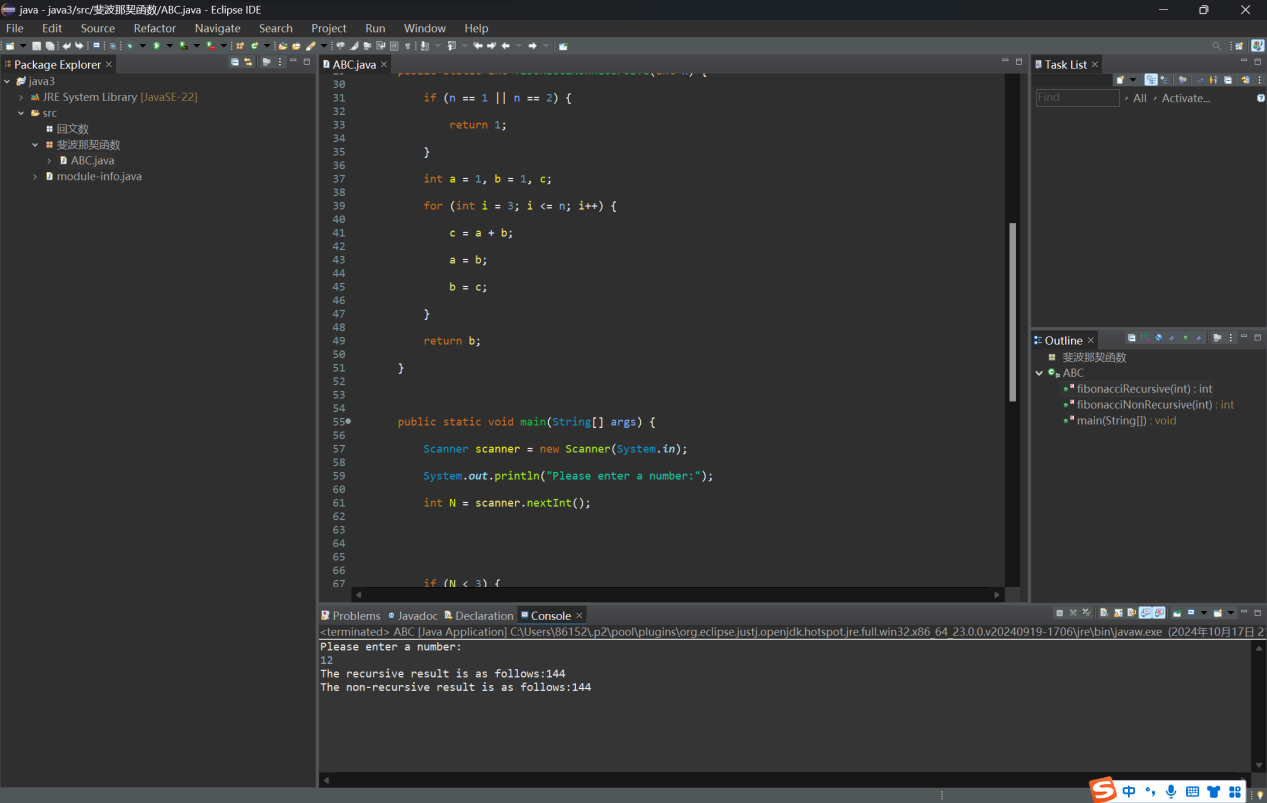
数字筛选：编写逻辑来遍历所有可能的五位和六位十进制数，并使用上述的回文数判断函数来筛选出回文数。

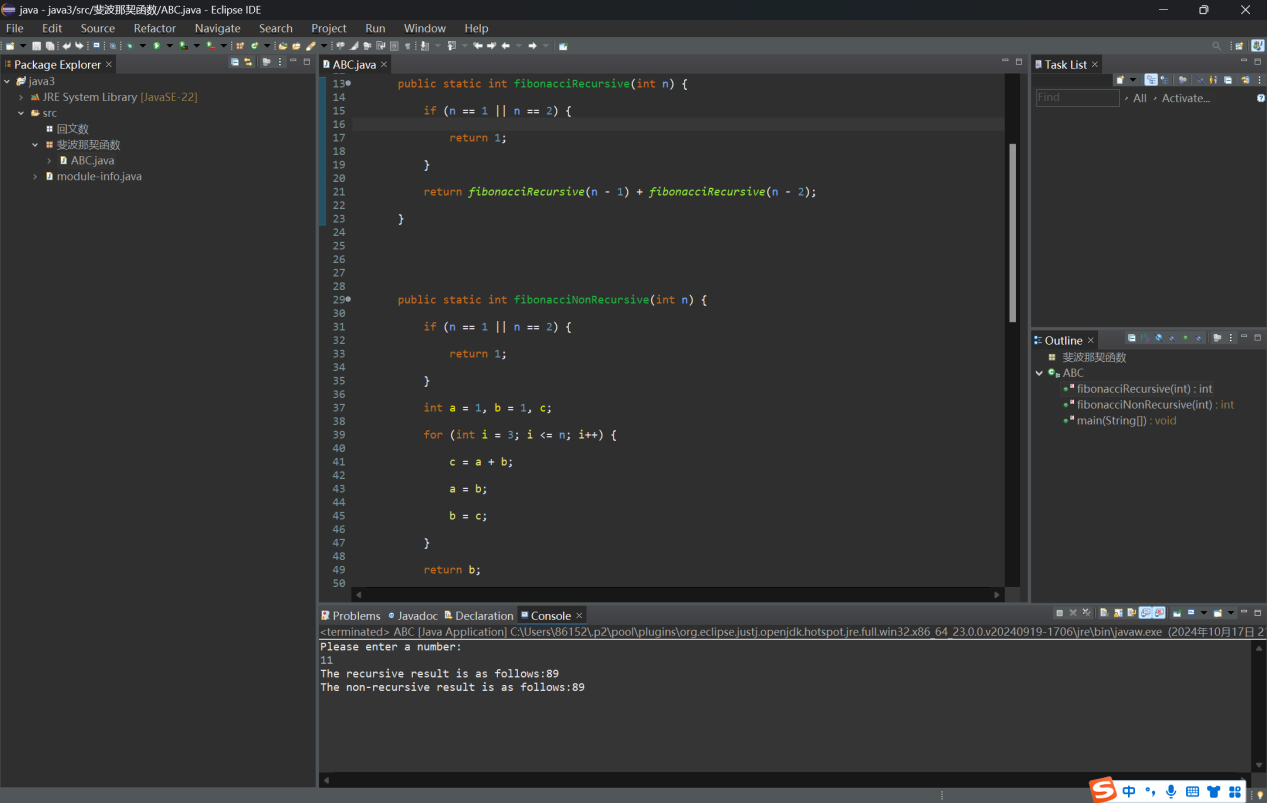
数字之和计算：对于筛选出的每一个回文数，计算其各位数字之和，并检查是否等于用户输入的正整数n。

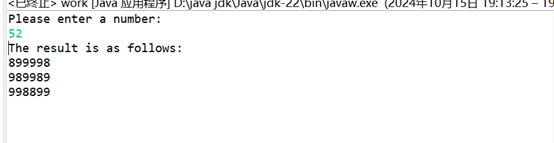
结果输出：按照指定的输出格式，打印出所有满足条件的回文数。

1. **实验结果 (程序的执行结果)**

**实验一**









01 **import** java.util.Scanner;

02

03

04

05 public class FibonacciCalculator

06 {

07

08

09

10

11

12 public **static int** fibonacciRecursive(**int** n)

13 {

14

15 **if** (n == 1 || n == 2)

16 {

17

18 **return** 1;

19

20 }

21

22 **return** fibonacciRecursive(n - 1) + fibonacciRecursive(n - 2);

23

24 }

25

26

27

28

29

30 public **static int** fibonacciNonRecursive(**int** n)

31 {

32

33 **if** (n == 1 || n == 2)

34 {

35

36 **return** 1;

37

38 }

39

40 **int** a = 1, b = 1, c;

41

42 **for** (**int** i = 3; i <= n; i++)

43 {

44

45 c = a + b;

46

47 a = b;

48

49 b = c;

50

51 }

52

53 **return** b;

54

55 }

56

57

58

59 public **static void** main(String[] args)

60 {

61

62 Scanner scanner = **new** Scanner(System.in);

63

64 System.out.println("Please enter a number:");

65

66 **int** N = scanner.nextInt();

67

68

69

70

71

72 **if** (N < 3)

73 {

74

75 System.out.println("Input number must be greater than or equal to 3.");

76

77 scanner.close();

78

79 **return**;

80

81 }

82

83

84

85

86

87 **int** recursiveResult = fibonacciRecursive(N);

88

89 System.out.println("The recursive result is as follows:" + recursiveResult);

90

91

92

93

94

95 **int** nonRecursiveResult = fibonacciNonRecursive(N);

96

97 System.out.println("The non-recursive result is as follows:" + nonRecursiveResult);

98

99

100

101 scanner.close();

102

103 }

104

105 }

106

107

**实验二**

import java.util.Scanner;

public class PalindromeAndSum {

   public static boolean isPalindrome(int number) {

       String str = Integer.*toString*(number);

       int len = str.length();

       for (int i = 0; i < len / 2; i++) {

           if (str.charAt(i) != str.charAt(len - i - 1)) {

               return false;

           }

       }

       return true;

   }

   public static int sumOfDigits(int number) {

       int sum = 0;

       while (number > 0) {

           sum += number % 10;

           number /= 10;

       }

       return sum;

   }

   public static void main(String[] args) {

       Scanner scanner = new Scanner(System.***in***);

       System.***out***.println("Please enter a number:");

       int n = scanner.nextInt();

       System.***out***.println("The result is as follows:");

       for (int i = 10000; i <= 999999; i++) {

           if (*sumOfDigits*(i) == n) {

               if (*isPalindrome*(i)) {

                   System.***out***.println(i);

               }

           }

       }

       scanner.close();

   }

}

​