МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Институт информационных технологий (ИТ) Кафедра инструментального и прикладного программного обеспечения (ИиППО)

Дисциплина «Программирование на языке Джава»

ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАНЯТИЮ №5

Выполнил студент группы ИНБО-02-20 Деревянкин Н.А.

Принял Степанов П.В.

Практическая работа выполнена «___»_____2021 г.

«_____» «___»_____2021 г.

Отметка о выполнении

Москва – 2021 г.

СОДЕРЖАНИЕ

Цель работы

Изучение работы с рекурсией.

Задание

1. Треугольная последовательность Дана монотонная последовательность, в которой каждое натуральное число k встречается ровно k раз: 1, 2, 2, 3, 3, 3, 4, 4, 4, 4,… По данному натуральному n выведите первые n членов этой последовательности. Попробуйте обойтись только одним циклом for.

2. От 1 до n Дано натуральное число n. Выведите все числа от 1 до n.

3. От A до B Даны два целых числа A и B (каждое в отдельной строке). Выведите все числа от A до B включительно, в порядке возрастания, если A < B, или в порядке убывания в противном случае.

4. Заданная сумма цифр Даны натуральные числа k и s. Определите, сколько существует k-значных натуральных чисел, сумма цифр которых равна d. Запись натурального числа не может начинаться с цифры 0. В этой задаче можно использовать цикл для перебора всех цифр, стоящих на какой-либо позиции.

5. Сумма цифр числа Дано натуральное число N. Вычислите сумму его цифр. При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется).

6. Проверка числа на простоту Дано натуральное число n>1. Проверьте, является ли оно простым. Программа должна вывести слово YES, если число простое и NO, если число составное. Алгоритм должен иметь сложность O(logn). Указание. Понятно, что задача сама по себе нерекурсивна, т.к. проверка числа n на простоту никак не сводится к проверке на простоту меньших чисел. Поэтому нужно сделать еще один параметр рекурсии: делитель числа, и именно по этому параметру и делать рекурсию.

7. Разложение на множители Дано натуральное число n>1. Выведите все простые множители этого числа в порядке неубывания с учетом кратности. Алгоритм должен иметь сложность O(logn)

8. Палиндром Дано слово, состоящее только из строчных латинских букв. Проверьте, является ли это слово палиндромом. Выведите YES или NO. При решении этой задачи нельзя пользоваться циклами, в решениях на питоне нельзя использовать срезы с шагом, отличным от 1.

9. Без двух нулей Даны числа a и b. Определите, сколько существует последовательностей из a нулей и b единиц, в которых никакие два нуля не стоят рядом.

10. Разворот числа Дано число n, десятичная запись которого не содержит нулей. Получите число, записанное теми же цифрами, но в противоположном порядке. При решении этой задачи нельзя использовать циклы, строки, списки, массивы, разрешается только рекурсия и целочисленная арифметика. Фунция должна возвращать целое число, являющееся результатом работы программы, выводить число по одной цифре нельзя.

11. Количество единиц Дана последовательность натуральных чисел (одно число в строке), завершающаяся двумя числами 0 подряд. Определите, сколько раз в этой последовательности встречается число 1. Числа, идущие после двух нулей, необходимо игнорировать. В этой задаче нельзя использовать глобальные переменные и параметры, передаваемые в функцию. Функция получает данные, считывая их с клавиатуры, а не получая их в виде параметров.

12. Вывести нечетные числа последовательности Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Выведите все нечетные числа из этой последовательности, сохраняя их порядок. В этой задаче нельзя использовать глобальные переменные и передавать какиелибо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция не возвращает значение, а сразу же выводит результат на экран. Основная программа должна состоять только из вызова этой функции.

13. Вывести члены последовательности с нечетными номерами Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Выведите первое, третье, пятое и т.д. из введенных чисел. Завершающий ноль выводить не надо. В этой задаче нельзя использовать глобальные переменные и передавать какиелибо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция не возвращает значение, а сразу же выводит результат на экран. Основная программа должна состоять только из вызова этой функции.

14. Цифры числа слева направо Дано натуральное число N. Выведите все его цифры по одной, в обычном порядке, разделяя их пробелами или новыми строками. При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется). Разрешена только рекурсия и целочисленная арифметика

15. Цифры числа справа налево Дано натуральное число N. Выведите все его цифры по одной, в обратном порядке, разделяя их пробелами или новыми строками. При решении этой задачи нельзя использовать строки, списки, массивы (ну и циклы, разумеется). Разрешена только рекурсия и целочисленная арифметика.

16. Количество элементов, равных максимуму Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Определите, какое количество элементов этой последовательности, равны ее наибольшему элементу. В этой задаче нельзя использовать глобальные переменные. Функция получает данные, считывая их с клавиатуры, а не получая их в виде параметра. В программе на языке Python функция возвращает результат в виде кортежа из нескольких чисел и функция вообще не получает никаких параметров. В программе на языке C++ результат записывается в переменные, которые передаются в функцию по ссылке. Других параметров, кроме как используемых для возврата значения, функция не получает. Гарантируется, что последовательность содержит хотя бы одно число (кроме нуля)

17. Максимум последовательности Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Определите наибольшее значение числа в этой последовательности. В этой задаче нельзя использовать глобальные переменные и передавать какиелибо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция возвращает единственное значение: максимум считанной последовательности. Гарантируется, что последовательность содержит хотя бы одно число (кроме нуля).

Выполнение работы

Все алгоритмы были выполнены при помощи рекурсии. Самым важным в выполнении является задать условие выхода, в противном случае наш алгоритм зациклится.

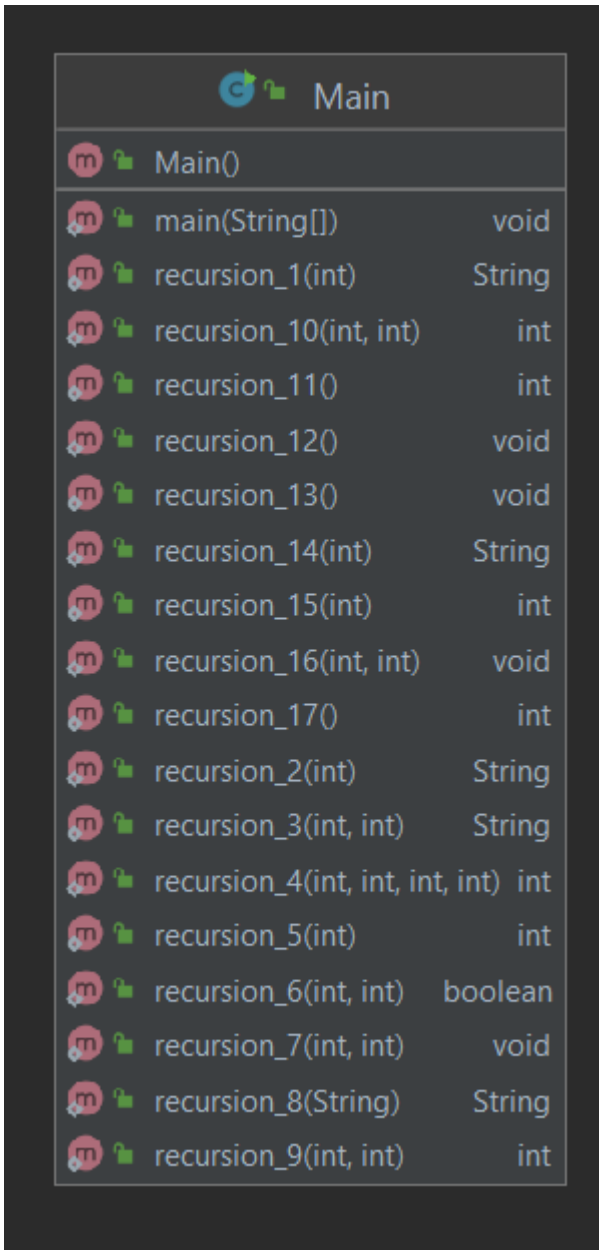

Рисунок 1 – UML Диаграмма

Код выполненной работы

Здесь в нескольких скриншотах можно увидеть, как выглядит код полученного задания и его вывод.

```
public static String recursion_1(int n) {
    int s = 0, j = 0;
    if (n == 1) {
        return "1";
    }
    else {
        for (int i = 1; s < n; i++) {
            s += i;
            j = i;
        }
        System.out.print(recursion_1( n: n-1) + " " + j);
    }
    return "";
}
```

Рисунок 2 – Алгоритм 1

```
public static String recursion_2(int n) {
    if (n == 1) {
        return "1";
    }
    return recursion_2( n: n - 1) + " " + n;
}
```

Рисунок 3 – Алгоритм 2

```java
public static String recursion_3(int a, int b) {
    if (a > b) {
        if (a == b) {
            return Integer.toString(a);
        }
        return a + " " + recursion_3( a: a - 1, b);
    }
    else {
        if (a == b) {
            return Integer.toString(a);
        }
        return a + " " + recursion_3( a: a + 1, b);
    }
}
```

Рисунок 4 – Алгоритм 3

```java
public static int recursion_4(int length, int sum, int k, int s) {
    int result = 0;
    if (length == k) {
        if (sum == s) {
            return 1;
        } else {
            return 0;
        }
    }
    int c = (length == 0 ? 1 : 0);
    for (int i = c; i < 10; i++) {
        result += recursion_4( length: length + 1,  sum: sum + i, k, s);
    }
    return result;
}
```

Рисунок 5 – Алгоритм 4

```java
public static int recursion_5(int n) {
    if (n < 10) {
        return n;
    }
    else {
        return n % 10 + recursion_5( n: n / 10);
    }
}
```

Рисунок 6 – Алгоритм 5

```java
public static boolean recursion_6(int n, int i) {
    if (n < 2) {
        return false;
    }
    else if (n == 2) {
        return true;
    }
    else if (n % i == 0) {
        return false;
    }
    else if (i < n / 2) {
        return recursion_6(n, i: i + 1);
    } else {
        return true;
    }
}
```

Рисунок 7 – Алгоритм 6

```java
public static void recursion_7(int n, int k) {
    if (k > n / 2) {
        System.out.println(n);
        return;
    }
    if (n % k == 0) {
        System.out.println(k);
        recursion_7( n: n / k, k);
    }
    else {
        recursion_7(n, ++k);
    }
}
```

Рисунок 8 – Алгоритм 7

```java
public static String recursion_8(String s) {
    if (s.length() == 1) {
        return "YES";
    }
    else {
        if (s.substring(0, 1).equals(s.substring(s.length() - 1))) {
            if (s.length() == 2) {
                return "YES";
            }
            return recursion_8(s.substring(1, s.length() - 1));
        }
        else {
            return "NO";
        }
    }
}
```

Рисунок 9 – Алгоритм 8

```java
public static int recursion_9(int a, int b) {
    if (a > b + 1) {
        return 0;
    }
    if (a == 0 || b == 0) {
        return 1;
    }
    return recursion_9(a,  b: b - 1) + recursion_9( a: a - 1,  b: b - 1);
}
```

Рисунок 10 – Алгоритм 9

```java
public static int recursion_10(int n, int i) { return (n==0) ? i : recursion_10( n: n/10,  i: i*10 + n%10 ); }
```

Рисунок 11 – Алгоритм 10

```java
public static int recursion_11() {
    Scanner input = new Scanner(System.in);
    int n = input.nextInt();
    int m = input.nextInt();
    if (n == 1) {
        if (m == 1) {
            return recursion_11() + n + m;
        }
        else {
            int k = input.nextInt();
            if (k == 1) {
                return recursion_11() + n + m + k;
            } else {
                return n + m + k;
            }
        }
    }
    else {
        if (m == 1) {
            return recursion_11() + n + m;
        }
        else {
            return n + m;
        }
    }
}
```

Рисунок 12 – Алгоритм 11

```java
public static void recursion_12() {
        Scanner input = new Scanner(System.in);
        int n = input.nextInt();
        if (n > 0) {
            if (n % 2 == 1) {
                System.out.println(n);
                recursion_12();
            }
            else {
                recursion_12();
            }
        }
}
```

Рисунок 13 – Алгоритм 12

```java
public static void recursion_13() {
    Scanner input = new Scanner(System.in);
    int n = input.nextInt();
    if (n > 0) {
        int m = input.nextInt();
        System.out.println(n);
        if (m > 0) {
            recursion_13();
        }
    }
}
```

Рисунок 14 -Алгоритм 13

```java
public static String recursion_14(int n) {
    if (n < 10) {
        return Integer.toString(n);
    }
    else {
        return recursion_14( n: n / 10) + " " + n % 10;
    }
}
```

Рисунок 15 – Алгоритм 14

```java
public static int recursion_15(int n) {
    if (n < 10) {
        return n;
    }
    else {
        System.out.print(n % 10 + " ");
        return recursion_15( n: n / 10);
    }
}
```

Рисунок 16 – Алгоритм 15

```java
public static void recursion_16(int max, int count) {
    Scanner input = new Scanner(System.in);
    int n = input.nextInt();
    if (n > 0) {
        if (n > max) {
            recursion_16(n,  count: 1);
        }
        else if (n == max) {
            recursion_16(max, ++count);
        }
        else {
            recursion_16(max, count);
        }
    }
    else {
        System.out.println(count);
    }
}
```

Рисунок 17 – Алгоритм 16

```java
public static int recursion_17() {
    Scanner input = new Scanner(System.in);
    int n = input.nextInt();
    if (n == 0) {
        return 0;
    }
    else {
        return Math.max(n, recursion_17());
    }
}
```

Рисунок 18 – Алгоритм 17

Вывод

В результате выполнения данной практической работы я потренировал свой навык пользования рекурсивными функциями, а также научился пользоваться режимом отладки, который очень помог при выполнении заданий.