Государственное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«Донецкий национальный технический университет»

Д.09.03.04- ПОИС.21-20/5847.ОП

***Кафедра*** искусственного интеллекта  
 и системного анализа

ОТЧЕТ

по учебной практике

Руководиель практики:

\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доцент О.М. Копытова

(дата, подпись)

Нормоконтроль:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ асс. А.П. Семёнова

(дата, подпись)

Исполнитель:

\_\_\_\_\_\_\_\_\_\_\_\_\_ ст.гр. ПИ-20г Е.М. Евсеев

(дата, подпись)

Донецк — 2021

ЛИСТ ЗАДАНИЯ

Вариант 9

Изучение среды разработки Eclipse и архитектуры ввода-вывода Java

Задание:

1. Пользуясь справочным руководством (doc/Введение-в-среду-разработки-Eclipse.html) установить среду разрабоки, создать простейшее консольное приложение Java, выполнить его запуск и пошаговую трассировку с просмотром значений всех используемых переменных и объектов.
2. Изучить интерфейсы классов ввода-вывода целых, вещественных, строк, а также средства форматированного ввода-вывода.
3. Реализовать задание в соответствии с номером варианта.

Задание 1

Отредактировать заданное предложение, удаляя из него слова, которые встречаются в предложении заданное число раз.

Задание 2

Для каждого столбца прямоугольной матрицы подсчитать сумму входящих в него элементов определить, имеется ли столбцы с одинаковой суммой. Подсчитать количество таких пар столбцов.

Задание 3

Даны действительные числа х, ().Вычислить с точностью сумму бесконечного ряда и указать количество учтенных слагаемых:

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc74705675)

[1 РАЗДЕЛ 4](#_Toc74705676)

[3.1 Постановка задачи 4](#_Toc74705677)

[3.2 Алгоритм решения 4](#_Toc74705678)

[3.3 Экранные формы 6](#_Toc74705679)

[3.4 Листинг 7](#_Toc74705680)

[2 РАЗДЕЛ 8](#_Toc74705681)

[2.1 Постановка задачи 8](#_Toc74705682)

[2.2 Алгоритм решения 8](#_Toc74705684)

[2.3 Экранные формы 10](#_Toc74705685)

[2.4 Листинг 11](#_Toc74705686)

[3 РАЗДЕЛ 12](#_Toc74705687)

[3.1 Постановка задачи 12](#_Toc74705688)

[3.2 Алгоритм решения 12](#_Toc74705689)

[3.3 Экранные формы 15](#_Toc74705690)

[3.4 Листинг 16](#_Toc74705691)

[ЗАКЛЮЧЕНИЕ 17](#_Toc74705692)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 18](#_Toc74705693)

ВВЕДЕНИЕ

Java — объектно-ориентированный язык программирования, разработанный компанией Sun Microsystems. Приложения Java обычно компилируются в специальный байт-код, поэтому они могут работать на любой виртуальной Java-машине (JVM) независимо от компьютерной архитектуры. Дата официального выпуска — 23 мая 1995 года.

Цели работы:

1. Целью работы является изучение средств проектирования, программирования, отладки программ на языке программирования Java с использованием среды разработки Eclipse.
2. Получение навыков составления алгоритмов и программ обработки последовательностей на примере задания вычисления суммы бесконечного ряда.
3. Реализация элементарных математических вычислений с использованием операторов языка программирования Java.
4. Организация элементарного ввода/вывода и осуществление контроля вводимых данных средствами языка Java.

1 РАЗДЕЛ

* 1. Постановка задачи

Задание

Отредактировать заданное предложение, удаляя из него слова, которые встречаются в предложении заданное число раз.

Входные данные*:*

str – строка, предложение;

n – целочисленное число, кол-во повторений слов.

Промежуточные данные:

words – массив строк, хранящий в себе слова предложения;

word – строка, слово из массива words;

occurrences – ассоциативный массив из пар значений строка-число, хранящий в себе слово и кол-во его повторений;

oldCount – целочисленное число, старое значение кол-ва повторений слова;

a – пара значений из ассоциативного массива occurrences.

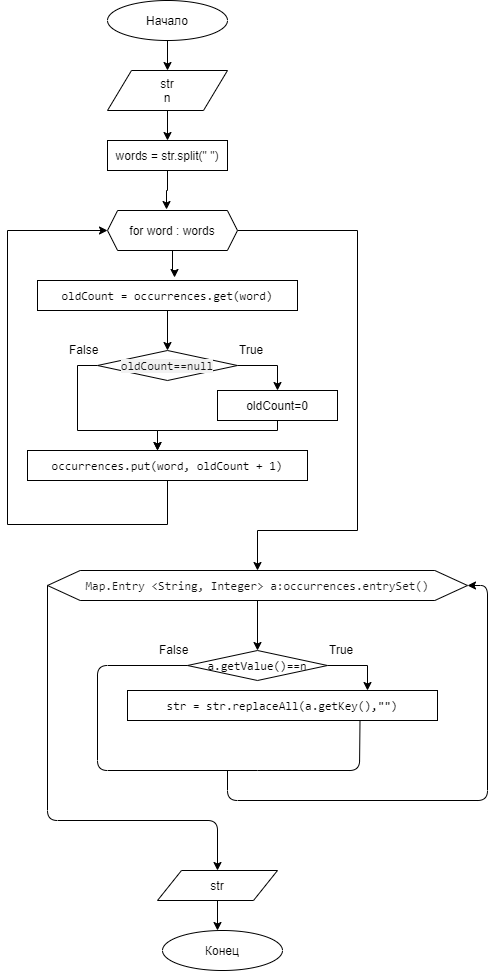
Выходные данные:

str – строка, отредакированное предложение.

* 1. Алгоритм решения

После ввода пользователем предложения и количества повторений, при помощи функции split() разбиваем строку на массив слов. Создаем ассоциативный массив из пар слово-повторения, и при повторении слова увеличиваем счётчик в паре. В другом цикле сравниваем счетчик каждого слова с заданным пользователем числом повторений, и при эквивалентности убираем из предложения это слова с помощью функции replaceAll(). Выводим отредактированное предложение.

Блок-схема:



* 1. Экранные формы

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.1 – Пример 1

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.2– Пример 2

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.3 – Пример 3

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1.4 – Пример 4

* 1. Листинг

1. **import** **java.util.HashMap**;
2. **import** **java.util.Scanner**;
3. **import** **java.util.Map**;
4. **public** **class** **Main** {
5. **public** **static** **void** main(String[] args) {
6. Scanner in = **new** Scanner(System.in);
7. System.out.println(*"Введите предложение:"*);
8. String str = in.nextLine();
9. System.out.println(*"Введите кол-во повторяемых слов для их удаления:"*);
10. **int** n = in.nextInt();
11. String[] words = str.split(*" "*);
12. Map<String, Integer> occurrences = **new** HashMap<String, Integer>();
13. **for** (String word : words) {
14. Integer oldCount = occurrences.get(word);
15. **if** (oldCount == **null**) {
16. oldCount = 0;
17. }
18. occurrences.put(word, oldCount + 1);
19. }
20. **for** (Map.Entry<String, Integer> a : occurrences.entrySet()) {
21. **if** (a.getValue() == n) {
22. str = str.replaceAll(a.getKey(), *""*);
23. }
24. }
25. System.out.println(str);
26. }
27. }
28. РАЗДЕЛ

2.1 Постановка задачи

Задание

Для каждого столбца прямоугольной матрицы подсчитать сумму входящих в него элементов определить, имеется ли столбцы с одинаковой суммой. Подсчитать количество таких пар столбцов.

Входные данные*:*

m, n – целочисленные числа, размеры матрицы;

array[m][n] – целочисленная матрица.

Промежуточные данные:

i, j – целочисленные числа, номер строки и столбца матрицы;

s – целочисленное число, сумма столбца;

w – целочисленное число, счетчик повторений.

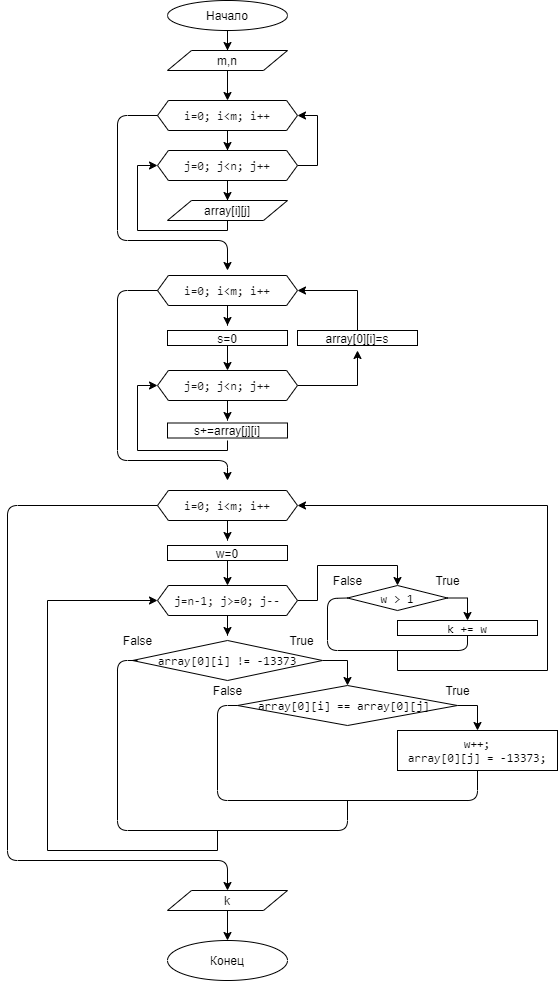
Выходные данные:

k – целочисленное число, количество столбцов с одинаковой суммой.

* 2. Алгоритм решения

После ввода пользователем матрицы в новом цикле находим сумму каждого столбца и записываем ее в первую строку соответствующего столбца. В другом цикле сравниваем данные суммы друг с другом, и при совпадении увеличиваем счетчик повторений. Сравнив все суммы столбцов выводим количество повторений.

Блок-схема:



* 1. Экранные формы

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2.1 – Ввод матрицы и результат программы

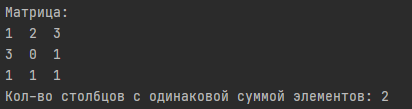


Рисунок 2.2 – Результат программы

* 1. Листинг

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

**public** **class** **Main** {

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

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

System.out.println(*"Введите кол-во строк:"*);

**int** m = in.nextInt();

System.out.println(*"Введите кол-во столбцов:"*);

**int** n = in.nextInt();

**int**[][] array = **new** **int**[m][n];

**for** (**int** i = 0; i < m; i++) {

**for** (**int** j = 0; j < n; j++) {

System.out.printf(*"Введите элемент [%d][%d]:\n"*, i + 1, j + 1);

array[i][j] = in.nextInt();

}

}

System.out.printf(*"Матрица: \n"*);

**for** (**int** i = 0; i < m; i++) {

**for** (**int** j = 0; j < n; j++) {

System.out.printf(*"%d "*, array[i][j]);

}

System.out.printf(*"\n"*);

}

**int** s;

**for** (**int** i = 0; i < n; i++) {

s = 0;

**for** (**int** j = 0; j < m; j++) {

s += array[j][i];

}

array[0][i] = s;

}

**int** w = 0, k = 0;

**for** (**int** i = 0; i < n; i++) {

w = 0;

**for** (**int** j = n - 1; j >= 0; j--) {

**if** (array[0][i] != -13373) {

**if** (array[0][i] == array[0][j]) {

w++;

array[0][j] = -13373;

}

}

}

**if** (w > 1) k += w;

}

System.out.printf(*"Кол-во столбцов с одинаковой суммой элементов: %d"*, k);

}

}

1. РАЗДЕЛ
   1. Постановка задачи

Задание

Даны действительные числа х, ().Вычислить с точностью сумму бесконечного ряда и указать количество учтенных слагаемых:

Входные данные*:*

0 < e < 1, x – вещественные числа, эпсилум и икс;

Промежуточные данные:

first, second – вещественные числа, текущий и предыдущий элементы;

h – функция, возвращаемая вещественное число, отношение текущего к предыдущему элементу.

Выходные данные:

summ – вещественное число, сумма;

k – целочисленное число, количество слагаемых.

* 1. Алгоритм решения

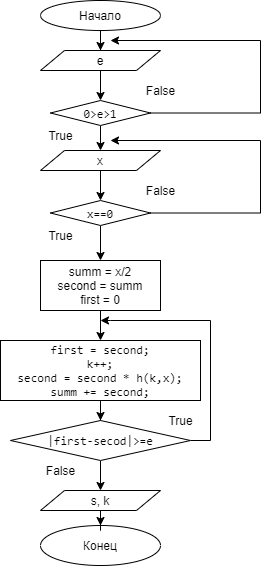
Найдем формулу, с помощью которой мы определим элементы суммы ряда. Для этого возьмем уравнение суммы ряда и найдем отношение текущего к предыдущему элементу:

Таким образом, отношение будет иметь следующий вид:

Используя эту формулу и текущий элемент, найдем следующий. Но для этого нам нужен самый первый элемент, который мы найдем по первоначальному уравнению, подставив начальный коэффициент:

Итак, пользователь вводит ɛ, после чего мы, зная первый элемент, находим второй, вычисляем сумму ряда и модуль разности двух последних элементов. Если модуль разности меньше или равен ɛ, то выводим на экран сумму, иначе продолжаем искать тот элемент, модуль разности с предыдущим элементом которого будет равна или меньше ɛ, после чего выводим сумму всех элементов.

Запишем алгоритм в виде блок-схемы:



Результат контрольного просчета для первых 5 итераций

1)Методом подбора:

2) С помощью отношения текущего к предыдущему элементу:

* 1. Экранные формы

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.1 – Пример 1

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.2 – Пример 2

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.3 – Пример 3

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3.4 – Пример 4

* 1. Листинг

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | **package** com.company;  **import** **java.util.Scanner**;  **public** **class** **main3** {  *//отношение текущего к предыдущему элементу*  **public** **static** **double** h(**int** k,**double** x){  **return** (-(Math.pow(x,2)/(4\*(Math.pow(x,2)+k))));  }  **public** **static** **void** main(String[] args) {  Scanner in = **new** Scanner(System.in);  System.out.println(*"Введите эпсилум (0<e<1):"*);  **double** e = in.nextDouble();  **while** (!(e>0&&1>e)){  System.out.println(*"Неверно. Введите эпсилум:"*);  e = in.nextDouble();  }  System.out.println(*"Введите икс (x!=0):"*);  **double** x = in.nextDouble();  **while** (!(x!=0)){  System.out.println(*"Неверно. Введите икс:"*);  x = in.nextDouble();  }  **int** k=0;  **double** summ= x/2, second=summ, first=0 ;  **while** (Math.abs(first-second)>=e)  {  first = second;  k++;  second = second \* h(k,x);  summ += second;  }  System.out.printf(*"Сумма = %f\nКол-во слагаемых - %d"*, summ,k+1);  }  } |

ЗАКЛЮЧЕНИЕ

В процессе выполнения заданий по учебной практике были исследованы средства проектирования, программирования, отладка программ на языке программирования Java с использованием среды разработки Eclipse. Изучены интерфейсы классов ввода-вывода целых, вещественных, строк а также средства форматированного ввода-вывода.

При реализации заданий были получены практические навыки работы со строчными типами данными, их обработкой с помощью функций Java, матричными структурами данных. Приобретены умения составления алгоритмов и программ обработки последовательностей на примере задачи с вычислением суммы бесконечного ряда. Реализованы элементарные математические вычисления с использованием операторов языка программирования Java. Организованы элементарный ввод/вывод и осуществление контроля вводимых данных средствами языка Java.

Таким образом, был получен основной опыт взаимодействия с актуальным и функциональным языком программирования Java. Изучен синтаксис и основные моменты устройства данного языка, его широкая область применения.

Java — язык программирования общего назначения. Имеет си-подобный синтаксис. Используется для создания приложений в разных областях: от веба до разработки игр, от мобильного ПО до программ для корпораций и научных институтов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Эккель Б.. Философия Java. Библиотека программиста. 4-е изд. – СПб.: Пи- тер, 2009. – 640 с.

2. Хорстманн К. Java 2. Библиотека профессионала, том 1. Основы. 7-е изд.: Пер. с англ. - М.: "Вильямс", 2006. - 896 с.

3. Хорстманн К. Java 2. Библиотека профессионала, том 2. Тонкости програм- мирования. Пер. с англ. - М.: "Вильямс", 2002. - 1120 с.

4. Щилдт Г. Холмс Д. Искусство программирования на Java. Пер. с англ. - М.: "Вильямс", 2005. - 335 с.

5. Арнольд К., Гослинг Д.. Язык программирования Java. Пер. с англ. - СПб.: Питер, 1997. - 304 с.

6. Монахов В. Язык программирования Java и среда NetBeans, 3-е издание. БХВ=Петербург, 2011, 704 с.

7. Казарин С., Клишин А. Среда разработки Java- приложений Eclipse - М., 2008. 77 с.

8. Хемраджани А. Гибкая разработка приложений на Java с помощью Spring, Hubernate и Eclipse.:Пер. с англ. - М.: "Вильямс", 2008. - 352 с.