Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 6

тема «Ветвящиеся алгоритмы»

по дисциплине «Информатика»

Вариант 18

Выполнил: студент группы

ИСТ-23-1б

Романов Ростислав Николаевич

Проверил: ассистент каф. ВММБ

Нетбай Г.В.

Пермь, 2024

Оглавление

[Задача 1 3](#_Toc120982012)

[1.1. Постановка задачи 3](#_Toc120982013)

[1.2. Решение задачи, код программы 3](#_Toc120982014)

[1.3. Тестирование работы программы 3](#_Toc120982015)

[Задача 2 5](#_Toc120982016)

[2.1 Постановка задачи 5](#_Toc120982017)

[2.2 Решение задачи, код программы 5](#_Toc120982018)

[2.3 Тестирование работы программы 5](#_Toc120982019)

[Задача 3 8](#_Toc120982020)

[3.1 Постановка задачи 8](#_Toc120982021)

[3.2 Решение задачи, код программы 8](#_Toc120982022)

[3.3 Тестирование работы программы 9](#_Toc120982023)

[Задача 4 11](#_Toc120982024)

[4.1 Постановка задачи 11](#_Toc120982025)

[4.2 Решение задачи, код программы 11](#_Toc120982026)

[4.3 Тестирование работы программы 12](#_Toc120982027)

[Задача 5 14](#_Toc120982028)

[5.1 Постановка задачи 14](#_Toc120982029)

[5.2 Решение задачи, код программы 14](#_Toc120982030)

[5.3 Тестирование работы программы 14](#_Toc120982031)

[Задача 6 16](#_Toc120982032)

[6.1 Постановка задачи 16](#_Toc120982033)

[6.2 Решение задачи, код программы 16](#_Toc120982034)

[6.3 Тестирование работы программы 16](#_Toc120982035)

[Задача 7 18](#_Toc120982036)

[7.1 Постановка задачи 18](#_Toc120982037)

[7.2 Решение задачи, код программы 18](#_Toc120982038)

[7.3 Тестирование работы программы 18](#_Toc120982039)

[Задача 8 19](#_Toc120982040)

[8.1 Постановка задачи 19](#_Toc120982041)

[8.2 Решение задачи, код программы 19](#_Toc120982042)

[8.3 Тестирование работы программы 19](#_Toc120982043)

[Задача 9 21](#_Toc120982044)

[9.1 Постановка задачи 21](#_Toc120982045)

[9.2 Решение задачи, код программы 21](#_Toc120982046)

[9.3 Тестирование работы программы 21](#_Toc120982047)

[Задача 10 24](#_Toc120982048)

[10.1 Постановка задачи 24](#_Toc120982049)

[10.2 Решение задачи, код программы 24](#_Toc120982050)

[10.3 Тестирование работы программы 24](#_Toc120982051)

# Задача 1

## 1.1. Постановка задачи

Найти сумму первых членов ряда и найти сумму членов ряда, которые меньше заданного с клавиатуры числа:

**

## 1.2. Решение задачи, код программы

package INFIRMATIKA.laba\_4.src;  
  
import java.util.Scanner;  
  
public class Task1 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 System.*out*.print("Введите число x: ");  
 double x = scanner.nextDouble();  
  
 System.*out*.print("Введите количество членов ряда (N): ");  
 int N = scanner.nextInt();  
  
 System.*out*.print("Введите число M: ");  
 double M = scanner.nextDouble();  
  
 double sumOfFirstN = *sumOfFirstN*(x, N);  
 double sumLessThanM = *sumLessThanM*(x, M, N);  
  
 System.*out*.println("Сумма первых " + N + " членов ряда: " + sumOfFirstN);  
 System.*out*.println("Сумма членов ряда, которые меньше " + M + ": " + sumLessThanM);  
 }  
  
 public static double sumOfFirstN(double x, int N) {  
 double sum = 0, term;  
 for (int n = 1; n <= N; n++) {  
 sum += Math.*pow*(x, 2 \* n - 1) / (Math.*cos*(n) + Math.*sin*(n));  
 }  
 return sum;  
 }  
  
 public static double sumLessThanM(double x, double M, int N) {  
 double sum = 0, term;  
 for (int n = 1; n <= N; n++) {  
 term = Math.*pow*(x, 2 \* n - 1) / (Math.*cos*(n) + Math.*sin*(n));  
 if (term < M) {  
 sum += term;  
 }  
 }  
 return sum;  
 }  
}

## 1.3. Тестирование работы программы

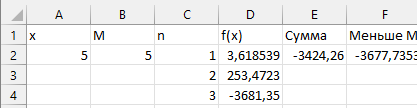
Для проверки задачи в MS Excel создана таблица данных в которой в ячейку А1 записано первое число, в ячейку A2 – второе число , в ячейку A3 – третье число. В ячейки B записаны формулы для вычисления значений, в ячейку С1 записана сумма значений элементов, а в ячейку D1 записана формула для вычисления суммы элементов, меньших числа M.

D2==$A$2^(2\*C2-1)/(COS(C2)+SIN(C2))

E2==СУММ(D:D)

F2==СУММЕСЛИ(D:D;"<B2";D:D)

На рис. 1 представлен вид решения в MS Excel.



Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| №  п.п | Ответ из Java | Ответ из MS Excel |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 2

## 2.1 Постановка задачи

Дано натуральное число. Определить, сколько раз в нем встречается минимальная цифра числа и вычислить факториал от этого количества (например, для числа 100 23 цифра 0 – минимальная, встречается 2 раза, 2! = 2, для числа 46 636 – ответ 1, для числа 11 315 – ответ 6).

## 2.2 Решение задачи, код программы

import java.util.Scanner;  
  
public class Task2 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 System.out.print("Введите натуральное число: ");  
 int number = scanner.nextInt();  
  
 int minDigit = findMinDigit(number);  
 int count = countMinDigitOccurrences(number, minDigit);  
 long factorial = computeFactorial(count);  
  
 System.out.println("Минимальная цифра числа: " + minDigit);  
 System.out.println("Количество вхождений минимальной цифры: " + count);  
 System.out.println("Факториал от количества вхождений: " + factorial);  
 }  
  
 public static int findMinDigit(int number) {  
 int minDigit = 9;  
 while (number > 0) {  
 int digit = number % 10;  
 if (digit < minDigit) {  
 minDigit = digit;  
 }  
 number /= 10;  
 }  
 return minDigit;  
 }  
  
 public static int countMinDigitOccurrences(int number, int minDigit) {  
 int count = 0;  
 while (number > 0) {  
 int digit = number % 10;  
 if (digit == minDigit) {  
 count++;  
 }  
 number /= 10;  
 }  
 return count;  
 }  
  
 public static long computeFactorial(int n) {  
 if (n == 0 || n == 1) {  
 return 1;  
 }  
 long factorial = 1;  
 for (int i = 2; i <= n; i++) {  
 factorial \*= i;  
 }  
 return factorial;  
 }  
}

## 2.3 Тестирование работы программы

Для проверки задачи в MS Excel создана таблица данных в которой в ячейку А2 записаны цифры из числа, в B3- минимальная цифра, С3 – количество минимальных цифр, D2 – факториал от кол-ва цифр в числе.

На рис. 2 представлен вид решения в MS Excel.

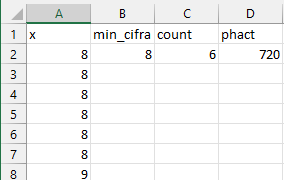


рис.2

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| №  п.п | Решение Java | Решение MS Excel |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 3

## 3.1 Постановка задачи

Написать программы, которые вычисляют выражения:

, , 

## 3.2 Решение задачи, код программы

import java.lang.Math;  
  
public class Task3 {  
 public static void main(String[] args) {  
 double result1 = calculateExpression1();  
 double result2 = calculateExpression2();  
 double result3 = calculateExpression3();  
  
 System.out.println("Результат выражения 1: " + result1);  
 System.out.println("Результат выражения 2: " + result2);  
 System.out.println("Результат выражения 3: " + result3);  
 }  
  
 public static double calculateExpression1() {  
 double sum = 0;  
 for (int i = 1; i <= 8; i++) {  
 double innerSum = 0;  
 for (int j = 1; j <= i; j++) {  
 innerSum += Math.pow(Math.log(j) + Math.log(i), 2);  
 }  
 sum += innerSum;  
 }  
 return sum;  
 }  
  
 public static double calculateExpression2() {  
 double product = 1;  
 for (int i = 1; i <= 5; i++) {  
 double innerProduct = 1;  
 for (int j = 1; j <= i; j++) {  
 innerProduct \*= Math.pow(j, Math.log(i + 1));  
 }  
 product \*= innerProduct;  
 }  
 return product;  
 }  
  
 public static double calculateExpression3() {  
 double product = 1;  
 for (int i = 1; i <= 8; i++) {  
 double innerSum = 0;  
 for (int j = i; j <= 2 \* i - 1; j++) {  
 for (int k = i + j; k <= 2 \* (i + j); k++) {  
 innerSum += (2 \* Math.sin(j) - 3 \* Math.cos(i - 0.5 \* k));  
 }  
 }  
 product \*= innerSum;  
 }  
 return product;  
 }  
}

## 3.3 Тестирование работы программы

Для проверки задачи (1 и 2) в MS Excel создана таблица данных в которой в ячейки 1 строки введены значения , а в значения 1 столбца значения .Во второй таблице в столбце E записаны формулы для вычисления произведений строки, а в ячейке Е10 сумма произведений.

1. B2=(LN($A2)+LN(B$1))^2
2. B2

Для проверки задачи(3) в MS Excel была создана таблица данных, в которой созданы таблички для вычисления суммы для каждого значения . Сумма вычисляется с помощью данного выражения, которое растянуто на определенный диапазон в зависимости от значений j и k. В ячейке J2 записана формула для нахождения произведения этих сумм.

G3=(2\*TAN($F3+1)-3\*LN(SIN($D$2-G$2)))

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение MS Excel |
| 1 |  |  |
| 2 |  |
| 3 | Смотри рис.3.3. |

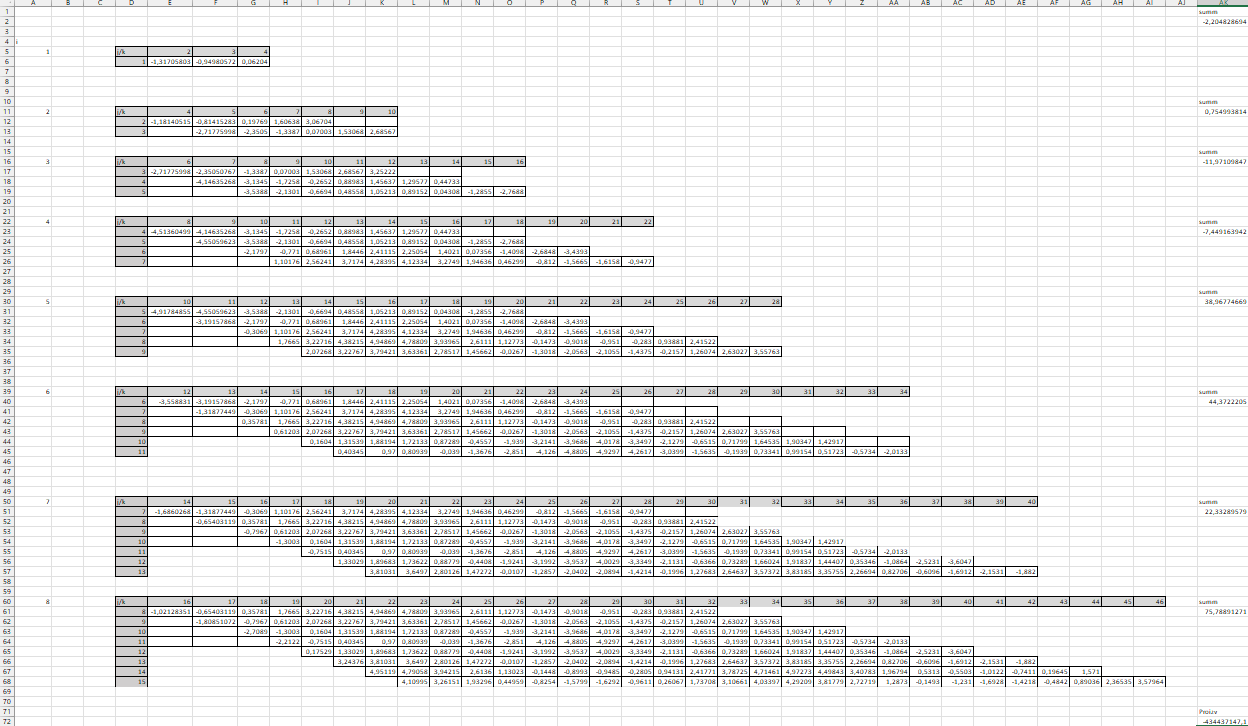


рис.3.3.-Проверка 3 номера на MS Excel

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 4

## 4.1 Постановка задачи

Пользователь вводит десятичную дробь (например, 0,2345 – у дроби нет целой части, если пользователь введет число, где есть целая часть, то должно выплыть сообщение об ошибке). Написать программу перевода десятичной дроби из десятичной системы счисления в систему счисления с основанием, которое пользователь вводит с клавиатуры (вводим ограничения на системы счисления, в которых есть буквенное обозначение цифр, так же не может быть введено 0 и 1 как основание системы счисления). После перевода сделать проверку, определить погрешность, если она есть. Пользователь может выбирать систему счисления до бесконечности, т.е. необходимо предусмотреть внешний цикл с вопросом к пользователю о необходимости продолжать перевод из одной системы счисления в другую.

## 4.2 Решение задачи, код программы

import java.util.Scanner;  
  
public class Task4 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
  
 boolean continueConversion = true;  
  
 while (continueConversion) {  
 System.out.print("Введите десятичную дробь (от 0 до 1): ");  
 String decimalFractionStr = scanner.next();  
 double decimalFraction = Double.parseDouble(decimalFractionStr);  
  
 if (decimalFraction >= 1 || decimalFraction <= 0) {  
 System.out.println("Ошибка: введенное число должно быть десятичной дробью (от 0 до 1).");  
 continue;  
 }  
  
 System.out.print("Введите основание системы счисления (больше 1 и не должно быть равно 1): ");  
 int base = scanner.nextInt();  
 scanner.nextLine(); *// Считываем символ новой строки из буфера ввода* if (base <= 1) {  
 System.out.println("Ошибка: основание системы счисления должно быть больше 1 и не равно 1.");  
 continue;  
 }  
  
 String result = convertDecimalToBase(decimalFraction, base);  
 System.out.println("Результат перевода: " + result);  
  
 double convertedBack = convertBaseToDecimal(result, base);  
 double error = Math.abs(decimalFraction - convertedBack);  
 System.out.println("Погрешность: " + error);  
  
 System.out.print("Желаете продолжить перевод (да/нет)? ");  
 String response = scanner.nextLine();  
 if (!response.equalsIgnoreCase("да")) {  
 continueConversion = false;  
 }  
 }  
 }  
  
 public static String convertDecimalToBase(double decimalFraction, int base) {  
 StringBuilder result = new StringBuilder("0.");  
 while (decimalFraction != 0) {  
 decimalFraction \*= base;  
 int digit = (int) decimalFraction;  
 result.append(digit);  
 decimalFraction -= digit;  
 }  
 return result.toString();  
 }  
  
 public static double convertBaseToDecimal(String number, int base) {  
 double result = 0;  
 for (int i = 2; i < number.length(); i++) {  
 int digit = Character.getNumericValue(number.charAt(i));  
 result += digit / Math.pow(base, i - 1);  
 }  
 return result;  
 }  
}

## 4.3 Тестирование работы программы

Для проверки задачи в онлайн-калькуляторе переводим число в другие СС.

Тестирование работы программы и проверка результатов решения

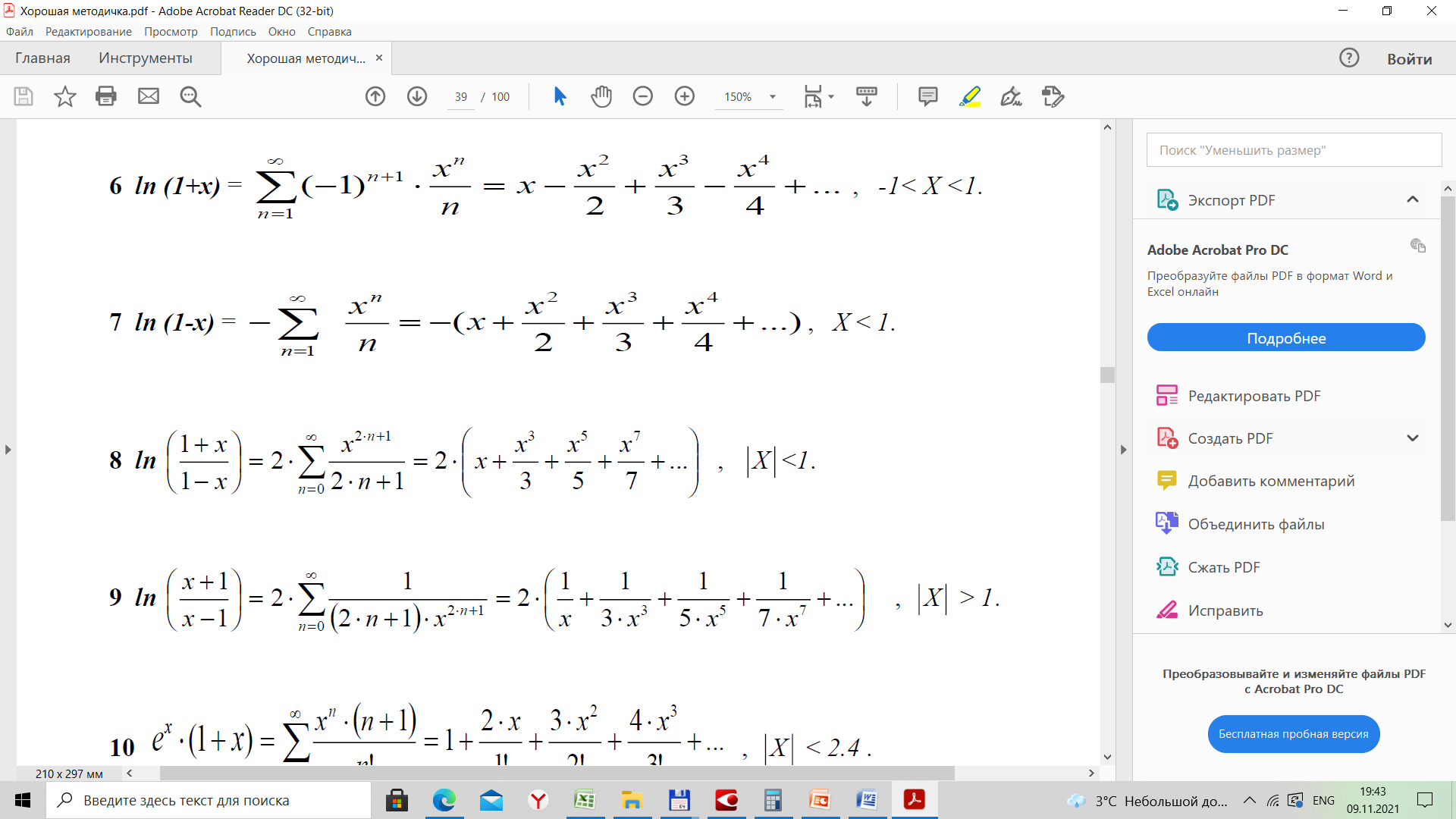
|  |  |  |
| --- | --- | --- |
| №  п.п. | Решение Java | Решение калькулятора |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и онлайн-калькулятора совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 5

## 5.1 Постановка задачи

Организовать цикл для нахождения функции , через ряд Маклорена с погрешностью 10-2 – 10-4, остановка итерационной процедуры , где  – это погрешность. Вывести значение суммы ряда и итоговую погрешность для всех вариантов остановки итерационной процедуры.



## 5.2 Решение задачи, код программы

public class Task5 {  
 public static void main(String[] args) {  
 double x = 0.5; *// Значение аргумента x* double e = 1e-2; *// Начальная погрешность* double sum = 0; *// Начальное значение суммы ряда* int n = 1; *// Начальное значение порядка члена ряда* while (true) {  
 double term = Math.*pow*(-1, n + 1) \* Math.*pow*(x, n) / n; *// Вычисляем очередной член ряда* sum += term; *// Добавляем член ряда к сумме* double ln1x = Math.*log1p*(x); *// Значение ln(1+x) по встроенной функции* double error = Math.*abs*(ln1x - sum); *// Вычисляем текущую погрешность* System.*out*.println("При n = " + n + ", сумма ряда: " + sum + ", погрешность: " + error);  
  
 if (error < e) {  
 break; *// Если погрешность меньше заданной, завершаем цикл* }  
  
 n++; *// Увеличиваем порядок члена ряда* }  
 }  
}

## 5.3 Тестирование работы программы

Для проверки задачи в MS Excel создана таблица данных в которой в ячейку А2 записано значение переменной x, в диапазонах B:B, F:F, J:J – значения переменной n. В ячейки C2, G2, K2 записаны погрешности от 0,0001 до 0, 01. В диапазонах D:D, H:H, L:L записана формула, по которой вычисляется данное выражение. В ячейках E2, I2, M2 записаны формулы, по которой вычисляется итоговые суммы при трех погрешностях.

D2)=(-1)^B2\*(A$2^(2\*B2+1))/(2\*B2+1) E2) =СУММ(D2:D20)

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| №  п.п. | Решение Java | Решение MS Excel |
| 1 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 6

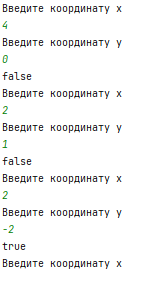
## 6.1 Постановка задачи

Переделать программу (класс) задания 6 лабораторной работы 5 о попадании точки в область в класс без метода main с названием Oblast. Метод main заменить на метод Oblast c входными данными в виде координат произвольной точки пространства и выходными данными типа boolean (true – если точка попала в область, false – если точка не попала в область). Создать программу, взаимодействующую с классом Oblast (без использования наследования), в которой пользователь в цикле проверяет попадание точек в область до бесконечности, т.е. необходимо предусмотреть цикл с вопросом к пользователю о необходимости проверки точки.

## 6.2 Решение задачи, код программы

import java.util.Scanner;  
  
public class Task6 {  
 public static void main(String[] args) {  
 MainProgram mainProgram = new MainProgram();  
 mainProgram.runProgram();  
 }  
}  
  
class Oblast {  
 public static double k(double x1, double y1, double x2, double y2) {  
 return (y1 - y2) / (x1 - x2);  
 }  
  
 public static double b(double x1, double y1, double x2, double y2) {  
 return y1 - x1 \* k(x1, y1, x2, y2);  
 }  
  
 public static boolean check(double x, double y, double xc, double yc, double r) {  
 double distanceSquared = (x - xc) \* (x - xc) + (y - yc) \* (y - yc);  
 return distanceSquared <= r \* r;  
 }  
  
 public static double find\_y(double x1, double y1, double x2, double y2, double x) {  
 return k(x1, y1, x2, y2) \* x + b(x1, y1, x2, y2);  
 }  
}  
  
class Figure extends Oblast {  
 public static boolean big\_circle(double x, double y) {  
 if (check(x, y, 0, 0, 6) && y <= 0) {  
 return true;  
 } else {  
 return false;  
 }  
 }  
  
 public static boolean figura1\_negrativ(double x, double y) {  
 if (y >= find\_y(-4, 0, -3, -3, x) && y >= find\_y(-3, -3, 1, 0, x) && y <= 0) {  
 return true;  
 } else {  
 return false;  
 }  
 }  
  
 public static boolean circle\_small\_negative(double x, double y) {  
 if (check(x, y, 1, -3, 2)) {  
 return true;  
 } else {  
 return false;  
 }  
 }  
  
 public static boolean circle\_small(double x, double y) {  
 if (check(x, y, 0, -3, 1) || check(x, y, 2, -3, 1)) {  
 return true;  
 }  
 return false;  
 }  
  
 public static boolean figura3\_negative(double x, double y) {  
 if (y >= find\_y(1, 0, 4, -3, x) && x <= 4 && y <= 0) {  
 return true;  
 } else {  
 return false;  
 }  
 }  
}  
  
class MainProgram {  
 public void runProgram() {  
 Scanner scanner = new Scanner(System.in);  
 while (true) {  
 System.out.println("Введите координату х");  
 double x = scanner.nextDouble();  
 System.out.println("Введите координату у");  
 double y = scanner.nextDouble();  
  
 boolean status = answer(x, y);  
 System.out.println(status);  
 }  
 }  
   
 public boolean answer(double x, double y) {  
 if (Figure.big\_circle(x, y)) {  
 if (Figure.figura1\_negrativ(x, y) || Figure.figura3\_negative(x, y)) {  
 return false;  
 }  
 if (Figure.circle\_small\_negative(x, y)) {  
 if (Figure.circle\_small(x, y)) {  
 return true;  
 } else {  
 return false;  
 }  
 } else {  
 return true;  
 }  
 } else {  
 return false;  
 }  
 }  
}

## 6.3 Тестирование работы программы



Программа работает корректно, что подтверждает факт правильного написания кода.

# Задача 7

## 7.1 Постановка задачи

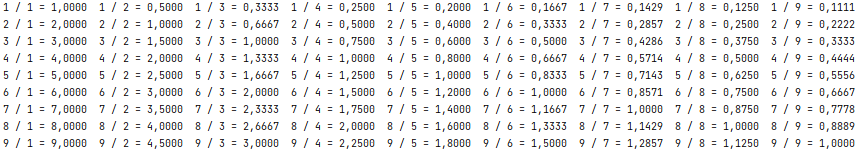
Напечатать полную таблицу деления (не целочисленного) в виде:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 / 1 = 1 | 1 / 2 = 0,5 | ... | 1 / 9 = 0,1111 |
| 2 / 1 = 2 | 2 / 2 = 1 | ... | 2 / 9 = 0,2222 |
| ... | ... | ... | ... |
| 9 / 1 = 9 | 9 / 2 = 4,5 | ... | 9 / 9 = 1 |

## 7.2 Решение задачи, код программы

public class Task7 {  
 public static void main(String[] args) {  
 printDivisionTable();  
 }  
  
 public static void printDivisionTable() {  
 for (int i = 1; i <= 9; i++) {  
 for (int j = 1; j <= 9; j++) {  
 double result = (double) i / j;  
 System.out.print(i + " / " + j + " = ");  
 System.out.printf("%.4f", result); *// Округляем результат до 4 знаков после запятой* System.out.print("\t");  
 }  
 System.out.println();  
 }  
 }  
}

## 7.3 Тестирование работы программы



Программа работает корректно, что подтверждает факт правильного написания кода.

# Задача 8

## 8.1 Постановка задачи

Переделать программу (класс) задания 1 лабораторной работы 4 о нахождении значений 2-х функций в класс без метода main с названием FunctionMy. Метод main заменить на метод FunctionMy c входными данными. Создать программу, взаимодействующую с классом FunctionMy, в которой пользователь в цикле находит сумму 10 значений функции изменяя только один параметр функции в цикле, остальные параметры, которые входя в формулу, считаются константами. Взаимодействие с классом FunctionMy сделать в виде наследования.

## 8.2 Решение задачи, код программы

import java.util.Scanner;  
  
class FunctionMy {  
 double t = 3.14;  
  
 double D(double y, double n) {  
 return y \* y + (0.5 \* n + 4.8) / (Math.sin(y) + Math.E);  
 }  
  
 double L(double x, double y, double z, double n) {  
 return Math.abs((Math.cos(Math.pow(y \* Math.pow(x + 2 \* z, 3), Math.E)) + Math.pow(t, 5) \* ((x + 1) / (2 + y))))   
 / (Math.sin(x \* y + Math.E + Math.pow(y, z)) + 5 \* z);  
 }  
}  
  
class MainProgram extends FunctionMy {  
 public void calculateFunctions() {  
 Scanner input = new Scanner(System.in);  
  
 for (int i = 1; i <= 10; i++) {  
 System.out.println("Итерация " + i);  
  
 System.out.print("Введите значение n: ");  
 double n = input.nextDouble();  
  
 System.out.print("Введите значение y: ");  
 double y = input.nextDouble();  
  
 System.out.print("Введите значение x: ");  
 double x = input.nextDouble();  
  
 System.out.print("Введите значение z: ");  
 double z = input.nextDouble();  
  
 System.out.println("Значение функции D(y) при y = " + y + ", и n = " + n +" равно " + D(y, n));  
 System.out.println("Значение функции L(x, y, z) при x = " + x + ", y = " + y + " и z = " + z + " равно " + L(x, y, z, n));  
 }  
 }  
}  
  
public class Task8 {  
 public static void main(String[] args) {  
 MainProgram mainProgram = new MainProgram();  
 mainProgram.calculateFunctions();  
 }  
}

## 8.3 Тестирование работы программы

|  |  |
| --- | --- |
| № п.п. | Решение Java |
| 1 |  |
| 2 |  |

# Задача 9

## 9.1 Постановка задачи

Написать программу вычисление корня р-й степени (степень вводиться с клавиатуры) в рамках итерационной процедуры . Для определения используется итерационная процедура на основе формулы Ньютона , , при этом . Остановка итерационной процедуры , где  – точность вычисления. В рамках программы определить число итраций, которые потребовались для отыскания корня р-й степени в рамках цикла с параметром для точности от 10-2 до 10-6, шаг 10-1. Организовать форматированный вывод результатов в виде: Точность Корень Число итераций.

## 9.2 Решение задачи, код программы

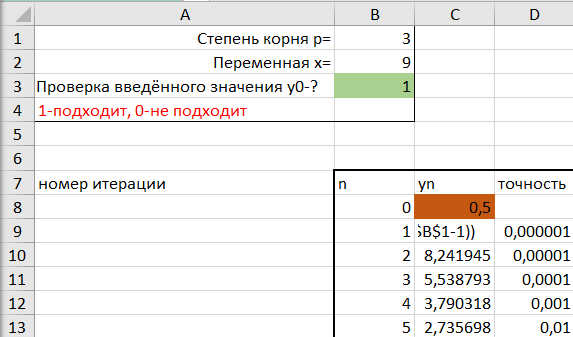
import java.util.Scanner;  
  
public class Task9 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.in);  
 System.out.print("Введите число x: ");  
 double x = scanner.nextDouble();  
 System.out.print("Введите степень p: ");  
 double p = scanner.nextDouble();  
  
 System.out.println("Точность\tКорень\t\tЧисло итераций");  
 for (double epsilon = 0.01; epsilon >= 1e-6; epsilon /= 10) {  
 double y0 = Math.exp(Math.log(x \* (p + 1)));  
 double yn = y0;  
 int iterations = 0;  
 while (true) {  
 double yn1 = (1 / p) \* ((p - 1) \* yn + x / Math.pow(yn, p - 1));  
 iterations++;  
 if (Math.abs(yn1 - yn) <= epsilon) {  
 System.out.printf("%.6f\t%.6f\t%d\n", epsilon, yn1, iterations);  
 break;  
 }  
 yn = yn1;  
 }  
 }  
 }  
}

## 9.3 Тестирование работы программы

Для проверки задачи в MS Excel создана таблица данных, в которой в ячейку B1 записано значение степени корня p, в B2 – значение переменной x, в C8 –начальное значение y0. В B3 записана формула для проверки y0. В диапазоне B8:B13 записано число итераций n. В диапазоне C8:C13 записаны значения корня p-й степени. В диапазоне D9:D13 записаны погрешности.

B3)=ЕСЛИ(C8<EXP(LN(B2\*(B1+1)));1;0)

C9)=1/$B$1\*(($B$1-1)\*C8+$B$2/C8^($B$1-1))



Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение MS Excel |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.

# Задача 10

## 10.1 Постановка задачи

Разработать алгоритм приближённого вычисления площади криволинейной фигуры, ограниченной осью абсцисс, графиком заданной функции , и вертикальными прямыми, т.е. . Каждый отрезок функции представляется в виде параболы c длиной отрезка (шагом)  (см. рис. ниже). Затем площадь под кривой вычисляется по формуле . Вычислить значения площади под кривой при n равном 10, 100, 1000, 10000 в рамках цикла по n. Оценить погрешность решения при разных шагах по сравнению с точным аналитическим решением. Организовать форматированный вывод результатов в виде:

n h S Аналитическое решение Погрешность.

## 10.2 Решение задачи, код программы

public class Task10 {  
 public static void main(String[] args) {  
 double a = 1.0; *// Начальная точка* double b = 2.0; *// Конечная точка* double analyticSolution = computeAnalyticSolution(a, b); *// Аналитическое решение* System.out.println("n\t\t\t h\t\t\t S\t\t\t Аналитическое решение\t\t\t Погрешность");  
 for (int n : new int[]{10, 100, 1000, 10000}) {  
 double h = (b - a) / n; *// Шаг* double numericalSolution = computeNumericalSolution(a, b, n); *// Численное решение* double error = Math.abs(analyticSolution - numericalSolution); *// Погрешность* System.out.printf("%d\t\t%.6f\t\t%.6f\t\t%.6f\t\t\t\t%.6f\n", n, h, numericalSolution, analyticSolution, error);  
 }  
 }  
  
 *// Функция, вычисляющая аналитическое решение* public static double computeAnalyticSolution(double a, double b) {  
 return 4.0 / 3.0 - Math.cos(b) - b \* Math.sin(b) + Math.cos(a) + a \* Math.sin(a);  
 }  
  
 *// Функция, вычисляющая численное решение* public static double computeNumericalSolution(double a, double b, int n) {  
 double h = (b - a) / n;  
 double sum = 0;  
  
 for (int i = 1; i <= n; i++) {  
 double x0 = a + (i - 1) \* h;  
 double x1 = a + i \* h;  
 double midpoint = (x0 + x1) / 2.0;  
 sum += h / 6 \* (f(x0) + 4 \* f(midpoint) + f(x1));  
 }  
  
 return sum;  
 }  
  
 *// Функция, вычисляющая значение функции в точке x* public static double f(double x) {  
 return Math.pow(Math.cos(x) + Math.pow(Math.cos(x), 2), 2);  
 }  
}

## 10.3 Тестирование работы программы

Далее в таблице представлено тестирование работы программы с проверкой решения задачи на языке Java с решением задачи в онлайн-калькуляторе.

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение онлайн-калькулятора |
| 1 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Excel совпадает. Данный факт подтверждает правильность написанного кода программы. Неопределенностей при решении задачи выявлено не было, возможно данная ситуация связанна с малым количеством проверок данных.