Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчёт по лабораторной работе № 4

тема «Линейные алгоритмы в Java»

по дисциплине «Информатика»

Выполнил: студент группы ИСТ-22-1б Золотарев И.О.

11 вариант

Пермь, 2022

**Содержание**

[Задание 1 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149029)

[1.1. Постановка задачи 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149030)

[1.2. Решение задачи, код программы 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149031)

[1.3. Тестирование работы программы с проверкой 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149032)

[Задание 2 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149033)

[2.1. Постановка задачи 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149034)

[2.2. Решение задачи, код программы 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149035)

[2.3. Тестирование работы программы с проверкой 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149036)

[Задание 3 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149037)

[3.1. Постановка задачи 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149038)

[3.2. Решение задачи, код программы 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149039)

[3.3. Тестирование работы программы с проверкой 4](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149040)

[Задание 4 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149041)

[4.1. Постановка задачи 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149042)

[4.2. Решение задачи, код программы 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149043)

[4.3. Тестирование работы программы с проверкой 3](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149044)

[Задание 5 5](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149045)

[5.1. Постановка задачи 5](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149046)

[5.2. Решение задачи, код программы 5](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149047)

[5.3. Тестирование работы программы с проверкой 5](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149048)

[Задание 6 7](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149049)

[6.1. Постановка задачи 8](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149050)

[6.2. Решение задачи, код программы 8](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149051)

[6.3. Тестирование работы программы с проверкой 8](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149052)

[Задание 7 7](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149053)

[7.1. Постановка задачи 7](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149054)

[7.2. Решение задачи, код программы 7](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149055)

[7.3. Тестирование работы программы с проверкой 8](file:///C:\Users\ivanx\Downloads\laba4report.docx#_Toc118149056)

# Задание 1

## 1.1. Постановка задачи

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

**

## 1.2. Решение задачи, код программы

import java.util.Scanner;  
import static java.lang.Math.\*;  
  
public class oliver3 {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 double m = sc.nextDouble();  
 double n = sc.nextDouble();  
 double x = sc.nextDouble();  
 double sum1 = 0;  
 double sum2 = 0;  
 double i = 0;  
 while (i < n){  
 sum1 += *pow*(-1,n-1)\*(*pow*(x-1,2\*n-1))/*getFactorial*(2\*n-1);  
 i += 2;  
 }  
 while (n > 1){  
 if (*pow*(-1,n-1)\*(*pow*(x-1,2\*n-1))/*getFactorial*(2\*n-1) < m)  
 sum2 += (*pow*(-1,n-1)\*(*pow*(x-1,2\*n-1))/*getFactorial*(2\*n-1));  
 n -= 2;  
 }  
 System.*out*.println(sum1);  
 System.*out*.println(sum2);  
 }  
  
 public static double getFactorial(double f) {  
 if (f <= 1) {  
 return 1;  
 }  
 else {  
 return f \* *getFactorial*(f - 1);  
 }  
 }  
}

# Задание 2

## 2.1. Постановка задачи

Дано натуральное число. Определить, сколько раз в нем встречается максимальная цифра (например, для числа 132 233 ответ равен 3, для числа 46 336 – 2, для числа 12 345 – 1).

## 2.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int n; int p; int maxdigit = 0;  
 int count = 0;  
 n = sc.nextInt();  
 while (n > 0) {  
 p = n % 10;  
 if (p > maxdigit) {  
 maxdigit = p;  
 count = 0;  
 }  
 if (p == maxdigit) count++;  
 n = n / 10;  
 }  
 System.*out*.println(count);  
 }  
}

# Задание 3

## 3.1. Постановка задачи

3. Написать программы, которые вычисляют выражения:

, , 

## 3.2. Решение задачи, код программы

1)

import static java.lang.Math.\*;  
  
public class oliver3 {  
 public static void main(String[] args) {  
 double sum=0;  
 for (int i = 1; i <= 8; i++) {  
 for (int j =1; j<=8; j++){  
 sum += *pow*(j+2\*i-1,2);  
 }  
 }  
 System.*out*.println(sum);  
 }  
}

2)

import static java.lang.Math.\*;  
  
public class oliver3 {  
 public static void main(String[] args) {  
 double sum=1;  
 for (int i = 1; i <= 5; i++) {  
 for (int j = 1; j<=i; j++){  
 sum \*= *sin*(i+j);  
  
 }  
 }  
 System.*out*.println(sum);  
 }  
}

3)

import static java.lang.Math.\*;  
  
  
public class oliver3 {  
 public static void main(String[] args) {  
 double i,j,k,a = 0,b = 0,c=1;  
 for (i=1;i<=8;i++){  
 for(j=i;j<=2\*i;j++){  
 for (k=1;k<=j;k++){  
 a += (i+*pow*(j-k,i+j));  
 }  
 b += a;  
 }  
 c \*= b;  
 }  
 System.*out*.println(c);  
 }  
}

# Задание 4

## 4.1. Постановка задачи

Пользователь вводит целое десятичное число. Написать программу перевода целого десятичного числа 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. 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.

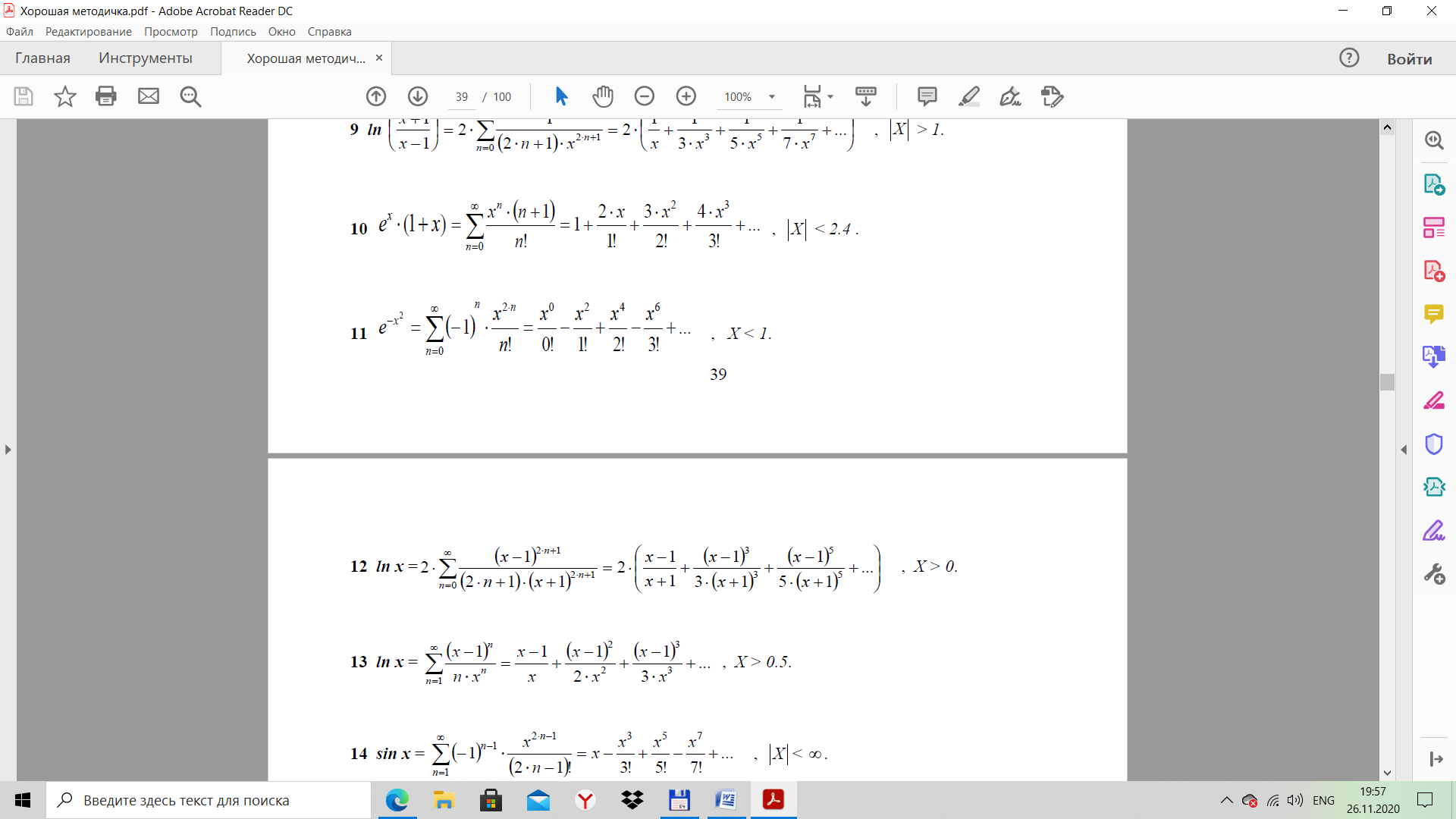
## 4.2. Решение задачи, код программы

import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("input number");  
 int number = sc.nextInt();  
 boolean ans = *getAns*();  
 while (ans) {  
 int ans1 = Integer.*parseInt*(*getInBase*(number));  
 System.*out*.println(ans1);  
 ans = *getAns*();  
 }  
 }  
 public static String getInBase(int number) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("input base");  
 int base = sc.nextInt();  
 if (base < 2 || base >= 11 || number < 0) {  
 throw new IllegalArgumentException();  
 }  
 StringBuilder num = new StringBuilder();  
 while (number > 0) {  
 num.insert(0, number % base);  
 number = number / base;  
 }  
 return num.toString();  
 }  
 public static boolean getAns(){  
 System.*out*.println("change the scale? type: true or false");  
 Scanner sc = new Scanner(System.*in*);  
 return sc.nextBoolean();  
 }  
}

# Задание 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

import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("input number");  
 int number = sc.nextInt();  
 boolean ans = *getAns*();  
 while (ans) {  
 int ans1 = Integer.*parseInt*(*getInBase*(number));  
 System.*out*.println(ans1);  
 ans = *getAns*();  
 }  
 }  
 public static String getInBase(int number) {  
 Scanner sc = new Scanner(System.*in*);  
 System.*out*.println("input base");  
 int base = sc.nextInt();  
 if (base < 2 || base >= 11 || number < 0) {  
 throw new IllegalArgumentException();  
 }  
 StringBuilder num = new StringBuilder();  
 while (number > 0) {  
 num.insert(0, number % base);  
 number = number / base;  
 }  
 return num.toString();  
 }  
 public static boolean getAns(){  
 System.*out*.println("change the scale? type: true or false");  
 Scanner sc = new Scanner(System.*in*);  
 return sc.nextBoolean();  
 }  
}

# Задание 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 static java.lang.Math.\*;  
import java.util.Scanner;  
  
public class Main {  
  
 public static class Oblast{  
 public static boolean CalculateHIt(double x, double y)  
 {  
 return ((x >= 0 && x \* x + y \* y <= 36) && ((*pow*(x - 2, 2) + *pow*(y - 3, 2) >= 4) || (*pow*(x - 2, 2) + *pow*(y - 4, 2) <= 1))  
 && (((*pow*(x - 2, 2) + *pow*(y + 1, 2) <= 1) || (*pow*(x - 2, 2) + *pow*(y + 3, 2) <= 1))||((!(y < -0.2 \* x + 1) || !(x < 5)) || (!(y > 0.4 \* x - 5)))));  
 }  
 }  
 public static boolean getAns(){  
 System.*out*.println("check dots? 1 - yes , else - no");  
 Scanner sc = new Scanner(System.*in*);  
 int ans = sc.nextInt();  
 return ans == 1;  
 }  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 boolean ans = *getAns*();  
 while (ans){  
 System.*out*.println("input x and y");  
 double x = sc.nextDouble();  
 double y = sc.nextDouble();  
 System.*out*.println(Oblast.*CalculateHIt*(x,y));  
 ans = *getAns*();  
 }  
 }  
}

# Задание 7

## 7.1. Постановка задачи

. Напечатать полную таблицу сложения в виде:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 + 1 = 2 | 2 + 1 = 3 | ... | 9 + 1 = 9 |
| 1 + 2 = 3 | 2 + 2 = 4 | ... | 9 + 2 = 11 |
| ... | ... | ... | ... |
| 1 + 9 = 10 | 2 + 9 = 11 | ... | 9 + 9 = 18 |

## 7.2. Решение задачи, код программы

public class Main {  
 public static void main(String[] args) {  
 for(int i = 1; i < 10; i++){  
 for (int j = 1; j < 10; j++){  
 System.*out*.print(j + "+"+ i +"="+ (i + j)+ " ");  
 }  
 System.*out*.println();  
 }  
 }  
}

# Задание 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 static java.lang.Math.\*;  
import java.util.Scanner;  
  
public class Main {  
 public static class FunctionMy {  
 public static double FunctionMy1(double y){  
 double m = 1;  
 return ((*pow*(m,2)+2.8\*m+0.355)/(*cos*(2\*y) + 3.6));  
 }  
 public static double FunctionMy2(double y){  
 double x = 1, d = 1, a = 1,c = 1;  
 return ((*pow*(*cos*(y + *sqrt*(x)),y)+2.4\*d\**cbrt*(y-x))/(*pow*(*exp*(1),y)+*pow*(*exp*(1),*pow*(y,x))\**sqrt*(*abs*(a+x\**pow*(c,4)))));  
 }  
 }  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int n = 0;  
 double sum = 0;  
 while (n < 10){  
 System.*out*.println("input y");  
 double y = sc.nextDouble();  
 sum += FunctionMy.*FunctionMy1*(y) + FunctionMy.*FunctionMy2*(y);  
 n++;  
 }  
 System.*out*.println(sum);  
 }  
}

# Задание 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 static java.lang.Math.\*;  
import java.util.Scanner;  
  
public class Main {  
 public static void main(String[] args) {  
 double ep1 = 0.01; double ep2 = 0.001; double ep3 = 0.0001; double ep4 = 0.00001; double ep5 = 0.000001; int n = 0;  
 Scanner sc = new Scanner(System.*in*);  
 double x = sc.nextDouble();  
 double p = sc.nextDouble();  
 double y0 = *exp*(*log*(x\*(p+1))/p)\*0.9;  
 *hueta*(ep1, x, y0, p, n);  
 *hueta*(ep2, x, y0, p, n);  
 *hueta*(ep3, x, y0, p, n);  
 *hueta*(ep4, x, y0, p, n);  
 *hueta*(ep5, x, y0, p, n);  
 }  
 public static void hueta(double ep,double x,double y,double p,double n){  
 double y\_i = 1/p\*((p - 1)\*y+x/*pow*(y, p - 1));  
 n++;  
 if (*abs*(y\_i - y) > ep) *hueta*(ep, x, y\_i, p, n);  
 else System.*out*.println(ep + ", " + (y\_i) + ", " + n);  
 }  
}

# Задание 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 прямоугольника c длиной отрезка (шагом)  (см. рис. ниже). Затем площадь под кривой вычисляется по формуле  – правый прямоугольник, т.к. высота прямоугольника берется как значение функции в крайней правой координате каждого прямоугольника. 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.

|  |  |
| --- | --- |
|  |  |
| а | б |

Рис. Разбиение площади под кривой на прямоугольники с шагом h:

а – общий вид; б – i-й правый прямоугольник

## 10.2. Решение задачи, код программы

import static java.lang.Math.\*;  
  
public class Main {  
 public static void main(String[] args) {  
 double sum;  
 double h;  
 for(double n = 10;n <= 10000; n \*= 10){  
 h = 1/n;  
 sum = 0;  
 for (int i = 0; i < n; i++){  
 sum += (*cos*(*pow*(n-h\*i,2) + 5 \* (n-h\*i) + 6)+ 5) \* h;  
 }  
 System.*out*.println(n + " " + h + " " + sum + " 5.18" + " "+ *abs*(sum - 5.18));  
 }  
  
 }  
}