Министерство науки и высшего образования РФ

ФГАОУ ВО Пермский национальный исследовательский

политехнический университет

Кафедра «Вычислительная математика, механика и биомеханика»

Отчет по лабораторной работе № 5

тема «Методы и циклы»

по дисциплине «Информатика»

Выполнил: студент группу ИСТ-22-1б Зверев А.А.

Проверил: Нетбай Георгий Владимирович

Пермь, 2022

**Содержание**

[Задание 1 3](#_Toc118317384)

[1.1. Постановка задачи 3](#_Toc118317385)

[1.2. Решение задачи, код программы 3](#_Toc118317386)

[1.3. Тестирование работы программы с проверкой 3](#_Toc118317387)

[Задание 2 5](#_Toc118317388)

[2.1. Постановка задачи 5](#_Toc118317389)

[2.2. Решение задачи, код программы 5](#_Toc118317390)

[2.3. Тестирование работы программы с проверкой 5](#_Toc118317391)

[Задание 3 7](#_Toc118317392)

[3.1. Постановка задачи 7](#_Toc118317393)

[3.2. Решение задачи, код программы 7](#_Toc118317394)

[3.3. Тестирование работы программы с проверкой 7](#_Toc118317395)

[Задание 4 8](#_Toc118317396)

[4.1. Постановка задачи 8](#_Toc118317397)

[4.2. Решение задачи, код программы 8](#_Toc118317398)

[4.3. Тестирование работы программы с проверкой 9](#_Toc118317399)

[Задание 5 9](#_Toc118317400)

[5.1. Постановка задачи 9](#_Toc118317401)

[5.2. Решение задачи, код программы 10](#_Toc118317402)

[5.3. Тестирование работы программы с проверкой 11](#_Toc118317403)

[Задание 6 13](#_Toc118317404)

[6.1. Постановка задачи 13](#_Toc118317405)

[6.2. Решение задачи, код программы 13](#_Toc118317406)

[6.3. Тестирование работы программы с проверкой 13](#_Toc118317407)

[Задание 7 15](#_Toc118317408)

[7.1. Постановка задачи 15](#_Toc118317409)

[7.2. Решение задачи, код программы 16](#_Toc118317410)

[7.3. Тестирование работы программы с проверкой 16](#_Toc118317411)

[Задание 8 17](#_Toc118317412)

[8.1. Постановка задачи 17](#_Toc118317413)

[8.2. Решение задачи, код программы 17](#_Toc118317414)

[8.3. Тестирование работы программы 18](#_Toc118317415)

[Задание 9 19](#_Toc118317416)

[9.1. Постановка задачи 19](#_Toc118317417)

[9.2. Решение задачи, код программы 19](#_Toc118317418)

[9.3. Тестирование работы программы 20](#_Toc118317419)

# 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 static int Factorial(int a){  
 int result = 1;  
 for (int i = 1; i <= a; i++){  
 result \*= i;  
 }  
 return result;  
}  
public static double num1\_1(int x, int n){  
 double s;  
 s = x;  
 for (int i = 3; i<= n; i += 2){  
 s += (*pow*(x,i))/(*Factorial*(n));  
 }  
System.*out*.println("Сумма первых N членов ряда");  
return s;  
}  
public static double num1\_2(int x, int m){  
 double s;  
 s = x;  
 int n;  
 n = 3;  
 while (s < m){  
 s += (*pow*(x,n))/(*Factorial*(n));  
 n += 2;  
 }  
 System.*out*.println("Сумма членов ряда, которые меньше M");  
 return s;  
}

public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 int n,k,x;  
 System.*out*.println("Input x, n & m");  
 x = in.nextInt();  
 n = in.nextInt();  
 int m = in.nextInt();  
 System.*out*.println(*num1\_1*(x,n));  
 System.*out*.println(*num1\_2*(x,m));

}

## 1.3. Тестирование работы программы с проверкой

Далее в таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java.

В ячейке A2 лежит формула для нахождения s:) =СУММ(C2:C12), в ячейке B2 вводимое зн. X в ячейках C2:C12 прописана функция каждого слагаемого :) =($B$2^D2)/ФАКТР(D2) и копирование вниз и зн. n вводимое.

Таблица 1

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение |
| 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.

# Задание 2

## 2.1. Постановка задачи

Написать программу, которая вычисляет количество месяцев необходимых для того, чтобы приобрести компьютер стоимостью  руб. Известно, что ежемесячная зарплата составляет  руб., причем каждый квартал, зарплата увеличивается на 5%. Желаемый компьютер каждый четный месяц дорожает на 3,15% от его стоимости и раз в пять месяцев дешевеет на 2,3%.

## 2.2. Решение задачи, код программы

import java.util.Scanner;  
import static java.lang.Math.\*;

public static int num2(double n, double k){  
 int month;  
 double s;  
 month = 1;  
 s = k;  
 while (s < n){  
 if (month % 3 == 0){  
 k \*= 1.05;  
 }  
 if (month % 2 == 0){  
 n \*= 1.0315;  
 }  
 if (month % 5 == 0){  
 n \*= 0.977;  
 }  
 s += k;  
 month ++;  
 }  
 return month;  
}

public static void main(String[] args) {  
 System.*out*.println("Введите номер задачи от 1 до 9");  
 Scanner in = new Scanner(System.*in*);

double n,k;

System.*out*.println("Input n (PC) and k (Zp)");  
 n = in.nextInt();  
 k = in.nextInt();  
 System.*out*.println(*num2*(n,k) + " Months need to buy pc");

}

## 2.3. Тестирование работы программы с проверкой

Для проверки задачи в MS Excel создана таблица данных в которой в ячейках A2:A6 сумма зарплат, стоимость пк в изменении B2:B6, и D2:D6 изменяемость заработной платы.

На рис. 1 представлен вид решения в MS Excel.

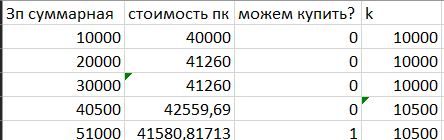


Рис. 1. Решение задачи в MS Excel

Далее в таблице 2 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением задачи в MS Excel.

Таблица 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.

# Задание 3

## 3.1. Постановка задачи

Написать программы, которые вычисляют выражения:

, , 

## 3.2. Решение задачи, код программы

import java.util.Scanner;  
import static java.lang.Math.\*;

public static int num3\_1(){  
 int f = 0;  
 int sum = 0;  
 for (int i = 1; i <= 8; i++){  
 for (int j = 1; j <= i; j++){  
 sum += *pow*((j + i),2);  
 }  
 }  
 return f;  
}  
public static int num3\_2(){  
 int f =1;  
 int umnozh = 1;  
 for (int i = 1; i <=5; i++){  
 for (int j = 1; j <= i; j++){  
 umnozh \*= j;  
 }  
 }  
 return f;  
}  
public static double num3\_3(){  
 int i,j,k;  
 double f = 1;  
 double umnozh = 0;  
 double s = 0;  
 for (i = 1; i <= 8; i++){  
 for (j = 1; j <= 2\*i-1; j++){  
 for (k = i + j; k <= 2\*(i+j); k++){  
 s = 2\*j-3\*(i-0.5\*k);  
 }  
 umnozh += s;  
 }  
 f \*= umnozh;  
 }  
 f = 0;  
 return f;  
}

public static void main(String[] args) {

System.*out*.println("f1 = " + *num3\_1*());  
System.*out*.println("f2 = " + *num3\_2*());  
System.*out*.println("f3 = "+ *num3\_3*());

}

## 3.3. Тестирование работы программы с проверкой

2 часть задания  можно представить как.

Таблица 3

Тестирование работы программы и проверка результатов решения

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

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.

# Задание 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;  
import static java.lang.Math.\*;  
 public static void num4(double a, int sc){  
 //double n =0.1;  
 double i;  
 double result = 0;  
 for (int j = 1;j <= 10; j++ ){  
 i = a\*sc;  
 result += (int)i \* *pow*(0.1,j);  
 a = i - (int)i;  
 }  
 System.*out*.println(result);  
}

public static void main(String[] args) {

System.*out*.println("Input number");  
double a = in.nextDouble();  
int ask = 1;  
while (ask == 1){  
 System.*out*.println("Input sc");  
 int sc = in.nextInt();  
 if (sc >=2 && sc <= 10 && a<1 && a>0){  
 *num4*(a,sc);  
 }  
 else {System.*out*.println("Not correct input");}  
 System.*out*.println("Do u want to change sc 1-yes, else - number no");  
 ask = in.nextInt();  
}

}

## 4.3. Тестирование работы программы с проверкой

Таблица 4

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

# 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 java.lang.Math.\*;  
import java.util.\*;  
  
public class lab6\_6 {  
 public static class Oblast {  
 public static boolean num5\_1figure(double x, double y) {  
 boolean out;  
 if ((!(y > 3.0 \* x + 24.0) || !(x > -8.0) || !(x < -6.0)) && (!(y > -6.0 \* x - 30.0) || !(x > -6.0) || !(x < -5.0)) && (!(y > x + 5.0) || !(x > -5.0) || !(x < -2.0)) && (!(y > 8.0 \* x + 26.0) || !(x < -2.0) || !(x > -4.0)) && !(y < -6.0) && (!(y < 2.0 \* x + 10.0) || !(x > -8.0) || !(x < -6.0)) && (!(y < -x - 8.0) || !(x < -6.0) || !(x > -8.0)) && !(x <= -8.0) && !(y >= 6.0) && !(x >= -2.0)) {  
 out = true;  
 } else if ((x != -6.0 || y != 6.0) && (x != -2.0 || y != 2.0) && (y != -6.0 || !(x <= -4.0) || !(x >= -8.0)) && (x != -8.0 || y != 0.0)) {  
 out = false;  
 } else {  
 out = true;  
 }  
 return out;  
 }  
  
 public static boolean num5\_2figure(double x, double y) {  
 boolean out = false;  
 if ((!(y > 10.0 \* x + 14.0) || !(x > -2.0) || !(x < -1.0)) && (!(y > -2.0 \* x + 2.0) || !(x >= -1.0) || !(x <= 2.0)) && (!(y > 8.0 \* x - 18.0) || !(x >= 2.0) || !(x <= 3.0)) && (!(y > -(1.0 \* x) + 11.0) || !(x > 1.0) || !(x < 6.0)) && (!(y < 0.0 \* x - 4.0) || !(x < 6.0) || !(x > 0.0)) && (!(y > -3.0 \* x - 4.0) || !(x > 0.0) || !(x < 1.0)) && (!(y < 0.0 \* x - 6.0) || !(x < 1.0) || !(x > -2.0)) && !(x <= -2.0) && !(x >= 6.0) && !(y >= 6.0) && !(y <= -7.0)) {  
 out = true;  
 } else if ((x != -2.0 || y != -6.0) && (x != -1.0 || y != 4.0) && (x != 2.0 || y != -2.0) && (x != 3.0 || y != 6.0) && (x != 6.0 || y != 1.0) && (x != 0.0 || y != -4.0) && (x != 1.0 || y != -7.0)) {  
 out = false;  
 } else {  
 out = true;  
 }  
 return out;  
 }  
  
 public static boolean infigure(double x, double y) {  
 return *num5\_1figure*(x, y) || *num5\_2figure*(x, y);  
 }  
  
 public static boolean Checkout() {  
 System.*out*.println("check dots? 1 - yes , 0 - no");  
 Scanner in = new Scanner(System.*in*);  
 int ans = in.nextInt();  
 return ans == 1;  
 }  
  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 boolean checkout = *Checkout*();  
 while (checkout) {  
 System.*out*.println("input x and y");  
 double x = in.nextDouble();  
 double y = in.nextDouble();  
 System.*out*.println(Oblast.*infigure*(x, y));  
 checkout = *Checkout*();  
 }  
 }  
 }  
}

## 6.3. Тестирование работы программы с проверкой

Для проверки нашей программы будем использовать приложенный график и графический редактор Paint.

Красная точка находится в 1 фигуре и имеет координаты (-6;0), зеленая точка находится во 2 фигуре и имеет координаты (0;-2), синяя точка находится вне фигур и имеет координаты (-2;6). Графическое отражение точек на областях изображено на рисунке 6.

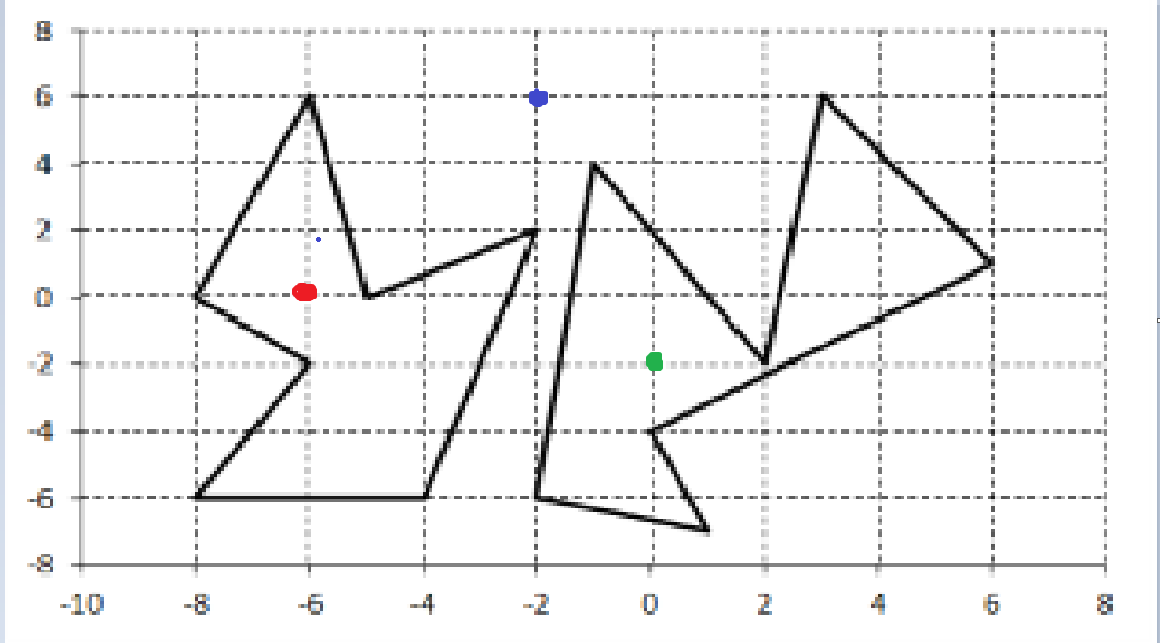


Рис. 6. Решение задачи с использованием paint

Далее в таблице 5 представлено тестирование работы программы с проверкой решения задачи на языке Java с графическим отражением в paint.

Таблица 5

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Отражение в paint |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |

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

# Задание 7

## 7.1. Постановка задачи

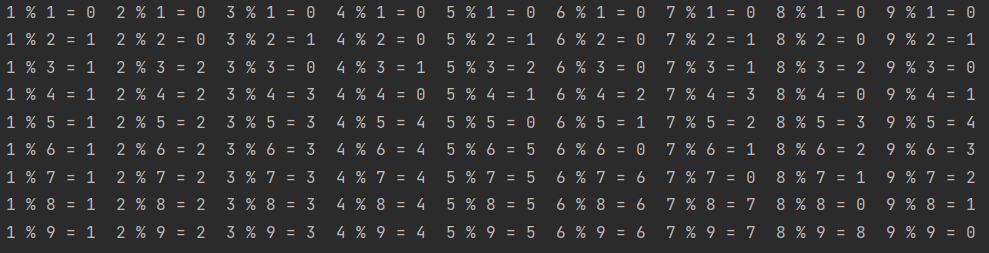
Напечатать полную таблицу остатков от деления в виде:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 % 1 = 0 | 2 % 1 = 0 | ... | 9 % 1 = 0 |
| 1 % 2 = 1 | 2 % 2 = 0 | ... | 9 % 2 = 1 |
| ... | ... | ... | ... |
| 1 % 9 = 1 | 2 % 9 = 2 | ... | 9 % 9 = 0 |

## 7.2. Решение задачи, код программы

public static void num7(){  
 int i = 1;  
 for (int j = 1; j <= 9; j += 1){  
 System.*out*.println(i + " % " + j + " = "+ (i % j) + " " +  
 (i+1) + " % " + j + " = "+ ((i+1) % j) + " " +  
 (i+2) + " % " + j + " = "+ ((i+2) % j) + " " +  
 (i+3) + " % " + j + " = "+ ((i+3) % j) + " " +  
 (i+4) + " % " + j + " = "+ ((i+4) % j) + " " +  
 (i+5) + " % " + j + " = "+ ((i+5) % j) + " " +  
 (i+6) + " % " + j + " = "+ ((i+6) % j) + " " +  
 (i+7) + " % " + j + " = "+ ((i+7) % j) + " " +  
 (i+8) + " % " + j + " = "+ ((i+8) % j) + " ");  
 }  
}

## 7.3. Тестирование работы программы



# 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.\*;  
  
public class lab6\_8 {  
 public static class FunctionMy {  
 public static double FunctionMy1(double x) {  
 double y = 1;  
 return ((*sqrt*(*abs*(x \* x - 6))) - (*sqrt*(*abs*(y \* y - 5))))  
 / (1 + ((x \* x) / (*pow*(y, 3) + 1)) + ((y \* y) / (*pow*(x, 3) + 1)));  
 }  
  
 public static double FunctionMy2(double x) {  
 double z = 1;  
 return *pow*(x, 3) \* (*pow*(*atan*(z), 3) + *E*);  
 }  
 }  
 public static void main(String[] args) {  
 Scanner in = new Scanner(System.*in*);  
 double sum = 0;  
 for (int n = 1; n <= 10; n++) {  
 System.*out*.println("Input x");  
 double x = in.nextDouble();  
 sum += FunctionMy.*FunctionMy1*(x) + FunctionMy.*FunctionMy2*(x);  
 }  
 System.*out*.println(sum);  
 }  
}

## 8.3. Тестирование работы программы с проверкой

Для проверки задачи в MS Excel создана таблица данных в которой в ячейку А2 записана переменная y, в ячейку B2 – d, в ячейку A4 – x, в ячейку B4 – a , в ячейку C4 – b. В ячейки C2 и D4 записаны формулы для вычисления значения функций R и L.

Формулы для вычисления функций R и L:

C2) =(SIN(A2)^2+0,3\*B2)/(EXP(A2)+LN(B2))

D4)=(КОРЕНЬ((3+A4)^6+LN(ABS(A4-КОРЕНЬ(B4\*A4\*C4^3))))+(ATAN(B4-A4^5))^4)/(EXP(EXP(A4+1))-КОРЕНЬ(ABS((COS(ABS(-A4+5)))^2))).

На рис. 1 представлен вид решения в MS Excel.

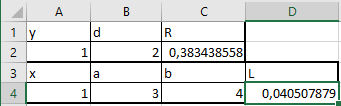


Рис. 1. Решение задачи в MS Excel

Далее в таблице 1 представлено тестирование работы программы с проверкой решения задачи на языке Java с решением задачи в MS Excel.

Таблица 6

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.

# 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 java.util.Scanner;  
import static java.lang.Math.\*;

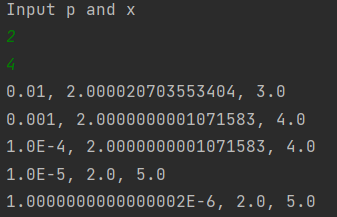
public static void nutonfunc(double e,double x,double y,double p,double n){  
 double y\_i = 1/p\*((p - 1)\*y+x/*pow*(y, p - 1));  
 n++;  
 if (*abs*(y\_i - y) > e) *nutonfunc*(e, x, y\_i, p, n);  
 else System.*out*.println(e + ", " + (y\_i) + ", " + n);  
}  
public static void num9(double x, double p){  
 int n = 0;  
 double y0 = *exp*(*log*(x\*(p+1))/p)\*0.9;  
 for (double e = 0.01; e > 0.000001; e \*= 0.1) {  
 *nutonfunc*(e,x,y0,p,n);  
 }  
}

public static void main(String[] args) {

System.*out*.println("Input p and x");  
double p = in.nextDouble();  
double x1 = in.nextDouble();  
*num9*(x1,p);

}

## 9.3. Тестирование работы программы



# 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 трапеции 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.

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

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

а – общий вид; б – i-я трапеция

## 10.2. Решение задачи, код программы

import java.util.Scanner;  
import static java.lang.Math.\*;

public static void num10(){  
 double sum;  
 double h;  
 for(double n = 10;n <= 10000; n \*= 10){  
 h = 1/n;  
 sum = 0;  
 for (int i = 1; i < n; i++){  
 sum += (((*pow*(*sin*(n-h\*i),2))+*cos*(*pow*(n-h\*i,2)+5)+6+(*pow*(*sin*(n-h\*(i-1)),2))+*cos*(*pow*(n-h\*(i-1),2)+5)+6)/2) \* h;  
 }  
 System.*out*.println(n + " " + h + " " + sum + " 5.18" + " "+ *abs*(sum - 5.18));  
 }  
}

## 10.3. Тестирование работы программы

