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 группы ИСТ-22-1б Братчиков З.С

Проверил: Нетбай Георгий Владимирович

Пермь, 2022

# Задание 1

## Постановка задачи

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



## Решение задачи, код программы

import *java.util.Scanner*;  
  
import static *java.lang.Math*.\*;  
  
public class *z6\_1* {  
 public static void main(*String*[] *args*) {  
 double sum, scrm, cr, chis, znam;  
 chis = 0;  
 znam = 1;  
 sum = 0;  
 scrm = 0;  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.println("введите N");  
 int N = input.nextInt();  
 *System*.out.println("введите M");  
 int M = input.nextInt();  
 for (int i = 1; i <= N; i++){  
 chis += *cos*(i);  
 if (i % 2 != 0) {  
 znam \*= *sin*(i);  
 } else {  
 znam \*= *cos*(i);  
 }  
 cr = chis / znam;  
 sum += cr;  
 if (cr < M) {  
 scrm += cr;  
 }  
 }  
 *System*.out.printf("Сумма первых " + N + " членов ряда = %5.2f\n",sum);  
 *System*.out.printf("Сумма членов ряда, которые меньше " + M + " = %5.2f\n",scrm);  
 }  
}

## 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 находится формула для нахождения S:) =СУММ(E2:E6), в ячейке B2:) =COS(D2), а в ячейках B3:B5:) =COS(D3)+B2; в ячейках C2:) =ЕСЛИ(ОСТАТ(D2;2)=0;cos(D2);SIN(D2)), в ячейках С3:С6:) =ЕСЛИ(ОСТАТ(D3;2)=0;COS(D3);SIN(D3))\*C2 , в столбце D значения х для каждой иттерации цикла, в столбце E:) =B2/C2

Таблица 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.

# Задание 2

## 2.1. Постановка задачи

У студента имеются накопления S руб. Ежемесячная стипендия составляет А руб., а расходы на проживание превышают ее и составляют В руб. Рост цен ежемесячно увеличивает расходы на % равный номеру месяца в году, а каждый три месяца требуется покупка учебников и канцтоваров на M руб. Определить, сколько месяцев сможет прожить студент, используя только накопления и стипендию, а так же за сколько месяцев он сможет накопить долг в 2S руб.

## 2.2. Решение задачи, код программы

import *java.util.Scanner*;  
  
public class *z6\_2* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 double S, A, B, M;  
 *System*.out.print("Введите ваши накопления: ");  
 S = input.nextDouble();  
 *System*.out.print("Введите вашу стипендию: ");  
 A = input.nextDouble();  
 *System*.out.print("Введите расходы на жилье (Расходы > стипендия): ");  
 B = input.nextDouble();  
 *System*.out.print("Введите расходы на канц.товары на 3 мес: ");  
 M = input.nextDouble();  
  
 double traty, m;  
 double sredstva = S;  
  
 for (m = 1; m <= 20; m++) {  
 S += A;  
 B \*= (1.0 + m / 100);  
 S -= B;  
  
 if (m % 3 == 0) {  
 S += A;  
 traty = (B + M) \* (1 + m / 100);  
 S -= traty;  
 }  
 if (S <= 0) {  
 *System*.out.println("На ваши сбережения и стипендию вы сможете прожить: " + m + " месяцев");  
 break;  
 }  
 }  
 for (m = 1; m <= 20; m++) {  
 S += A;  
 B \*= (1.0 + m / 100);  
 S -= B;  
  
 if (m % 3 == 0) {  
 S += A;  
 traty = (B + M) \* (1 + m / 100);  
 S -= traty;  
 }  
 if (*Math*.*abs*(S) >= 2 \* sredstva) {  
 *System*.out.println("долг, в 2 раза больший ваших накоплений вы наберете через " + m + "месяцев");  
 break;  
 }  
 }  
 }  
}

## 2.3. Тестирование работы программы с проверкой

Для проверки задачи в MS Excel создана таблица данных в которой в ячейках A3:A17 сумма сбережений в данный месяц, в ячейке B2 – начальные накопления, в столбце С – стипендия, в столбце D – плата за жилье, в столбце Е – траты на канцтовары, в столбце F – номер месяца.

На рис. 1 представлен вид решения в MS Excel.

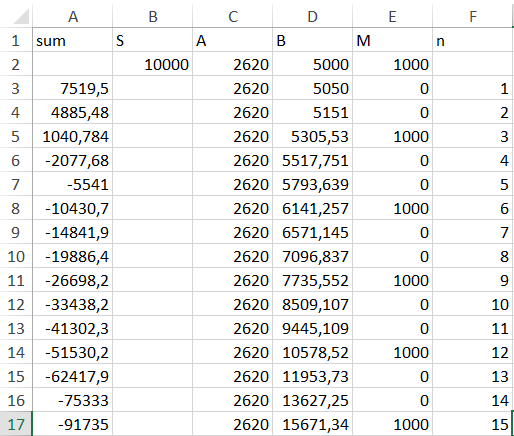


Рис. 1. Решение задачи в MS Excel

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



## 3.2. Решение задачи, код программы

import static *java.lang.Math*.\*;  
  
public class *z\_6\_3\_1* {  
 public static void main(*String*[] *args*) {  
 double sum\_1 = 0;  
 for (int i = 1; i <= 8; i++) {  
 for (int j = 1; j <= i; j++) {  
 sum\_1 += *pow*(j, 2);  
 }  
 }  
 *System*.out.println(sum\_1);  
 }  
}

public class *z\_6\_3\_2* {  
 public static void main(*String*[] *args*) {  
 int pr = 1;  
 int sum = 0;  
 for (int i = 1; i <= 8; i++) {  
 pr = 1;  
 for (int j = 1; j <= 3; j++) {  
 pr = pr\*(j+i);  
 }  
 sum += pr;  
 }  
 *System*.out.println(sum);  
 }  
}

public class *z\_6\_3\_3* {  
 public static void main(*String*[] *args*) {  
 int sum\_1 = 0;  
 int sum\_2 = 0;  
 int pr = 1;  
 for (int i = 1; i <= 8; i++) {  
 sum\_1 = 0;  
 for (int j = 1; j <= i; j++) {  
 pr = 1;  
 for (int k = 1; k <= (2 \* i + j); k++) {  
 pr = pr \* ((2 \* j) - (3 \* (i - k)));  
 }  
 sum\_1 += pr;  
 }  
 sum\_2 += sum\_1;  
 }  
 *System*.out.println(sum\_2);  
 }  
}

## 3.3. Тестирование работы программы с проверкой

Таблица 3

Тестирование работы программы и проверка результатов решения

|  |  |  |
| --- | --- | --- |
| № п.п. | Решение Java | Решение |
| 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.ArrayList*;  
import *java.util.Scanner*;  
import *java.util.StringJoiner*;  
  
public class *z\_6\_4*{  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.println("Хотите выполнить перевод? (true || false)");  
 boolean ans = input.nextBoolean();  
  
 while (ans) {  
 *ArrayList*<*Integer*> nums = new ArrayList<>();  
 *System*.out.print("Введите число в 10-ой СС: ");  
 double num = input.nextDouble();  
 *System*.out.print("Введите систему счисления: ");  
 double new\_system = input.nextInt();  
 double m;  
 int q;  
  
 if (new\_system == 0 || new\_system == 1 || new\_system > 10) {  
 *System*.out.println("ведена не верная система счисления");  
 break;  
 } else {  
 if (num / 10 >= 0.1) {  
 *System*.out.println("введено число, в котором присутствует целая часть");  
 break;  
 } else {  
  
 m = num \* new\_system;  
 q = (int) *Math*.*floor*(m);  
 nums.add(q);  
 for (int i = 1; i <= 15; i++) {  
 m = (m - q) \* new\_system;  
 q = (int) *Math*.*floor*(m);  
 nums.add(q);  
 }  
 }  
 }  
 *StringJoiner* joiner = new StringJoiner("");  
 for (*Integer* integer : nums) {  
 joiner.add(integer.toString());  
 }  
 *String* new\_number = joiner.toString();  
 *System*.out.println("Значение " + num + " в СС с основанием " + new\_system + " = 0," + new\_number);  
 *System*.out.println("Хотите переводить дальше? (true || false)");  
 ans = input.nextBoolean();  
 }  
 *System*.out.println("конец выполнения программы");  
 }  
}

## 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.util.Locale*;  
import *java.util.Scanner*;  
import static *java.lang.Math*.\*;  
import static *java.lang.Math*.\*;  
  
public class *z\_6\_6* {  
 public static boolean Oblast(double *x*, double *y*) {  
 int counter = 0;  
 if ((*x* >= -3 & *x* <= -1 & ((*y* <= (*sqrt*(3 - *pow*(*x*, 2) - 2 \* *x*) + 1)) || (*y* >= -(*sqrt*(3 - *pow*(*x*, 2) - 2 \* *x*) + 1)))) ||  
 (*x* >= -1 & *x* <= 2 & (((*y* >= (1.5 \* *x* + 2)) & (*y* <= 5)) || ((*y* >= -0.25 \* *x* + 0.5) & (*y* <= 1.5 \* *x* + 2) & (*y* <= -*x* + 2)))) ||  
 ((*x* >= -1) & (*x* <= 0) & (*y* >= -1) & (*y* <= -*x* - 1)) ||  
 ((*x* >= -1) & (*x* <= 1) & (*y* <= -1) & (*y* >= (-1.5 \* *x* - 2.5)) & (*y* <= (-3 \* *x* - 1)))) {  
 counter++;  
 if (*x* > -2 & *x* < -1 & *y* > (-*x* - 1) & *y* < (-0.25 \* *x* + 0.5)) {  
 counter--;  
 }  
 } else {  
 return false;  
 }  
 if ((*x* > 2) & (*x* < 4) & (((*y* > 0.5 \* *x* - 2) & (*y* < -0.5 \* *x* + 2)) || ((*y* > -*x* + 1) & (*y* > 2 \* *x* - 8) & (*y* < 0.5 \* *x* - 2)))) {  
 counter = 0;  
 if (((*x* >= 2) & (*x* <= 4) & (*y* >= 1) & (*y* <= 3 \* *x* - 5) & (*y* <= -3 \* *x* + 13)) ||  
 ((*x* >= 2) & (*x* <= 4) & (*y* <= 1) & (*y* >= -0.5 \* *x* + 2)) ||  
 (*x* >= 2 & *x* <= 6 & *y* <= (*sqrt*(-12 - *pow*(*x*, 2) + 8 \* *x*) - 1) & *y* >= (-(*sqrt*(-12 - *pow*(*x*, 2) + 8 \* *x*)) - 1)) ||  
 ((*x* >= 3) & (*x* <= 5) & (*y* >= (-0.5 \* *x* - 2.5)) & (*y* <= *x* - 7) & (*y* <= -2 \* *x* + 5)) ||  
 ((*x* >= 4) & (*x* <= 7) & (*y* >= -3) & (*y* <= *x* - 7) & (*y* <= -2 \* *x* + 11)) ||  
 ((*x* >= 4) & (*x* <= 7) & (*y* <= -3) & (*y* >= -2 \* *x* + 5) & (*y* >= *x* - 10))) {  
 counter++;  
 }  
 }  
 return counter > 0;  
 }  
  
 public static void main(*String*[] *args*) {  
 double x = 0, y = 0;  
 *Scanner* input = new Scanner(*System*.in);  
 boolean stop = false;  
 while (!stop) {  
 *System*.out.println("Введите \"stop\" если хотите закончить выполнение программы.\n" +  
 "Нажмите \"Enter\" если хотите продолжить.");  
 *String* in = input.nextLine().toLowerCase(*Locale*.ROOT);  
 if (in.equals("stop")) {  
 stop = true;  
 } else {  
 *System*.out.print("Введите x: ");  
 x = *Double*.*parseDouble*(input.nextLine());  
 *System*.out.print("Введите y: ");  
 y = *Double*.*parseDouble*(input.nextLine());  
 if (*Oblast*(x, y)) {  
 *System*.out.println("Точка попала в область :)");  
 } else {  
 *System*.out.println("Точка не попала в область");  
 }  
 }  
 }  
 }  
}

## 6.3. Тестирование работы программы с проверкой

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 | Решение в MS Whiteboard |
| 1 |  |  |
| 2 |  |  |

Сравнение решения задачи с использованием двух прикладных пакетов показала, что решения задачи в Java и MS Whiteboard 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

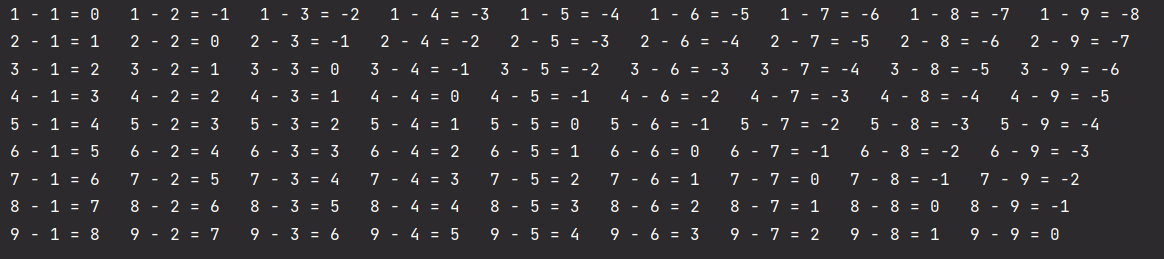
Напечатать полную таблицу вычитания в виде:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 – 1 = 0 | 1 – 2 = –1 | ... | 1 – 9 = –8 |
| 2 – 1 = 1 | 2 – 2 = 0 | ... | 2 – 9 = –7 |
| ... | ... | ... | ... |
| 9 – 1 = 8 | 9 – 2 = 7 | ... | 9 – 9 = 0 |

## 7.2. Решение задачи, код программы

public class *z\_6\_7* {  
 public static void main(*String*[] *args*) {  
 int[][] arr = new int[9][9];  
 for (int i = 1; i <= 9; i++) {  
 for (int j = 1; j <= 9; j++) {  
 *System*.out.print(i + " - " + j + " = " + (i - j) + " ");  
 }  
 *System*.out.println();  
 }  
 }  
}

## 7.3. Тестирование работы программы



# Задание 8

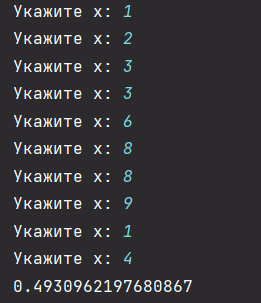
## 8.1. Постановка задачи

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

mport *java.util.Scanner*;  
import static *java.lang.Math*.\*;  
  
public class *z\_6\_8* {  
 public static class *FunctionMy* {  
 public static double FunctionMy1(double *u*) {  
 double y = 1;  
 return (*Math*.*sin*(2 \* *u*) / *log*(2 \* y + *u*));  
 }  
  
 public static double FunctionMy2(double *x*) {  
 double z = 1, b = 1, a = 1;  
 return (*sqrt*(*abs*(*x*) + *pow*(*cos*(*x*), 3) + *pow*(z, 4))) /  
 (*log*(*x*) - *asin*(b \* *x* - a));  
 }  
 }  
  
 public static void main(*String*[] *args*) {  
 *Scanner* in = new Scanner(*System*.in);  
 double sum = 0;  
 double summ = 0;  
 for (int n = 1; n <= 10; n++) {  
 *System*.out.print("Укажите x: ");  
 double x = in.nextDouble();  
 sum += *FunctionMy*.*FunctionMy1*(x);  
 summ += *FunctionMy*.*FunctionMy2*(x);  
 }  
  
 *System*.out.println(sum);  
 *System*.out.println(summ);  
 }  
}

## 8.3. Тестирование работы программы с проверкой



# Задание 9

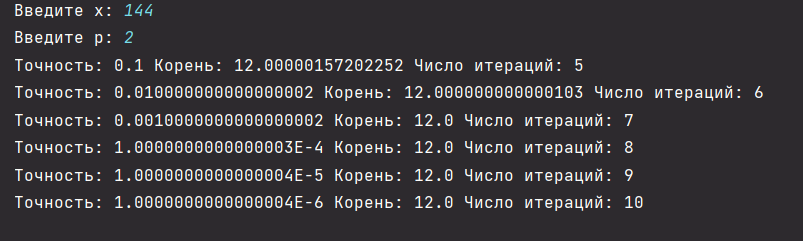
## 9.1. Постановка задачи

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.*\*;  
  
import static *java.lang.Math*.\*;  
  
public class *z\_6\_9* {  
 public static void main(*String*[] *args*) {  
 *Scanner* input = new Scanner(*System*.in);  
 *System*.out.print("Введите x: ");  
 double x = input.nextDouble();  
 *System*.out.print("Введите p: ");  
 double p = input.nextDouble();  
 double y = *log*(x \* (p + 1)) / p;  
 int counter = 0;  
 for (double e = *pow*(10, -1); e > *pow*(10, -6); e \*= 0.1) {  
 while (true) {  
 double ly = y;  
 y = 1 / p \* ((p - 1) \* y + x / *pow*(y, (p - 1)));  
 counter++;  
 if (*abs*(y - ly) < e) {  
 *System*.out.println("Точность: " + e + " Корень: " + y + " Число итераций: " + counter);  
 break;  
 }  
 }  
 }  
 }  
}

## 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 static *java.lang.Math*.\*;  
  
public class *z\_6\_10* {  
 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*(n - h \* i) - *sin*(*pow*(n - h \* i, 2))) \* h;  
 }  
 *System*.out.printf("%-9.0f\n%e\n%f\n5.18\n%f\n\n", n, h, sum, *abs*(sum - 5.18));  
 }  
 }  
}

## 10.3. Тестирование работы программы

