Образец оформления отчёта по Лабораторной работе.

|  |  |
| --- | --- |
| **К Г Э У** | МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«КАЗАНСКИЙ ГОСУДАРСТВЕННЫЙ ЭНЕРГЕТИЧЕСКИЙ УНИВЕРСИТЕТ»**  (ФГБОУ ВО «КГЭУ») |

**Кафедра Информатики и информационных управляющих систем**

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7**

**ПОСТРОЕНИЕ ПРОГРАММ МОДУЛЬНОЙ СТРУКТУРЫ. ОБЪЯВЛЕНИЕ И ОПРЕДЕЛЕНИЕ ФУНКЦИЙ.**

|  |  |
| --- | --- |
| **Исполнитель:** | Банницин Дмитрий |
| **Группа:** | ТРП-3-22 |
| **Вариант:** | 35 |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

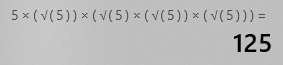
Казань -2022

**Задания для самостоятельной работы**

**Задача №1**

Аналогично Примеру 6, составить отдельные программы для вычисления функций, заданных в Лабораторной работе № 3 (Задание 1). В главной программе для фактического аргумента cгенерировать случайные числа для двух разных значений в диапазон от M до N и подсчитать для них значения соответствующей функции.

**Контрольный пример**

****

**Код**

#include <iostream>

#include <random>

using namespace std;

double getAkPlus(int k, int t);

double getAkMinus(int k, int t);

double getZ(int t, int n, int m);

int main()

{

int t, n, m;

random\_device random\_device;

std::mt19937 generator(random\_device());

uniform\_int\_distribution<> distribution(-9, 4);

cout << "t = ";

cin >> t;

m = distribution(generator);

cout << "m = " << m << endl;

uniform\_int\_distribution<> distribution2(-9, m);

n = distribution2(generator);

cout << "n = " << n << endl;

getZ(t, n ,m);

}

double getAkPlus(int k, int t)

{

if (k == 0)

{

return t;

}

else

{

return getAkPlus(k - 1, t) \* sqrt(t);

}

}

double getAkMinus(int k, int t)

{

if (k == 0)

{

return t;

}

else

{

return getAkMinus(k + 1, t) \* sqrt(t);

}

}

double getZ(int t, int n, int m)

{

if (n > m || m >= 4)

{

cout << "Incorrect value of m or n";

return 0;

}

if (t < 0)

{

cout << "Z(t) = " << t;

}

else if (t >= 0 && t < 10)

{

if (n > 0)

cout << "Z(t) = " << getAkPlus(n, t);

else

cout << "Z(t) = " << getAkMinus(n, t);

}

else

{

if (m > 0)

cout << "Z(t) = " << getAkPlus(m, t);

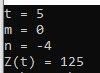
else

cout << "Z(t) = " << getAkMinus(m, t);

}

}

**Решение**

****

**Задача №2**

Написать функцию ShiftRight3(A, B, C), выполняющую правый циклический сдвиг: значение А переходит в В, значение В – в C, значение C – в A. Переменные A, B, C – вещественные параметры, являющиеся одновременно входными и выходными. С помощью этой функции выполнить правый циклический сдвиг для трёх наборов случайных чисел.

**Код**

#include <iostream>

using namespace std;

void ShiftRight3(float \*A, float \*B, float \*C);

int main()

{

float A, B, C;

A = rand();

B = rand();

C = rand();

cout << "A = " << A << endl;

cout << "B = " << B << endl;

cout << "C = " << C << endl;

ShiftRight3(&A, &B, &C);

cout << "-----------------\n";

cout << "A = " << A << endl;

cout << "B = " << B << endl;

cout << "C = " << C << endl;

}

void ShiftRight3(float \*A, float \*B, float \*C)

{

float temp = \*A;

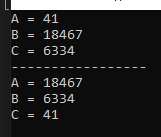
\*A = \*B;

\*B = \*C;

\*C = temp;

}

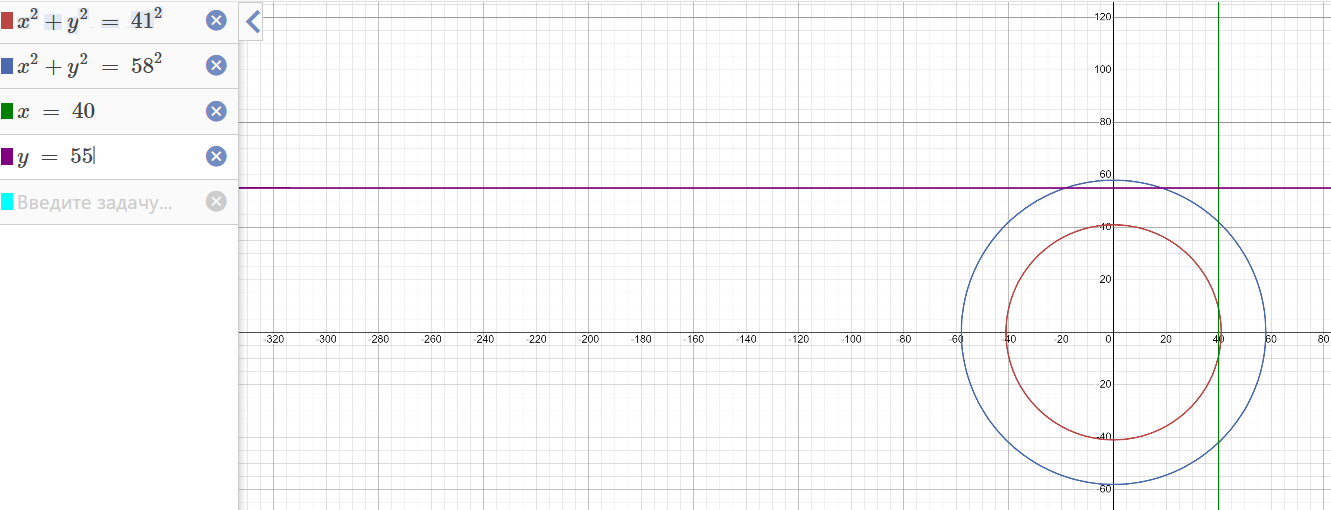
**Решение**

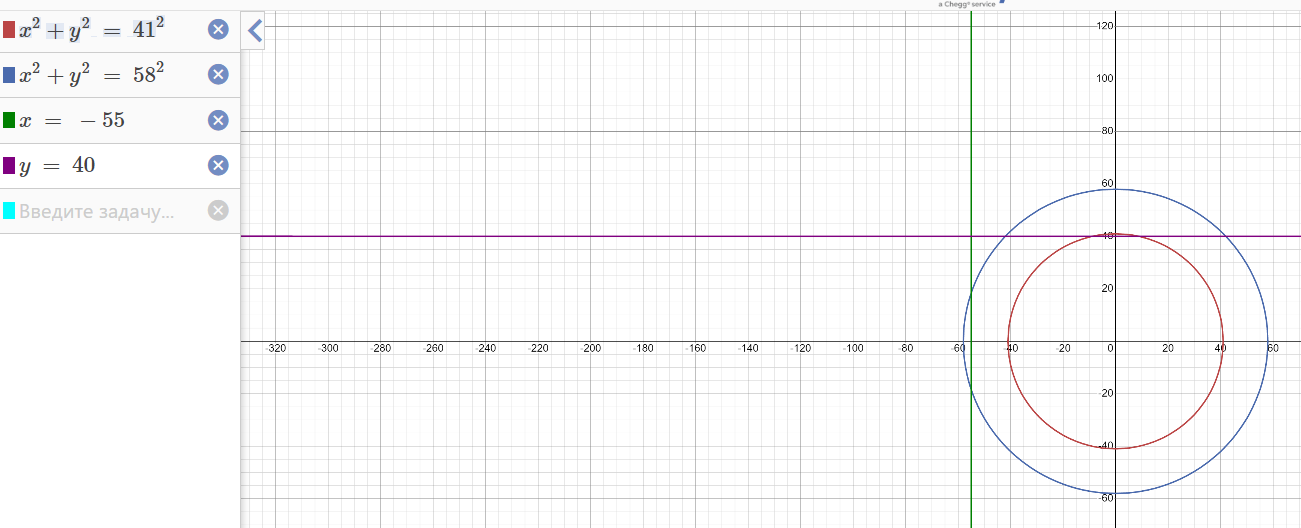
****

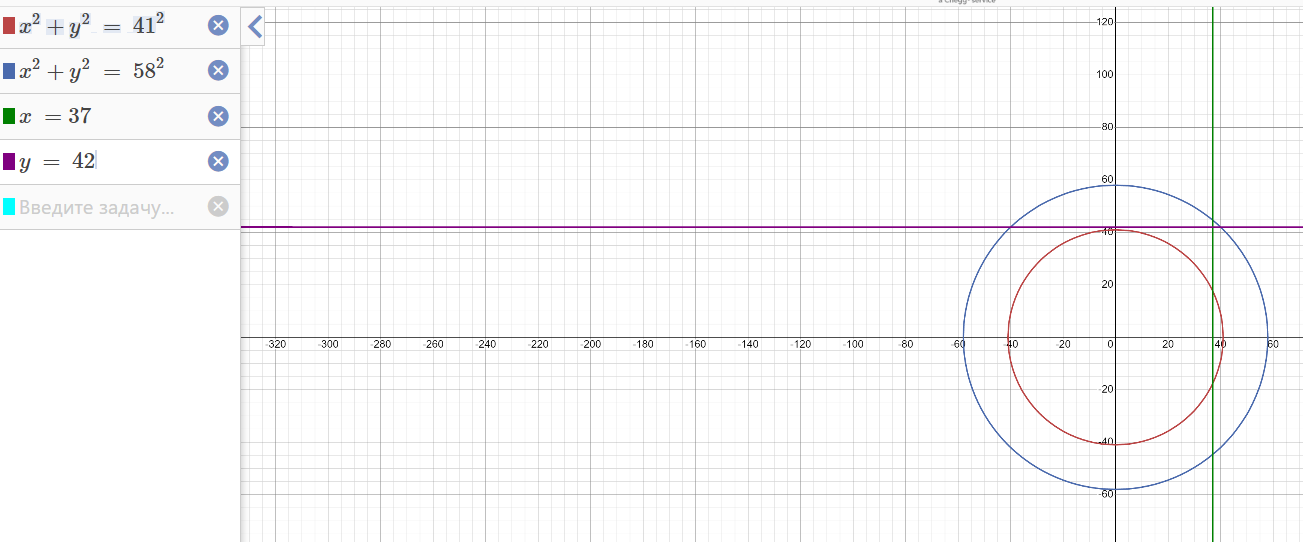
**Задача №3**

Смоделировать игру «Мишень», суть которой в следующем. Компьютер задаёт невидимую игроку мишень в виде кольца, образованного окружностями с центром в начале координат и радиусами R1 и R2. Игрок вводит координаты точки x, y. Игра завершается, когда произошло попадание в мишень и компьютер сообщает о победе игрока, или когда игрок ошибочно вводит хотя бы один нецифровой символ. Если попадания нет, то компьютер сообщает игроку, куда он попал: «внутрь мишени», «ниже | выше мишени», «левее | правее мишени» - и, в случае внешнего положения точки, сообщает её минимальное расстояние d до кольца. Для каждого случая попадания в указанные области и для определения расстояния d разработать отдельные функции.

**Контрольный пример**

****





**Код**

#include <iostream>

using namespace std;

void Hit(bool &win);

void Inside();

void Outside(int x, int y, int R2);

int GetD(int x, int y, int R2);

int main()

{

int R1, R2, x, y;

bool win = false;

R1 = rand() % 100;

R2 = rand() % 50 + R1;

cout << "R1 = " << R1 << endl;

cout << "R2 = " << R2 << endl;

do

{

cout << "x = "; cin >> x;

cout << "y = "; cin >> y;

if (pow(x, 2) + pow(y, 2) <= pow(R2, 2) && pow(x, 2) + pow(y, 2) > pow(R1, 2))

{

Hit(win);

}

else if (pow(x, 2) + pow(y, 2) <= pow(R2, 2))

{

Inside();

}

else

{

Outside(x, y, R2);

}

cout << "-------------\n";

} while (!win);

}

void Hit(bool &win)

{

cout << "hit\n";

win = true;

}

void Inside()

{

cout << "miss, inside the target\n";

}

int GetD(int x, int y, int R2)

{

return sqrt(pow(x, 2) + pow(y, 2)) - R2;

}

void Outside(int x, int y, int R2)

{

int d;

d = GetD(x, y, R2);

cout << "miss\n";

cout << "d = " << d << endl;

if (x < 0 && y < 0)

{

cout << "left|bottom\n";

}

else if (x < 0 && y > 0)

{

cout << "left|top\n";

}

else if (x > 0 && y > 0)

{

cout << "right|top\n";

}

else if (x > 0 && y < 0)

{

cout << "right|bottom\n";

}

else if (x > 0 && y == 0)

{

cout << "right\n";

}

else if (x < 0 && y == 0)

{

cout << "left\n";

}

else if (x == 0 && y < 0)

{

cout << "bottom\n";

}

else if (x == 0 && y >> 0)

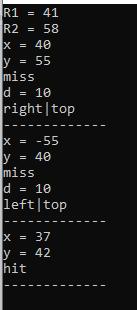
{

cout << "top\n";

}

}

**Решение**

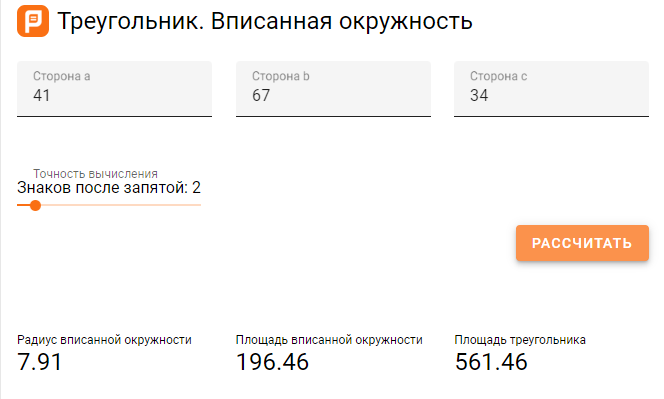
****

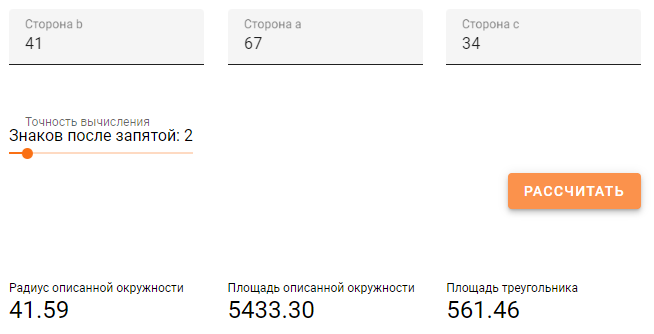
**Домашнее задание.**

**Задание №1.**

Треугольник задан тремя сторонами a, b, c. Составить функции для определения по ним радиусов вписанной и описанной окружностей. Найти сумму и разность полученных радиусов для трёх треугольников, стороны которых задаются случайным образом (использовать или создать собственную функцию для проверки существования треугольника).

**Контрольный пример**

****



**Код**

#include <iostream>

#include <list>

using namespace std;

const double PI = acos(-1);

struct radiuses {

float R\_inside;

float R\_outside;

};

class Triangle

{

private:

int A;

int B;

int C;

float half\_p;

float square;

public:

radiuses\* rad = new radiuses();

public:

Triangle(int a, int b, int c)

{

A = a;

B = b;

C = c;

half\_p = getHalf\_P(a, b, c);

square = getSquare(a, b, c);

getRadiuses(\*rad);

};

private:

void getRadiuses(radiuses &rad)

{

rad.R\_inside = sqrt((half\_p - A)\*(half\_p - B)\*(half\_p - C)/half\_p);

rad.R\_outside = A \* B \* C / 4 / square;

}

float getHalf\_P(int a, int b, int c)

{

return (a + b + c) / 2;

}

float getSquare(int a, int b, int c)

{

return sqrt(half\_p \* (half\_p - a) \* (half\_p - b) \* (half\_p - c));

}

};

bool TriangleExists(int a, int b, int c);

void getSum(Triangle t1, Triangle t2, Triangle t3);

int main()

{

int a, b, c;

Triangle \*trs[3];

for (size\_t i = 0; i < 3; )

{

a = rand() % 100;

b = rand() % 100;

c = rand() % 100;

if (TriangleExists(a, b, c))

{

cout << "a = " << a << endl;

cout << "b = " << b << endl;

cout << "c = " << c << endl;

trs[i] = new Triangle(a, b, c);

cout << "inside radius = " << trs[i]->rad->R\_inside << endl;

cout << "outside radius = " << trs[i]->rad->R\_outside << endl;

cout << "-------------------\n";

i++;

}

}

getSum(\*trs[0], \*trs[1], \*trs[2]);

}

bool TriangleExists(int a, int b, int c) {

if (a + b > c && a + c > b && b + c > a)

return true;

else

return false;

}

void getSum(Triangle t1, Triangle t2, Triangle t3)

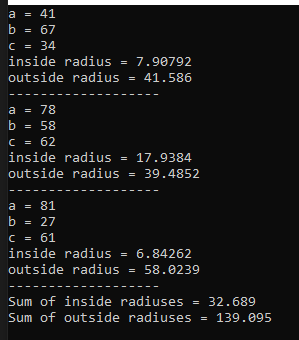
{

cout << "Sum of inside radiuses = " << t1.rad->R\_inside + t2.rad->R\_inside + t3.rad->R\_inside << endl;

cout << "Sum of outside radiuses = " << t1.rad->R\_outside + t2.rad->R\_outside + t3.rad->R\_outside << endl;

}

**Решение**

****

**Задание №2.**

Дано действительное число x. Вычислить:

Разработать программу, которая выводила бы на экран следующую картинку:



**Код**

#include <iostream>

#include <time.h>

using namespace std;

struct coordinates {

pair <float, float> coord1;

pair <float, float> coord2;

pair <float, float> coord3;

};

coordinates getCoords();

coordinates get\_ab(coordinates coords);

void drawTriangle(coordinates ab);

int main()

{

srand(time(0));

coordinates coords = getCoords();

coordinates ab = get\_ab(coords);

drawTriangle(ab);

}

coordinates getCoords() {

coordinates coords;

coords.coord1 = make\_pair(rand() % 15 + 15, rand() % 8);

coords.coord2 = make\_pair(rand() % 15, rand() % 8 + 16);

coords.coord3 = make\_pair(rand() % 15 + 30, rand() % 8 + 8);

return coords;

}

coordinates get\_ab(coordinates coords) {

coordinates ab;

float a1 = float((coords.coord2.second - coords.coord1.second)) / (coords.coord2.first - coords.coord1.first);

float b1 = coords.coord1.second - a1 \* coords.coord1.first;

ab.coord1 = make\_pair(a1, b1);

float a2 = float((coords.coord3.second - coords.coord2.second)) / (coords.coord3.first - coords.coord2.first);

float b2 = coords.coord2.second - a2 \* coords.coord2.first;

ab.coord2 = make\_pair(a2, b2);

float a3 = float((coords.coord1.second - coords.coord3.second)) / (coords.coord1.first - coords.coord3.first);

float b3 = coords.coord1.second - a3 \* coords.coord1.first;

ab.coord3 = make\_pair(a3, b3);

return ab;

}

void drawTriangle(coordinates ab) {

for (size\_t i = 0; i < 24; i++)

{

for (size\_t j = 0; j < 44; j++)

{

if (i > ab.coord1.first \* j + ab.coord1.second &&

i < ab.coord2.first \* j + ab.coord2.second &&

i > ab.coord3.first \* j + ab.coord3.second

)

cout << " ";

else

cout << ".";

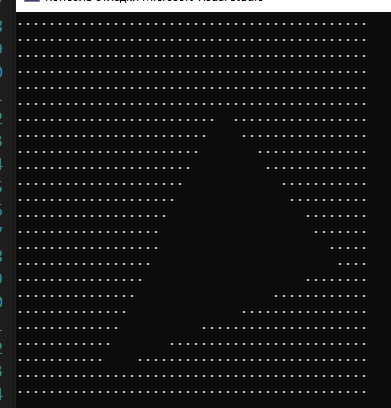
}

cout << endl;

}

}

**Решение**

****