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 № 5**

**ПОСТРОЕНИЕ ПРОГРАММ С АЛГОРИТМАМИ ЦИКЛИЧЕСКОЙ СТРУКТУРЫ. СЧЁТНЫЙ ОПЕРАТОР ЦИКЛА.**

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

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

Казань -2022

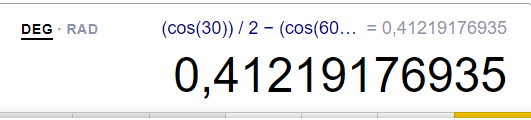
**Задания для самостоятельной работы**

**Задача №1**

Составить программу для вычисления конечных сумм рядов. При необходимости выводить рекуррентные соотношения, избегать возведения в степень (-1).



**Контрольный пример**

****

**Код**

#include <iostream>

using namespace std;

int factorial(int i);

int main()

{

float pi = acos(-1);

float x = 0;

float U = -1;

float S;

int N = 0, i;

float cosix, fact;

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

cout << "N = "; cin >> N;

for (i = 1, S = 0; i <= N; i++)

{

U = -U;

cosix = cos(i \* x / 180 \* pi);

fact = factorial(i \* 2);

S += U \* cosix / fact;

}

cout << "\n S(" << x << ',' << N << ")= " << S << endl;

system("pause");

return 0;

}

int factorial(int i)

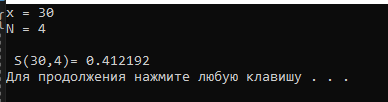
{

if (i == 0) return 1;

else return i \* factorial(i - 1);

}

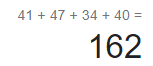
**Решение**

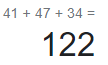
****

**Задача №2**

У прилавка магазина выстроилась очередь из n покупателей. Время обслуживания продавцом i-го покупателя равно ti (i = 1, …, n). Пусть даны натуральное n и действительные t1, … ,tn. Получить n с ,Κ с 1 , где сi – время пребывания i-го покупателя в очереди (i = 1, …, n). Указать номер покупателя, для обслуживания которого продавцу потребовалось самое малое время.

**Контрольный пример**

****

****

****

**Код**

#include <iostream>

using namespace std;

int main()

{

int n = 0;

int time = 0;

int minTime = 9999;

int minNum = 9999;

cout << "n = "; cin >> n;

int\* times = new int[n];

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

times[j] = rand() % 60;

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

cout << "time" << j << " = " << times[j] << endl;

for (int i = 1; i < n+1; i++)

{

time = 0;

for (int j = 0; j < i - 1; j++)

time += times[j];

cout << "\nc" << i << " = " << time;

}

for (int j = 0; j < n; j++)

{

if (times[j] < minTime)

{

minTime = times[j];

minNum = j+1;

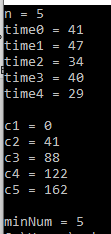
}

}

cout << "\n\nminNum = " << minNum;

}

**Решение**

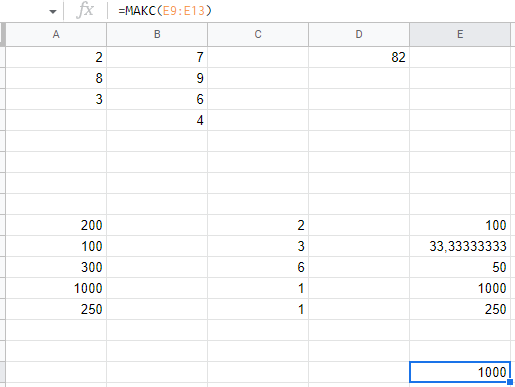
****

**Домашнее задание.**

**Задание №1.**

В 10 шкатулках находятся деньги (200, 100, 300, 1000, 250, 150, 900, 50, 700, 650) долларов. Игроки разложили свои фишки в любую из шкатулок следующим образом (2, 3, 6, 1, 1, 0, 4, 2, 0, 1). После того, как фишки разложены, шкатулки начинают по очереди вскрывать, деньги делятся между игроками по количеству фишек, шкатулка остаётся пустой. Процесс вскрытия шкатулок и распределения денег прекращается, как только встречается шкатулка, в которой нет ни одной фишки. Определить содержание шкатулок после прекращения игры. Оценить максимальное значение выигрыша, если опустошено хотя бы две шкатулки.

**Контрольный пример**



**Код**

#include <iostream>

using namespace std;

int main()

{

int boxes[]{200, 100, 300, 1000, 250, 150,

900, 50, 700, 650};

int chips[]{ 2, 3, 6, 1, 1, 0, 4, 2, 0, 1 };

int maxwin = 0;

for (int i = 0; chips[i] != 0; i++)

{

if (maxwin < boxes[i]/chips[i])

maxwin = boxes[i] / chips[i];

boxes[i] = 0;

}

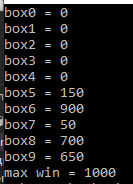
for (int i = 0; i < 10; i++)

cout << "box" << i << " = " << boxes[i] << endl;

cout << "max win = " << maxwin;

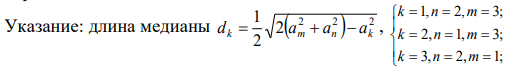
}

**Решение**

****

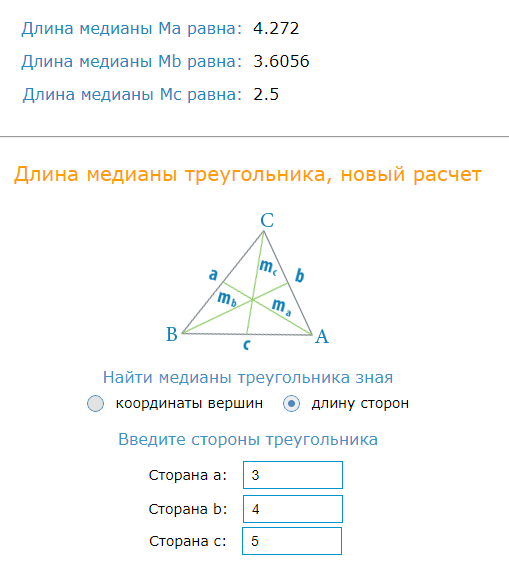
**Задание №2.**

Треугольник задан тремя сторонами a1, а2, а3. Определить длину самой короткой медианы.



**Контрольный пример**

X =



**Код**

#include <iostream>

using namespace std;

float getMedian(int m, int a1, int a2, int a3);

int main()

{

setlocale(LC\_ALL, "Russian");

int a1, a2, a3;

cout << "a1 = "; cin >> a1;

cout << "a2 = "; cin >> a2;

cout << "a3 = "; cin >> a3;

float minWidth = 99999;

float medians[3];

medians[0] = getMedian(1, a1, a2, a3);

medians[1] = getMedian(2, a1, a2, a3);

medians[2] = getMedian(3, a1, a2, a3);

if (medians[0] == 0 || medians[1] == 0 || medians[2] == 0)

return 0;

for (int i = 1; i < 4; i++)

{

switch (i)

{

case 1:

if (medians[0] < minWidth)

minWidth = medians[0];

break;

case 2:

if (medians[1] < minWidth)

minWidth = medians[1];

break;

case 3:

if (medians[2] < minWidth)

minWidth = medians[2];

break;

}

}

for (int i = 0; i < 3; i++)

{

cout << "m" << i+1 << " = " << medians[i] << endl;

}

cout << "min median = " << minWidth << endl;

}

float getMedian(int m, int a1, int a2, int a3)

{

float result = 0;

switch (m)

{

case 1:

if(2 \* (pow(a3, 2) + pow(a2, 2)) < pow(a1, 2))

cout << "Подкоренное выражение меньше нуля";

else

result = sqrt(2 \* (pow(a3, 2) + pow(a2, 2)) - pow(a1, 2)) / 2;

break;

case 2:

if (2 \* (pow(a3, 2) + pow(a1, 2)) < pow(a2, 2))

cout << "Подкоренное выражение меньше нуля";

else

result = sqrt(2 \* (pow(a3, 2) + pow(a1, 2)) - pow(a2, 2)) / 2;

break;

case 3:

if (2 \* (pow(a1, 2) + pow(a2, 2)) < pow(a3, 2))

cout << "Подкоренное выражение меньше нуля";

else

result = sqrt(2 \* (pow(a1, 2) + pow(a2, 2)) - pow(a3, 2)) / 2;

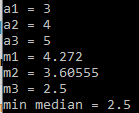
break;

}

return result;

}

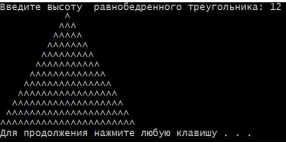
**Решение**

****

**Задание №3.**

Нарисовать равнобедренный треугольник из символов ^. Высоту выбирает пользователь.

**Контрольный пример**



(из методички)

**Код**

#include <iostream>

void printPir(int n);

using namespace std;

int main()

{

int n = 0;

cout << "n = "; cin >> n;

printPir(n);

}

void printPir(int n)

{

cout << endl;

int maxSize = 1;

for (; maxSize < n\*2; maxSize += 2);

for (size\_t i = 1; i < n+1; i++)

{

string str = "";

for (size\_t j = 0; j < (maxSize - 1) / 2 - i; j++, str += " ");

str += "^";

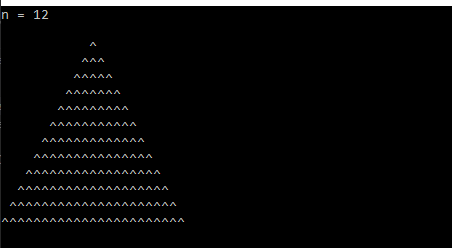
for (size\_t j = 1; j < i \* 2 - 1; j++, str += "^");

cout << str << endl;

}

}

**Решение**

****