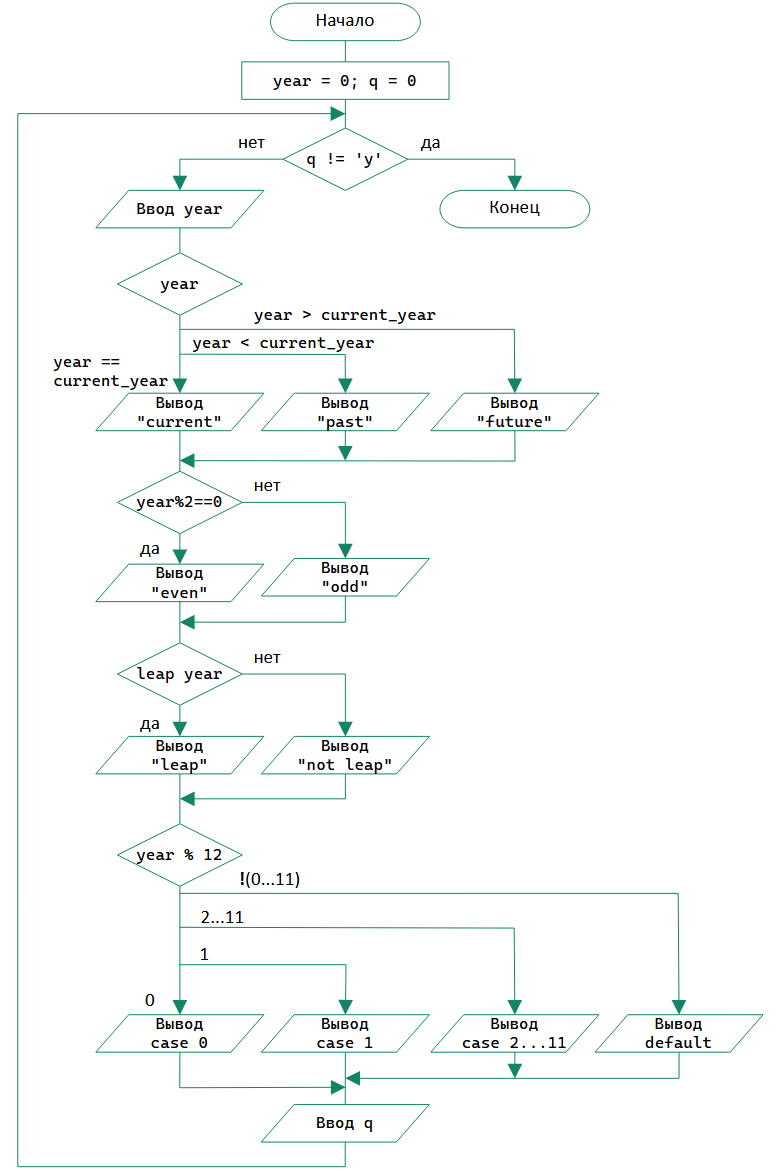
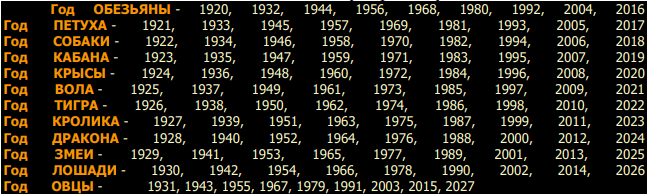
**30333-2 Калевич Сергей Михайлович Лабораторная работа №2 вариант 4**

**ЗАДАНИЯ 1, 2. if, switch (схема)**



Пользователь вводит год (число).

Проанализировать число и вывести результат:

* Это прошлое, настоящее (текущий год) или будущее;
* Это четный или нечетный год;
* Високосный или нет;
* Вывести знак этого года по японскому гороскопу. 

**Ход выполнения**

lab2\_tasks1-2.cpp

// Lab #2

// task #1, #2

#include <iostream>

#include <Windows.h>

#include <cmath>

using namespace std;

void periodOfYear(int y1)

{

const int current\_year = 2023;

if (y1 == current\_year)

cout << "current" << endl;

else if (y1 < current\_year)

cout << "past" << endl;

else

cout << "future" << endl;

}

void evenYear(int y2)

{

// a ? b : c

cout << ((y2 % 2 == 0) ? "even" : "odd") << endl;

}

void leapYear(int y3)

{

if ((y3 % 400 == 0) || ((y3 % 4 == 0) && !(y3 % 100 == 0)))

cout << " leap " << endl;

else

cout << " not leap " << endl;

}

void horoscope(int y4)

{

int x = y4 % 12;

switch (x)

{

case 0: cout << " Monkey " << endl; break;

case 1: cout << " Rooster " << endl; break;

case 2: cout << " Dog " << endl; break;

case 3: cout << " Boar " << endl; break;

case 4: cout << " Rat " << endl; break;

case 5: cout << " Ox " << endl; break;

case 6: cout << " Tiger " << endl; break;

case 7: cout << " Rabits " << endl; break;

case 8: cout << " Dragon " << endl; break;

case 9: cout << " Snake " << endl; break;

case 10: cout << " Horse " << endl; break;

case 11: cout << " Sheep " << endl; break;

default: cout << " unknown year " << endl;

}

}

void test()

{

int array[] = {1932, 1921, 2018, 1972, 2021, 1926,

1947, 1963, 2000, 2025, 1942, 2027};

const int length = sizeof(array) / sizeof(array[0]);

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

{

cout << array[i] << ' ';

horoscope(array[i]);

}

}

void main()

{

cout << "Lab 2 is start!" << endl;

test();

int year = 0;

char q = 0;

while (q != 'y')

{

cout << "Input year ";

cin >> year;

cout << "This year is "; periodOfYear(year);

cout << "This year is "; evenYear(year);

cout << "This year is "; leapYear(year);

cout << "This year is "; horoscope(year);

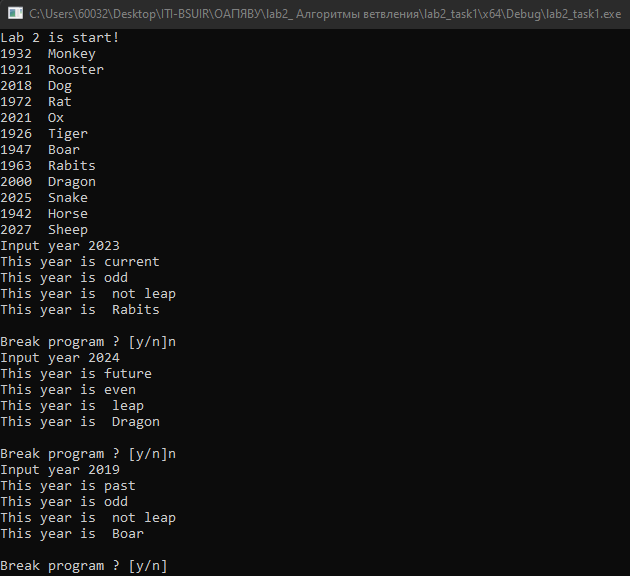
cout << endl << "Break program ? [y/n]";

cin >> q;

}

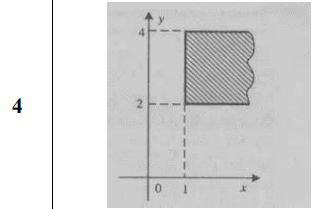
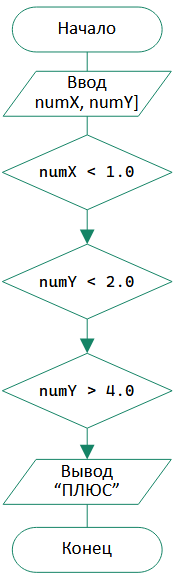
}

**Результат:**



**ЗАДАНИЕ 3. Тернарная операция**

Пользователь вводит 2 вещественных числа. Вывести слово "ПЛЮС", если точка с такими координатами лежит в указанной области (неровный край области означает, что область продолжается в бесконечность):

**Ход выполнения**

lab2\_task3\_var4.cpp

// Lab #2

// Task #3

// Variant #4

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <Windows.h>

using namespace std;

const double xEdge{ 1.0 };

const double yBottom{ 2.0 };

const double yTop{ 4.0 };

double numX{};

double numY{};

void test()

{

cout << "Ожидаются плюсы" << endl;

double array[][2] =

{

{1.0, 2.0},

{1.0, 4.0},

{2.0, 2.0},

{1.0, 3.0},

{2.7, 3.1},

{100.2, 3.5},

{0.0, 0.0},

{0.9, 1.3},

{1.2, 1.2},

{1.2, 4.5}

};

double x{};

double y{};

int length = sizeof(array) / sizeof(array[0]);

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

{

x = array[i][0];

y = array[i][1];

cout << x << ' ' << y << ' ';

x < xEdge ? printf("МИНУС X\n")

: y < yBottom ? printf("МИНУС yBottom\n")

: y > yTop ? printf("МИНУС yTop\n") : printf("ПЛЮС\n");

if (i == 5) cout << "\nОжидаются минусы" << endl;

}

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

test();

cout << "\nВведите два числа" << endl;

cin >> numX >> numY;

numX < xEdge ? printf("")

: numY < yBottom ? printf("")

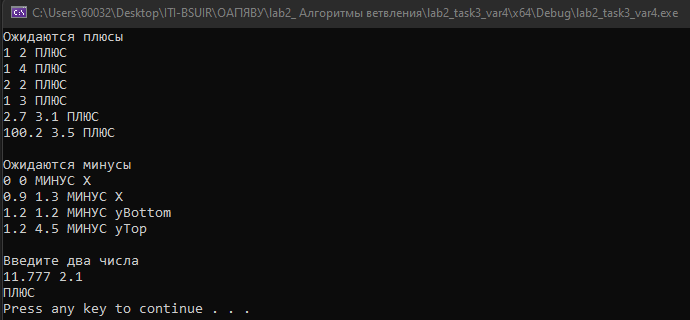
: numY > yTop ? printf("") : printf("ПЛЮС\n");

system("pause");

return 0;

}

**Результат:**

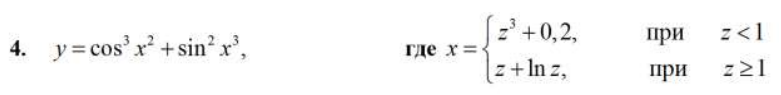


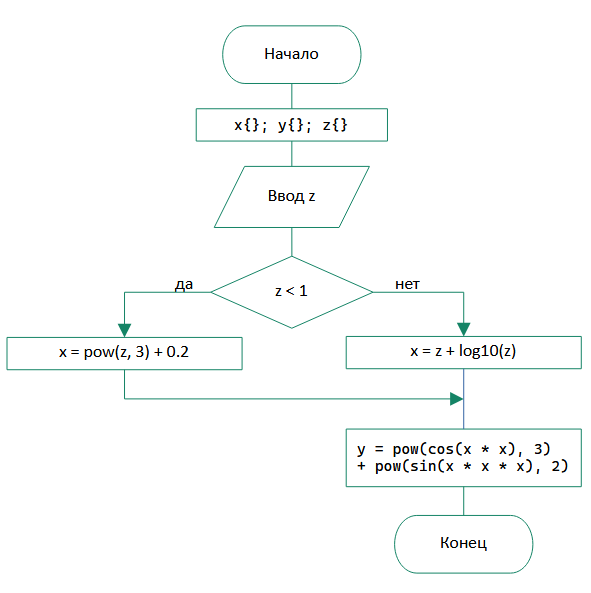
**ЗАДАНИЕ 4: Использовать оператор IF**

Составить программу для определения значения функции.

Обязательно предусмотреть вывод сообщения о том, по какой ветви происходило вычисление значения аргумента функции х.

Самостоятельно определить исключительные ситуации и предусмотреть диагностические сообщения, если нормальное завершение программы невозможно (например, деление на ноль).





**Ход выполнения**

lab2\_task4\_var4.cpp

// Lab #2

// Task #4

// Variant #4

#define \_USE\_MATH\_DEFINES

#include <iostream>

#include <Windows.h>

#include <cmath>

using namespace std;

double zLess1(double z)

{

cout << "\nВычисление при z меньше единицы" << endl;

return pow(z, 3) + 0.2;

}

double zEqualGreater1(double z)

{

cout << "\nВычисление при z равной или больше единицы" << endl;

return z + log10(z);

}

double yFormula(double z)

{

double x{};

double y{};

if (z < 1) x = zLess1(z);

else x = zEqualGreater1(z);

y = pow(cos(x \* x), 3) + pow(sin(x \* x \* x), 2);

cout << "при z = " << z << " y = " << y << endl;

return y;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

double z{};

yFormula(0.99);

yFormula(0.0);

yFormula(-0.52);

yFormula(1.0);

yFormula(1.57);

yFormula(100.35);

cout << "\nВведите число " << endl;

cin >> z;

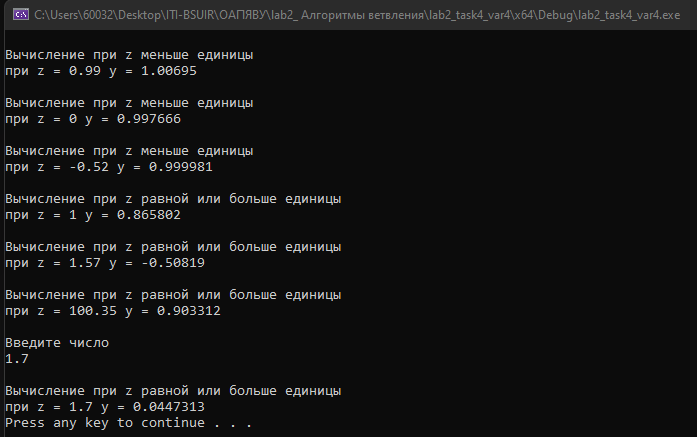
yFormula(z);

system("pause");

return 0;

}

**Результат:**



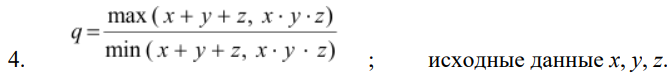
**ЗАДАНИЕ 5: ВЕТВЯЩИЙСЯ АЛГОРИТМ**

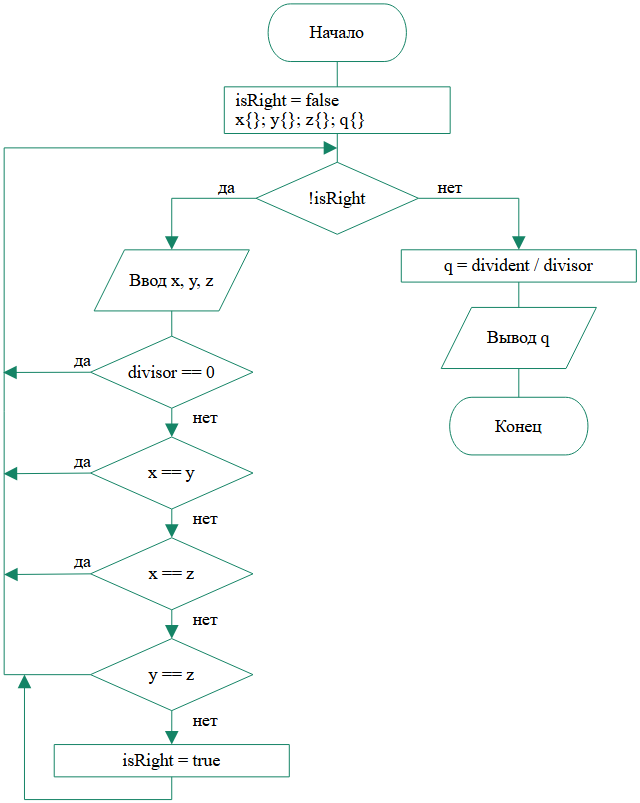
Составить программу для определения значения функции.

Считать, что при вычислении функции нахождение минимального и максимального элементов производится только среди различных значений.

В противном случае предусмотреть повторный ввод исходных данных.

Самостоятельно определить исключительные ситуации и предусмотреть диагностические сообщения, если нормальное завершение программы невозможно (например, деление на ноль).





**Ход выполнения**

lab2\_task5\_var4.cpp

// Lab #2

// Task #5

// Variant #4

#include <iostream>

#include <Windows.h>

#include <cmath>

using namespace std;

double divident(double x, double y, double z)

{

return fmax(x + y + z, x \* y \* z);

}

double divisor(double x, double y, double z)

{

return fmin(x + y + z, x \* y \* z);

}

double qFormula()

{

bool isFirstIteration = true;

bool isRight = false;

double x{};

double y{};

double z{};

double q{};

cout << "Введите три разных числа" << endl;

while (!isRight)

{

if (isFirstIteration) isFirstIteration = false;

else cout << "Введите другие числа" << endl;

cin >> x >> y >> z;

divisor(x, y, z) == 0 ? printf("Делитель равняется нулю\n")

: isRight = true;

if (!isRight) continue;

x == y ? printf("Числа должны быть разными\n", isRight = false)

: x == z ? printf("Числа должны быть разными\n", isRight = false)

: y == z ? printf("Числа должны быть разными\n", isRight = false)

: isRight = true;

}

q = divident(x, y, z) / divisor(x, y, z);

cout << "q = " << q << endl;

return q;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

qFormula();

system("pause");

return 0;

}

Выводы

Изучены и применены на практике условные операторы, циклы, тернарный оператор, оператор switch. На их основе научились программировать ветвящиеся алгоритмы.