Балтийский государственный технический университет  
«ВОЕНМЕХ» им. Д. Ф. Устинова

Кафедра O7 «Информационные системы и программная инженерия»

**Практическая работа №2**по дисциплине «Информатика: Основы программирования»  
на тему «Ветвления и циклы»  
  
Вариант 17

Выполнил:  
Студент Усов Д.А.  
Группа Е123Б  
  
Преподаватель:  
Лестенко Н.А.

Санкт-Петербург  
2022 г.

1. Вычислить значение функции , используя условную операцию ?:

*Исходные данные:*

Аргументы функции m и n. Так как значения m и n могут быть любыми, объявим соответствующие переменные типа double.

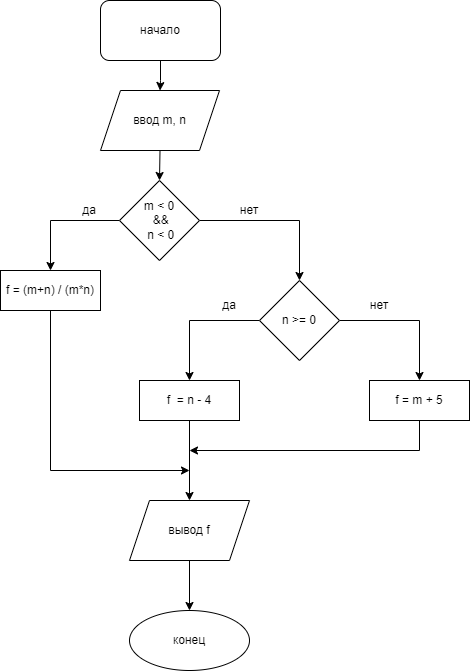
*Результирующие данные:*

Значение функции f , соответствующая переменная тоже будет типа double.

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| m = -1, n = -2 | -1.5 | -1.500000 |
| m = 5, n = 4 | 0 | 0.000000 |
| m = 4, n = -3 | 9 | 9.000000 |
| M = 0, n = -6 | 5 | 5.000000 |

Схема программы



Текст программы

#include <stdio.h>  
  
int main(){  
 double m,n,f; /\* объявление переменных \*/  
 printf("m = ");  
 scanf("%lf",&m); /\* ввод аргументов функции m,n \*/  
 printf("n = ");  
 scanf("%lf",&n);  
 f = m < 0 && n < 0 ? (m+n)/(m\*n) : n >= 0 ? n- 4 : m + 5;  
 /\* вложенное условие: при ложности условия первого усл. оператора,  
 программа переходит во второй усл. оператор\*/  
 printf("f = %lf",f); /\* вывод результата \*/  
 return 0;  
}

1. Вычислить значение функции

*Исходные данные:*

Аргументы x и y — действительные числа, тип double*.*

*Результирующие данные:*

A – результат функции, тип double.

*Предварительные вычисления:*

Чтобы можно было вычислить значение функции, должны быть выполнены следующие условия: знаменатель не должен быть равен 0, y ≠ 0.

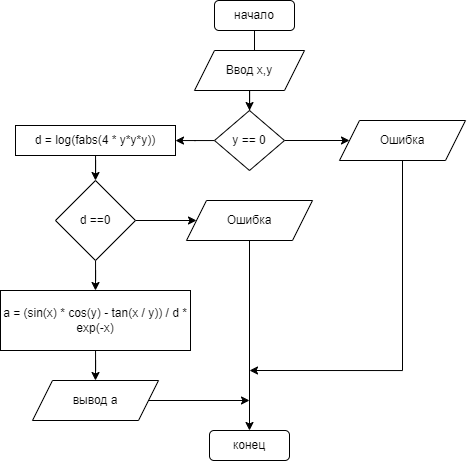
*Вспомогательные переменные:*

d – знаменатель дроби, переменная типа double.

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| x = 1, y = 2 | -0.095158958 | -0.095158958 |
| x = 5, y = 0 | Функция не определена | Функция не определена |
| x = -6, y = -8 | -51.443123889 | -51.443123889 |
| x = 3, y = -90 | -0.000099959 | -0.000099959 |

Схема программы:



нет

да

нет

да

Текст программы

#include <stdio.h>  
#include <math.h>  
  
int main() {  
 double x, y, a, d; // объявление переменных  
 printf("x = ");  
 scanf("%lf", &x);  
 printf("y = ");  
 scanf("%lf", &y); // ввод аргументов функции  
 if (y == 0) { // проверка на соответствие ООФ  
 printf("Функция не определена"); // вывод сообщения об ошибке  
 } else {  
 d = log(fabs(4 \* y\*y\*y)); // знаменатель  
 if (d == 0) { // проверка на соответствие ООФ  
 printf("Функция не определена"); // вывод сообщения об ошибке  
 } else {  
 a = (sin(x) \* cos(y) - tan(x / y)) / d \* exp(-x);  
  
 printf("a = %.9lf", a); // вычисление и вывод результата  
 }  
 }  
 return 0;  
}

1. Определить, лежат ли три точки с координатами (x1, y1), (x2, y2), (x3, y3) на одной прямой. Если не лежат, вычислить расстояния до каждой точки из начала координат. Если лежат, получить общее уравнение этой прямой вида Ax+By+C = 0, где A ≥ 0.

*Исходные данные:*

x\_1, y\_1 – координаты первой точки, тип double;

x\_2, y\_2 – координаты первой точки, тип double;

x\_3, y\_3 – координаты первой точки, тип double;

*Результирующие данные:*

Ответ на вопрос (“Yes”/”No”);

d\_1, d\_2, d\_3 – расстояния до каждой точки из начала координат, тип double;

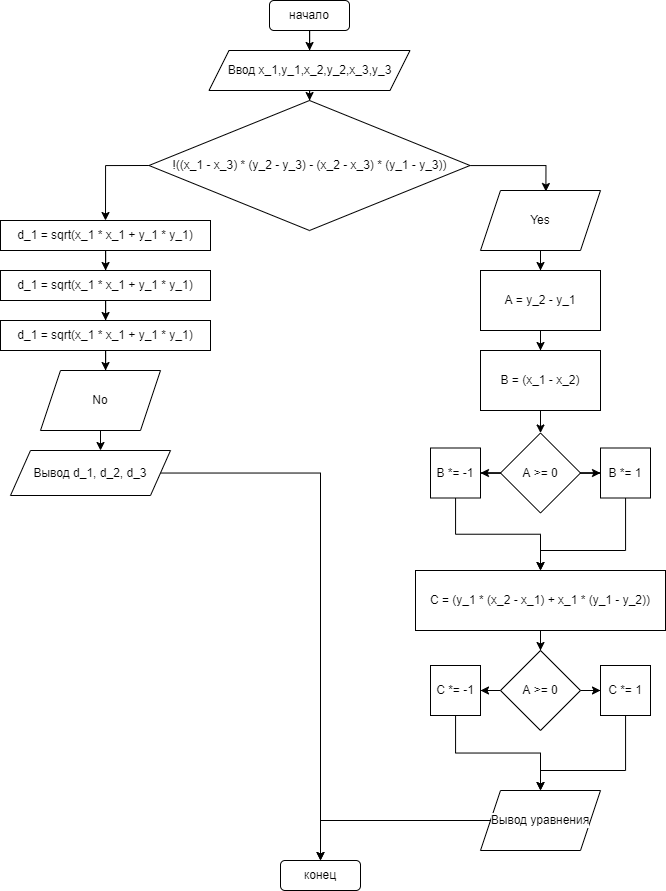
уравнение этой прямой вида Ax+By+C = 0

*Вспомогательные переменные:*

Переменные A, B, С – служат для значений коэффициентов A, B, C, тип - double;

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| *Enter x\_1 y\_1 x\_2 y\_2 x\_3 y\_3: -2 2 2 -2 1 -1* | Yes  4x + 4y + 0 = 0 | Yes  4x + 4y + 0 = 0 |
| Enter x\_1 y\_1 x\_2 y\_2 x\_3 y\_3:-2 -2 2 -2 1 1 | No  Distance for p\_1: 2.82843, p\_2: 2.82843, p\_3: 1.41421 | No  Distance for p\_1: 2.82843, p\_2: 2.82843, p\_3: 1.41421 |
| Enter x\_1 y\_1 x\_2 y\_2 x\_3 y\_3:-6 -3 8 1 4.5 0 | Yes  4x - 14y - 18 = 0 | Yes  4x - 14y - 18 = 0 |
| Enter x\_1 y\_1 x\_2 y\_2 x\_3 y\_3:-6 -2.5 8 0.5 4.5 -0.5 | No  Distance for p\_1: 6.5, p\_2: 8.01561, p\_3: 4.52769 | No  Distance for p\_1: 6.5, p\_2: 8.01561, p\_3: 4.52769 |

Схема программы:

нет

да

да

нет

нет

да

Текст программы

#include <stdio.h>  
#include <math.h>  
  
int main() {  
 double x\_1, x\_2, x\_3, y\_1, y\_2, y\_3, A, B, C, d\_1, d\_2, d\_3;  
 printf("Enter x\_1 y\_1 x\_2 y\_2 x\_3 y\_3:");  
 scanf("%lf %lf %lf %lf %lf %lf", &x\_1, &y\_1, &x\_2, &y\_2, &x\_3, &y\_3);  
 if (!((x\_1 - x\_3) \* (y\_2 - y\_3) - (x\_2 - x\_3) \* (y\_1 - y\_3))) { // Проверка на то лежат ли точки на одной прямой  
 printf("Yes\n"); // Ответ на вопрос  
 A = y\_2 - y\_1; // Вычисление коэффициента A  
 B = (x\_1 - x\_2) \* (A >= 0 ? 1 : -1); // Вычисление коэффициента B  
 C = (y\_1 \* (x\_2 - x\_1) + x\_1 \* (y\_1 - y\_2))\* (A >= 0 ? 1 : -1); // Вычисление коэффициента C  
 printf("%gx %c %gy %c %g = 0", fabs(A), B < 0 ? '-' : '+', fabs(B), C < 0 ? '-' : '+', fabs(C)); // Вывод уравнения  
 } else {  
 d\_1 = sqrt(x\_1 \* x\_1 + y\_1 \* y\_1); // Вычисление расстояния до начала координат от 1 точки  
 d\_2 = sqrt(x\_2 \* x\_2 + y\_2 \* y\_2); // Вычисление расстояния до начала координат от 2 точки  
 d\_3 = sqrt(x\_3 \* x\_3 + y\_3 \* y\_3); // Вычисление расстояния до начала координат от 3 точки  
 printf("No\n"); // Ответ на вопрос  
 printf("Distance for p\_1: %g, p\_2: %g, p\_3: %g", d\_1, d\_2, d\_3); // Вывод расстояний  
 }  
 return 0;  
}

1. Получить таблицу температур по Цельсию от 0° до 100° и их эквивалентов по шкале Фаренгейта, используя для перевода формулу . Использовать управляющую инструкцию *for*.

*Исходные данные:*

Счетчик, обозначим t\_c, типа char будет достаточно.

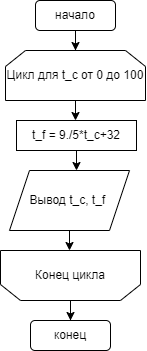
*Результирующие данные:*

t\_f – температура в фаренгейтах, тип double

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
|  | 0 32  1 33.8  2 35.6  3 37.4  4 39.2  5 41  6 42.8  7 44.6  8 46.4  9 48.2  10 50  11 51.8  12 53.6  13 55.4  14 57.2  15 59  16 60.8  17 62.6  18 64.4  19 66.2  20 68  21 69.8  22 71.6  23 73.4  24 75.2  25 77  26 78.8  27 80.6  28 82.4  29 84.2  30 86  31 87.8  32 89.6  33 91.4  34 93.2  35 95  36 96.8  37 98.6  38 100.4  39 102.2  40 104  41 105.8  42 107.6  43 109.4  44 111.2  45 113  46 114.8  47 116.6  48 118.4  49 120.2  50 122  51 123.8  52 125.6  53 127.4  54 129.2  55 131  56 132.8  57 134.6  58 136.4  59 138.2  60 140  61 141.8  62 143.6  63 145.4  64 147.2  65 149  66 150.8  67 152.6  68 154.4  69 156.2  70 158  71 159.8  72 161.6  73 163.4  74 165.2  75 167  76 168.8  77 170.6  78 172.4  79 174.2  80 176  81 177.8  82 179.6  83 181.4  84 183.2  85 185  86 186.8  87 188.6  88 190.4  89 192.2  90 194  91 195.8  92 197.6  93 199.4  94 201.2  95 203  96 204.8  97 206.6  98 208.4  99 210.2  100 212 | 0 32  1 33.8  2 35.6  3 37.4  4 39.2  5 41  6 42.8  7 44.6  8 46.4  9 48.2  10 50  11 51.8  12 53.6  13 55.4  14 57.2  15 59  16 60.8  17 62.6  18 64.4  19 66.2  20 68  21 69.8  22 71.6  23 73.4  24 75.2  25 77  26 78.8  27 80.6  28 82.4  29 84.2  30 86  31 87.8  32 89.6  33 91.4  34 93.2  35 95  36 96.8  37 98.6  38 100.4  39 102.2  40 104  41 105.8  42 107.6  43 109.4  44 111.2  45 113  46 114.8  47 116.6  48 118.4  49 120.2  50 122  51 123.8  52 125.6  53 127.4  54 129.2  55 131  56 132.8  57 134.6  58 136.4  59 138.2  60 140  61 141.8  62 143.6  63 145.4  64 147.2  65 149  66 150.8  67 152.6  68 154.4  69 156.2  70 158  71 159.8  72 161.6  73 163.4  74 165.2  75 167  76 168.8  77 170.6  78 172.4  79 174.2  80 176  81 177.8  82 179.6  83 181.4  84 183.2  85 185  86 186.8  87 188.6  88 190.4  89 192.2  90 194  91 195.8  92 197.6  93 199.4  94 201.2  95 203  96 204.8  97 206.6  98 208.4  99 210.2  100 212 |

Схема программы:



Текст программы

#include <stdio.h>  
  
int main(){  
 char t\_c; // Переменная счетчик  
 double t\_f; // Температура в фаренгейтах  
  
 for (t\_c = 0; t\_c<=100; t\_c++){  
 t\_f = 9./5\*t\_c+32; // Вычисление температуры  
 printf("%d\t%g\n", t\_c, t\_f); // Вывод температур  
 }  
  
 return 0;  
}

1. Вычислить сумму кубов всех четных чисел, лежащих в диапазоне [X, Y], где X и Y – вводимые с клавиатуры натуральные числа.

*Исходные данные:*

Переменные x, y, тип unsigned long long.

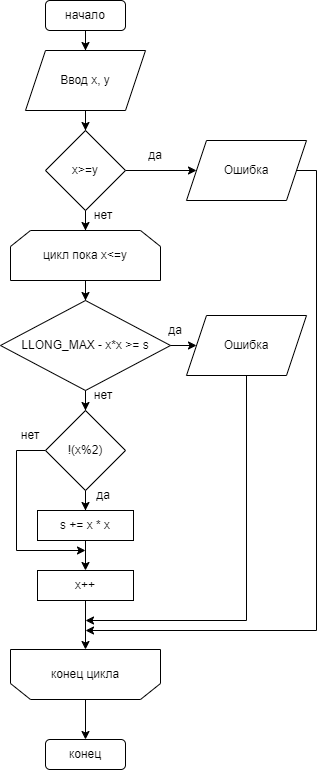
*Результирующие данные:*

Переменная s – сумма кубов всех четных чисел в диапазоне, тип unsigned long long или сообщение об ошибке.

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| *Enter x y:*  *1 5* | 20 | 20 |
| Enter x y:  5 2 | Ошибка | Ошибка |
| Enter x y:  555 5555555555 | Ошибка | Ошибка |

Схема программы:



Текст программы

#include <stdio.h>  
#include <limits.h>  
  
int main(){  
 unsigned long long x,y, s = 0;  
 printf("Enter x y:\n");  
 scanf("%lld %lld", &x,&y);  
 if (x>=y){ // Проверка на то что первое число меньше второго  
 printf("The first number must be less than the second");  
 return 1;  
 }  
  
 while(x<=y){  
 if (LLONG\_MAX - x\*x >= s) { // Проверка на переполнение  
 if (!(x%2)) { // Проверка на четность  
 s += x \* x;  
 }  
 } else{  
 printf("You have entered too large a range or too large numbers");  
 return 2;  
 }  
 x++;  
 }  
 printf("%lld", s);  
 return 0;  
}

1. С клавиатуры вводится число N. Определить, может ли оно быть двоичным (т.е. состоять только из 0 и 1).

*Исходные данные:*

Переменная c в которую будут сохраняться символы по одному, тип int

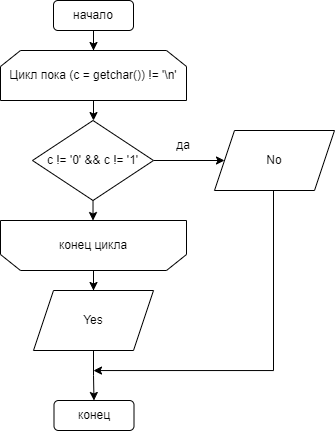
*Результирующие данные:*

Ответ на вопрос (“Yes”/”No”);

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| *84512* | No | No |
| 1000110 | Yes | Yes |

Схема программы:



Текст программы

#include <stdio.h>  
  
int main ()  
{  
 int c;  
 while ((c = getchar()) != '\n') // Получение символа из вводимой строки  
 if (c != '0' && c != '1'){ // Проверка на то является ли символ 0 или 1  
 printf("No");  
 return 1;  
 }  
 printf("Yes");  
 return 0;  
}

1. Вычислить значение суммы бесконечного ряда с заданной точностью ε=10-6 и значение функции (для проверки) . Использовать рекуррентные зависимости для вычисления значений слагаемых.

*Исходные данные:*

Аргумент функции. Тип аргумента в задании не указан, поэтому переменная х будет типа double.

*Результирующие данные:*

Значение суммы s тоже будет типа double.

*Вспомогательные переменные:*

n – индекс слагаемого – целое число типа int, a – значение текущего слагаемого – вещественное число типа double.

*Предварительные вычисления:*

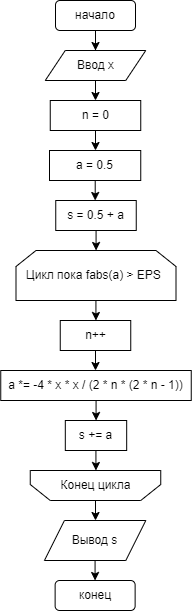
n-ное слагаемое ,

предшествующее слагаемое

коэффициент пропорциональности

индекс первого слагаемого в рекуррентной последовательности n=0, первое слагаемое

Схема программы:



Текст программы

#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
  
#define EPS 1e-6  
  
int main(void) {  
 double x, a, s;  
 int n = 0; /\*индекс первого слагаемого 0\*/  
 printf("x=");  
 scanf("%lf", &x);  
 a = 0.5;  
 s = 0.5 + a;  
/\*пока слагаемое по модулю больше заданной точности\*/  
 while (fabs(a) > EPS) {  
 n++; /\*увеличиваем индекс слагаемого\*/  
 a \*= -4 \* x \* x / (2 \* n \* (2 \* n - 1));  
/\*вычисляем очередное слагаемое домножением  
предыдущего на коэффициент пропорциональности\*/  
 s += a; /\*накапливаем сумму\*/  
 }  
 printf("s=%lf\n cos(%lf)^2=%lf\n",  
 s, x, cos(x) \* cos(x));  
 system("pause");  
 return 0;  
}

*Результаты тестирования*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый результат** | **Результат работы программы** |
| x = 0 |  | s = 1.000000 |
| x = 5 |  | s = 0.080464 |

1. Вычислить значение суммы бесконечного ряда с заданной точностью ε=10-6 и значение функции (для проверки) учесть, что . Использовать рекуррентные зависимости для вычисления значений слагаемых по частям.

*Исходные данные:*

Аргумент функции. Тип аргумента в задании не указан, поэтому переменная х будет типа double.

*Результирующие данные:*

Значение суммы s тоже будет типа double.

*Вспомогательные переменные:*

n – индекс слагаемого – целое число типа int, a – значение текущего слагаемого – вещественное число типа double.

*Предварительные вычисления:*

n-ое слагаемое

предшествующее слагаемое

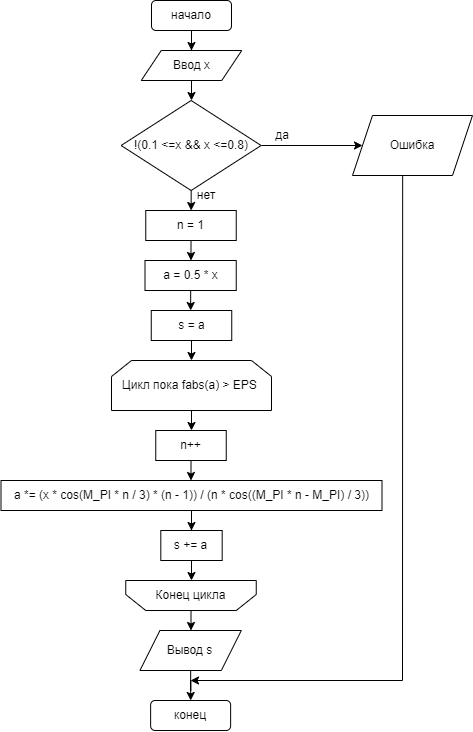
коэффициент пропорциональности s

индекс первого слагаемого в рекуррентной последовательности

n=1,

первое слагаемое

Схема программы:



Текст программы

#include <stdio.h>  
#include <math.h>  
  
#define EPS 1e-6  
  
int main(void) {  
 double x, a, s;  
 int n = 1; /\*индекс первого слагаемого 0\*/  
 printf("x=");  
 scanf("%lf", &x);  
 if (!(0.1 <=x && x <=0.8)) /\*если x не принадлежит отрезку [0.1;0.8]\*/  
 { /\*выводим сообщение об этом\*/  
 printf("Invalid value x\n");  
  
 return 1; /\*и прерываем программу\*/  
 }  
 a = 0.5 \* x;  
 s = a;  
 while (fabs(a) > EPS) {/\*пока слагаемое по модулю больше заданной точности\*/  
 n++; /\*увеличиваем индекс слагаемого\*/  
 a \*= (x \* cos(M\_PI \* n / 3) \* (n - 1)) / (n \* cos((M\_PI \* n - M\_PI) / 3));  
 /\*вычисляем очередное слагаемое домножением  
 предыдущего на коэффициент пропорциональности\*/  
 s += a; /\*накапливаем сумму\*/  
 }  
 printf("s=%.6lf\n y(%lf)=%.6lf\n",  
 s, x, -0.5 \* log(1 - 2 \* x \* cos(M\_PI / 3) + x\*x));  
  
 return 0;  
}

*Результаты тестирования*

|  |  |  |
| --- | --- | --- |
| **Исходные данные** | **Ожидаемый результат** | **Результат работы программы** |
| x=1 | Ошибка | Ошибка |
| x=0.5 |  | s=0.143841 |
| x=0.3 |  | s=0.117861 |

1. Дано целое число а. Найти все такие простые числа p, чтобы дробь была сократима на p. Простым называется натуральное число N, не имеющее других делителей, кроме 1 и самого N.

*Исходные данные:*

Аргумент функции число a, тип long.

*Результирующие данные:*

p – найденное простое число, тип long.

*Вспомогательные переменные:*

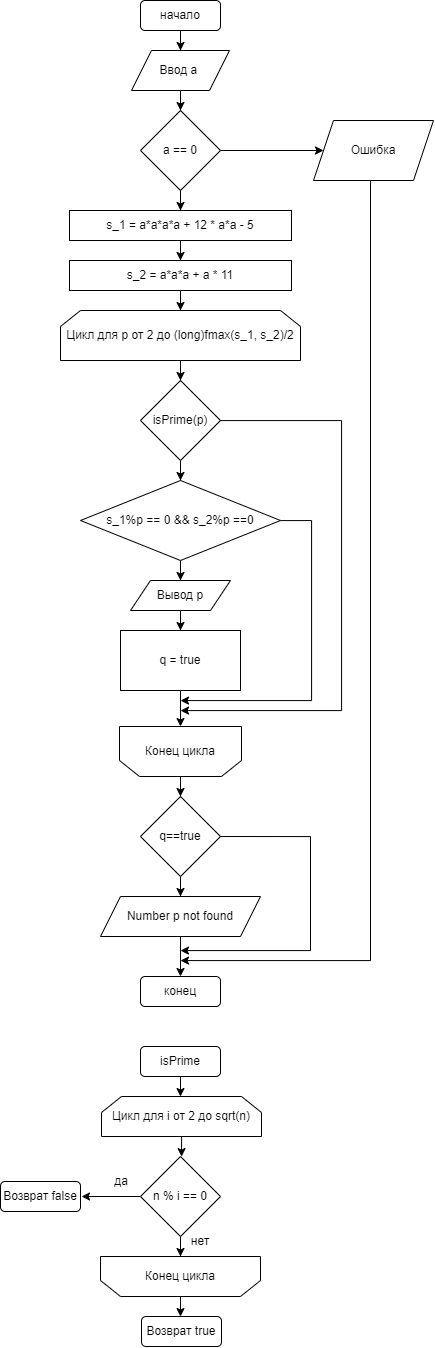
s\_1, s\_2 – числитель и знаменатель, тип long;

q – определяет нашли ли мы хоть одно простое число, тип bool

Таблица тестирования:

|  |  |  |
| --- | --- | --- |
| **Входные данные** | **Ожидаемый результат** | **Результат работы программы** |
| Enter a:1 | 2 | 2 |
| Enter a:0 | Ошибка | Ошибка |
| Enter a:-5 | 2  5 | 2  5 |

Схема программы:



Текст программы

#include <stdio.h>  
#include <stdlib.h>  
#include <stdbool.h>  
#include <math.h>  
  
bool isPrime(long);  
  
int main() {  
 long s\_1, s\_2;  
 long p, a;  
 bool q;  
  
 printf("Enter a:");  
 scanf("%ld", &a);  
 if (a == 0) {  
 printf("The number a cannot be equal to 0\n"); // Иначе знаменатель будет равен 0  
 return 1;  
 }  
  
 s\_1 = a\*a\*a\*a + 12 \* a\*a - 5; // Вычисляем числитель  
 s\_2 = a\*a\*a + a \* 11; // Вычисляем знаменатель  
 printf("%ld/%ld\n", s\_1, s\_2);  
 for (p =2; p <= (long)fmax(s\_1, s\_2)/2; p++){  
 if (isPrime(p)){ // Проверка числа на простое число  
 if (s\_1%p == 0 && s\_2%p ==0){  
 printf("%ld\n", p); // Выводим p  
 q = true; // Если мы нашли хотя бы одно значение p  
 }  
 }  
 }  
 if (q) return 0;  
 printf("Number p not found\n");  
 return 2;  
  
}  
  
bool isPrime(long n) {  
 long i;  
 for (i = 2; i <= sqrt(n); i++) {  
 if (n % i == 0) {  
 // n не простое, так как делится на i  
 return false;  
 }  
 }  
//n простое.  
 return true;  
}