

Министерство образования и науки РФ  
ФГБОУ ВО ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
Кафедра «Информационная безопасность систем и технологий»

ОТЧЁТ  
о лабораторной работе №2  
Использование ветвлений и циклов в программах C++

Дисциплина: Языки программирования

Группа: 18ПИ1

Выполнил: Асаян А.В.

Количество баллов:

Дата сдачи:

Проверил: к.т.н., доцент Лупанов М.Ю.

Пенза, 2019

## 1 Цель работы

1.1 Освоить реализацию ветвлений и циклов с помощью операторов языка C++.

## 2 Задания к практической работе

2.1 Составить алгоритм вычисления среднего арифметического последовательности чисел с плавающей точкой. Реализовать алгоритм в виде программы на языке C++. Формат вводимых данных: целое число N (длина последовательности), числа с плавающей точкой (N чисел). Формат выводимых данных: значение среднего арифметического.

2.2 Составить алгоритм поиска заданного числа в последовательности чисел. Поиск прекращать, когда в последовательности встретится число 0. Искомое число не должно быть нулем. Реализовать алгоритм в виде программы на языке C++. Формат вводимых данных: искомое целое число, последовательность целых чисел. Формат выводимых данных: слово «найдено» и значение искомого числа или фраза «не найдено».

2.3 Составить алгоритм вычисления суммы положительных членов целочисленной последовательности. Суммирование прекращать, когда в последовательности встретится число 0. Реализовать алгоритм в виде программы на языке C++. Формат вводимых данных: последовательность целых чисел. Формат выводимых данных: целочисленное значение вычисленной суммы.

2.4 Составить алгоритм, определяющий является ли число простым. Реализовать алгоритм в виде программы на языке C++. Формат вводимых данных: одно целое число. Формат выводимых данных: слово «Да» или «Нет».

2.5 Задание повышенной сложности. Составить алгоритм, ищущий в последовательности простые числа. Поиск прекращать, когда в последовательности встретится число 0. Формат вводимых данных: последовательность целых чисел. Формат выводимых данных: слово «простое» при обнаружении каждого простого числа.

### 3 Результат выполнения работы

3.1 Программа получает целое число-длину последовательности. Запускается цикл для ввода чисел последовательности с плавающей точкой и поиска их суммы. Цикл работает до тех пор, пока не будет посчитана сумма всех элементов последовательности. После завершения цикла найденная сумма делится на длину последовательности, результат деления-это среднеарифметическое. На рисунке 1 изображена блок-схема для данного алгоритма.



Рисунок 1 — Среднеарифметическое последовательности чисел.

Код программы на языке C++:

```
#include <stdio.h>
#include <iostream>
```

```

#include <cmath>
using namespace std;
int main(int argc, char **argv)
{
    int n,i=0;
    double x,sr,s=0;
    cout<< "Длина последовательности"<<endl;
    cin >> n;
    n=abs(n);
    while( i < n) {
        cout << "x равен"<<endl;
        cin >> x;
        s=s+x;
        i++;
    }
    sr=s/n;
    cout<<"Среднеарифметическое " <<n<<"
чисел\n"<<sr<<endl;
    return 0;
}

```

Согласно заданию, длина последовательности — это некоторое целое число  $N$ , так как отрицательные числа принадлежат множеству целых, пользователь может ввести отрицательное целое число в качестве длины последовательности, поэтому программа за длину последовательности принимает модуль числа  $N$ . Результат работы программы для последовательности из 5 чисел (1.25, 12.6, 4.7, 5.0, 2.3) представлен на рисунке 2. На рисунке 3 представлена проверка результата работы программы.

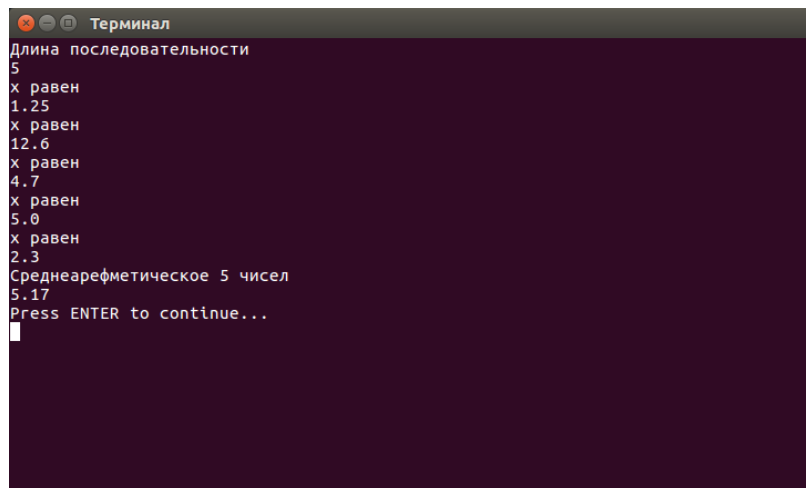


Рисунок 2 — Результат поиска среднего арифметического.

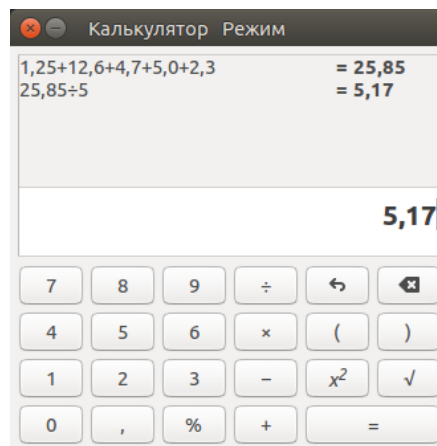


Рисунок 3 — Проверка результата работы первой программы.

3.2 Программа получает искомое число, оно проверяется на неравенство нулю. Если проверка не пройдена и введённое число равно 0, на экран выводится сообщение о том, что число не должно быть равно нулю и программа завершается. Если проверка пройдена, то запускается цикл для ввода чисел последовательности и поиска искомого. Введённое число последовательности проверяется на равенство нулю, если оно равно нулю, то выводится сообщение «Не найдено» и программа завершается. Если число последовательности не равно нулю, то оно сравнивается с искомым. Если число равно искомому, то выводится сообщение « Найдено \*искомое число\*» и цикл поиска завершается, так как число найдено. После чего программа завершается. Если число не равно искомому, то цикл продолжает работать до

того момента, пока пользователь не встретит ноль или искомое число. На рисунке 4 изображена блок-схема для данного алгоритма.

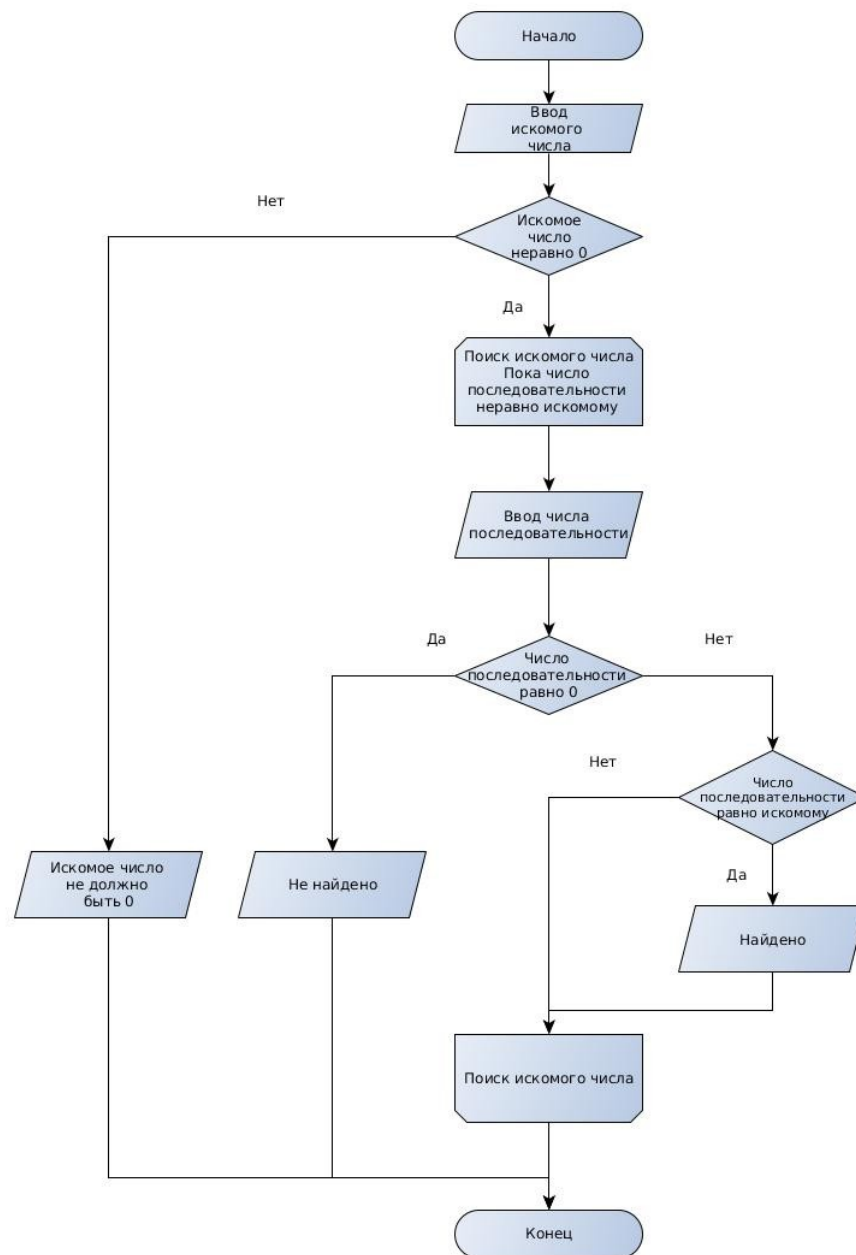


Рисунок 4 — Искомое число.

Код программы на языке C++:

```
#include <stdio.h>
#include <iostream>
using namespace std;
int main(int argc, char **argv)
{
```

```

        int n,x;

        cout<< "Введите искомое число
последдвательности"<<endl;
        cin >> n;
        if(n!=0) {
            while(x!=n) {
                cout<< "Введите число
последовательности"<<endl;
                cin>>x;
                if (x==0) {
                    cout<<"Не найдено"<< endl;
                    break;
                }
                if (x==n)
                    cout<<"Найдено "<< x<< endl;
            }

        }
        else
            cout<< "Число не должно быть равно нулю"<<endl;
        return 0;
    }

```

На рисунках 5,6,7 представлены 3 результата работы программы.

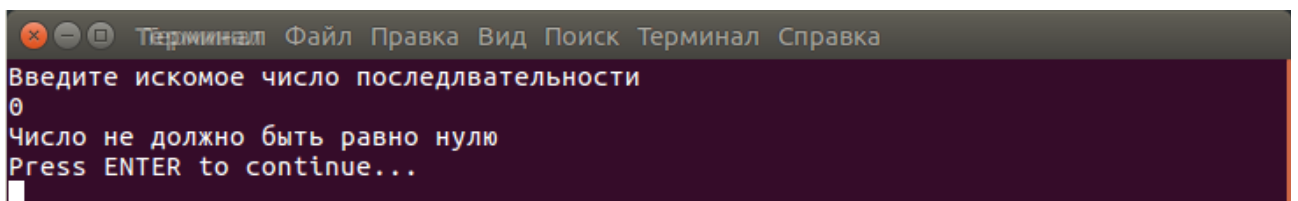


Рисунок 5 — Результат работы второй программы, если искомое число равно 0.

```
Терминал Файл Правка Вид Поиск Терминал Справка
Введите искомое число последлвательности
3
Введите число последовательности
1
Введите число последовательности
2
Введите число последовательности
0
Не найдено
Press ENTER to continue...
█
```

Рисунок 6 — Результат работы второй программы при встрече 0 в последовательности.

```
Терминал Файл Правка Вид Поиск Терминал Справка
Введите искомое число последлвательности
-8
Введите число последовательности
4
Введите число последовательности
5
Введите число последовательности
6
Введите число последовательности
1
Введите число последовательности
9
Введите число последовательности
-3
Введите число последовательности
-8
Найдено -8
Press ENTER to continue...
█
```

Рисунок 7 — Результат работы второй программы при нахождении искомого числа.

3.3 Программа начинается с цикла, который работает пока в последовательности чисел не встретится ноль. В цикле вводятся числа последовательности и проверяются на соответствие условию. Если число последовательности больше нуля, то сумма положительных членов последовательности увеличивается на это число и цикл продолжает работу.



Если число меньше нуля, то цикл продолжает свою работу, но сумма не увеличивается. Когда цикл завершается, выводится найденная сумма и программа завершается. На рисунке 8 представлена блок-схема для данного алгоритма.

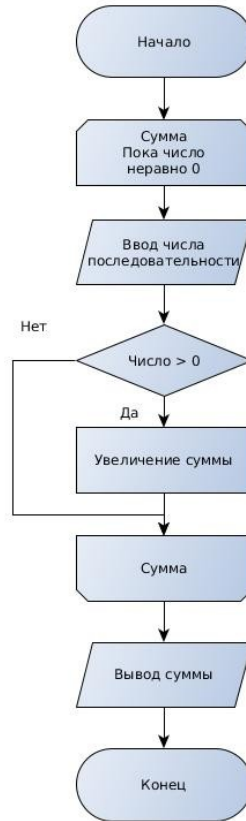


Рисунок 8 — Сумма положительных членов последовательности.

Код программы на языке C++:

```

#include <stdio.h>
#include <iostream>
using namespace std;
int main(int argc, char **argv)
{
    int x=1,s=0;
    while(x!=0) {
        cout<< "Введите число
последовательности"<<endl;
        cin>>x;
    }
}
  
```

```

        if (x>0)
            s=s+x;
    }
    cout<<s<<endl;

    return 0;
}

```

Результат работы программы для последовательности чисел 8, 5, 4, 3, 7, -8, -9, 5, 0 представлен на рисунке 9.

```

Терминал  Файл  Правка  Вид  Поиск  Терминал  Справка
Введите число последовательности
8
Введите число последовательности
5
Введите число последовательности
4
Введите число последовательности
3
Введите число последовательности
7
Введите число последовательности
-8
Введите число последовательности
-9
Введите число последовательности
5
Введите число последовательности
0
32
Press ENTER to continue...

```

Рисунок 9 — Результат работы третьей программы.

На рисунке 10 представлена проверка результатов работы программы в калькуляторе.

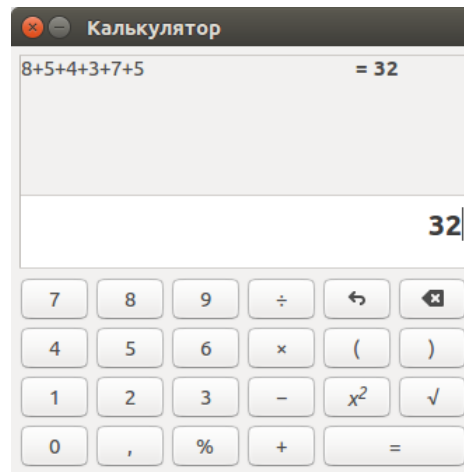


Рисунок 10 — Проверка результатов работы третьей программы.

3.4 Программа получает целое число, затем открывается цикл, который работает, пока делитель меньше или равен проверяемому числу. В цикле остаток от деления проверяемого числа на делитель проверяется, если он равен 0, то количество делителей увеличивается на один и делитель увеличивается на один. Если остаток от деления не равен 0, то делитель увеличивается на один. После завершения цикла, проверяется количество делителей, если оно равно 2, то выводится сообщение «Да», если не равно 2, то выводится «Нет», программа завершается. На рисунке 11 изображена блок-схема для данного алгоритма.

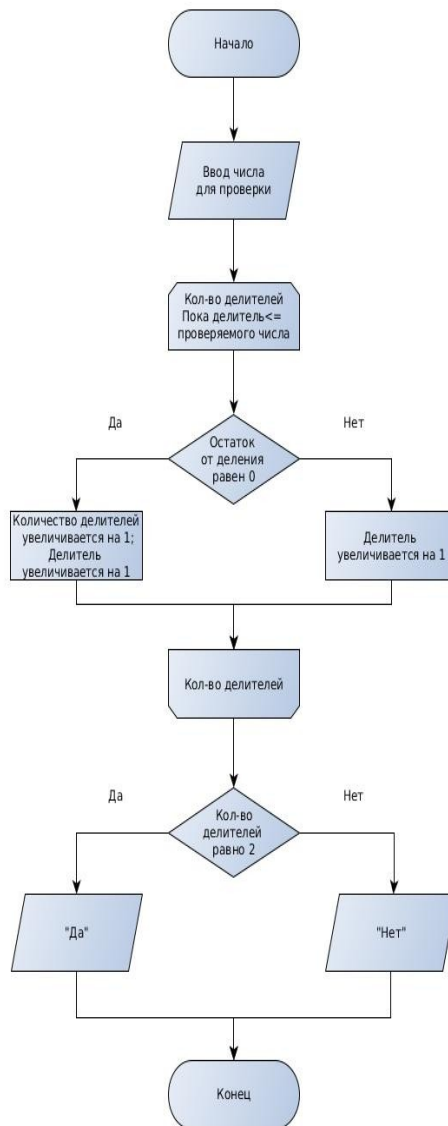


Рисунок 11 — Проверка простого числа.

Код программы на языке C++:

```

#include <stdio.h>
#include <iostream>
using namespace std;
int main()
{
    int i=1,a=0;
    int x;
    cout<<"Введите число для проверки "<< endl;

```

```

cin>>x;
while (i<=x) {
    if(x%i==0) {
        a=a+1;
        i++;
    } else {
        i++;
    }
}

if(a==2)
    cout<<"Да"<<endl;
else
    cout<<"Нет"<<endl;
return 0;
}

```

Результаты работы программы представлены на рисунках 12 и 13.

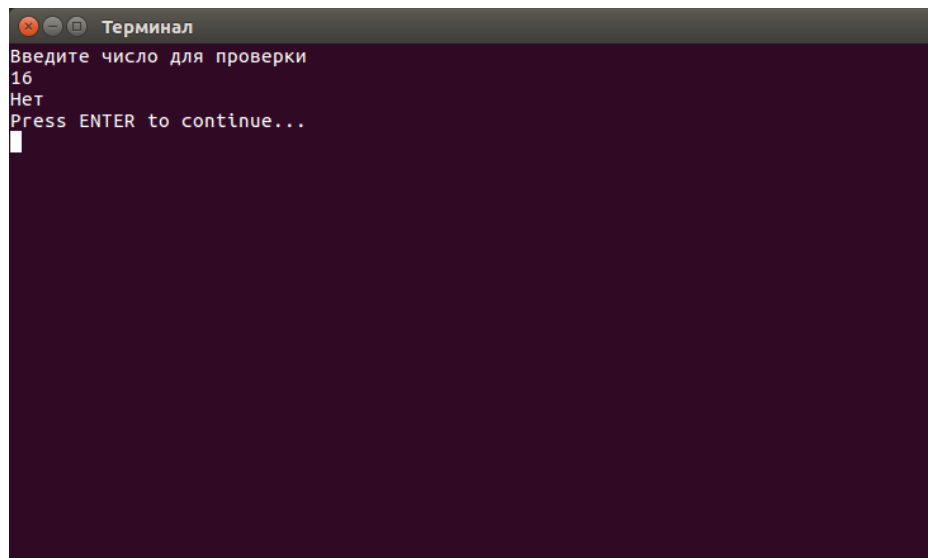


Рисунок 12 — Результат работы третьей программы, если проверяемое число не простое.

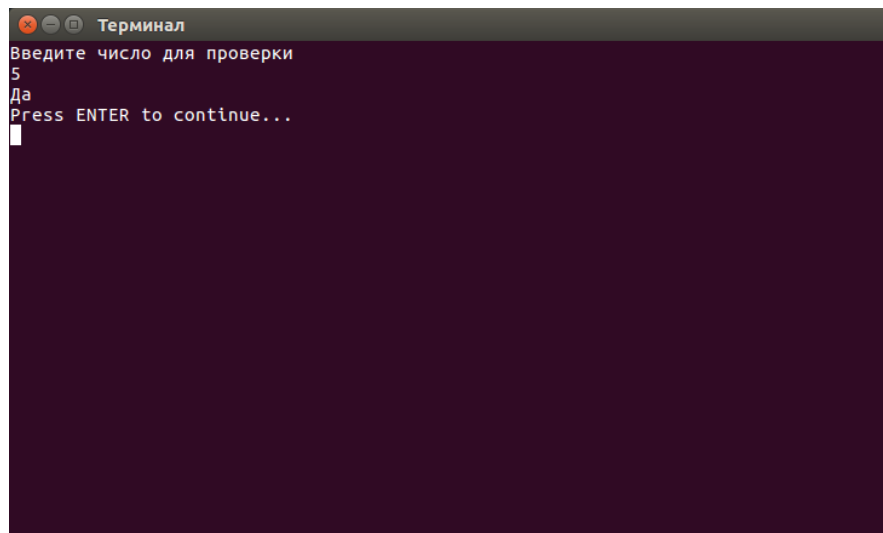


Рисунок 13 — Результат работы программы, если проверяемое число простое.

3.5 В алгоритме запускается цикл с постусловием, который работает до тех пор пока не встретит в последовательности 0. После открытия цикла вводятся числа последовательности, после открывается ещё один цикл для поиска количества делителей, который работает пока делитель меньше или равен числу последовательности. В цикле проверяется остаток от деления числа последовательности на делитель, если он равен 0, то количество делителей увеличивается на один и делитель увеличивается на один. После завершения цикла по поиску количества делителей, проверяется равенство количества делителей 2. Если оно равно 2, то на экран выводится сообщение о том, что число простое и первый цикл продолжает работу, если неравно 2, то цикл продолжает работу, не выводя сообщений. Первый цикл и вся программа завершаться, когда будет введено число 0. Блок-схема этого алгоритма представлена на рисунке 14.

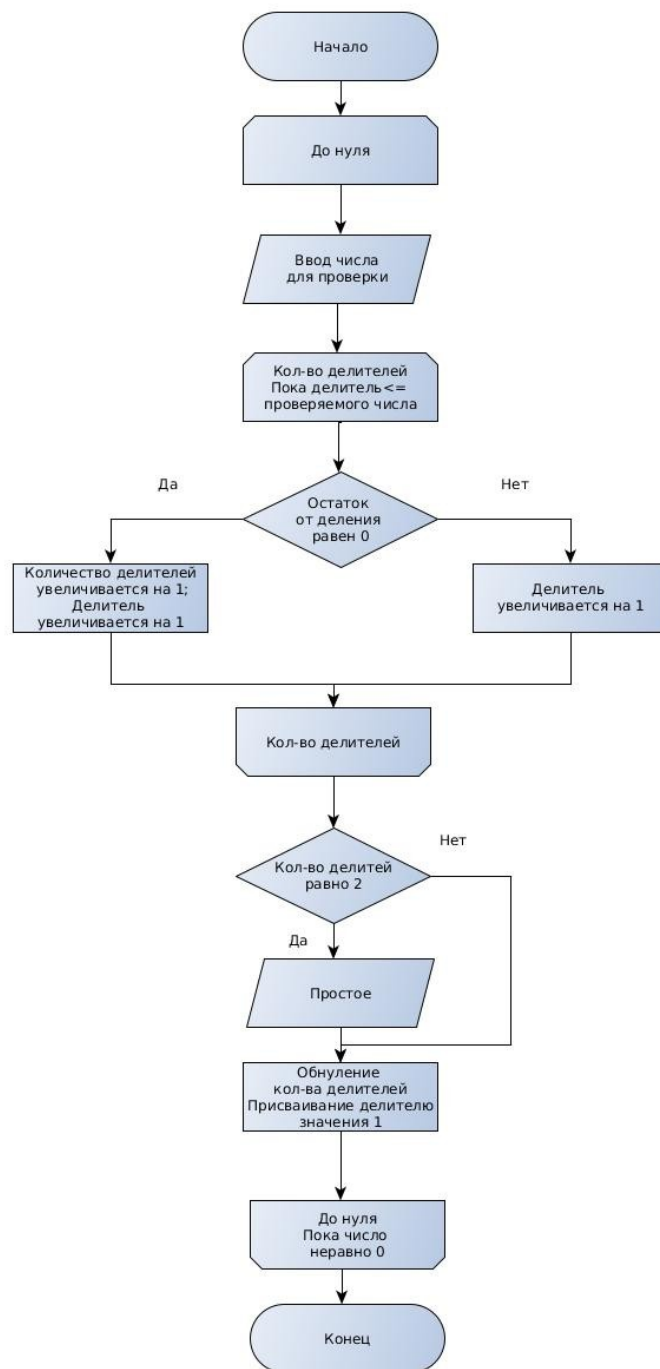


Рисунок 14 — Поиск простых чисел в последовательности.

Код программы на языке C++:

```

#include <stdio.h>
#include <iostream>
using namespace std;
int main(int argc, char **argv)
{

```

```

    {
        int i=1,a=0;
        int x;
        do {
            cout<<"Введите число для проверки "<<
endl;

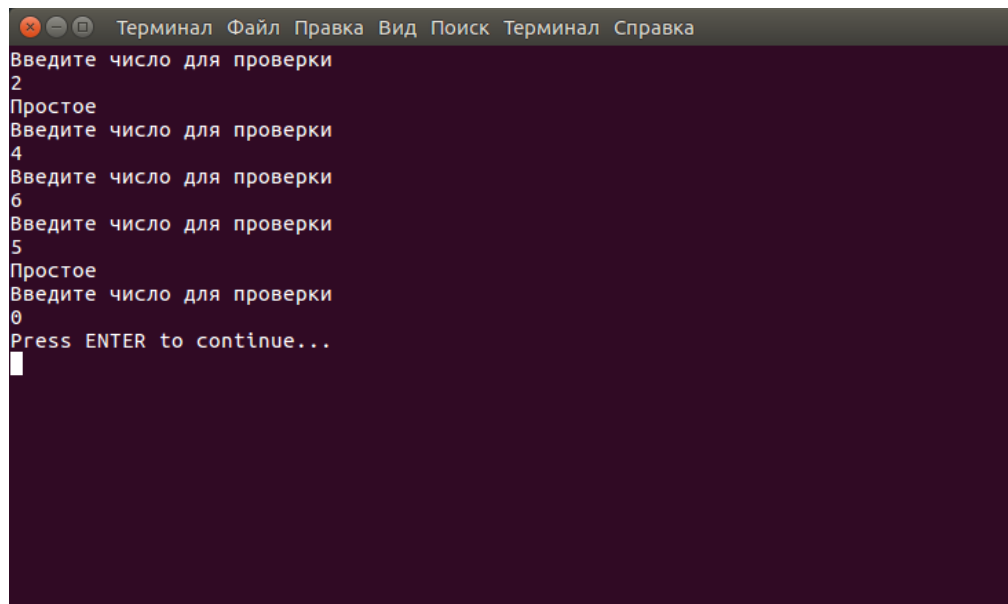
            cin>>x;
            while (i<=x) {
                if(x%i==0) {
                    a=a+1;
                    i++;
                } else {
                    i++;
                }

            }
            if(a==2) {
                cout<<"Простое"<<endl;
            }
            i=1;
            a=0;
        } while(x!=0);
    }
    return 0;
}

```

Результат работы программы при работе с последовательностью чисел 2, 4, 6, 5, 0 представлен на рисунке 15.





```
Терминал Файл Правка Вид Поиск Терминал Справка
Введите число для проверки
2
Простое
Введите число для проверки
4
Введите число для проверки
6
Введите число для проверки
5
Простое
Введите число для проверки
0
Press ENTER to continue...
█
```

Рисунок 15 — Результат работы пятой программы.

#### 4 Вывод

В процессе выполнения лабораторной работы были изучены алгоритмы ветвления и циклы, была освоена реализация ветвления и циклов в языке C++, были получены практические навыки по использованию инструкции ветвления if-else в языке C++, по использованию циклов с постусловием и предусловием при помощи инструкций while и do — while и инструкции for в языке C++.