**Лабораторная работа № 2**

Отладка программ по использованию простых типов данных: целых, вещественных, символьных, логических. Отладка программ по работе с простыми и совершенными числами.

Гвоздиков Данила Алексеевич
М3З-105Бк-20

## Анализ требований

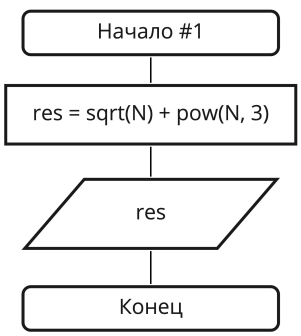
Во приведенных ниже заданиях данные не вводятся с помощью клавиатуры, а заранее определены в программе. Все вычисления производятся над неотрицательными числами.

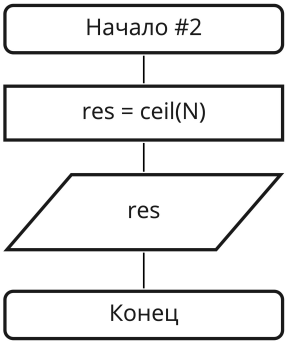
1. Программа должна определять сумму двух произвольных встроенных функций. Произвольные функции заранее предопределены (корень и куб). Результат округляется до 4-го знака после запятой.
2. Полученные вычисления предыдущей программы округляются до целого числа и выводятся.
3. Вычисляется сумма объемов шаров. На вход подается R, дельтаR, k. Все объемы шаров с радиусами попадающими в диапазон (R, R + дельтаR, R + 2 * дельтаR, R + N * дельтаR, k*R) суммируются. Выводится сумма объемов.
4. На вход подаются три числа. На выходе выводится максимальное число из трех (вычисления производить без использования встроенной функции Max) и положительность всех трех чисел.
5. Даны значения T = true, L1 = false. Вывести значения L1, L2, L3, где по порядку: L2 = T and L1; L1 = !L2; L3 = L1 and L2, T;
6. Даны значения: *A=7, B=9, C=5.* Требуется определить значение выражения *(A>B) AND (B=A+2) OR NOT (C <> B).* Полученное значение нужно изменить любым способом.
7. Напечатать буквы латинского алфавита в строку и в столбик.
8. Подсчитать общее количество символов строки и число определенной буквы в этой же строке.
9. Определить, является ли число простым.
10. Определить все простые числа, не превосходящие N
11. Разложить N на простые множители.
12. Определить все простые числа в интервале от N до M.
13. Определить простое число не превосходящее число N.
14. Определить, является ли число N совершенным.
15. Определить, являются ли числа M и N взаимнопростыми.
16. Определить, является ли число N палиндромом.

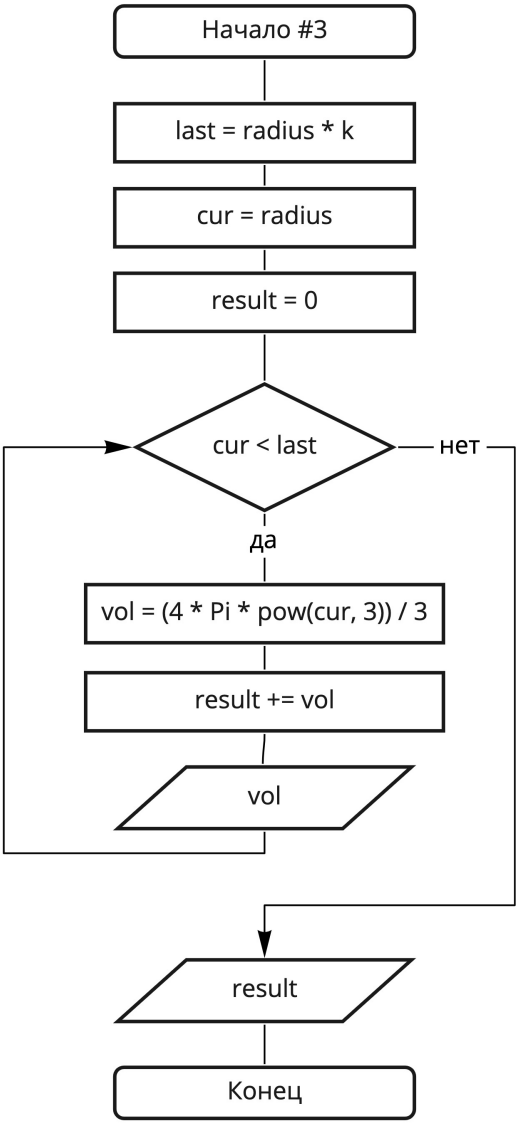
## Определение спецификаций

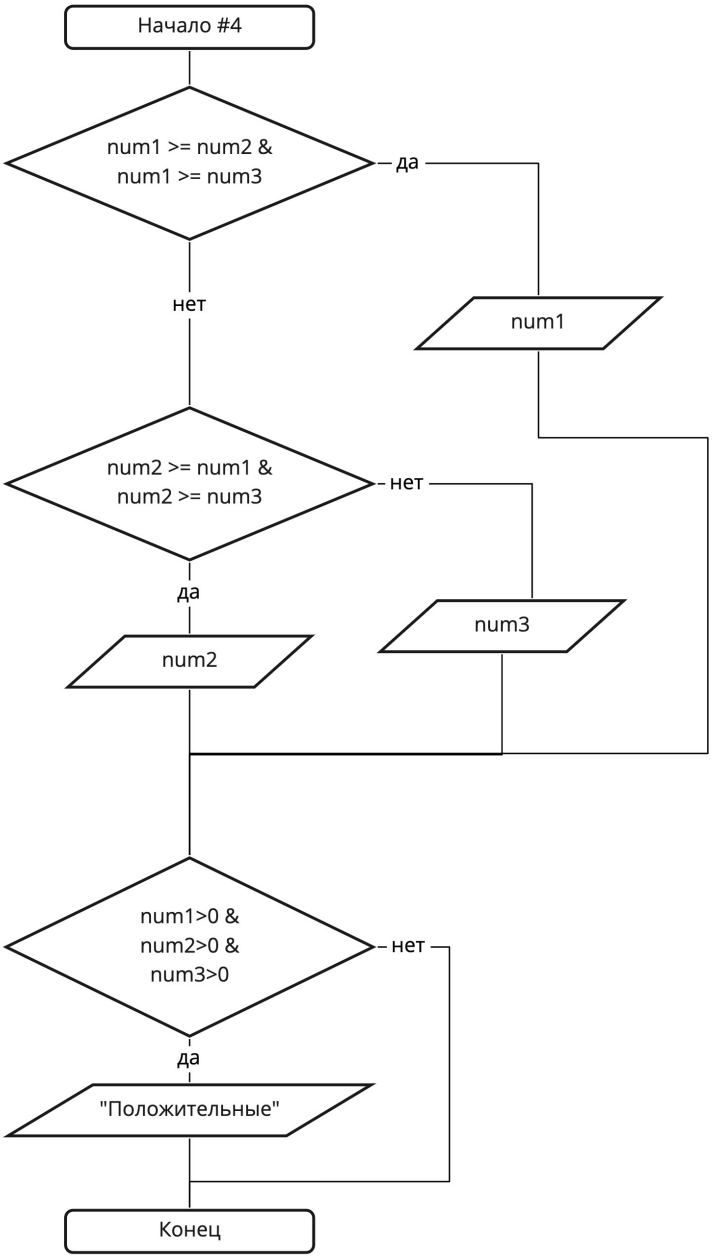
Должно работать на положительных натуральных числах.
Сложность вычисления палиндрома - O(n / 2), где n - длина строки.

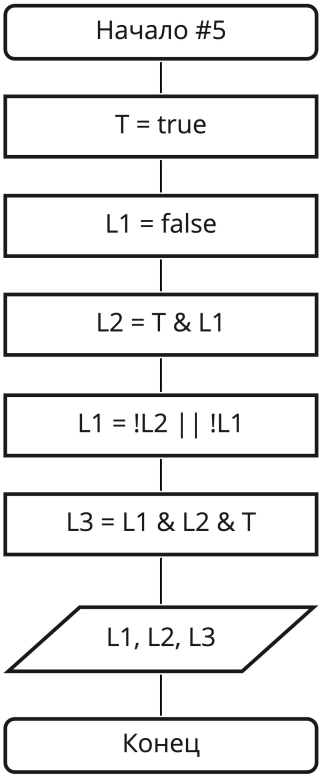
## Проектирование

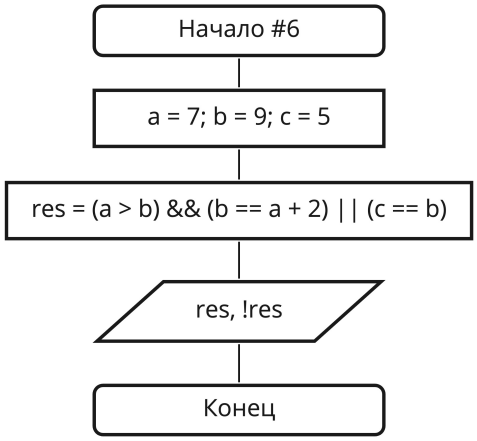

Начало #1
res = sqrt(N) + pow(N, 3)
res
Конец

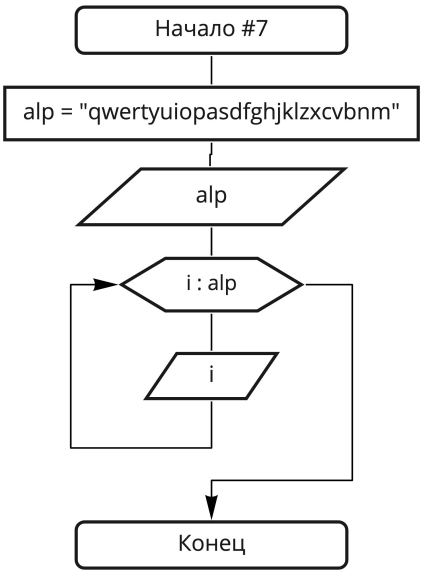

Начало #2
res = ceil(N)
res
Конец

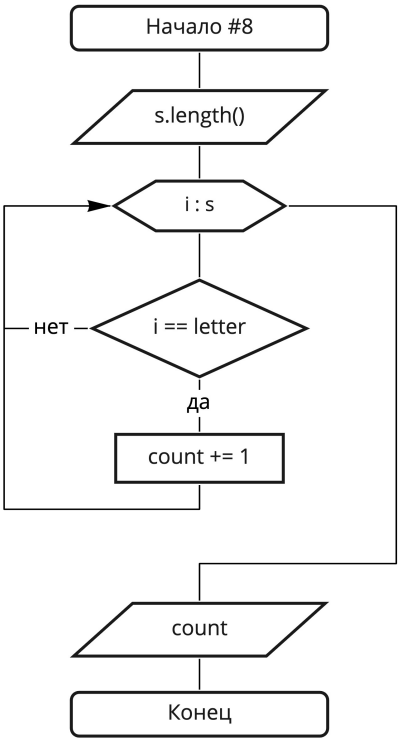

Начало #3
last = radius * k
cur = radius
result = 0
cur < last
нет
да
vol = (4 * Pi * pow(cur, 3)) / 3
result += vol
vol
result
Конец

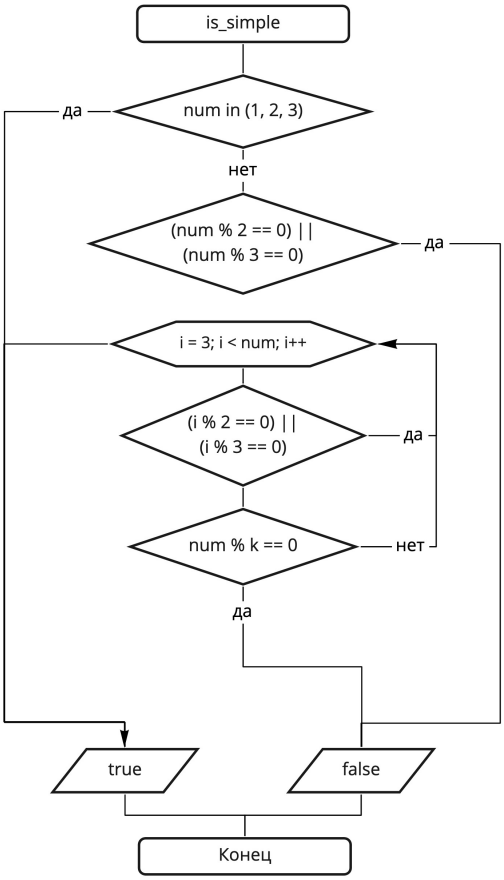

Начало #4
num1 >= num2 &
num1 >= num3
да
нет
num1
num2 >= num1 &
num2 >= num3
нет
да
num3
num2
num1>0 &
num2>0 &
num3>0
нет
да
"Положительные"
Конец

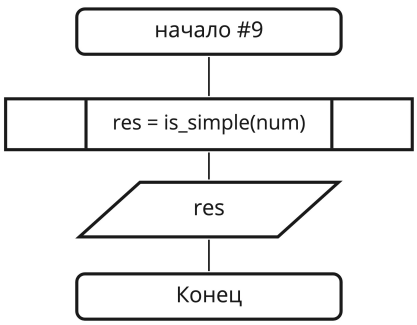

Начало #5
T = true
L1 = false
L2 = T & L1
L1 = !L2 || !L1
L3 = L1 & L2 & T
L1, L2, L3
Конец

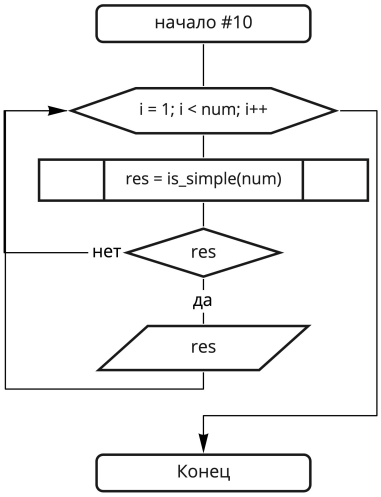

Начало #6
a = 7; b = 9; c = 5
res = (a > b) && (b == a + 2) || (c == b)
res, !res
Конец

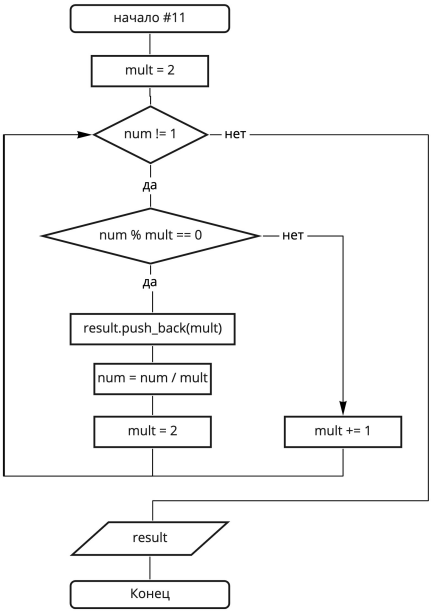

Начало #7
alp = "qwertyuiopasdfghjklzxcvbnm"
alp
i : alp
i
Конец

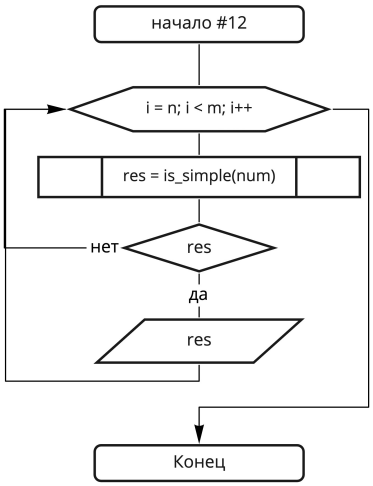

Начало #8
s.length()
i : s
нет
i == letter
да
count += 1
count
Конец

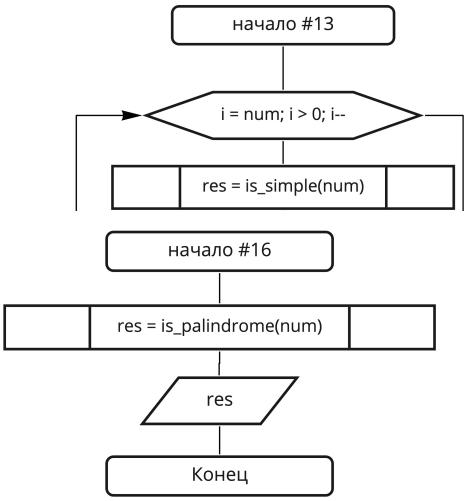

is_simple
да
num in (1, 2, 3)
нет
(num % 2 == 0) ||
(num % 3 == 0)
да
i = 3; i < num; i++
(i % 2 == 0) ||
(i % 3 == 0)
да
num % k == 0
нет
да
true
false
Конец

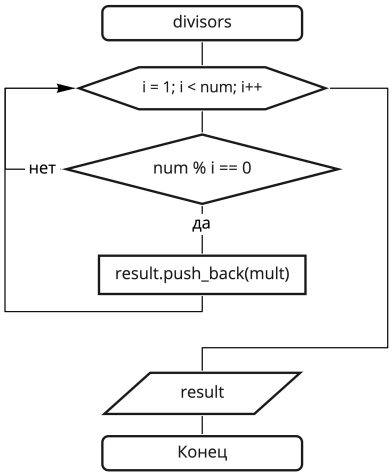

начало #9
res = is_simple(num)
res
Конец

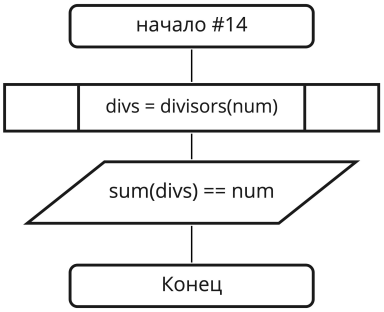

начало #10
i = 1; i < num; i++
res = is_simple(num)
нет
res
да
res
Конец

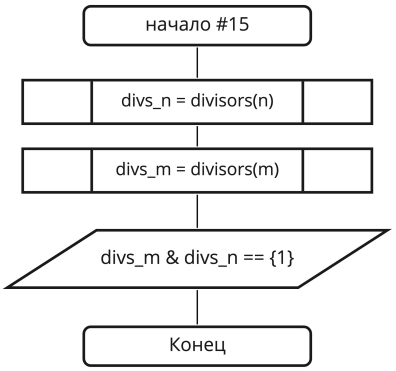

начало #11
mult = 2
num != 1
нет
да
num % mult == 0
нет
да
result.push_back(mult)
num = num / mult
mult = 2
mult += 1
result
Конец

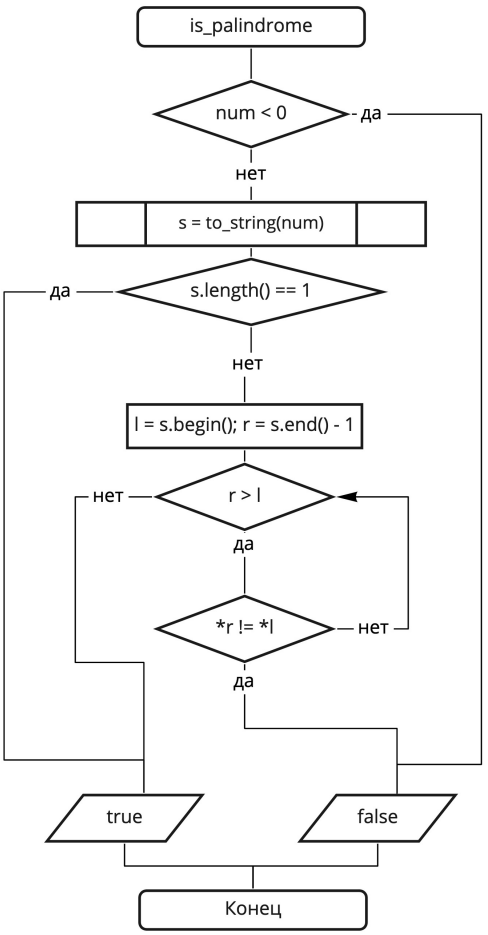

начало #12
i = n; i < m; i++
res = is_simple(num)
нет
res
да
res
Конец


начало #13
i = num; i > 0; i--
res = is_simple(num)
начало #16
res = is_palindrome(num)
res
Конец


divisors
i = 1; i < num; i++
нет
num % i == 0
да
result.push_back(mult)
result
Конец


начало #14
divs = divisors(num)
sum(divs) == num
Конец


начало #15
divs_n = divisors(n)
divs_m = divisors(m)
divs_m & divs_n == {1}
Конец


is_palindrome
num < 0
да
нет
s = to_string(num)
да
s.length() == 1
нет
l = s.begin(); r = s.end() - 1
нет
r > l
да
*r != *l
нет
да
true
false
Конец

## Исходный код

```
//
// Created by Danila Gvozdikov on 24.12.2020.
//

#include <iostream>
#include <sstream>
#include <stdexcept>
#include <string>
#include <cmath>
#include <vector>
#include <set>

using namespace std;

void throwErr(const string& message) {
    throw runtime_error(message);
}

void throwLeqZero(const int value, const string& valueName) {
    if (value <= 0) {
        throwErr(valueName + " должна(ен) быть больше 0.");
    }
}

void throwLeqZero(const double value, const string& valueName) {
    if (value <= 0) {
        throwErr(valueName + " должна(ен) быть больше 0.");
    }
}


// sum of two functions
class TProgram1 {
public:
    double run(double num) const {
        // check that number is greater then 0;
        throwLeqZero(num, "Число");

        // then count sqrt of number + cube of number and return
it
        return sqrt(num) + pow(num, 3);
    }

};

// return integer part of a number from program 1
class TProgram2 {
```

```
public:
    int run(double num) {
        // ceil trunks all numbers after comma.
        return ceil(num);
    }
};

// Count volume of several spheres with R, R + deltaR, R +
2deltaR, ..., R*k
class TProgram3 {
public:
    void run(double radius, double delta, double k) const {
        // check input
        throwLeqZero(radius, "Радиус");
        throwLeqZero(delta, "Дельта");
        throwLeqZero(k, "Коэффициент k");

        // count required variables
        double last = radius * k;
        double cur = radius;
        long double result = 0.0;
        int i = 1;

        // and count volumes in cycle until we find radius bigger
than k*R
        while (cur < last) {
            double volume = (4 * M_PI * cur * cur * cur) / 3;
            cout << fixed;
            cout.precision(4);
            cout << "Объем шара #" << i++ << ": " << volume <<
endl;
            result += volume;
            cur += delta;
        }

        cout << "Сумма объемов: " << result << endl;
    }
};

// maximum of three numbers and check if all of them are greater
than 0
class TProgram4 {
public:
    void run(int num1, int num2, int num3) const {
        int max = 0;

        // find max number
        if (num1 >= num2 && num1 >= num3) {
```

```
            max = num1;
        } else if (num2 >= num1 && num2 >= num3) {
            max = num2;
        } else {
            max = num3;
        }

        cout << "Максимальное число: " << max << endl;

        // check if all are more then 0
        if (num1 > 0 && num2 > 0 && num3 > 0) {
            cout << "Все числа положительные." << endl;
        }
    }
};

// value of three variables L1, L2, L3
class TProgram5 {
public:
    void run() {
        bool T = true;
        bool L1 = false;
        bool L2 = T && L1;
        L1 = !L2 || !L1;
        bool L3 = L1 && L2 && T;
        cout << "L1 = " << L1 << "; L2 = " << L2 << "; L3 = " <<
L3 << endl;
    }
};

// value of boolean expression
class TProgram6 {
public:
    void run() {
        int a = 7;
        int b = 9;
        int c = 5;

        // replaced not (c <> b) with (c == b)
        bool res = (a > b) && (b == a + 2) || (c == b);
        cout << "Значение выражения: " << res << endl;
        cout << "Измененненное выражение: " << !res;
    }
};

// print alphabet
class TProgram7 {
public:
```

```
    void run() {
        // use simple string to print letters
        // actually i was lazy to print it in right order, but
there was no such condition anyway
        string alphabet = "qwertyuiopasdfghjklzxcvbnm";
        cout << "В строку: " << alphabet << endl;
        cout << "В столбик: " << endl;

        for (const auto &i : alphabet) {
            cout << i << endl;
        }
    }
};

// count letters
class TProgram8 {
public:
    void run(const string &s, const char &letter) {
        cout << "общее кол-во символов: " << s.length() << endl;

        size_t count = 0;
        for (const auto &i : s) {
            // case sensitive actually
            if (letter == i) {
                ++count;
            }
        }
        cout << "Число повторений символа \"" << letter << "\": "
<< count << endl;
    }
};

// returns all simple divisors of number as vector of integer.
vector<int> simplify(const int num) {
    if (num == 1) {
        return {1};
    }
    int proc = num;
    int mult = 2;
    vector<int> result;
    while (proc != 1) {
        if (proc % mult == 0) {
            proc = proc / mult;
            result.push_back(mult);
            mult = 2;
        } else {
            mult += 1;
        }
```

```
    }

    return result;
}

bool is_simple(const int num) {
    // 1, 2, 3 are simple numbers
    if (num == 1 || num == 2 || num == 3) {
        return true;
    }

    // if number can be divided by 2 or 3 it's simple
    if ((num % 2 == 0) || (num % 3 == 0)) {
        return false;
    }

    // simple divisors starts by 3
    for (int k = 3; k < num; k++) {
        // we can skip divisors that can be divided by 3 or 2 to
make faster counts
        if ((k % 3 == 0) || (k % 2 == 0)) {
            continue;
        }

        if (num % k == 0) {
            return false;
        }
    }
    return true;
}

// Check if the number is simple
class TProgram9 {
public:
    void run(int num) {
        cout << num << " - ";
        if (is_simple(num)) {
            cout << "простое число.";
        } else {
            cout << "не простое число";
        }
        cout << endl;
    }
};

// print all simple numbers less then N
class TProgram10 {
public:
```

```
    void run(int num) {
        for (int i = 1; i < num; i++) {
            if (is_simple(i)) {
                cout << "Простое число: " << i << "\n";
            }
        }
    }
};

// print simple multipliers
class TProgram11 {
public:
    void run(int num) {
        vector<int> simple = simplify(num);
        cout << "Простые множетели: ";
        for (const auto &i : simple) {
            cout << i << " ";
        }
        cout << endl;
    }
};

// print simple numbers in (n, m);
class TProgram12 {
public:
    void run(int n, int m) {
        for (int i = n; i < m; i++) {
            if (is_simple(i)) {
                cout << "Простое число: " << i << "\n";
            }
        }
        cout << endl;
    }
};

// find simple number less then num
class TProgram13 {
public:
    void run(int num) {
        if (num <= 1) {
            cout << "Таких натуральных чисел нет." << endl;
            return;
        }
        for (int i = num - 1; i > 1; i--) {
            if (is_simple(i)) {
                cout << "Простое число: " << i << endl;
                return;
            }
```

```
        }
    }

};

// returns vector of all divisors of number
vector<int> divisors(int num) {
    vector<int> result = {};
    for (int i = 1; i < num; i++) {
        if (num % i == 0) {
            result.push_back(i);
        }
    }
    return result;
}

// check if number is perfect
class TProgram14 {
public:
    void run(int num) {
        vector<int> divs = divisors(num);
        int sum = 0;
        for (const int &i : divs) {
            sum += i;
        }

        cout << num << " - ";
        if (sum == num) {
            cout << "совершенное.";
        } else {
            cout << "не совершенное";
        }
        cout << endl;
    }
};

// check if two numbers are simple to each other
class TProgram15 {
public:
    void run(int n, int m) {
        // get all divisors of n and m
        vector<int> divs_n = divisors(n);
        vector<int> divs_m = divisors(m);

        // then make sets from them
        set<int> divs_n_set(divs_n.begin(), divs_n.end());
        set<int> divs_m_set(divs_m.begin(), divs_m.end());
```

```
        set<int> res;
        // count intersection of two sets
        set_intersection(divs_n_set.begin(), divs_n_set.end(),
divs_m_set.begin(), divs_m_set.end(),
                         inserter(res, res.begin()));

        // if intersection contains only one number then it always
will be "1"
        // that means our numbers are simple to each other.
        cout << n << "и" << m << " - ";
        if (res.size() == 1) {
            cout << "взаимно простые.";
        } else {
            cout << "не взаимно простые";
        }
        cout << endl;
    }
};

class TProgram16 {
public:
    void run(int num) {
        cout << num << " - ";
        if (is_palindrome(num)) {
            cout << "палиндром";
        } else {
            cout << "не палиндром";
        }
        cout << endl;
    }

protected:
    bool is_palindrome(int num) {
        if (num < 0) {
            // numbers lower then 0 cannot be palindrome because
of "-" at the beginning of the string.
            return false;
        }
        // make string stream
        stringstream ss;
        // input our number to stream
        ss << num;
        // and save it as string
        string strnum = ss.str();
        size_t length = strnum.length();
        if (length == 1) {
            // strings with only one letter are always palindromes
            return true;
```

```
        }

        // make two iterators from our string
        auto left = strnum.begin();
        // the last one is '\0', so we should skip that.
        auto right = strnum.end() - 1;

        // then iterate while we do not reach the middle of the
string
        while (right > left) {
            // check if the letter of the left iterator equals the
letter of the right iterator
            if (*left != *right) {
                return false;
            }
            // move iterators to the neighbour symbol
            left++;
            right--;
        }
        return true;
    }
};

int main() {
    cout << "Program #1 - сумма двух функций (15)" << endl;
    double res = TProgram1().run(15);
    cout << "Результат: " << res;
    cout << endl << endl;

    cout << "Program #2 - целая часть" << endl;
    cout << "Целая часть - " << TProgram2().run(res);
    cout << endl << endl;

    cout << "Program #3 - сумма объемов шаров (R=10, delta=0.5,
k=5)" << endl;
    TProgram3().run(10.0, 0.5, 5);
    cout << endl << endl;

    cout << "Program #4 - максимум и положительность" << endl;
    TProgram4().run(9, 2, 15);
    cout << endl << endl;

    cout << "Program #5 - значение переменных L1, L2, L3" << endl;
    TProgram5().run();
    cout << endl << endl;

    cout << "Program #6 - значение выражения" << endl;
    TProgram6().run();
```

```
    cout << endl << endl;

    cout << "Program #7 - алфавит" << endl;
    TProgram7().run();
    cout << endl << endl;

    cout << "Program #8 - кол-во символов (\"London is the capital
of Great Britain\", 'i')" << endl;
    TProgram8().run("London is the capital of Great Britain",
'i');
    cout << endl << endl;

    cout << "Program #9 - простое число (27, 10)" << endl;
    TProgram9 prog9;
    prog9.run(27);
    prog9.run(10);
    cout << endl << endl;

    cout << "Program #10 - простые числа < N (100)" << endl;
    TProgram10().run(100);
    cout << endl << endl;

    cout << "Program #11 - разложение на простые множители (50)"
<< endl;
    TProgram11().run(50);
    cout << endl << endl;

    cout << "Program #12 - простые числа (N, M) - (5 - 40)" <<
endl;
    TProgram12().run(5, 40);
    cout << endl << endl;

    cout << "Program #13 - простое число < N (100)" << endl;
    TProgram13().run(100);
    cout << endl << endl;

    cout << "Program #14 - совершенное число (496, 50)." << endl;
    TProgram14 prog14;
    prog14.run(496);
    prog14.run(50);
    cout << endl << endl;

    cout << "Program #15 - взаимно простые числа ((14, 27), (100,
420))" << endl;
    TProgram15 prog15;
    prog15.run(14, 27);
    prog15.run(100, 420);
    cout << endl << endl;
```

```
    cout << "Program #16 - палиндром (1052, 176671, 88988)" <<
endl;
    TProgram16 prog16;
    prog16.run(1052);
    prog16.run(176671);
    prog16.run(88988);

    return 0;
}
```

## Тестирование

Тестирование с помощью пробного запуска. Результат:

```
Program #1 - сумма двух функций (15)
Результат: 3378.87

Program #2 - целая часть
Целая часть - 3379

Program #3 - сумма объемов шаров (R=10, delta=0.5, k=5)
Объем шара #1: 4188.7902
Объем шара #2: 4849.0483
Объем шара #3: 5575.2798
Объем шара #4: 6370.6263
Объем шара #5: 7238.2295
Объем шара #6: 8181.2309
Объем шара #7: 9202.7721
Объем шара #8: 10305.9947
Объем шара #9: 11494.0403
Объем шара #10: 12770.0505
Объем шара #11: 14137.1669
Объем шара #12: 15598.5311
Объем шара #13: 17157.2847
Объем шара #14: 18816.5692
Объем шара #15: 20579.5263
Объем шара #16: 22449.2975
Объем шара #17: 24429.0245
Объем шара #18: 26521.8488
Объем шара #19: 28730.9120
Объем шара #20: 31059.3558
Объем шара #21: 33510.3216
Объем шара #22: 36086.9512
Объем шара #23: 38792.3861
Объем шара #24: 41629.7679
Объем шара #25: 44602.2381
Объем шара #26: 47712.9384
Объем шара #27: 50965.0104
```

Объем шара #28: 54361.5957
Объем шара #29: 57905.8358
Объем шара #30: 61600.8724
Объем шара #31: 65449.8469
Объем шара #32: 69455.9012
Объем шара #33: 73622.1766
Объем шара #34: 77951.8149
Объем шара #35: 82447.9576
Объем шара #36: 87113.7463
Объем шара #37: 91952.3226
Объем шара #38: 96966.8280
Объем шара #39: 102160.4043
Объем шара #40: 107536.1929
Объем шара #41: 113097.3355
Объем шара #42: 118846.9737
Объем шара #43: 124788.2490
Объем шара #44: 130924.3030
Объем шара #45: 137258.2774
Объем шара #46: 143793.3137
Объем шара #47: 150532.5536
Объем шара #48: 157479.1385
Объем шара #49: 164636.2102
Объем шара #50: 172006.9102
Объем шара #51: 179594.3800
Объем шара #52: 187401.7614
Объем шара #53: 195432.1958
Объем шара #54: 203688.8249
Объем шара #55: 212174.7902
Объем шара #56: 220893.2335
Объем шара #57: 229847.2961
Объем шара #58: 239040.1198
Объем шара #59: 248474.8462
Объем шара #60: 258154.6167
Объем шара #61: 268082.5731
Объем шара #62: 278261.8569
Объем шара #63: 288695.6097
Объем шара #64: 299386.9731
Объем шара #65: 310339.0887
Объем шара #66: 321555.0981
Объем шара #67: 333038.1428
Объем шара #68: 344791.3645
Объем шара #69: 356817.9048
Объем шара #70: 369120.9052
Объем шара #71: 381703.5074
Объем шара #72: 394568.8529
Объем шара #73: 407720.0834
Объем шара #74: 421160.3403
Объем шара #75: 434892.7654

Объем шара #76: 448920.5002
Объем шара #77: 463246.6863
Объем шара #78: 477874.4653
Объем шара #79: 492806.9788
Объем шара #80: 508047.3684
Сумма объемов: 12810577.0833

Program #4 - максимум и положительность
Максимальное число: 15
Все числа положительные.

Program #5 - значение переменных L1, L2, L3
L1 = 1; L2 = 0; L3 = 0

Program #6 - значение выражения
Значение выражения: 0
Измененное выражение: 1

Program #7 - алфавит
В строку: qwertyuiopasdfghjklzxcvbnm
В столбик:
q
w
e
r
t
y
u
i
o
p
a
s
d
f
g
h
j
k
l
z
x
c
v
b

n
m

Program #8 - кол-во символов ("London is the capital of Great Britain", 'i')
общее кол-во символов: 38
Число повторений символа "i": 4

Program #9 - простое число (27, 10)
27 - не простое число
10 - не простое число

Program #10 - простые числа < N (100)
Простое число: 1
Простое число: 2
Простое число: 3
Простое число: 5
Простое число: 7
Простое число: 11
Простое число: 13
Простое число: 17
Простое число: 19
Простое число: 23
Простое число: 29
Простое число: 31
Простое число: 37
Простое число: 41
Простое число: 43
Простое число: 47
Простое число: 53
Простое число: 59
Простое число: 61
Простое число: 67
Простое число: 71
Простое число: 73
Простое число: 79
Простое число: 83
Простое число: 89
Простое число: 97

Program #11 - разложение на простые множители (50)
Простые множетели: 2 5 5

Program #12 - простые числа (N, M) - (5 - 40)
Простое число: 5
Простое число: 7
Простое число: 11
Простое число: 13
Простое число: 17
Простое число: 19
Простое число: 23
Простое число: 29
Простое число: 31
Простое число: 37

Program #13 - простое число < N (100)
Простое число: 97

Program #14 - совершенное число (496, 50).
496 - совершенное.
50 - не совершенное

Program #15 - взаимно простые числа ((14, 27), (100, 420))
14и27 - взаимно простые.
100и420 - не взаимно простые

Program #16 - палиндром (1052, 176671, 88988)
1052 - не палиндром
176671 - палиндром
88988 - палиндром