Лабораторная работа № 3

Отладка программ по работе с производящими функциями и числовыми рядами. Отладка программ по обработке числовых последовательностей.

Гвоздиков Данила Алексеевич
М3З-105Бк-20

Анализ требований

В заданиях 1-3 находится сумма первых членов ряда. На ввод подается два числа: N и X, где N - количество первых членов числового ряда, X - неизвестное число ряда. В заданиях 5-11 на вход подается последовательность целых чисел произвольной длины.

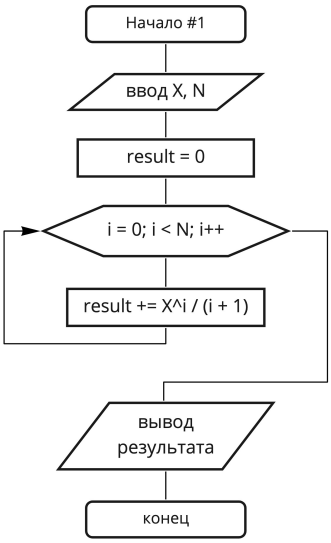
Задания:

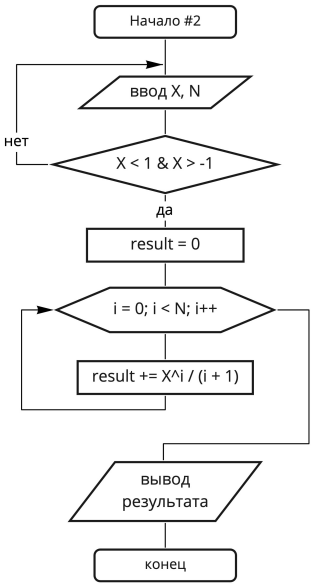
1. Вводится N и X, найти сумму первых N членов ряда *Y = 1 + X/2 + X^2/3 + …*
2. Вводится N и X, найти сумму первых N членов ряда *Y = 1 + X/2 + X^2/3 + …* при |X| < 1
3. Вводится N и X, найти сумму первых N членов ряда *Y = 1 - 1/X + 2/X^2 + …* при |X| > 1
4. Нужно вывести все трехзначные простые числа.
5. Вводится числовая последовательность до тех пора пока на ввод не подастся слово «end». Нужно найти сумму всех отрицательных чисел в последовательности и вывести его.
6. Вводится числовая последовательность до тех пора пока на ввод не подастся слово «end». Нужно найти и вывести наибольшее число из последовательности.
7. Вводится числовая последовательность до тех пора пока на ввод не подастся слово «end». Нужно подсчитать и вывести количество нулей в последовательности.
8. Вводится числовая последовательность до тех пора пока на ввод не подастся слово «end». Нужно найти и вывести наибольшее из всех четных чисел в последовательности.
9. Вводится числовая последовательность до тех пора пока на ввод не подастся слово «end». Нужно найти наибольшее из всех отрицательных чисел
10. Вводится числовая последовательность до тех пора пока на ввод не подастся число «0». Нужно определить, является ли последовательность знакопеременной и вывести ответ.
11. Вводится числовая последовательность до тех пора пока на ввод не подастся число «0». Нужно найти и вывести два наименьших числа.

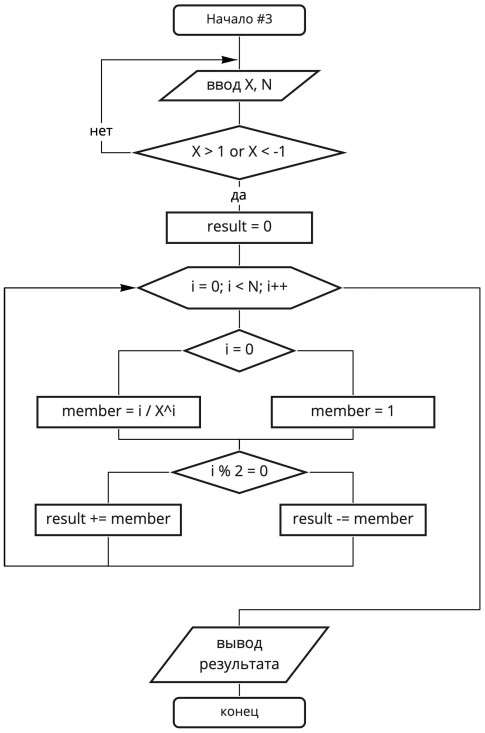
Определение спецификаций

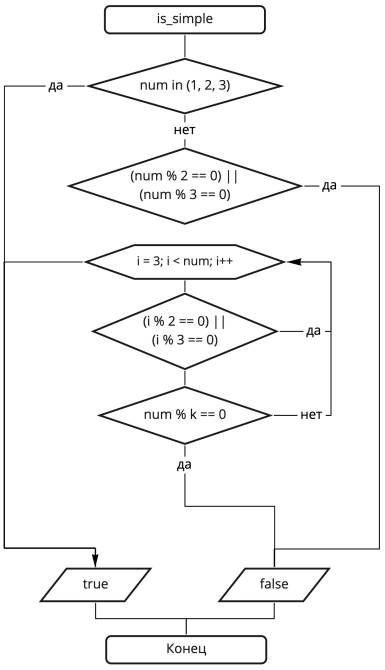
Сложность работы с числовыми последовательностями не больше O(2n).

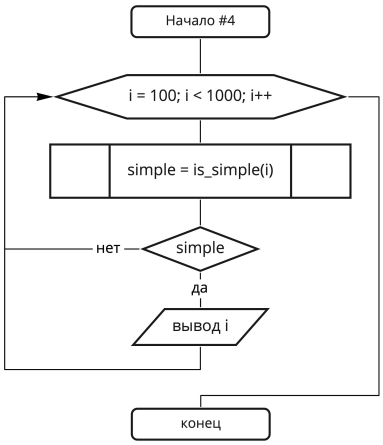
Проектирование

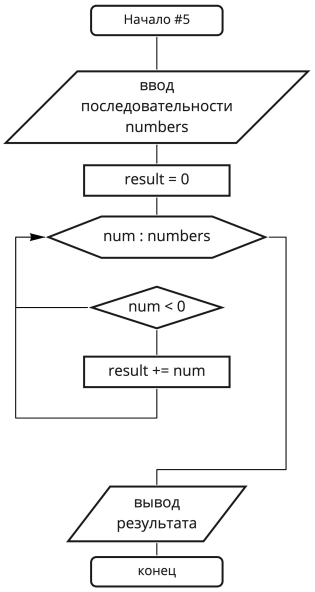

Начало #1
ввод X, N
result = 0
i = 0; i < N; i++
result += X^i / (i + 1)
вывод результата
конец

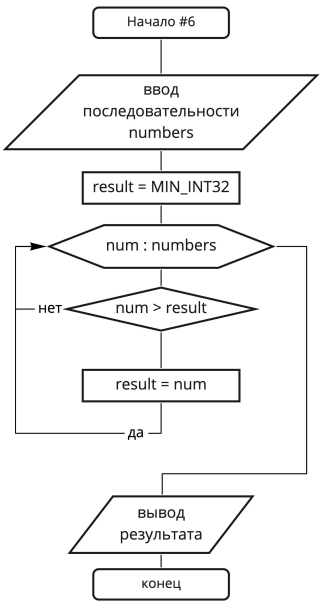

Начало #2
ввод X, N
нет
X < 1 & X > -1
да
result = 0
i = 0; i < N; i++
result += X^i / (i + 1)
вывод результата
конец

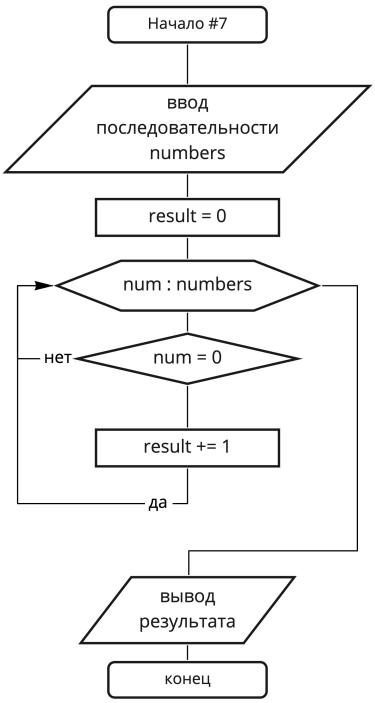

Начало #3
ввод X, N
нет
X > 1 or X < -1
да
result = 0
i = 0; i < N; i++
i = 0
member = i / X^i
member = 1
i % 2 = 0
result += member
result -= member
вывод результата
конец

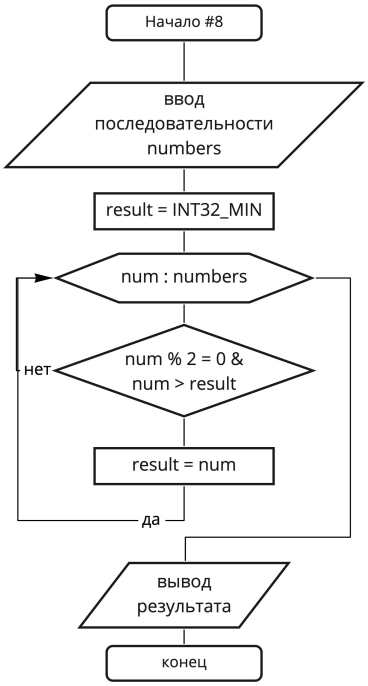

is_simple
да
num in (1, 2, 3)
нет
(num % 2 == 0) | |
(num % 3 == 0)
да
i = 3; i < num; i++
(i % 2 == 0) | |
(i % 3 == 0)
да
num % k == 0
нет
да
true
false
Конец

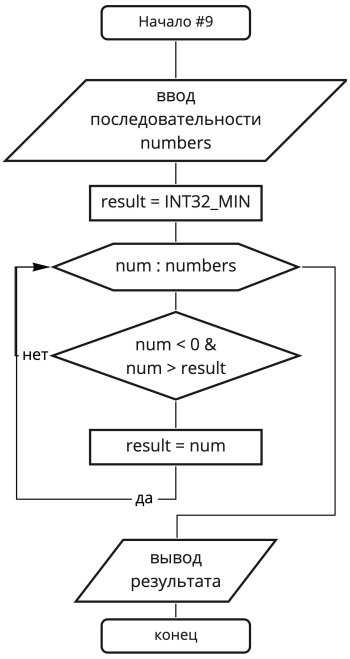

Начало #4
i = 100; i < 1000; i++
simple = is_simple(i)
нет
simple
да
вывод i
конец

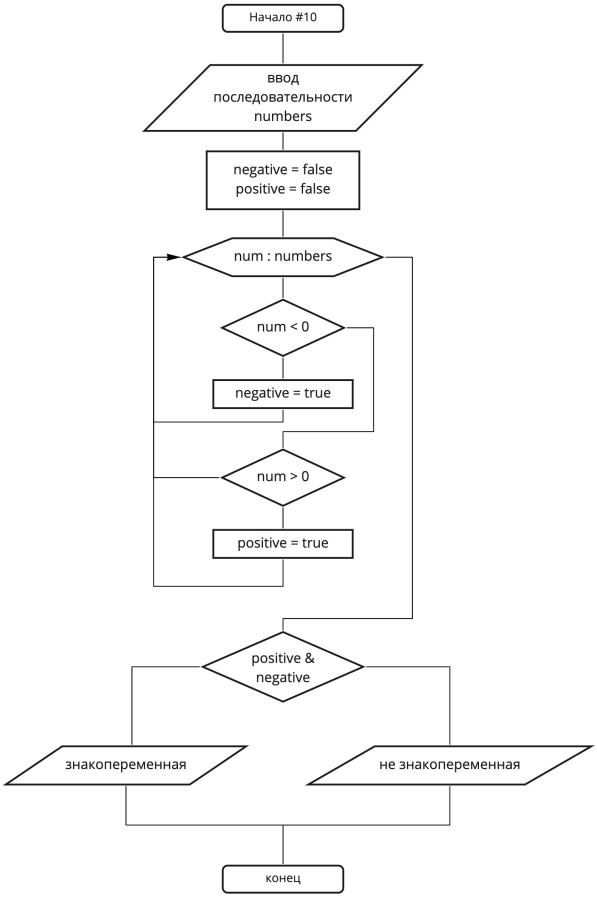

Начало #5
ввод последовательности numbers
result = 0
num : numbers
num < 0
result += num
вывод результата
конец

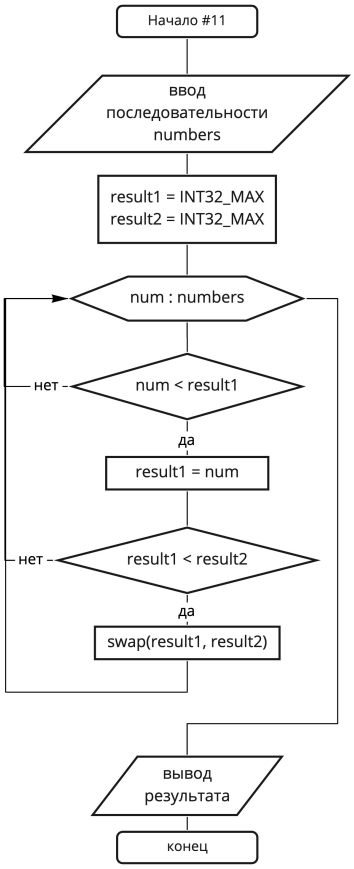

Начало #6
ввод последовательности numbers
result = MIN_INT32
num : numbers
нет
num > result
result = num
да
вывод результата
конец


Начало #7
ввод последовательности numbers
result = 0
num : numbers
нет
num = 0
result += 1
да
вывод результата
конец


Начало #8
ввод последовательности numbers
result = INT32_MIN
num : numbers
нет
num % 2 = 0 & num > result
result = num
да
вывод результата
конец


Начало #9
ввод последовательности numbers
result = INT32_MIN
num : numbers
нет
num < 0 & num > result
result = num
да
вывод результата
конец


Начало #10
ввод последовательности numbers
negative = false
positive = false
num : numbers
num < 0
negative = true
num > 0
positive = true
positive & negative
знакопеременная
не знакопеременная
конец


Начало #11
ввод последовательности numbers
result1 = INT32_MAX
result2 = INT32_MAX
num : numbers
нет
num < result1
да
result1 = num
нет
result1 < result2
да
swap(result1, result2)
вывод результата
конец

Исходный текст программы

```
#include <cmath>
#include <stdexcept>
#include <string>
#include <regex>
#include <iostream>


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.");
    }
}

double inputNum(const string& err_desc = "") {
    string input;
    cin >> input;

    basic_regex numreg("[-]?\\d+(\\.\\d+)?", regex_constants::ECMAScript);
    if (!regex_match(input.begin(), input.end(), numreg)) {
        throw runtime_error("Неверный формат числа.");
    }

    double result = 0;
    try {
        result = stod(input);
    } catch (exception) {
        throw runtime_error(err_desc);
    }
    return result;
}

double inputDouble(const string& err_desc = "") {
    return inputNum(err_desc);
}
```

```
int inputInt(const string& err_desc = "") {
    return int(inputNum(err_desc));
}


// Base class for program
// All programs below inherits base program but should provide:
//  - variables to input and variables for results
//  - method InputValues() to input required data
//  - method ValidateValues() to validate input that throws runtime_error when
data is incorrect.
//  - method ProcessValues() to cound result
//  - const method OutputResult() to print result
class TProgram {
public:
    void run() {
        while (true) {
            try {
                this->InputValues();  // input required variables
            } catch (runtime_error) {
                cout << "Некорректный тип введенных данных, ожидалось целое
число." << endl
                     << "Попробуйте еще раз." << endl;
                continue;
            }

            try {
                this->ValidateData();
            } catch (runtime_error err) {
                cout << "Введеные данные некорректны, ошибка: \"" << err.what()
<< "\"" << endl
                     << "Попробуйте ввести новые данные." << endl;
                continue;
            }

            break;
        }

        this->ProcessValues(); // count result from input
        cout << endl;  // just one empty string between input and output

        // Print fixed float values with 4 numbers after the comma.
        cout << fixed;
        cout.precision(4);
        this->OutputResult();  // print results
    };
```

```
protected:
    virtual void InputValues() {};
    virtual void ValidateData() const {};
    virtual void ProcessValues() {};
    virtual void OutputResult() const {};
};

// Find sum of N first members in row Y = 1 + X/2 + X^2/3 ....
class TProgram1 : public TProgram {
protected:
    int N = 0;
    double X = 0.0;
    long double result = 0.0;

    void InputValues() {
        cout << "Введите количество членов ряда N: ";
        N = inputInt();
        cout << "Введите X: ";
        X = inputDouble();
    }
    void ValidateData() const {
        throwLeqZero(N, "Кол-во членов ряда.");
    }
    void ProcessValues() {
        for (int i = 0; i < N; i++) {
            result += pow(X, i) / (i + 1);
        }
    }
    void OutputResult() const {
        cout << "Сумма первых " << N << " членов ряда при X = " << X << ": " <<
result;
    }
};

// Program2 is the same as Program1 but the different condition for X
class TProgram2 : public TProgram1 {
protected:
    void ValidateData() const {
        if (X >= 1 || X <= -1) {
            throw runtime_error("X должен быть меньше 1 по модулю.");
        }
    }
};

// Program3 is about row of numbers two, but different row and different
condition for X
class TProgram3 : public TProgram1 {
protected:
```

```
    void ValidateData() const {
        if (X <= 1 and X >= -1) {
            throw runtime_error("X должен быть больше 1 по модулю.");
        }
    }
    void ProcessValues() {
        double member;
        for (int i = 0; i < N; i++) {
            if (i == 0) {
                member = 1;
            } else {
                member = pow(X, i) / i;
            }

            i % 2 == 0 ? result += member : result -= member;
        }
    }
};

bool isSimple(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;
}

// Print all 3-digit simple numbers
class TProgram4 : public TProgram {
public:
```

```
    void run() {
        // Print fixed float values with 4 numbers after the comma.
        cout << fixed;
        cout.precision(4);
        this->OutputResult();  // print results
    }
protected:
    void OutputResult() const {
        cout << "Простые числа:\n";
        for (int i = 100; i < 1000; i += 1) {
            if (isSimple(i)) {
                cout << i << endl;
            }
        }
    }
};


// input several integer values until the finish words found in input.
vector<int> inputSeveralNumbers(const string& finishWord = "end") {
    vector<int> result;

    string input = "";
    cout << "Введите последовательность чисел (введите \"" << finishWord << "\" чтобы закончить): ";
    while (true) {
        cin >> input;
        if (input == finishWord) {
            break;
        }
        result.push_back(stoi(input));
    }

    return result;
}

// Find sum of all negative numbers
class TProgram5 : public TProgram {
protected:
    int result = 0;
    vector<int> numbers;

    virtual void InputValues() {
        numbers = inputSeveralNumbers();
    }
    virtual void ProcessValues() {
        for (const auto& n : numbers) {
            if (n < 0) {
```

```
                result += n;
            }
        }
    }
    virtual void OutputResult() const {
        cout << "Сумма отрицательных чисел: " << result;
    }
};

// All below programs work with row of numbers, so all of them has nearly same
input logic
// Program6 is about finding the greatest number in row.
class TProgram6 : public TProgram5 {
protected:
    void ProcessValues() {
        int biggest = numbers[0];
        for (int i = 1; i < numbers.size(); i++) {
            biggest = max(biggest, numbers[i]);
        }
        result = biggest;
    }
    void OutputResult() const {
        cout << "Самое большое число: " << result;
    }
};

// find the number of zeroes in the row
class TProgram7 : public TProgram5 {
protected:
    void ProcessValues() {
        for (int num : numbers) {
            result += num == 0;
        }
    }
    void OutputResult() const {
        cout << "Количество нулей: " << result;
    }
};

// find the greatest of all even numbers in row
class TProgram8 : public TProgram5 {
protected:
    bool wasFound = false;
    void ProcessValues() {
        result = INT32_MIN;
        for (int num : numbers) {
            if (num % 2 == 0) {
                result = max(num, result);
```

```
                wasFound = true;
            }
        }
    }
    void OutputResult() const {
        if (wasFound) {
            cout << "Самое большое из четных чисел: " << result;
        } else {
            cout << "Нет четных чисел.";
        }

    }
};

// find the greatest number of all negative ones in row
class TProgram9 : public TProgram5 {
protected:
    bool wasFound = false;
    void ProcessValues() {
        result = INT32_MIN;
        for (int num : numbers) {
            if (num < 0) {
                result = max(num, result);
                wasFound = true;
            }
        }
    }
    void OutputResult() const {
        if (wasFound) {
            cout << "Самое большое число из отрицательных чисел: " << result;
        } else {
            cout << "Нет отрицательных чисел.";
        }
    }
};

// Check if the row has negative and positive answers
class TProgram10 : public TProgram {
protected:
    bool positive = false;
    bool negative = false;
    vector<int> numbers;
    virtual void InputValues() {
        numbers = inputSeveralNumbers("0");
    }
    virtual void ProcessValues() {
        bool negative = false;
        bool positive = false;
```

```
        for (int num : numbers) {
            if (num < 0) {
                negative = true;
            } else {
                positive = true;
            }
            if (positive && negative) {
                break;
            }
        }
    }
    virtual void OutputResult() const {
        if (positive && negative) {
            cout << "Последовательность знакопеременная.";
        } else {
            cout << "Последовательность одного знака.";
        }
    }
};

// Find two the smallest numbers
class TProgram11 : public TProgram10 {
protected:
    int result1 = INT32_MAX;
    int result2 = INT32_MAX;
    void ProcessValues() {
        for (int num : numbers) {
            if (num <= result1) {
                result1 = num;
            }
            if (result1 <= result2) {
                int temp = result1;
                result1 = result2;
                result2 = temp;
            }
        }
    }
    void OutputResult() const {
        cout << "Два самых маленьких числа: " << result1 << ", " << result2;
    }
};

TProgram* ChooseProgram(int num) {
    switch (num) {
        case 1: return new TProgram1();
        case 2: return new TProgram2();
        case 3: return new TProgram3();
        case 4: return new TProgram4();
```

```
        case 5: return new TProgram5();
        case 6: return new TProgram6();
        case 7: return new TProgram7();
        case 8: return new TProgram8();
        case 9: return new TProgram9();
        case 10: return new TProgram10();
        case 11: return new TProgram11();
        default: return nullptr;
    }
}

int main() {

    // Для начала выбираем программу, которую мы хотим запустить
    cout << "Выбери программу:" << endl
         << "1 - Сумма первых членов ряда Y = 1 + X/2 + X^2/3 + ..." << endl
         << "2 - Сумма первых членов ряда Y = 1 + X/2 + X^2/3 + ... при |X| < 1"
<< endl
         << "3 - Сумма первых членов ряда Y = 1 + 1/X + 2/X^2 + ... при |X| > 1"
<< endl
         << "4 - Все простые трехзначные числа." << endl
         << "5 - Сумма всех отрицательных чисел в последовательности." << endl
         << "6 - Наибольшее число из последовательности." << endl
         << "7 - Количество нулей в последовательности" << endl
         << "8 - Наибольшее из четных чисел в последовательности" << endl
         << "9 - Наибольшее из отрицательных чисел в последовательности" << endl
         << "10 - Определение знакопеременности последовательности" << endl
         << "11 - Два наименьших числа из последовательности" << endl;

    //
    int progNum = 0;
    while (true) {
        cout << "Ввод: ";

        try {
            progNum = inputInt("Номер программы");
        } catch (...) {
            progNum = 0;
        }

        if (1 <= progNum && progNum <= 11) {
            break;
        }

        cout << "Ввод должен содержать целое число от 1 до 6. Попробуйте еще
раз." << endl;
    }
```

```
    cout << endl;
    TProgram* program = ChooseProgram(progNum);

    program->run();

    delete program;
    return 0;
}
```

Тестирование

Программа 1

Ввод: N=3 X=3
Ожидание: 5.5
Вывод: 5.5

Ввод: N=1 X=0
Ожидание: 1
Вывод: 1.0

Программа 2

Ввод: N=10, X=5
Ожидание: Ошибка
Вывод: Ошибка

Ввод: N=10, X=0
Ожидание: 1
Вывод: 1

Программа 3

Ввод: N=0 X=2
Ожидание: Ошибка
Вывод: Ошибка

Ввод: N=5 X=20
Ожидание: 0.9546
Вывод: 0.9546

Программа 4

Вывод:
101
103
107
109

113
127
131
137
139
149
151
157
163
167
173
179
181
191
193
197
199
211
223
227
229
233
239
241
251
257
263
269
271
277
281
283
293
307
311
313
317
331
337
347
349
353
359
367
373
379
383
389

397
401
409
419
421
431
433
439
443
449
457
461
463
467
479
487
491
499
503
509
521
523
541
547
557
563
569
571
577
587
593
599
601
607
613
617
619
631
641
643
647
653
659
661
673
677
683
691

701
709
719
727
733
739
743
751
757
761
769
773
787
797
809
811
821
823
827
829
839
853
857
859
863
877
881
883
887
907
911
919
929
937
941
947
953
967
971
977
983
991
997

Программа 5

Ввод: 0 -1 -5 1 5 end
Ожидание: -6
Вывод: -6

Ввод: 10 -5 kk end
Ожидание: ошибк
Вывод: ошибка

Программа 6

Ввод: 1 2 3 4 5 6 end
Ожидание: 6
Вывод: 6

Ввод: 10 -5 kk end
Ожидание: ошибка ввода
Вывод: ошибка

Программа 7

Ввод: 1 2 3 4 5 6 7 end
Ожидание: 0
Вывод: 0

Ввод: 1 0 9 0 -1 0 end
Ожидание: 3
Вывод: 3

Программа 8

Ввод: 0 2 9 3 end
Ожидание: 2
Вывод: 2

Ввод: 0 1 9 1 -10 end
Ожидание: -10
Вывод: -10

Программа 9

Ввод: 0 1 -9 -1 end
Ожидание: -9
Вывод: -9

Программа 10

Ввод: 9 2 1 0
Ожидание: не знакопеременная
Вывод: не знакопеременная

Ввод: 9 -1 2 19 -9 0
Ожидание: знакопеременная
Вывод: знакопеременная

Программа 11

Ввод: 1 2 3 4 5 0
Ожидание: 1 2
Вывод: 1 2

Ввод: 9 -10 9 2 4 1 0
Ожидание: -10 1
Вывод: -10 1