**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе № 3**

**по дисциплине «Программирование»**

Тема: Реализация программы с циклами для вычисления последовательности значений

|  |  |  |
| --- | --- | --- |
| Студент гр. 2309 |  | Савин П.А. |
| Преподаватель |  | Калмычков В.А. |

Санкт-Петербург

2022

**Формулировка задания.**

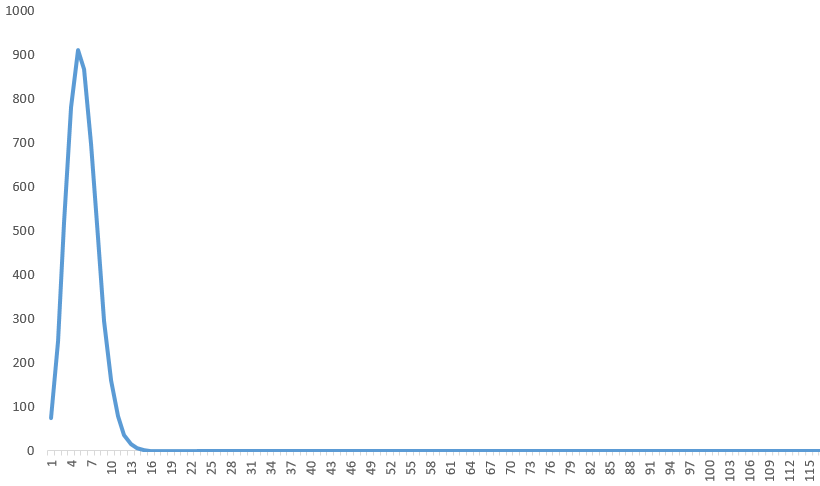
Дана следующая формула для суммы числового ряда:

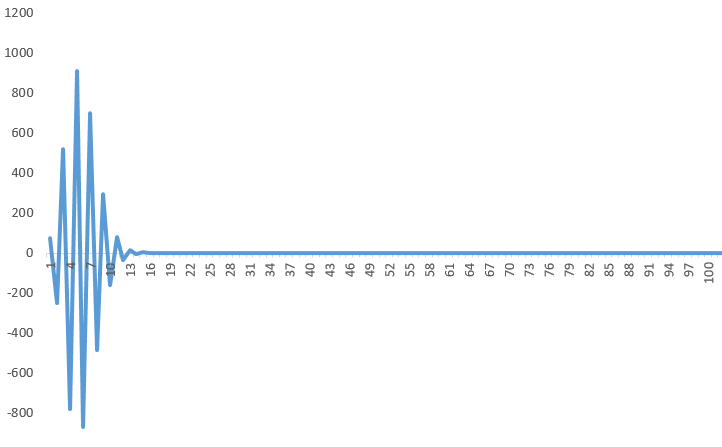
(1)

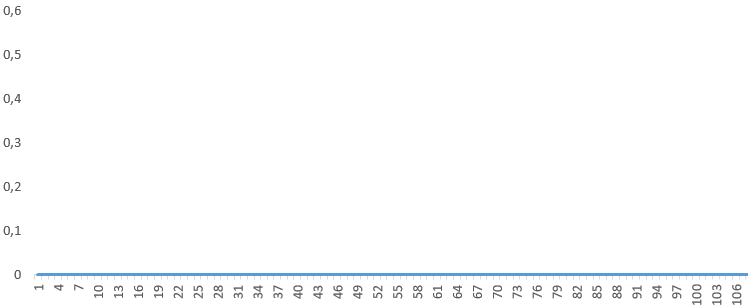
На вход подаются значения ε > 0 и x. Определить минимальное значение i > 0, для которого очередное слагаемое по модулю не превышает ε.

**Анализ задания.**

Несмотря на то, что ε ограничен только снизу, эта задача явно на очень малые значения, следовательно, ε << 1 (много меньше). Допустим, что для этой лабораторной работы значения ε принадлежат интервалу . Очевидно, что – параметрическое значение, определяемое пользователем. Также нужно обозначить, почему именно модуль слагаемого. Для этого посчитаем значения членов ряда и построим графики зависимости:

 Для x > 0

 Для x < 0

 Для x = 0

Соответственно, в зависимости от разных случаев могут быть как отрицательные значения, так и изначально нулевой числовой ряд, именно поэтому в условии стоит дополнение про модуль. Также очевидно, что при x = 0 весь числовой ряд, начиная с первого члена, будет меньше заданного ε.

**Математическая** **постановка.**

Дано:

Найти:

Решение:

Превратить числовой ряд в рекуррентное соотношение:

,

где  *.*

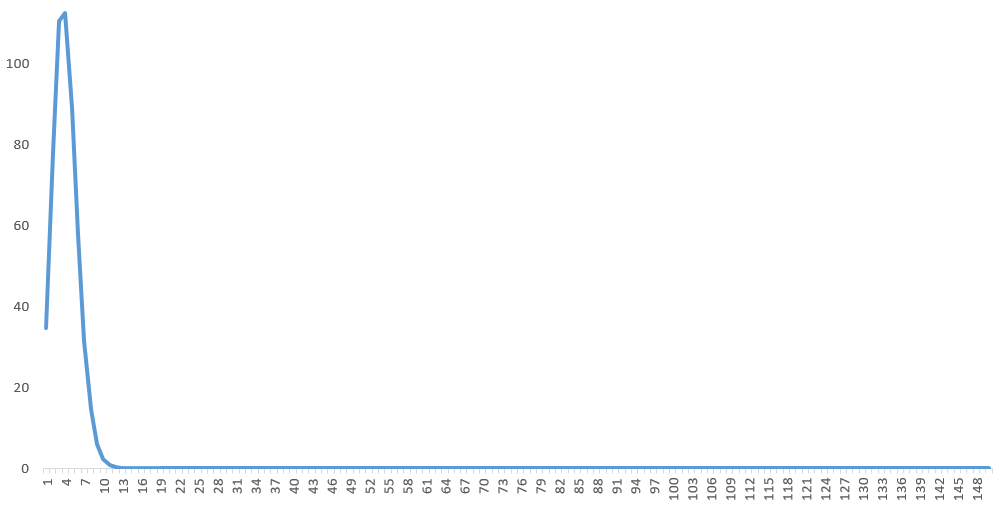
Вычислять до тех пор, пока не найдется такое, что оно меньше по модулю ε. Номер i, соответствующий последнему значению, и будет ответом.

Ответ: – натуральное число.

**Контрольный пример.**

Для контрольного примера были взяты значения и .

График зависимости числового ряда:

**

Ответ на задачу:

**

**Особенности реализации задания на компьютере.**

Помимо вывода ответа на задачу, требуется вывести кол-во вычислений, понадобившихся для поиска ответа (примечание: поскольку компьютер изначально не знает поведение числового ряда, при x = 0 ему все равно нужно вычислить хотя бы одно значение и получить ответ на вопрос).

Более того, поиск ответа нужно оформить в виде неразрывной таблицы следующего вида:

|  |  |  |  |
| --- | --- | --- | --- |
| № | i | ai | si |

где № - порядковый номер, i – номер члена, ai – значение i-ого члена, si – накапливаемая сумма (от i = 2 до текущего i). Соответственно, нужно придумать, как выделить пространство под вывод так, чтобы все числа были в один ряд и они могли выстраиваться, не разрывая границы таблицы появлением новых символов.

Опытным путем выведено, что больше 10000 осмысленных вычислений компьютер не сможет произвести ввиду стремления ряда к бесконечно малым величинам, также и из-за того, что тип double может обрабатывать числа вплоть до 322 порядка. Поэтому под вывод для порядкового номера и номера i выделено ровно 5 символов. Вопрос с выводом ai и si тоже был решен – в библиотеке iomanip есть манипуляторы scientific и showpos, которые в купе приведут все ai и si к общему виду – обязательный знак числа в начале, одна цифра под целую часть, за точкой ровно 6 цифр мантиссы (не больше, не меньше), символ экспоненты, снова знак, но порядка, и 2 или 3 цифры под порядок (меньше быть не может из-за формы scientific, больше - из-за пределов типа double). Получаем 14 символов – в упор для случая, когда порядок трехзначный. Выровняем все элементы по левому краю манипулятором left.

Но это не все задание. Требуется также оформить «логирование» работы программы в отдельном текстовом файле (с дублированием выведенной таблицы, если такое событие произошло), отклонять неверные значения ε и ограничить возможность пользователю вводить ε 3 попытками, по истечении которых программа завершает свою работу и предлагает попробовать еще раз.

**Разработка интерфейса пользователя.**

Макет O1: название программы, автор

Макет O2: «Попыток на ввод осталось: 3/2/1»

Макет O3: «Введите ненулевое положительное эпсилон в интервале (0; 10^-6): \_»

Макет I4: ±ddd.dddd или ±d.de±dd

Макет O5: Уведомление об успешной записи ε (в случае правильного значения)

Макет O6: Уведомление о неверном значении, напоминание о границах правильных значений (в случае неправильного значения)

Макет O7: Уведомление об истечении попыток, просьба перезагрузиться и попробовать снова (в случае 3 неверных попыток)

Макет O8: «Введите x: \_»

Макет I9: ±ddd.dddd или ±d.de±dd

Макет O10: «Таблица членов ряда, больших по модулю эпсилон, и их накапливаемых сумм:»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| № | i | ai | si |

(выводится, если x – ненулевое; под ней выводятся все вычисления, не включая пограничное)

Макет O11: вывод i и кол-ва вычислений

**Описание используемых данных.**

|  |  |  |
| --- | --- | --- |
| Тип | Имя | Назначение |
| double | e, x, a, sum | хранение введенных пользователем значений ε и x, соответственно; хранение значений для ai и si |
| unsigned int | i | счетчик попыток ввода, очередного члена числового ряда и кол-ва вычислений |

**Организация ввода-вывода.**

|  |  |  |  |
| --- | --- | --- | --- |
| Библиотека | Имя | Команда | Назначение |
| iostream | cin | >> | считывание введенных в консоли данных |
| cout | << | вывод данных на консоль |
|  | setlocale | смена локализации программы на русский |
| iomanip |  | setw | выделение пространства под вывод |
|  | left | выравнивание по левому краю |
|  | showpos | задание правила вывода чисел: знак целой части для любого числа |
|  | scientific | задание экспоненциального представления числа (знак, 1 цифра целой части, 6 цифр мантиссы, знак порядка, 2 или 3 цифры порядка) |
| ofstream | file | << | запись в файл через поток |

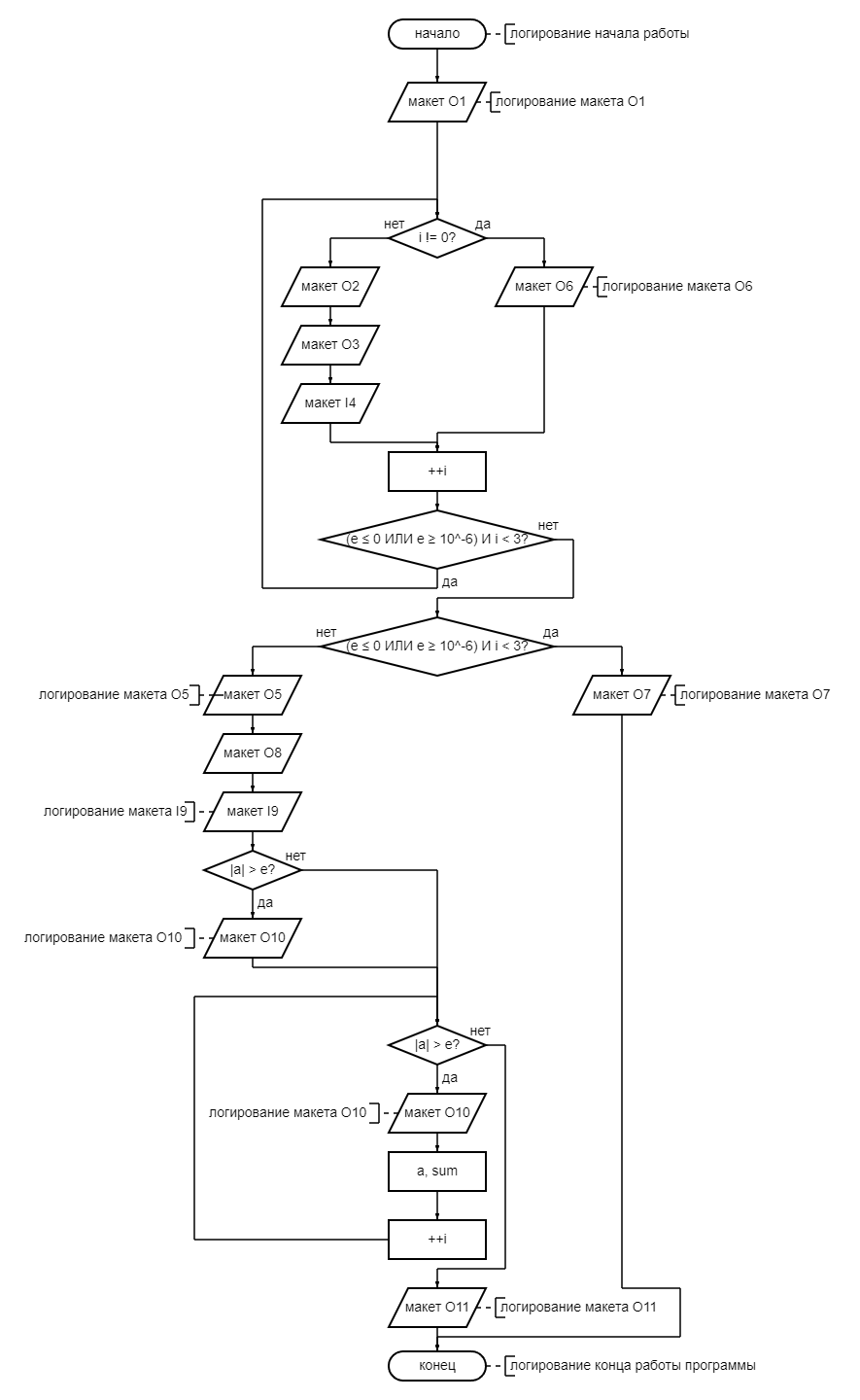
**Представление алгоритма решения задачи.**

1. Подключение необходимых библиотек (iostream, iomanip, fstream)
2. Заход в main, начало программы
3. Установка русской локализации
4. Открытие файла для вывода логов работы программы (или открытие со стиранием, если файл уже был в директории программы)
5. Вывод макета O1, логирование макета O1
6. Объявление переменной e, инициализация переменной i
7. Cделай:
   1. i != 0?
      1. Да: вывод макета O6, логирование макета O6
   2. Вывод макетов O2 и O3
   3. Считывание e по макету I4

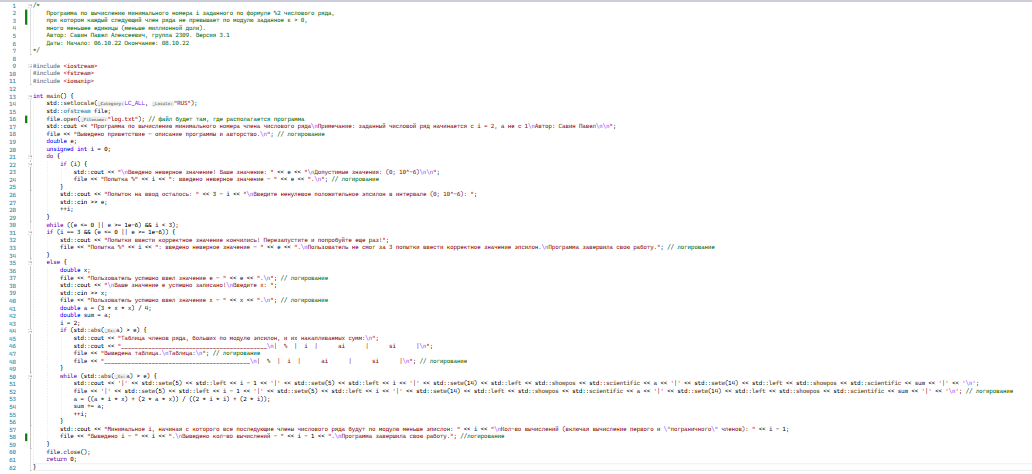
Пока (e <= 0 ИЛИ e >= 10^-6) И i < 3

1. (e <= 0 ИЛИ e >= 10^-6) И i < 3?
   1. Да: вывод макета O7, логирование макета O7
   2. Нет:
      1. Объявление переменной x
      2. Вывод макета O5, логирование макета O5
      3. Вывод макета O8
      4. Считывание x по макету I9
      5. Логирование записи x
      6. Инициализация переменных a и sum, переопределение i
      7. |a| > e?
         1. Да: вывод макета O10, логирование макета O10
      8. Пока |a| > e
         1. Вывод макета O10, логирование макета O10
         2. Перерасчет a и sum
      9. Вывод макета O11, логирование макета O11
2. Закрытие файла
3. Выход с main, завершение программы

**Блок-схема.**

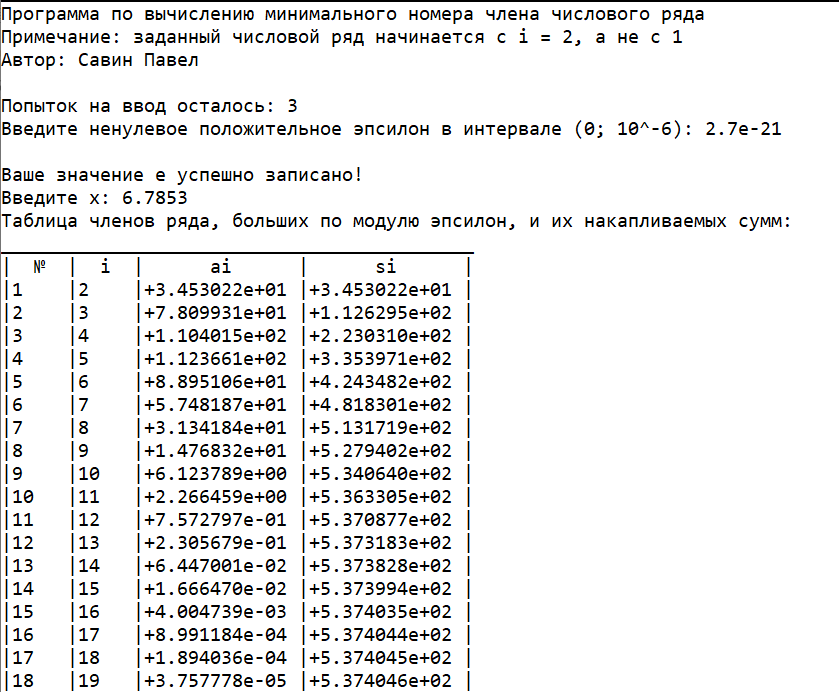


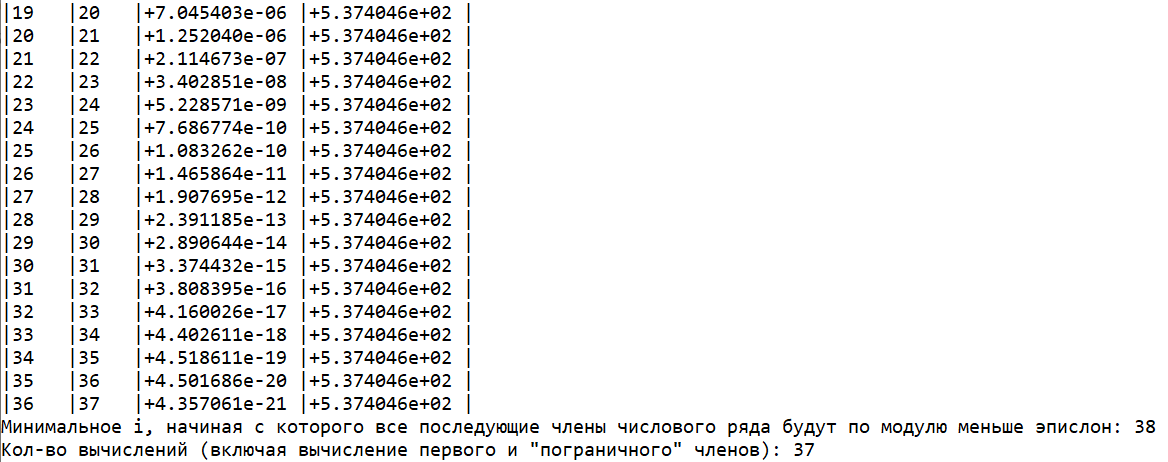
**Текст программы.**

****

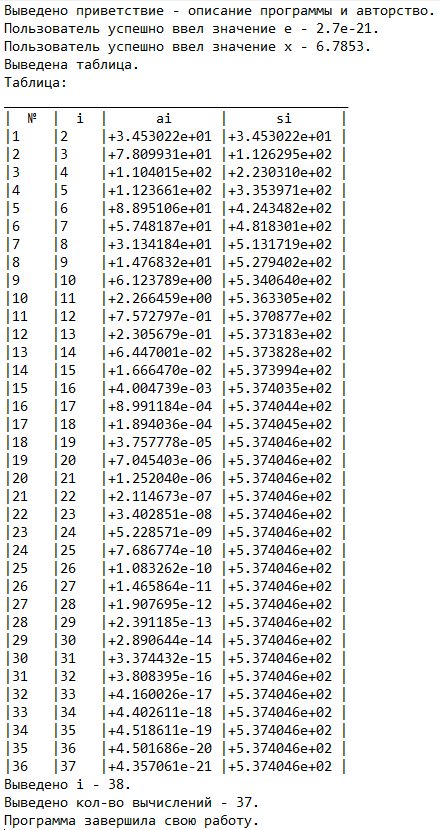
**Результаты работы программы.**

Консоль:

****

****

Лог-файл:



**Вывод о проделанной работе.**

Изучена методика работы с циклами do-while и while-do, получены базовые навыки работы с записью в файл, освоены новые манипуляторы вывода из библиотеки iomanip.