Национальный исследовательский ядерный университет
«МИФИ» Институт интеллектуальных кибернетических
систем

Кафедра №12 «Компьютерные системы и технологии»

ОТЧЕТ

О выполнении лабораторной работы №2

«Вычисление значений числовых рядов и функций с заданной точностью»

Студент: Титов Иван Андреевич.

Преподаватель: Уваров М.П.

Группа: Б23-901

Москва – г .2023

I.Подготовительная часть

Лабораторная работа № 2 «Вычисление значений числовых рядов и функций с заданной точностью»

Необходимо спроектировать и реализовать на языке C **две** программы, позволяющие вычислять значения некоторой заданной функции.

Программа № 1 должна обеспечивать возможность вычисления значения функции при определённых значениях параметров, указанных пользователем. При этом, пользователь должен иметь возможность указать количество членов ряда, которое необходимо использовать при вычислениях.

Ключевым компонентом программы № 1 должна быть некоторая функция, на вход которой передаются значения параметров и количество членов ряда, необходимое для проведения вычислений. Возвращаемым значением для указанной функции должно быть вычисленное значение.

Программа № 2 должна обеспечивать возможность вычисления значения функции при определённых значениях параметров, указанных пользователем. При этом, пользователь должен иметь возможность указать точность, с которой должно быть вычислено значение функции.

Ключевым компонентом программы № 2 должна быть некоторая функция, на вход которой передаются значения параметров и точность, с которой необходимо вычислить результат. В качестве результата функция должна возвращать вычисленное значение и количество членов ряда, которое потребовалось для обеспечения заданной точности (данное значение необходимо вернуть через параметр).

При этом, в обеих программах должно осуществляться вычисление значения функции не только при помощи разложения в ряд, но и с использованием функций стандартной библиотеки.

Кроме того, необходимо научиться:

- оценивать область сходимости ряда — определять диапазон значений входных параметров, при которых ряд сходится;
- оценивать погрешность вычислений — определять количество верных цифр в записи результата.

Вариант №65

Задание

Вычислить значение функции в точке при помощи разложения в ряд:

$$\ln\left(x+\sqrt{1+x^2}\right) = x + \sum_{n=1}^{\infty}(-1)^n \frac{(2n-1)!!}{(2n)!!}\frac{x^{2n+1}}{2n+1}$$

где $|x| \leq 1$.

Тип данных

Число с плавающей точкой двойной точности — double (спецификатор формата: `%lf`).

Цель: создать программу, которая удовлетворяет требованиям.

Требования:

1. Принимать на вход значения следующих типов данных для двух программ соответственно:
   А)Первая программа: x – double, n - unsigned int
   Б)Вторая программа: x – double, eps – double
   Для ввода значений double недопустим ввод других чисел кроме дробных (отрицательных и положительных), единицы (по условию лабораторной работы). Значение «0» для ввода недоступно из логических соображений (при подстановке нуля в формулу функция будет равна нулю).
   Для ввода значений double доступны любые числа (длиной до 4 цифр)(неотрицательные, т.к. речь идёт о вполне натуральной величине, а именно числе членов ряда). А состоят из максимум 4 цифр по той причине, что при больших n программа может работать некорректно (возможно переполнение памяти)(такая степень точности и не требуется)
2. Программа должна выводить значение функции при разложении в ряд с определённой погрешностью относительно математической функции

Примечания к программе

1. Для обеспечения безопасного ввода в программах предусмотрены соответствующие двум используемым переменным (double и unsigned int) функции inputD и inputI. В последнем случае применяется также вспомогательная функция range, сравнивающая заданное значение с константой «uintmax», максимально допустимым значением типа unsigned int.
   Стоит отметить, что безопасный ввод осуществляется по средствам посимвольного ввода (с помощью функции getch()) и работы с массивами. Безопасный ввод также допускает использование клавиш «Esc» (для отмены ввода), «BackSpace» (для удаления последнего введённого символа из массива), «Enter» (для завершения ввода). Все изменения в массиве отображаются динамически с помощью функций printf() и while. Таким образом, программа защищена от ввода некорректным символов и переполнения стека.
2. К сожалению, для unix систем не существует библиотеки <conio.h>. Соответственно невозможно применить функцию getch(). В Линукс существует библиотека со схожей функцией с аналогичным названием getch(), однако в программе она работала некорректно да и автор в ней не до конца разобрался. Поэтому для замены библиотеки <conio.h> был добавлен заголовочный файл («conio.c»), который используется в обеих программах и определяет функцию getch.
   Код заголовочного файла приведён ниже:

```c
#include <stdio.h>
#include <termios.h>
#include <unistd.h>
int getch(void) {
    struct termios oldattr, newattr;
    int ch;
    tcgetattr(STDIN_FILENO, &oldattr);
    newattr = oldattr;
    newattr.c_lflag &= ~(ICANON | ECHO);
    tcsetattr(STDIN_FILENO, TCSANOW, &newattr);
    ch = getchar();
    tcsetattr(STDIN_FILENO, TCSANOW, &oldattr);
    return ch;
}
```

II.Алгоритмическая часть

Первая программа содержит следующие функции: main(), exprow(), inputD(), inputI(), range()

Функции имеют свои специфический, закрепленные за ними операции (что обеспечивает функциям логическую завершенность):

Программа 1:

1. Main(): объявление основных переменных: n (число членов ряда при разложении), x (переменная x в формуле), ref (результат расчёта логарифма функцией из math.h), проверка корректности выполнения функций inputI и inputD, присуждение значений переменным n и x функциями inputI и inputD соответственно.
2. inputI(): безопасный ввод значений типа данных int, их динамический вывод в терминал, проверка длины числа сверкой с «uintmax» функции range ()
3. range (): сверка целого числа с «uintmax» (сначала по длине, потом – по значению)[1]
4. InputD(): безопасный ввод значений типа данных double (отрицательных и положительных, доступны для ввода только значения 1.0 и дробные (до 7 чисел после запятой), их динамический вывод в терминал, проверка длины числа сверкой с «uintmax» функции range ()
5. Exprow() – функция, относящаяся к первой программе, которая считает sum (текущую сумму) циклом for

Программа 2:

1. Main() - объявление основных переменных: i (счётчик необходимых для достижения требуемой погрешности членов ряда), x (заданное число функции), eps (требуемая погрешность), pribl (приближенное значение суммы ряда), проверка корректности выполнения функций inputD, присуждение значений переменным n и x функциями inputD.

[1] Функция range() теряет своё практическое значение, т.к. длина массива в функции inputI ограничена двумя символами, но при потенциальном расширении массива в inputI range становится необходимой

2. InputD(): безопасный ввод значений типа данных double (отрицательных и положительных, доступны для ввода только значения 1.0 и дробные (до 7 чисел после запятой), их динамический вывод в терминал
3. Exprow() - функция, относящаяся ко второй программе, которая считает sum (текущую сумму) циклом do-while, а также обновляет переменную i (количество циклов), подаваемую на вход функции

Следующие переменные имеют тип данных unsigned int: n, i – по следующим причинам:

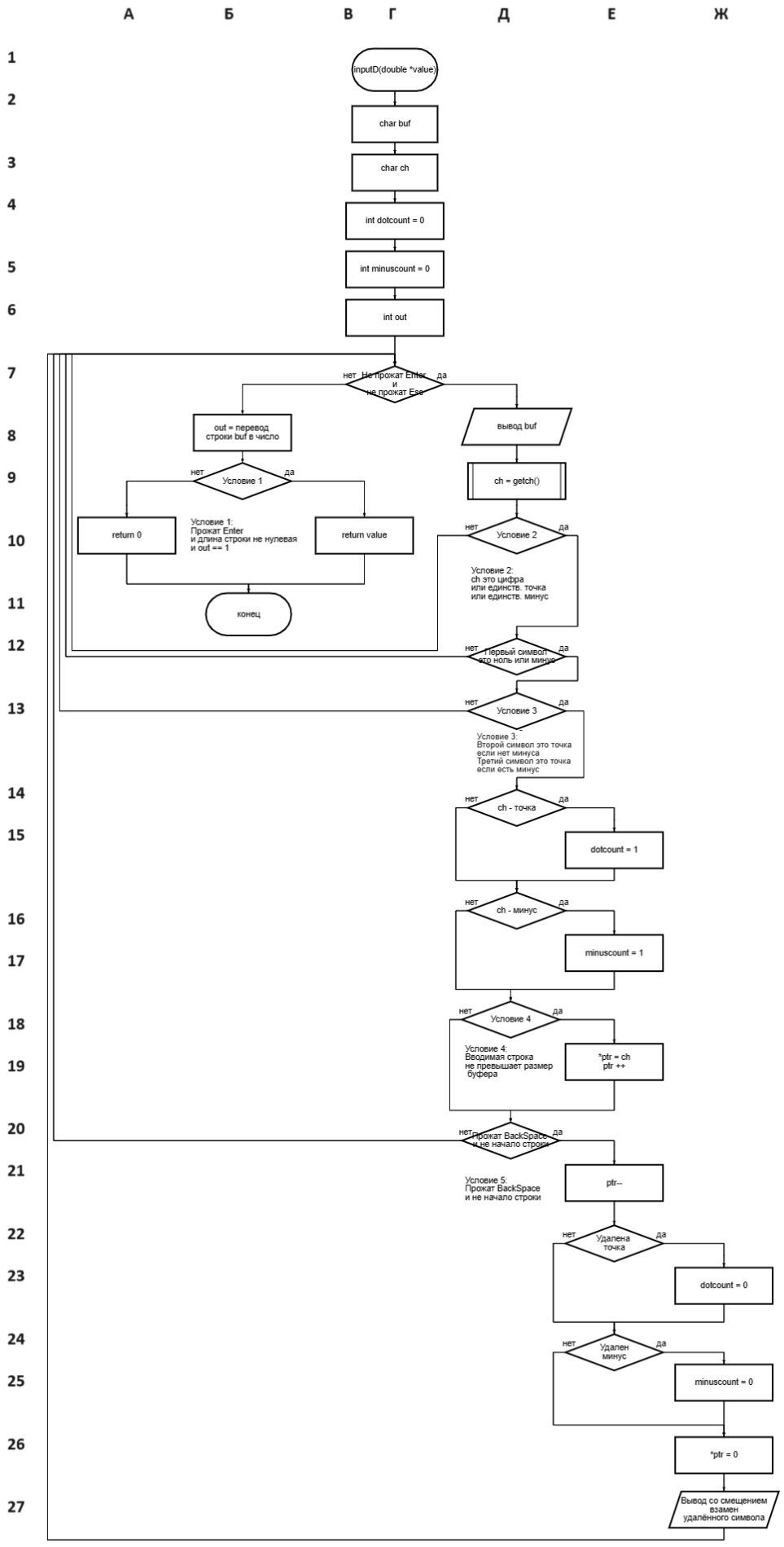
1. Int т.к. требуется совершать операции с целыми числами
2. Программа не предусмотрена для работы с большими числами, значит, нет необходимости в таких альтернативах с большей «вместимостью» (занимаемым местом в памяти), как double, long int и т.п.
3. Unsigned т.к. n и i – натуральные числа

Следующие переменные имеют тип данных char: ch, buf (массивы из функций ввода) – по следующей причине:

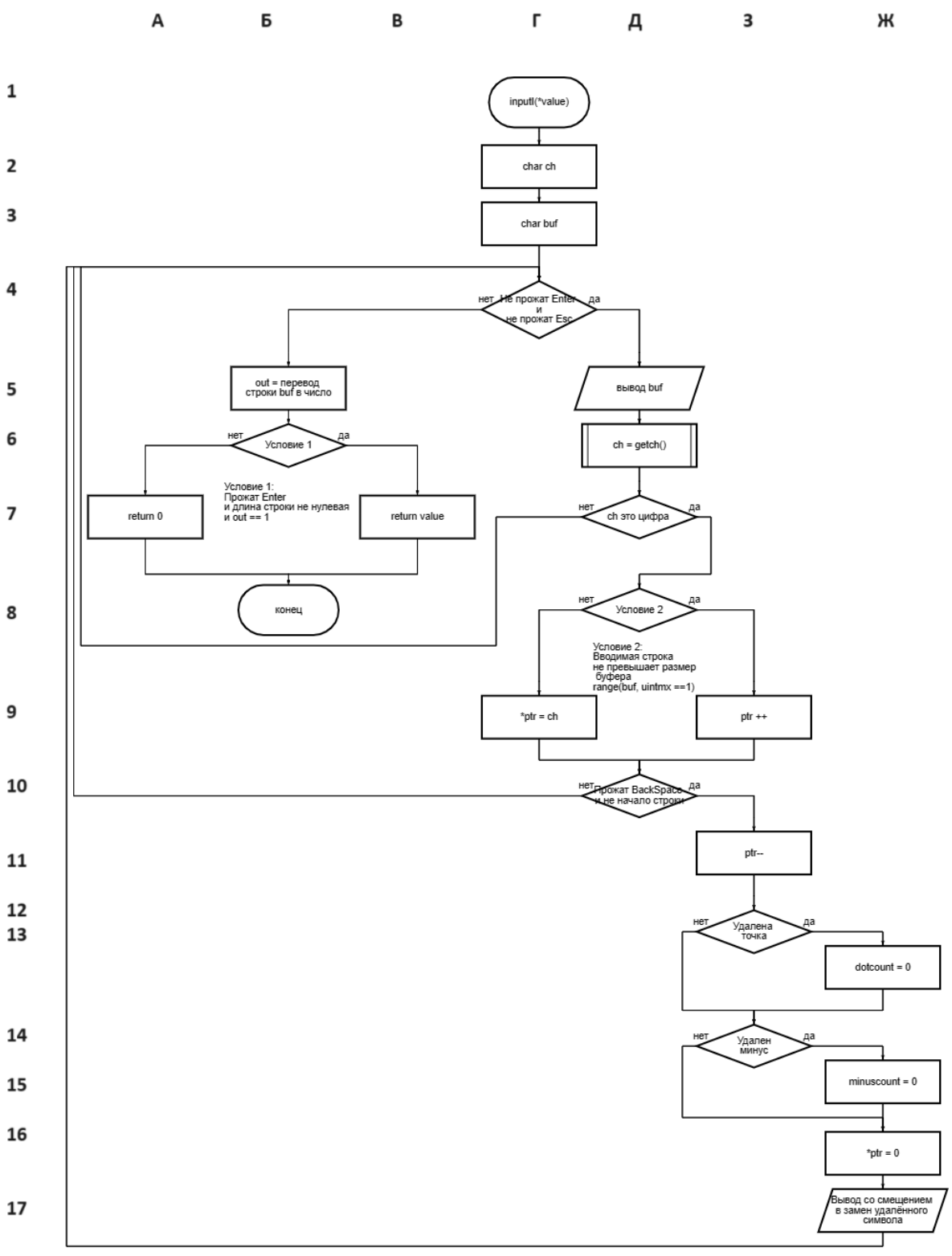
1. Для безопасного ввода запись в буфер введённых посимвольно значений осуществляется контролируемо, поэтому

По условию лабораторной задачи основной тип данных – double. Главные переменные (идущие в выходные значения): pribl, sum имеют тип данных double и используются в программах с точностью до 7 цифр после запятой.

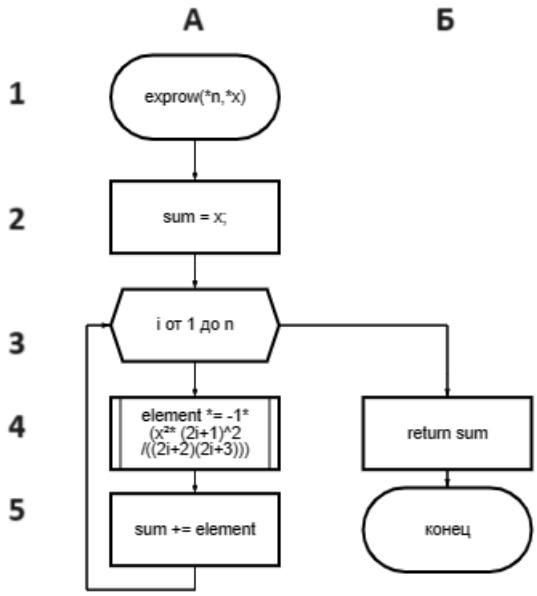
Блок-схема первой программы



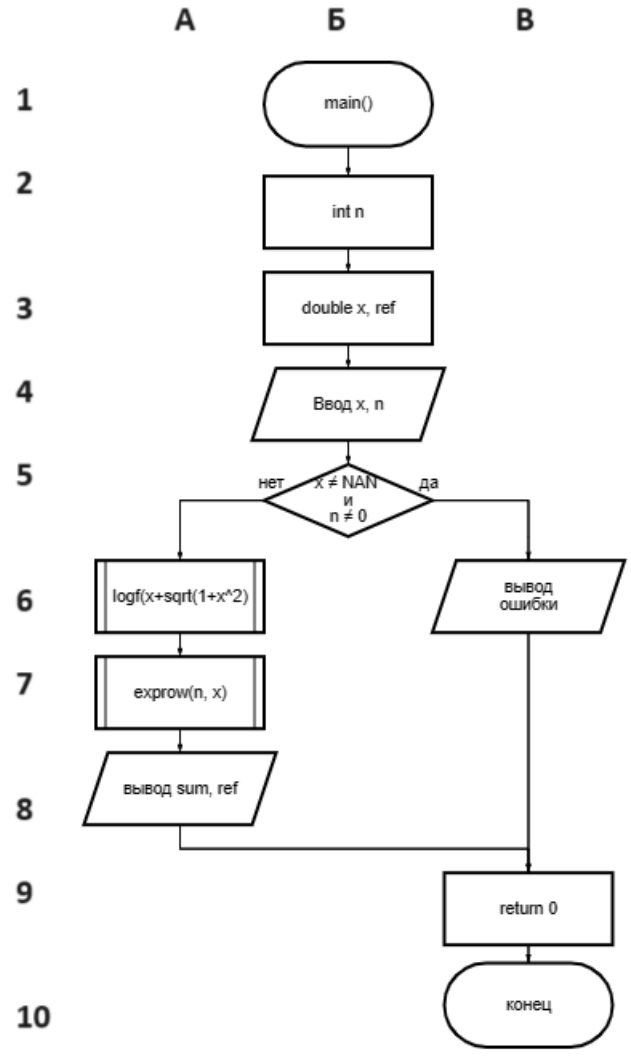
inputD(double *value)



inputI(int *value)

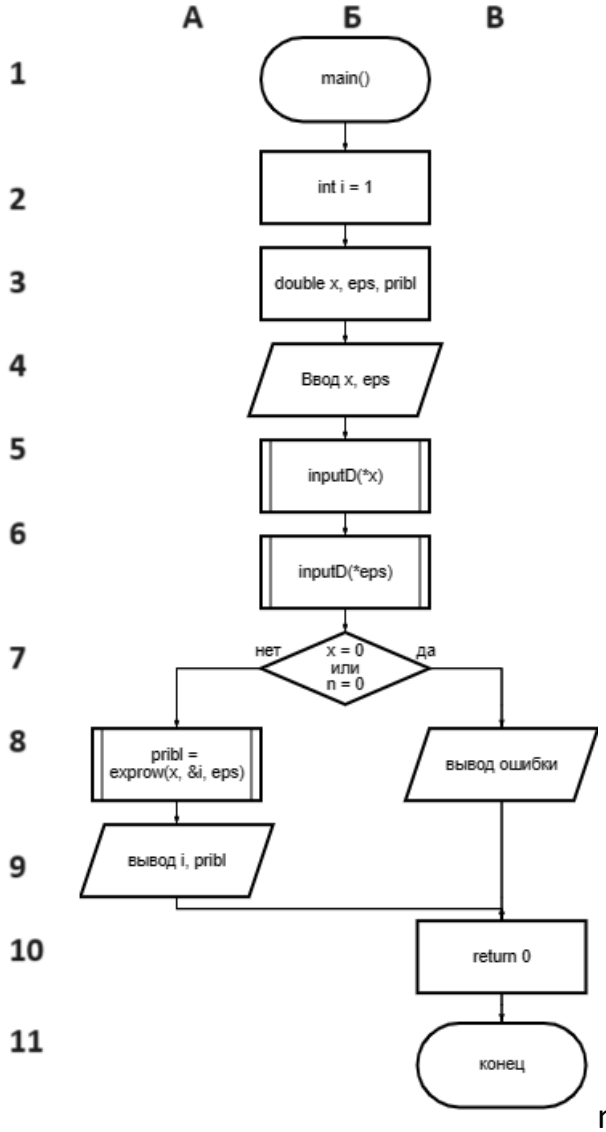


exprow(int n, double *x)

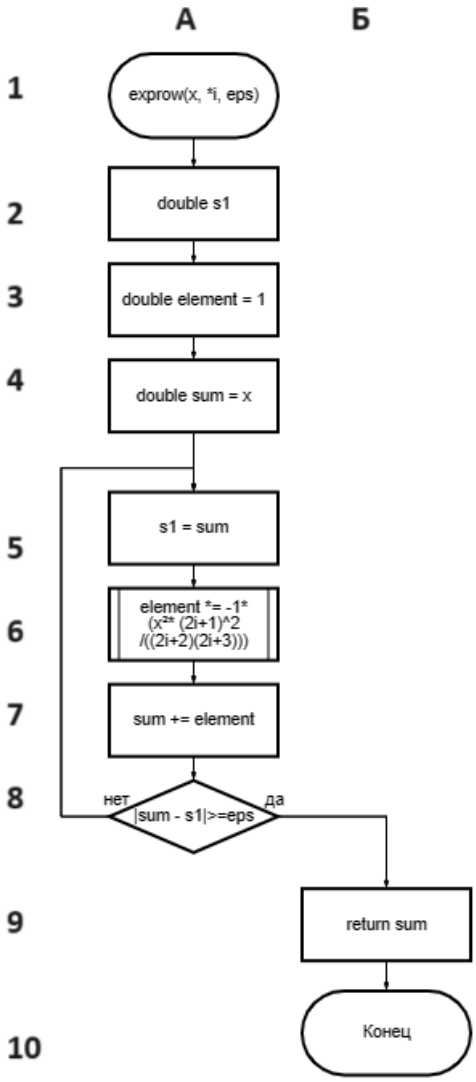


main()

Блок-схема второй программы



main()



exprow(double x, int *i, double x)

(inputD(double *value) аналогична версии из первой программы)

Код первой программы

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "conio.c"
#define Esc 27
#define uintmx "4294967295"
#define Enter 10
#define BackSp 127
#define minus 45
#define dot 46
#define zero 48
#define nine 57
#define one 49
int range(char* buf, const char* max) {
        if (strlen(buf) != strlen(max)) { //сравнение длин строк
                return (strlen(buf) < strlen(max));
        }
        else return (strcmp(max, buf) >= 0); //сравнение строк посимвольно
}
double inputD(double* value) {
    char ch; char buf[12] = {0};
    char* ptr = buf;
    int dotcount = 0, minuscount = 0, out;
    do {
        printf("\r%s", buf); //вывод строки в терминал с левого края
        ch = getch();           //считывание символа
        if (ch == one && ptr == buf){ //если единица, сразу вывод
                *value = 1.0;
                *ptr = ch;  printf("\r%s", buf);
                return *value;
        }
        if ((ch >= zero && ch <= nine) || (ch == dot && dotcount == 0) || (ch
== minus && ptr == buf && minuscount==0)) {
            if ((ptr == buf && ch != minus && ch != zero )||(ptr == buf + 1
&& ch != dot && *(ptr-1) != minus)) continue;
            if (ch == dot) dotcount = 1;
            if (ch == minus) minuscount = 1;
            if (ptr - buf < sizeof(buf) - 1) { // проверка заполненности
буфера с учётом нулевого символа в конце (поэтому -1)
                *ptr = ch;
                ptr++;
            }
        }
        if (ch == BackSp && ptr > buf) {        //прожат ли BackSp
            ptr--;
            if (*ptr == dot) dotcount = 0;      //удаление счётчика точек
(одна т. на число)
            if (*ptr == minus) minuscount = 0;  //удаление счётчика точек
(один минус на число)
            *ptr = 0;                           //очистка символа из буфера
            printf("\b \b");                    //удаление символа из
терминала
        }
    } while (ch != Enter && ch != Esc);
    out = (sscanf(buf, "%lf", value)); //строку в число double
    if (ch == Enter && strlen(buf) && out) {  //непустое, правильно считалось
        return *value;
    }
    else return 0;
}
int inputI(unsigned int *value) { //почти аналогично
        char ch; char buf[4] = {0};
```

```
        char* ptr = buf;
        do {
                printf("\r%s",buf);
                ch = getch();
                if (ch >= zero && ch <= nine) {
                        *ptr = ch;
                        if (range(buf, uintmx) == 1) ptr++;
                        else *ptr = 0;
                }
                if (ch == BackSp && ptr > buf) {
                        *--ptr = 0;
                        printf("\b \b");
                }
        } while (ch != Enter && ch != Esc);
        if (ch == Enter && strlen(buf)) {
                sscanf(buf, "%u", value);
                return *value;
        }
        return 0;
}
double exprow(int n, double x){
        int i;double sum,element,elmold;
        sum = x; element = 1;
        for (i = 1; i<=n; i++){
                element *= -1*((x*x)*(2*i+1)*(2*i+1)/((2*i+2)*(2*i+3)));
                //printf ("%lf\n", element);
                sum += element;
                //printf ("%lf\n", sum);
        }
        return sum;
}
int main(){
        int n;
        double x,ref;
        printf("Enter x\n");
        if (inputD(&x) == 0){
                printf("\nEnter cancelled\n");
                return 0;
        }
        printf("\nYour x is %.9lf\n", x);
        printf("Enter n\n");
        if (inputI(&n) == 0){
                printf("\nEnter cancelled\n");
                return 0;
        }
        printf("\nYour n is %u\n", n);
        ref = logf(x + sqrtf(1+(x*x))); //вычисление функций math.h
        printf ("\nAnswer with expansion in a row: %.9lf\nWith math.h
function: %.9lf\n", exprow(n,x), ref);
        return 0;
}
```

Код второй программы

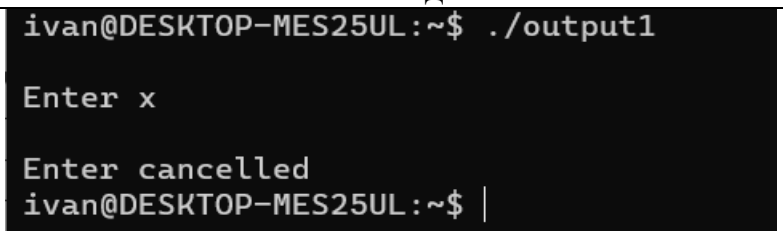
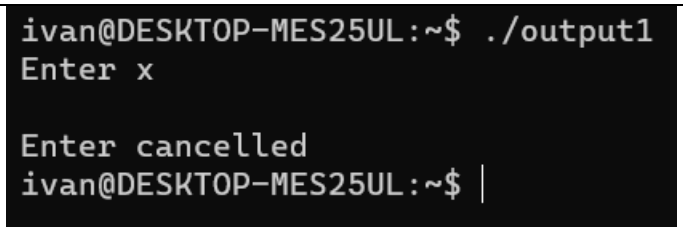
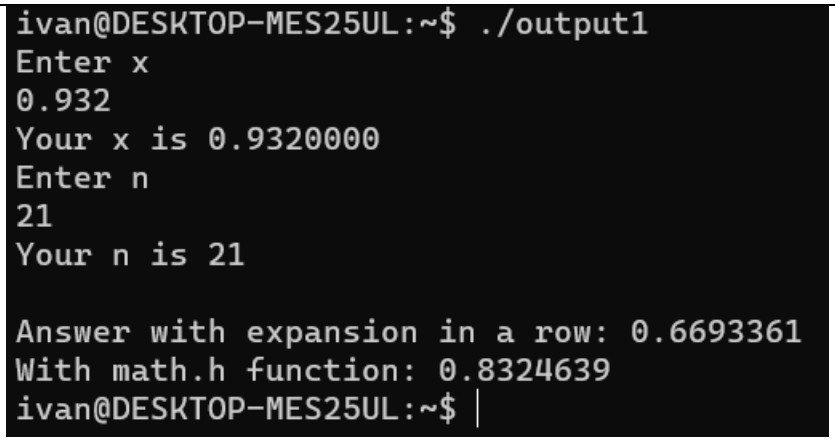
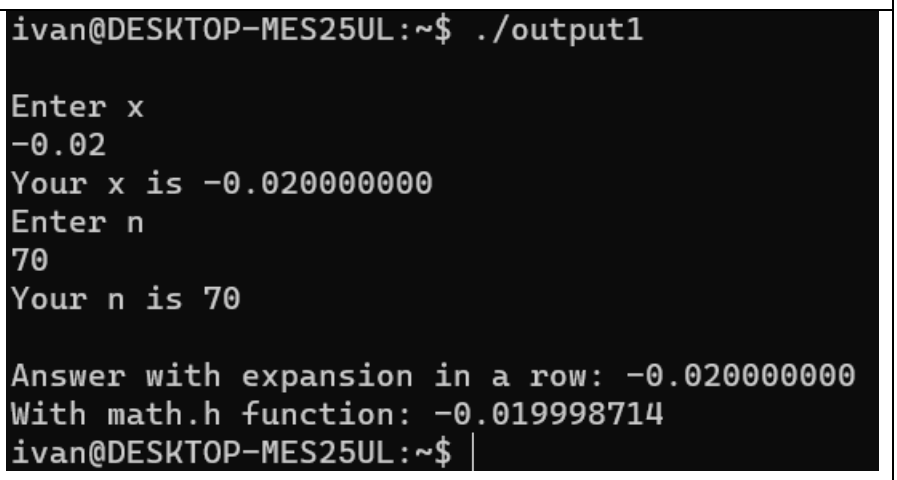
```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include "conio.c"
#define Esc 27
#define Enter 10
#define BackSp 127
#define minus 45
#define dot 46
#define zero 48
#define nine 57
```

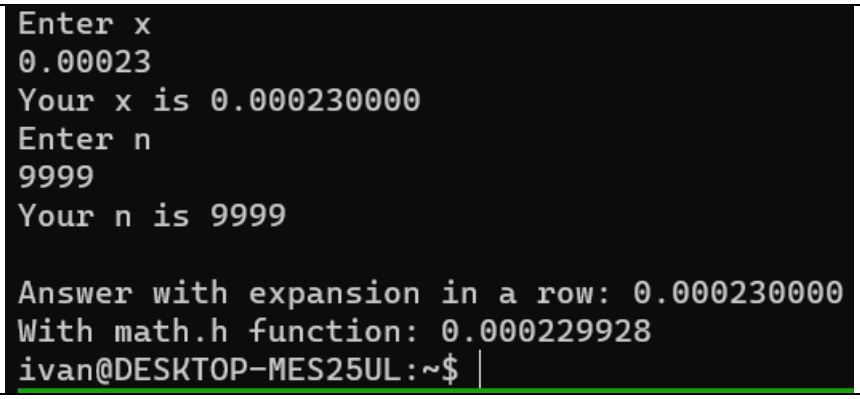
```c
#define one 49
double inputD(double* value) {
    char ch; char buf[12] = {0};
    char* ptr = buf;
    int dotcount = 0, minuscount = 0, out;
    do {
        printf("\r%s", buf); //вывод строки в терминал с левого края
        ch = getch();
        if (ch == one && ptr == buf){
                *value = 1.0;
                *ptr = ch;  printf("\r%s", buf);
                return *value;
        }
        if ((ch >= zero && ch <= nine) || (ch == dot && dotcount == 0) || (ch
== minus && ptr == buf && minuscount==0)) {
            if ((ptr == buf && ch != minus && ch != zero )||(ptr == buf + 1
&& ((ch != dot && *(ptr-1) != minus)||(ch!=zero && *(ptr-1) == minus)))||(ptr
=>
            if (ch == dot) dotcount = 1;
            if (ch == minus) minuscount = 1;
            if (ptr - buf < sizeof(buf) - 1) {
                *ptr = ch;
                ptr++;
            }
        }
        if (ch == BackSp && ptr > buf) {
            ptr--;
            if (*ptr == dot) dotcount = 0;
            if (*ptr == minus) minuscount = 0;
            *ptr = 0;
            printf("\b \b");
        }
    } while (ch != Enter && ch != Esc);
    out = (sscanf(buf, "%lf", value));
    if (ch == Enter && strlen(buf) && out) {
        return *value;
    }
    else return 0;
}
double exprow(double x,unsigned int *i, double eps){
        double element,sum,s1;
        sum = x; element = 1;
        do {
                s1 = sum;
                element *= -1*((x*x)*(2**i+1)*(2**i+1)/((2**i+2)*(2**i+3)));
                //printf("%lf", element);
                *i += 1;
                sum += element;
        } while (fabs(sum-s1)>=eps);
        return sum;
}
int main() {
        unsigned int i = 1;
        double x,eps,pribl;
        printf("Enter x\n");
        if (inputD(&x)==0){
                printf("Enter cancelled\n");
                return 0;
        }
        printf("\nYour x is %.9lf\n", x);
        printf("Enter eps\n");
        if (inputD(&eps)==0){
                printf("Enter cancelled\n");
                return 0;
```

```
        }
        printf("\nYour eps is %.9lf\n", eps);
        pribl = exprow(x, &i, eps);
        printf("\nAnswer with expansion in a row: %.9lf\nWith the following
num of iterations %u\n", pribl, i);
        return 0;
}
```

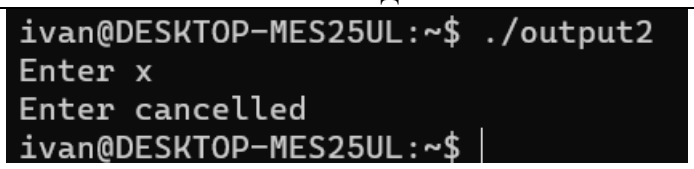
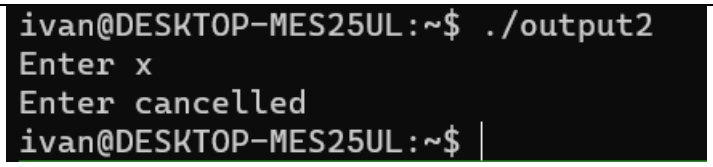
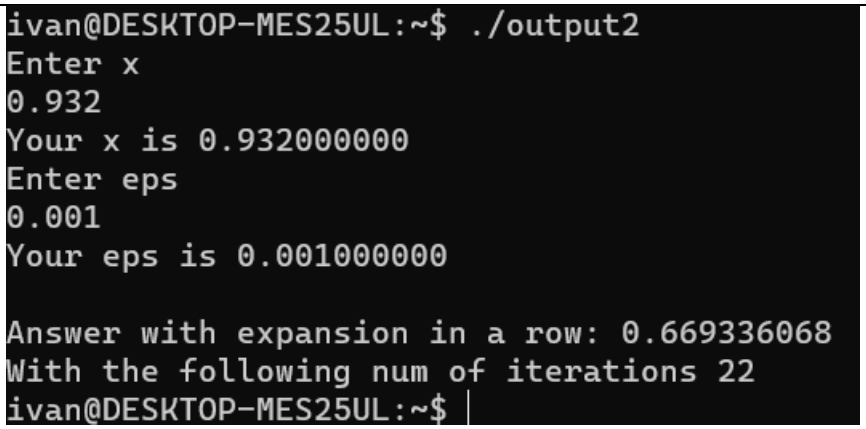
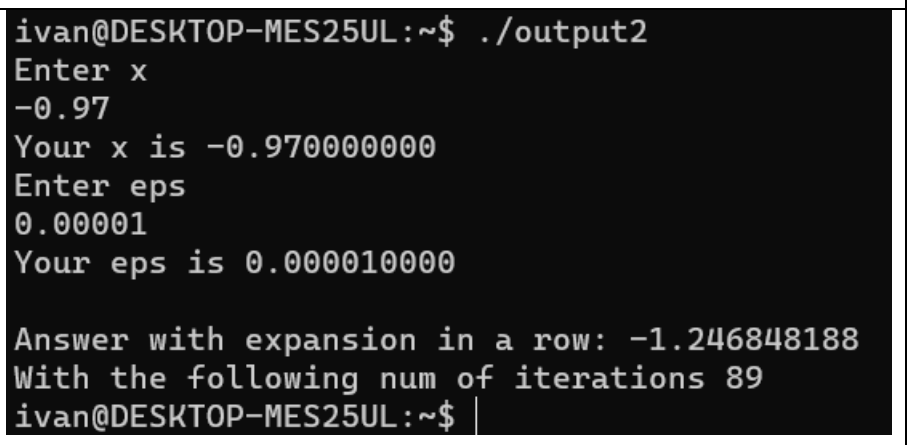
Результаты ввода/вывода

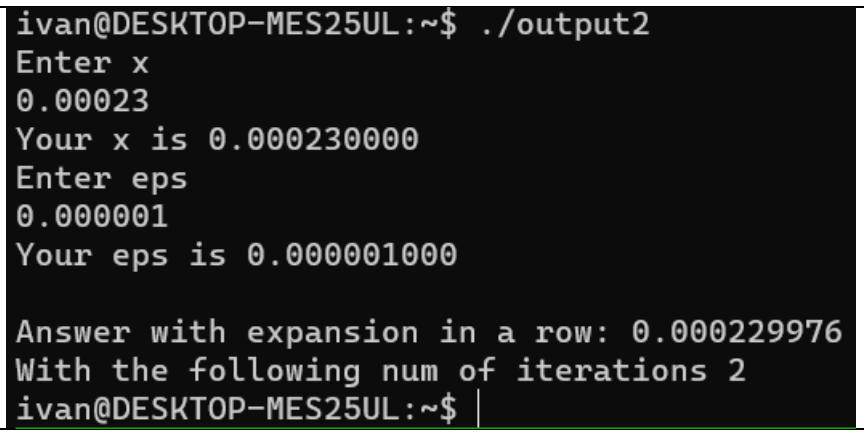
1 программа

Ввод (x/n)	Вывод
Слово/2слово	
323св334/32	
0.93a2c/21	
-0.02/70	

0.00023/9999	

2 программа

Ввод (x/eps)	Вывод
Слово/2слово	
323св334/32	
0.93a2c/0.00c1	
-0.97/0.00001	

0.00023/0.000001	

Выводы

В ходе работы над лабораторной работы №1 были освоены азы языка программирования C: безопасный ввод, примитивная работа со строками/массивами (без анализа задействованной памяти), сравнение между значениями типов данных, работа с нецелыми значениями

Помимо того, немаловажным является приведение интуитивных представлений о построении алгоритмов и соответственно блок-схем к начальному (но, тем не менее, хоть сколько-нибудь структурированному) пониманию этих объектов.

Создание и оформление лабораторной работы – сам по себе процесс очень важный, т.к. это есть по сути второй проект подобного рода, в котором необходимо правильно изложить свои мысли и знания, представить их и впоследствии завершённый продукт.

Конечно, ответы получились не совсем точными. Но хочется отметить, что это не есть проблема программы (исполнения в коде), а заданной функции и способа её представления в задании как разложения в ряд Тейлора. К сожалению, автор не способен исправить ошибки задания, но может лишь оценить их точность.

К сожалению, не всё то, что хотел реализовать автор изначально, было реализовано из-за очевидной сложности реализации некоторых функций, операций, преобразований и т.п. Но сам факт притязаний на овладение знаниями и навыками даёт мотивацию работать над изучением языка программирования и информатики в целом.

$$\log\left(x+\sqrt{1+x^2}\right)=-\sum_{k=1}^{\infty}\frac{(-1)^k\left(-1+x+\sqrt{1+x^2}\right)^k}{k}$$