Пермский национальный исследовательский политехнический университет

Лабораторные работы по

«Информатике»

за 1 семестр

вариант №4

Выполнил:

студент группы РИС-21-2бз (РИС-21-1бзу)

Дерябин Кирилл Николаевич

Проверила:

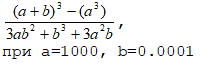
доцент кафедры ИТАС

Полякова О.А.

**Лабораторная работа №1. "Знакомство с Си++. Выполнение программы простой структуры".**

Цель: Знакомство со средой программирования, создание, отладка и выполнение простой программы, содержащей ввод/вывод информации и простейшие вычисления.

1. **Постановка задачи**
2. Вычислить значение выражения при различных вещественных типах данных (float и double). Вычисления следует выполнять с использованием промежуточных переменных. Сравнить и объяснить полученные результаты.



1. Вычислить значения выражений. Объяснить полученные результаты.



**2. Программа решения задания 1**

С float

#include <iostream>  
#include <math.h>

int main()

{

float a = 1000.0f;

float b = 0.0001f;

float chisl = pow(a + b, 3) - pow(a, 3);

float znamn = pow(3.f \* a \* b, 2) + pow(b, 3) + 3.f \* pow(a, 2) \* b;

std::cout << chisl / znamn << std::endl;

return 0;

}

С double

int main()

{

double a = 1000.0;

double b = 0.0001;

double chisl = pow(a + b, 3) - pow(a, 3);

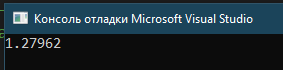
double znamn = pow(3.0 \* a \* b, 2) + pow(b, 3) + 3.0 \* pow(a, 2) \* b;

std::cout << chisl / znamn << std::endl;

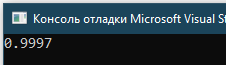
return 0;

}

**3. Результаты работы программы для типа данных float**

****

**4. Результаты работы программы для типа данных double**



**5. Объяснение результатов**

Программа с типом данных float, выполняет вычисления с меньшей точностью, и упускаются числа после запятой, чем с типом данных double. Поэтому ответы в программах отличаются.

**6. Программа решения задания 2**

#include <iostream>

int main()

{

int inn, inm, n, m, result;

std::cin >> inn >> inm;

n = inn;

m = inm;

result = n++ \* m;

std::cout << "1) " << result << std::endl;

n = inn;

m = inm;

result = n++ < m;

std::cout << "2) " << result << std::endl;

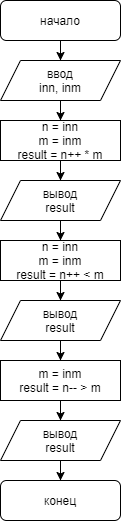
m = inm;

result = m-- > m;

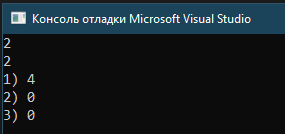
std::cout << "3) " << result << std::endl;

return 0;

}



**7. Результаты работы программы**

****

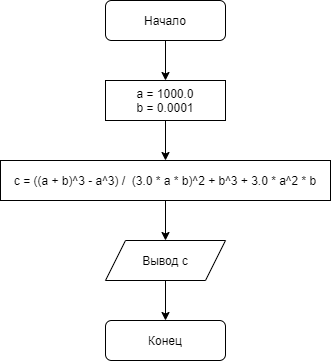
**8. Объяснение результатов**

В данной программе я сделал ввод чисел с m и n с консоли. Переменные inn и inm хранят исходные числа ввода. Переменные n и m требуются для того чтобы производить операции инкремента/декремента не модифицируя исходные данные. Значения m и n обновляются исходными данными перед выполнением указанного выражения. Переменная result хранит ответ выражения для вывода его в консоль.

Первое выражение n++ \* m равняется ответу 4, потому что постфиксный инкремент не влияет на результат умножения, так как оно выполняется после.

Второе выражение n++ < m равняется 0 (ложь), так как n после инкремента равняется 3, а m не изменилось. Следовательно, n больше m, что означает ложность выражения n < m.

Третье выражение m-- > m ложно, потому что (m – 1) == (m – 1).



**Лабораторная работа №2.** **"Использование основных операторов языка Си"**

**1. Постановка задачи**

Используя оператор цикла, найти сумму элементов, указанных в конкретном варианте. Результат напечатать, снабдив соответствующим заголовком.

**2. Текст программы.**

#include <iostream>

int main()

{

setlocale(LC\_ALL, "ru"); //меняю локаль для вывода русских символов в консоли

int j = 0;

for (int i = 21; i < 100; i++)

if (!(i % 3))

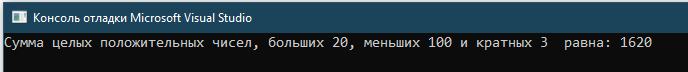
j += i;

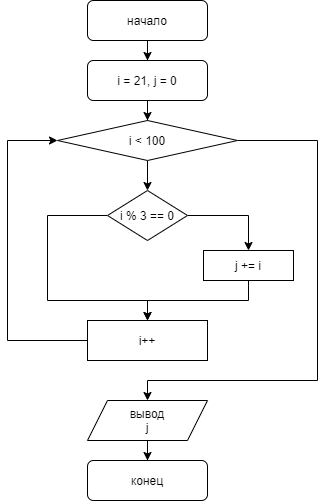
std::cout << "Сумма целых положительных чисел, больших 20, меньших 100 и кратных 3 равна: " << j << std::endl;

return 0;

}

**3. Результат решения конкретного варианта.**





**Анализ выполнения работы**

Установка русского языка в консоли выполняется с помощью функции setlocale, где первым параметром указываются все локали, а вторым параметром указывается язык. Кратность трем проверяется остатком от деления на 3, если остатка нет, значит число кратно. Кратные трем числа складываются в переменной j и выводятся в консоль по завершению цикла.

# Лабораторная работа №3 "Вычисление функций с использованием их разложения в степенной ряд"

**Цель:** Практика в организации итерационных и арифметических циклов.

**1. Постановка задачи.**

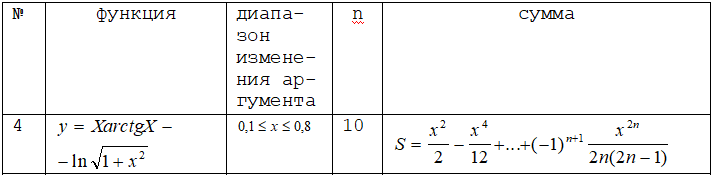
Для х изменяющегося от a до b с шагом (b-a)/k, где (k=10), вычислить функцию f(x), используя ее разложение в степенной ряд в двух случаях:

а) для заданного n;

б) для заданной точности E (E=0.0001).

Для сравнения найти точное значение функции.

**2. Вариант задания.**

****

**3. Математическая модель**

Член суммы представлен в виде двух сомножителей, один из которых вычисляется по рекуррентному соотношению, а другой непосредственно an=φ(x, n)\*сn(x,n), где сn=cn-1ψ(x,n).



**4. Программа.**

#include <iostream>

#include <math.h>

#define EPSILON 0.0001

int main()

{

int n = 10;

int k = 10;

float a = 0.1f;

float b = 0.8f;

float h = (b - a) / k;

float an, p, prev, SN, SE, Y;

for (float x = a; x <= b + h; x += h) {

p = x;

SN = 0.f;

int i;

for (i = 1; i <= n; i++) {

float tmp = 2.f \* i;

p \*= pow(-1.f, i + 1.f) \* ((pow(x, tmp)) / (tmp\*(tmp - 1.f)));

SN += p;

}

p = x;

an = 0.f;

prev = 1.f;

SE = 0.f;

for (i = 1; fabs(prev - an) > EPSILON; i++) {

p \*= x;

prev = an;

float tmp = 2.f \* i;

an = p \* pow(-1.f, i + 1.f) \* (pow(x, tmp) / (tmp \* (tmp - 1.f)));

SE += an;

}

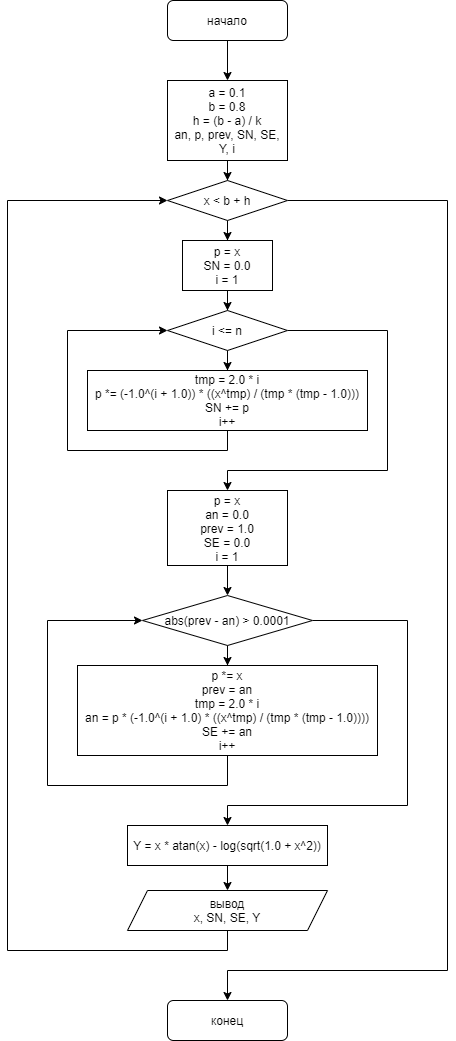
Y = x \* atan(x) - log(sqrt(1.f + x \* x));

std::cout << "X=" << x << "\tSN=" << SN << "\tSE=" << SE << "\tY=" << Y << std::endl;

}

return 0;

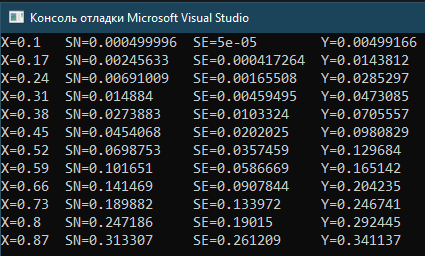
}



**Анализ выполнения работы**

Основная часть программы организованна в виде трёх циклов: основной цикл for, в который вложены for и while, который занимаются расчётами формул. Вывод данных в консоль производится с помощью iostream и оператора <<. Std::endl вставляет в вывод перенос каретки.

**5. Полученные результаты.**

****

# Лабораторная работа № 4 "Работа с одномерными массивами"

**Цель:** Получение навыков обработки одномерных массивов.

**1. Вариант задания.**

1) Сформировать одномерный массив целых чисел, используя датчик случайных чисел.

2) Распечатать полученный массив.

3) Удалить элементы, индексы которых кратны 3.

4) Добавить после каждого отрицательного элемента массива элемент со значением

| M[ i-1 ]+1|.

5) Распечатать полученный массив.

**2. Текст программы.**

#include <iostream>

#include <time.h>

int random()

{

return (rand() % 7) ? rand() : -rand();

}

int find\_element\_in\_array(int \*parr, int newsize, int element)

{

for (int i = 0; i < newsize; i++)

if (parr[i] == element)

return i;

return -1;

}

bool remove\_element\_from\_array(int \*parr, int \*pnewsize, int size, int element)

{

int ind = find\_element\_in\_array(parr, \*pnewsize, element);

if (ind == -1)

return false;

for (int i = ind; i < size - 1; i++)

parr[i] = parr[i + 1];

(\*pnewsize)--;

return true;

}

void print\_array(int \*p, int size)

{

for (int i = 0; i < size; i++)

std::cout << p[i] << " ";

std::cout << std::endl;

}

int main()

{

srand((unsigned int)time(NULL));

int i;

int array[10];

const int array\_size = sizeof(array) / sizeof(array[0]);

int newsize = array\_size;

//Сформировать одномерный массив целых чисел, используя датчик случайных чисел

for (i = 0; i < array\_size; i++)

array[i] = random();

//Распечатать полученный массив

print\_array(array, newsize);

//Удалить элементы, индексы которых кратны 3

for (i = 0; i < array\_size; i++)

if (!(array[i] % 3))

remove\_element\_from\_array(array, &newsize, array\_size, array[i]);

//Добавить после каждого отрицательного элемента массива элемент со значением |M[i - 1] + 1|

#define abs(n) ((n < 0) ? -n : n)

for (i = 0; i < newsize - 1; i++)

if (array[i] < 0)

array[i + 1] = abs(array[i - 1] + 1);

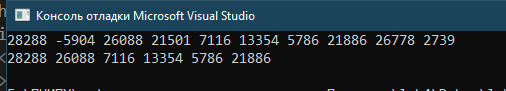
#undef abs

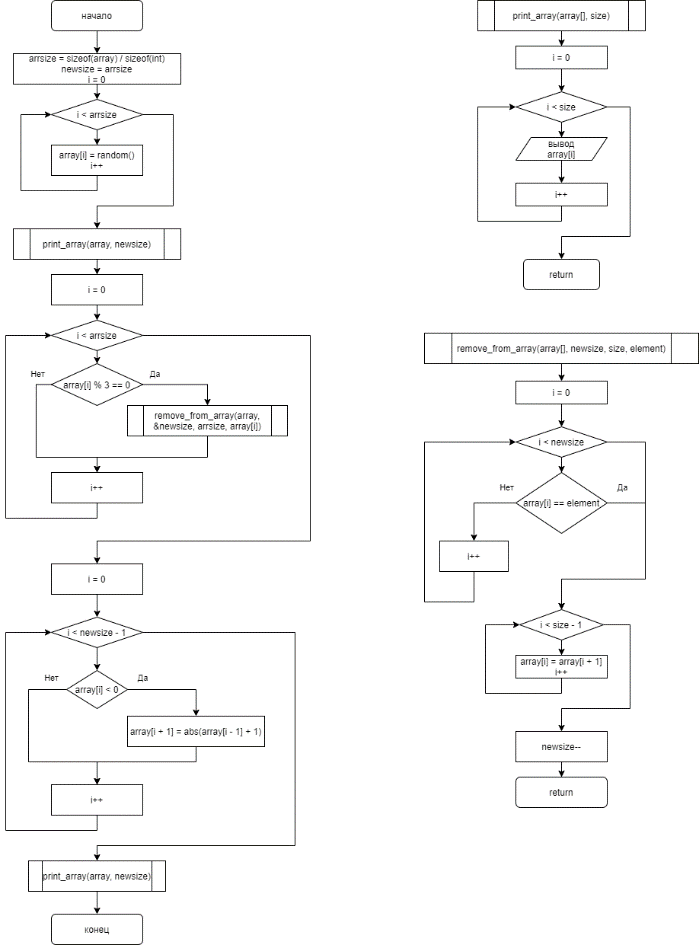
print\_array(array, newsize);

return 0;

}

**3. Результат решения конкретного варианта.**

****



**Анализ выполнения работы**

В данной работе был использован генератор случайных чисел, для него была использована функция rand(), которая выдает следующее случайное число в последовательности. Чтобы изменить последовательность, требуется вызвать srand с указанием своего числа, как правило времени или количества тактов.

# Лабораторная работа №5 "Функции и массивы"

**Цель:** Организовать обработку массивов с использованием функций, научиться передавать массивы как параметры функций.

**1. Постановка задачи.**

Используя функции, решить указанную в варианте задачу. Массив должен передаваться в функцию как параметр.

**2. Вариант задания**

Написать функцию для удаления строки из двумерного массива. Оставшиеся строки должны быть расположены плотно, недостающие элементы заменяются 0. С помощью разработанных функций исключить из массива строки с номерами от А до В.

**3. Текст программы.**

#include <iostream>

#define STRING\_LENGTH 32

bool fill\_string(char strings[][STRING\_LENGTH], int num\_of\_strings, int index)

{

if (index < 0 || index >= num\_of\_strings)

return false;

for (int i = 0; i < STRING\_LENGTH-1; i++)

strings[index][i] = '0';

return true;

}

bool delete\_string(char strings[][STRING\_LENGTH], int num\_of\_strings, const char \*string)

{

auto find\_string = [strings, num\_of\_strings](const char \*string) -> int {

for (int i = 0; i < num\_of\_strings; i++)

if (!strcmp(strings[i], string))

return i;

return -1;

};

int index;

if ((index = find\_string(string)) == -1) {

std::cout << "String not found!" << std::endl;

return false;

}

return fill\_string(strings, num\_of\_strings, index);

}

void print\_strings(char strs[][STRING\_LENGTH], int number\_of\_strings)

{

for (int i = 0; i < number\_of\_strings; i++)

std::cout << strs[i] << std::endl;

}

int main()

{

char strs[][STRING\_LENGTH] = {

"string0",

"string1",

"string2",

"string3",

"string4",

"string5",

"string6",

"string7",

"string8",

"string9"

};

std::cout << "Original strings array:\n";

const int number\_of\_strings = sizeof(strs) / STRING\_LENGTH;

print\_strings(strs, number\_of\_strings);

std::cout << "\n\nDeleting string 'string5':\n";

if (!delete\_string(strs, number\_of\_strings, "string5")) {

std::cout << "failed delete string!" << std::endl;

return 1;

}

print\_strings(strs, number\_of\_strings);

std::cout << "\n\nDeleting strings 1-5:\n";

int A = 1;

int B = 5;

for (; A < B; A++)

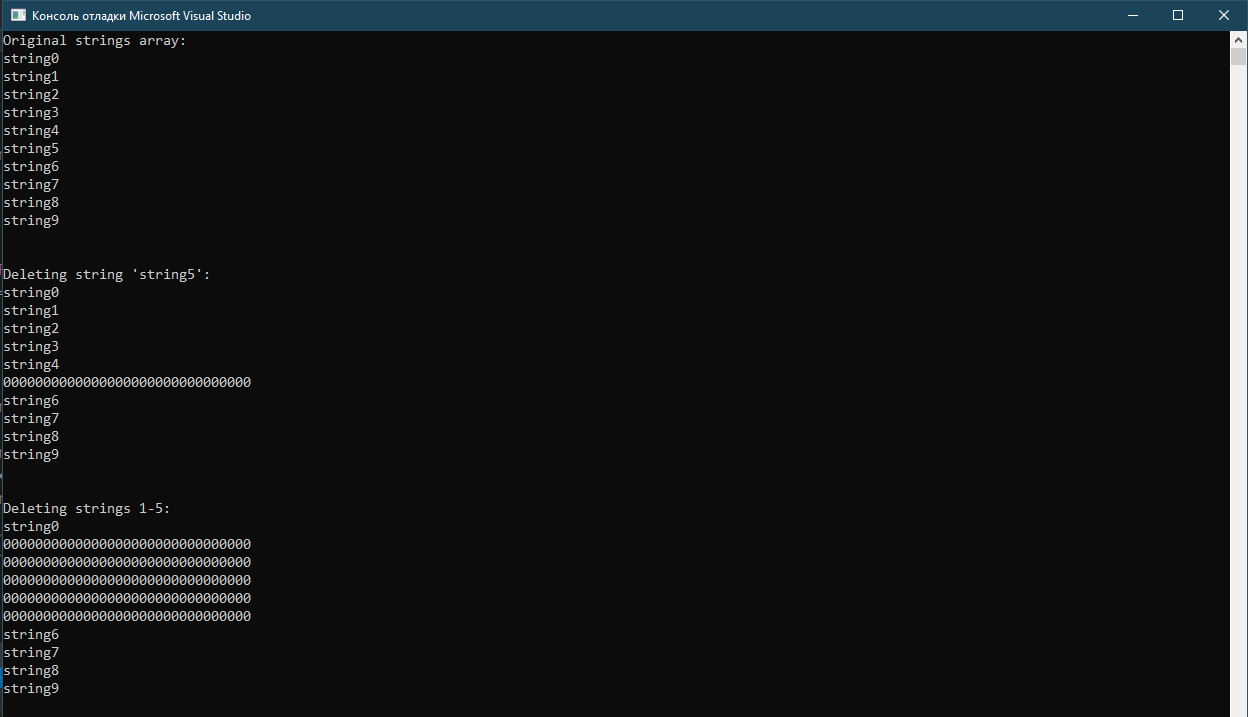
fill\_string(strs, number\_of\_strings, A);

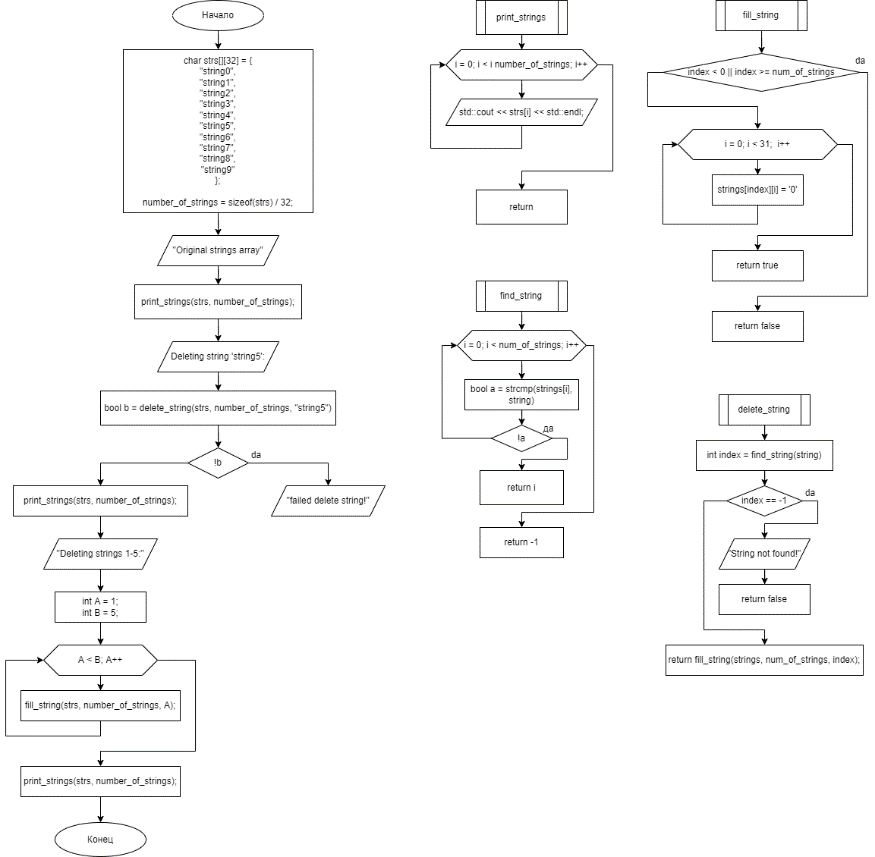
print\_strings(strs, number\_of\_strings);

return 0;

}

**4. Результат решения конкретного варианта.**



****

# Анализ выполнения работы

Для сравнения строк была использована функция из заголовочного файла string.h strcmp. Первым параметром передается строка которую требуется сравнить, а вторым параметром передается строка с которой нужно сравнивать. В результате функция выдает число 0, если обе строки полностью равны. Строки были объявлены в глобальном изначально инициализированном статическом массиве.

# Лабораторная работа № 6 "Строки"

**Цель:** Изучение символьных и строковых переменных и способов их обработки в языке Си.

**1. Постановка задачи для конкретного варианта.**

Задана строка, состоящая из символов. Символы объединяются в слова. Слова друг от друга отделяются одним или несколькими пробелами. В конце текста ставится точка. Текст содержит не более 255 символов. Выполнить ввод строки, используя функцию Gets(s) и обработку строки в соответствии со своим вариантом.

**2. Исходные данные.**

Напечатать все слова, которые содержат по одной цифре.

**3. Текст программы.**

// lab6.cpp

#include <stdio.h>

#include <string.h>

#include <ctype.h> //isdigit

const int max\_strings = 8;

bool digit\_contain(const char \*str)

{

for (int i = 0; i < strlen(str); i++)

if (isdigit(str[i]))

return true;

return false;

}

int main()

{

char buffers[max\_strings][32];

int i;

for (i = 0; i < max\_strings; i++) {

printf("Enter string ( %d / %d )\n", i + 1, max\_strings);

gets\_s(buffers[i], sizeof(buffers) / max\_strings);

}

printf("\nStrings contain numbers:\n");

for (i = 0; i < max\_strings; i++)

if (digit\_contain(buffers[i]))

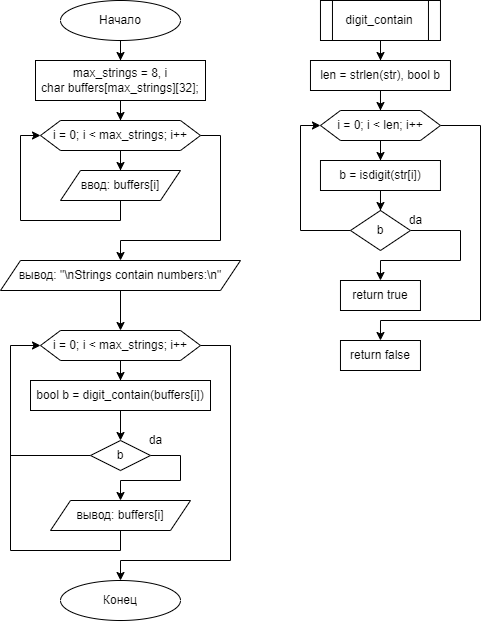
printf("%d: %s\n", i + 1, buffers[i]);

return 0;

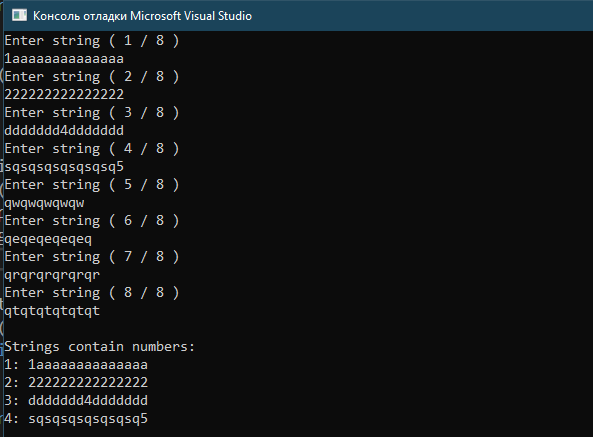
}

# Анализ выполнения работы

Для определения, содержится ли в строке число, была использована функция digit\_contain. Внутри функции имеется цикл, который пробегается по всем символам строки до длинны строки, и если символ является числом, функция возвращает true. Если строка не содержит числа, цикл завершается и функция возвращает false. Длина строки определяется с помощью функции strlen.



**4. Результаты выполнения программы.**

****

**Лабораторная работа №7. "Решение нелинейных уравнений".**

Цель: изучить 3 метода: половинного деления, итераций, Ньютона.

**Постановка задачи**

Известно, что корень есть в интервале [a,b] => [0.0, 1.5]. Нелинейная функция: x^5-cos(x^2)=0

**Текст программы «Метод половинного деления»**

#include <iostream>

#define EPSILON 0.0001

#define RANGE 9999

#define LEFT\_RANGE 0.0

#define RIGHT\_RANGE 1.5

#define MyFunc(x) (pow(x, 5) - cos(pow(x, 2)))

int main()

{

double a = LEFT\_RANGE;

double b = RIGHT\_RANGE;

double tempa = MyFunc(a);

double tempb = MyFunc(b);

int i = 0;

double midx, funcmidx;

while (((b - a) > EPSILON) && (i < RANGE)) {

midx = (a + b) / 2;

funcmidx = MyFunc(midx);

if ((tempa \* funcmidx) <= 0) {

b = midx;

tempb = funcmidx;

}

else {

a = midx;

tempa = funcmidx;

}

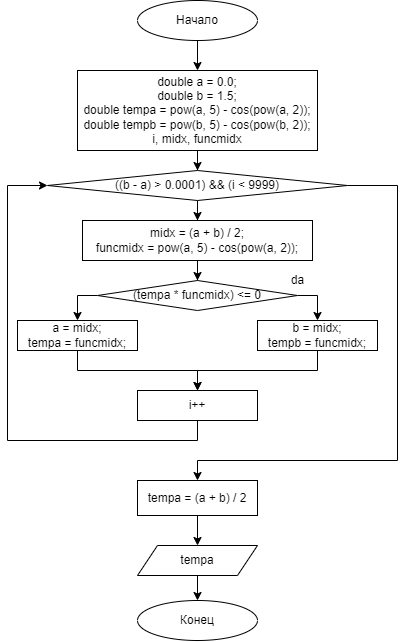
i++;

}

std::cout << "X = " << (a + b) / 2 << std::endl;

return 0;

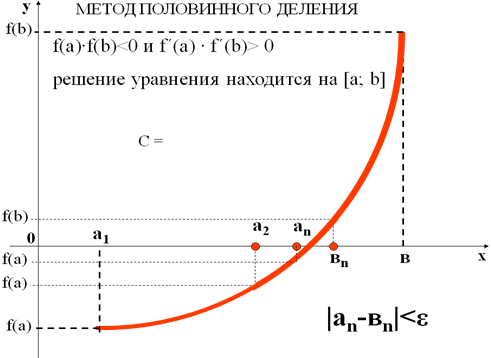
}

****

**Результат работы программы «Метод половинного деления»**



**Геометрическая интерпретация «Метод половинного деления»**

****

**Текст программы «Метод** **итераций»**

#include <iostream>

#define EPSILON 0.0001

#define RANGE 99999

#define LEFT\_RANGE 0.0

#define RIGHT\_RANGE 1.5

#define MyFunc(x) (pow(x, 5) - cos(pow(x, 2)))

#define MyFunc2(x) (pow(x, 5) - (5.0 \* pow(x, 4)) + (10.0 \* pow(x, 3)) - (10.0 \* (x\*x)) + (5.0 \* x) - 1.0 - (cos((x\*x) - 2.0 \* x + 1.0)))

int main()

{

float x0 = LEFT\_RANGE, x1 = RIGHT\_RANGE;

for (int i = 0; i < RANGE; i++) {

x1 = MyFunc2(x0);

float cc = fabs(x1 - x0);

//std::cout << "cc = " << cc << std::endl;

if (cc > EPSILON)

break;

x0 = x1;

}

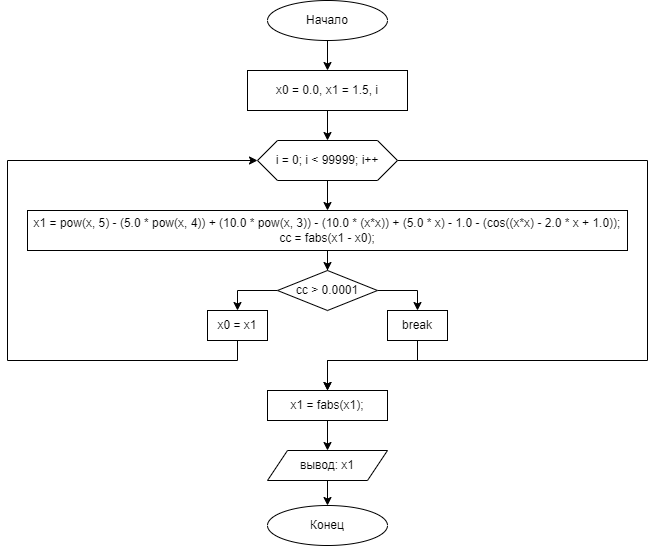
std::cout << "X = " << fabs(x1) << std::endl;

return 0;

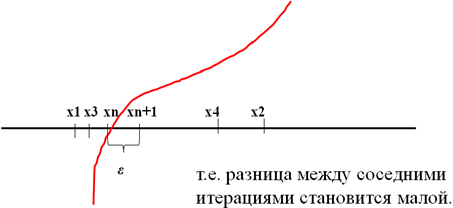
}

**Результат работы программы «Метод итераций»**



****

**Геометрическая интерпретация «****Метод итераций»**



**Текст программы «Метод** **Ньютона»**

#include <iostream>

#define EPSILON 0.0001

#define RANGE 10000000

#define LEFT\_RANGE 0.0

#define RIGHT\_RANGE 1.5

#define Func0(x) (pow(x, 5) - cos(pow(x, 2)))

#define Func1(x) ((5.0 \* pow(x, 4)) + (2.0 \* x) \* sin(x\*x))

#define Func2(x) ((20.0 \* pow(x, 3)) + (4.0 \* (x\*x)) \* cos(x\*x) + (2.0 \* sin(x\*x)))

int main()

{

double a = LEFT\_RANGE;

double b = RIGHT\_RANGE;

int n = 0;

double c = b;

if (Func0(a) \* Func2(a) > 0)

c = a;

do {

c = c - Func0(c) / Func1(c);

n++;

} while (fabs(Func0(c)) > EPSILON && n < RANGE);

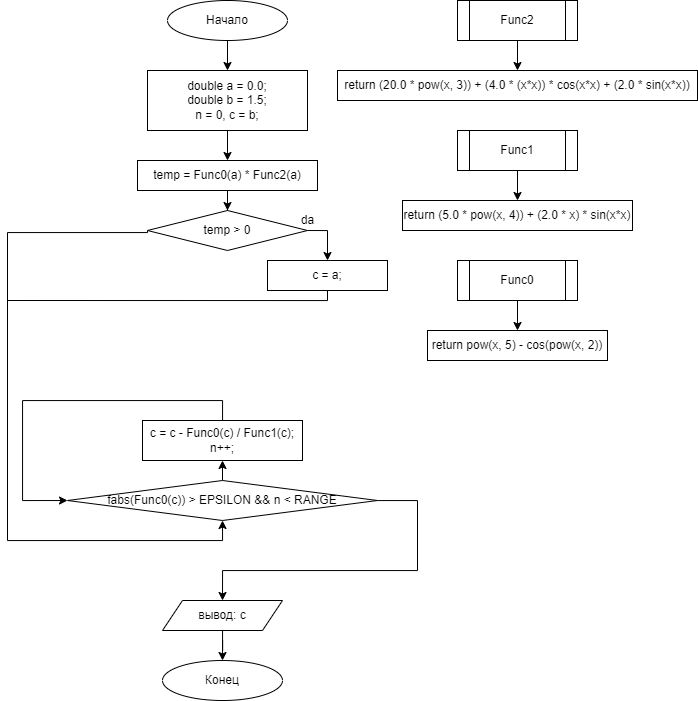
std::cout << "X = " << c << std::endl;

return 0;

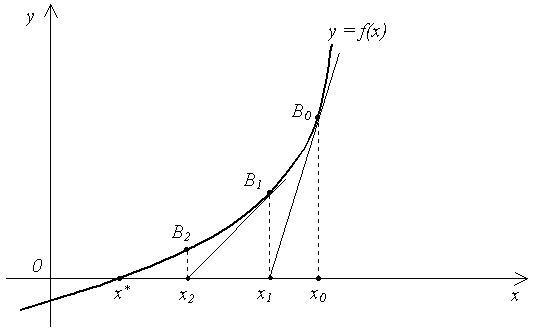
}

**Результат работы программы «****Метод Ньютона»**



****

**Геометрическая интерпретация «Метод Ньютона»**



С каждой новой касательной приближенное значение X становится всё ближе к действительному значению.