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

1. Постановка задачи

* Вычислить значение выражения при различных вещественных типах данных (float и double). Вычисления следует выполнять с использованием промежуточных переменных. Сравнить и объяснить полученные результаты.
* Вычислить значения выражений. Объяснить полученные результаты.

|  |  |  |
| --- | --- | --- |
| 4 | ,  при а=1000, b=0.0001 | 1. n++\*m 2. n++<m 3. m-- >m |

1. Код программы для задания 1

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

float a = 1000;

float b = 0.0001f;

float op1 = pow(a + b, 3);

float op2 = pow(a, 3);

float op3 = op1 - op2;

float op4 = 3 \* a \* b \* b + pow(b, 3) + 3 \* a \* a \* b;

float res = op3 / op4;

cout << "Result = " << res << endl;

double a1 = 1000;

double b1 = 0.0001;

double op11 = pow(a1 + b1, 3);

double op21 = pow(a1, 3);

double op31 = op11 - op21;

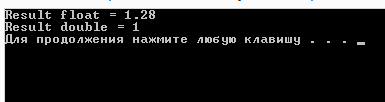
double op41 = 3 \* a1 \* b1 \* b1 + pow(b1, 3) + 3 \* a1 \* a1 \* b1;

double res1 = op31 / op41;

cout << "Result = " << res1;

}

1. Результат выполнения программы



1. Блок схема



1. Код программы для задания 2

#include <iostream>

using namespace std;

int main()

{

setlocale(LC\_ALL, "rus");

int n, m;

cout << "Введите n: ";

cin >> n;

cout << "Введите m: ";

cin >> m;

int copy\_n = n;

int copy\_m = m;

int res1 = copy\_n++ \* copy\_m;

cout << "n++ \* m = " << res1 << endl;

copy\_n = n;

bool res2 = copy\_n++ < m;

cout << "n++ < m = " << res2 << endl;

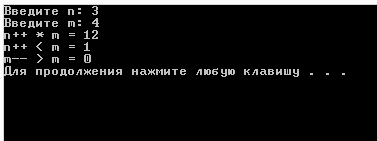
bool res3 = copy\_m-- > copy\_m;

cout << "m-- > m = " << res3 << endl;

system("pause");

}

1. Результат выполнения программы



# 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.

Найти сумму целых положительных чисел, больших 20, меньших 100 и кратных 3

1. Код программы

#include <iostream>

using namespace std;

int main()

{

int sum = 0;

for (int i = 20; i < 100; i++)

{

if (i % 3 == 0)

sum += i;

}

cout << "Summa = " << sum;

}

1. Результат выполнения



1. Блок схема



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

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 ε (ε=0.0001).

Для сравнения найти точное значение функции.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 |  |  | 10 |  |

1. Код программы

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

double a = 0.1, b = 0.8;

int n = 10, k = 10;

double h = (b - a) / k;

double eps = 0.0001;

for (double x = a; x <= b + 0.001; x += h)

{

double an = x \* x / 2;

double sn = an;

int i = 0;

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

{

an = -an \* x \* x \* i \* (2 \* i - 1) / ((i + 1) \* (2 \* i + 1));

sn += an;

}

an = x \* x / 2;

double se = an;

i = 1;

do

{

an = -an \* x \* x \* i \* (2 \* i - 1) / ((i + 1) \* (2 \* i + 1));

se += an;

i++;

} while (abs(an) > eps);

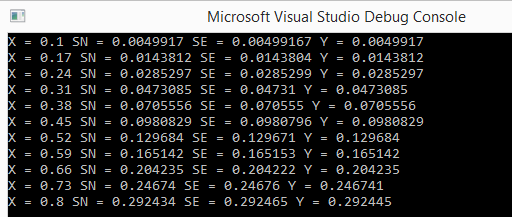
double y = x \* atan(x) - log(sqrt(1 + x \* x));

cout << "X = " << x << " SN = " << sn << " SE = " << se << " Y = " << y << endl;

}

}

1. Результат выполнения



1. Блок схема



# Лабораторная работа № 4 "Работа с одномерными массивами"

1. Постановка задачи

* Сформировать одномерный массив целых чисел, используя датчик случайных чисел.
* Распечатать полученный массив.
* Удалить элементы, индексы которых кратны 3.
* Добавить после каждого отрицательного элемента массива элемент со значением | M[ I-1 ]+1|.
* Распечатать полученный массив.

1. Код программы

#include <iostream>

#include <ctime>

using namespace std;

int main()

{

srand(time(0));

int a[100];

int n;

cout << "n = ";

cin >> n;

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

{

a[i] = rand() % 100 - 50;

}

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

{

cout << a[i] << " ";

}

cout << endl;

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

{

if (i % 2 == 0)

{

for (int j = i; j < n - 1; j++)

{

a[j] = a[j + 1];

}

n--;

}

}

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

{

cout << a[i] << " ";

}

cout << endl;

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

{

if (a[i] < 0)

{

for (int j = n; j > i; j--)

{

a[j] = a[j - 1];

}

a[i + 1] = a[i] + 1;

n++;

i++;

}

}

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

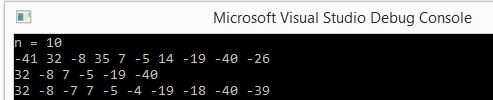
{

cout << a[i] << " ";

}

}

1. Результат выполнения



1. Блок схема





# Лабораторная работа №5 "Функции и массивы"

1. Постановка задачи

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

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.

2. Код программы

#include <iostream>

using namespace std;

void fill\_matrix(int a[100][100], int n, int m)

{

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

{

for (int j = 0; j < m; j++)

{

a[i][j] = rand() % 100;

}

}

}

void print\_matrix(int a[100][100], int n, int m)

{

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

{

for (int j = 0; j < m; j++)

{

cout << a[i][j] << "\t";

}

cout << endl;

}

}

void delete\_row(int a[100][100], int& n, int m, int row)

{

for (int i = row; i < n - 1; i++)

{

for (int j = 0; j < m; j++)

{

a[i][j] = a[i + 1][j];

}

}

for (int j = 0; j < m; j++)

{

a[n - 1][j] = 0;

}

n--;

}

int main()

{

int n, m;

cout << "n = ";

cin >> n;

cout << "m = ";

cin >> m;

int a[100][100];

fill\_matrix(a, n, m);

print\_matrix(a, n, m);

int start, end;

cout << "a = ";

cin >> start;

cout << "b = ";

cin >> end;

for (int i = start; i <= end; i++)

{

delete\_row(a, n, m, start);

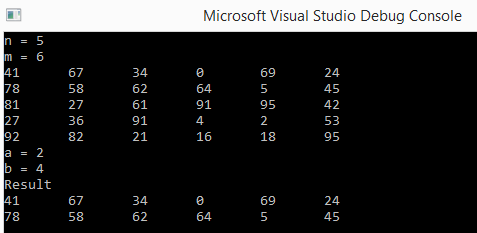
}

cout << "Result" << endl;

print\_matrix(a, n, m);

}

3. Результат выполнения



# Лабораторная работа №6 "Методы решения нелинейных уравнений"

1. Постановка задачи

Поиск корней функции на заданном интервале с помощью трех методов: метод половинного деления, метод Ньютона, метод итераций.

1. Код программы

Метод половинного деления

#include <iostream>

#include <cmath>

using namespace std;

double f(double x)

{

return sqrt(x) + x \* x - 10;

}

double poldel(double a, double b, double eps)

{

double c;

while (fabs(b - a) > eps)

{

c = (b + a) / 2;

if (f(a) \* f(c) < 0)

b = c;

else a = c;

}

return c;

}

int main()

{

setlocale(LC\_ALL, "rus");

double a = 0, b = 5, c, eps = 0.0001;

cout << "eps = " << eps << " a = " << a << " b = " << b << endl;

c = poldel(a, b, eps);

cout << "Корень = " << c << endl;

}

Метод Ньютона

#include <iomanip>

#include <cmath>

#include <iostream>

using namespace std;

double f(double x)

{

return sqrt(x) + x \* x - 10;

}

double df(double x)

{

return 1/(2\*sqrt(x)) + 2\*x;

}

double g(double x)

{

return x - f(x) / df(x);

}

double calc\_root(double x, double eps)

{

while (eps < fabs(f(x)))

{

x = g(x);

}

return x;

}

int main()

{

setlocale(LC\_ALL, "rus");

double x = 1;

double eps = 0.0001;

double root = calc\_root(x, eps);

cout << "Корень = " << root << endl;

}

Метод итераций

#include <cmath>

#include <iomanip>

#include <iostream>

using namespace std;

double f(double x)

{

return sqrt(x) + x \* x - 10;

}

double g(double x)

{

return x - 0.1 \* f(x);

}

double calc\_root(double x, double eps)

{

while (eps < fabs(f(x)))

{

x = g(x);

}

return x;

}

int main()

{

setlocale(LC\_ALL, "rus");

double x = 5;

double eps = 0.0001;

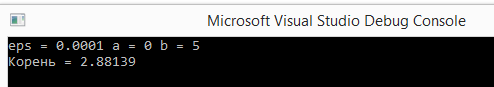
double root = calc\_root(x, eps);

cout << "Корень = " << root << endl;

}

1. Результат выполнения

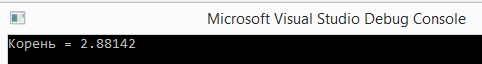
Метод половинного деления



Метод Ньютона



Метод итераций



1. Блок схемы

Метод половинного деления





Метод Ньютона







Метод итераций

