ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ

УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кафедра Інформатики

**Звіт**

з лабораторної роботи № 2

на тему «Обчислення рядів»

Виконав Перевірив

ст.гр.ІТІНФ-20-1 ас.каф. Інформатики

Самченко С.О. Пономаренко Р.П.

Харків 2021

**Мета роботи:** Застосування чисельних методів розв'язання нелінійних рівнянь у різноманітних практичних задачах. Аналіз результатів.

Варіант №21:

Методи: комбінований, ітерацій

1)Комбінований метод (метод хорд і дотичних) - це чисельний метод знаходження (одного) рішення x (із заданою точністю ε) нелінійного рівняння виду f(x) = 0.

**Суть комбінованого методу** полягає в розбитті відрізка [a; b] (за умови f (a) f (b) <0) на три відрізка за допомогою хорди і дотичній і виборі нового відрізка від точки перетину хорди з віссю абсцис до точки перетину дотичної з віссю абсцис, на якому функція змінює знак і містить Рішення.

Побудова хорд і дотичних триває до досягнення необхідної точності рішення ε.

Комбінований метод можна застосовувати для вирішення рівняння виду f(x) = 0 на відрізку [a; b], якщо жодна точка відрізка [a; b] не є ні стаціонарної, ні критичної, тобто f '(x) ≠ 0 і f' '(x) ≠ 0.

Умова початкової точки для методу хорд f (x) f '' (x) <0.

Умова початкової точки для методу дотичних f (x) f '' (x)> 0.

Спочатку знаходимо відрізок [a; b] такий, що функція f (x) двічі неперервно диференційовна і змінює знак на відрізку, тобто f (a) f (b) <0.

Далі застосовуємо ***алгоритм рішення.***

Вхідні дані: f (x), f '(x), f''(x), a, b, ε.

Якщо f'(a) f''(a) <0, то a = a - f(a) (a - b) / (f (a) - f (b))

інакше якщо f(a) f''(a)> 0, то a = a - f(a) / f'(a)

Якщо f '(b) f''(b) <0, то b = b - f(b) (b - a) / (f(b) - f(a))

інакше якщо f(b) f ''(b)> 0, то b = b - f(b) / f'(b)

Якщо | a - b | > 2ε, то йти до 1.

x = (a + b) / 2

Вихідні дані: x.

Значення x є рішенням з заданою точністю ε нелінійного рівняння виду f (x) = 0.

Якщо f (x) = 0, то x - точне рішення.

#include <iostream>

#include <cmath>

#include <iomanip>

// x^3 + 7x^2 + 4x - 12 = 0

using namespace std;

double f(double x)

{

double func = pow(x, 3) + 7 \* pow(x, 2) + 4 \* x - 12;

return func;

}

double f1(double x)

{

double func = 3 \* pow(x, 2) + 14 \* x + 4;

return func;

}

double f2(double x)

{

double func = 6 \* x + 14;

return func;

}

void diapazons(int a, int b) {

cout << "общий промeжуток: ";

for (int i = a; i < b + 1; i++) {

cout << i << " ";

}

cout << endl;

cout << "диапазоны, в которых находится х: ";

for (int x = a; x < b + 1; x++) {

int y = f(x);

if (y == 0) {

cout << "[" << x - 1 << " ; " << x + 1 << "] ";

}

}

cout << endl;

}

double ff(double a, double b) {

float e = 0.00000000001;

double xa(0), xb(0), t1, t2, s;

if ((f1(a) \* f2(a)) < 0)

{

xa = a;

xb = b;

}

else if ((f1(a) \* f2(a)) > 0)

{

xa = b;

xb = a;

}

while (1)

{

t1 = xa - (f(xa) / f1(xa));

t2 = xb - (f(xb) \* (xb - xa)) / (f(xb) - f(xa));

s = (t1 + t2) / 2;

if (abs(t2 - t1) < e)

{

return (t1 + t2) / 2;

printf("%s%10.8f%s%10.8f%s%10.8f%s", "Ax= ", xa, "\tBx= ", xb, "\tx= ", s, "\n");

}

else

{

printf("%s%10.8f%s%10.8f%s%10.8f%s", "Ax= ", xa, "\tBx= ", xb, "\tx= ", s, "\n");

xa = t1;

xb = t2;

}

}

}

int main() {

setlocale(LC\_ALL, "rus");

int a(-10), b(10);

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

diapazons(a, b);

cout << "diapazon" << endl;

cout << "[-7 ; -5]" << endl;

a = -7; b = -5;

ff(a, b);

cout << "[-3 ; -1]" << endl;

a = -3; b = -1;

ff(a, b);

cout << "[0 ; 2]" << endl;

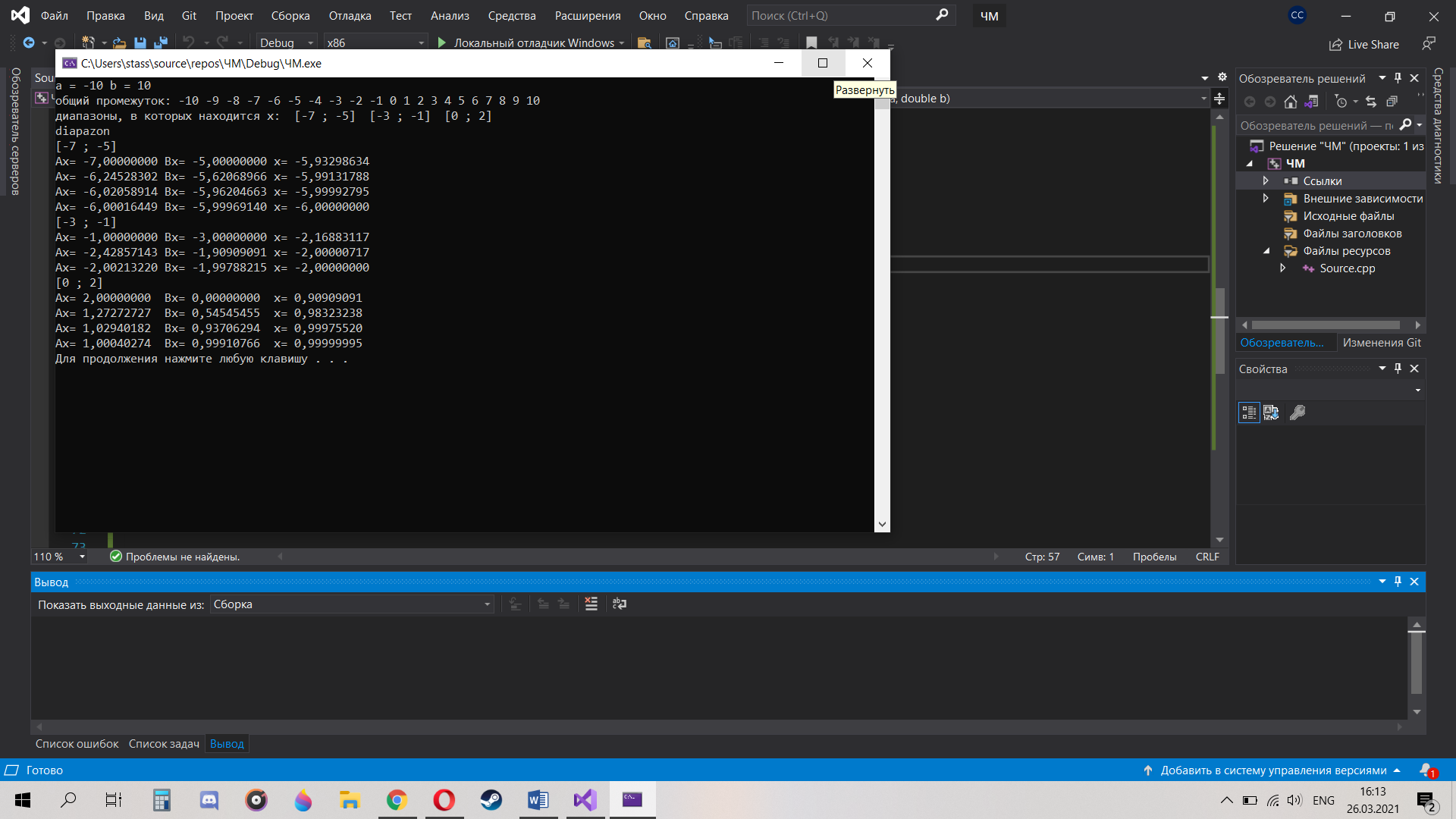
a = 0; b = 2;

ff(a, b);

system("pause");

return 0;

}



**Метод простої ітерації** - один з найпростіших чисельних методів розв'язання рівнянь. Метод заснований на принципі стискає відображення, який стосовно до чисельних методів в загальному вигляді також може називатися методом простої ітерації або методом послідовних наближень. Зокрема, для систем лінійних алгебраїчних рівнянь існує аналогічний метод ітерації.

Ідея методу простої ітерації полягає в тому, щоб рівняння f(x) = 0 привести до еквівалентного рівняння х = φ(х)

так, щоб відображення φ(x) було стискає. Якщо це вдається, то послідовність ітерацій xi+1 = φ(xi) сходиться. Таке перетворення можна робити різними способами. Зокрема, зберігає коріння рівняння виду φ(x) = x – λ(x)f(x), якщо λ(x) ≠ 0 на досліджуваному відрізку. Оптимальним вибором є λ(x) = 1/f’(x), що призводить до методу Ньютона, який є швидким, але вимагає обчислення похідної. Якщо в якості λ(x) вибрати константу того ж знака, що і похідна в околиці кореня, то ми отримуємо найпростіший метод ітерації.

#include <iostream>

#include <cmath>

#include <iomanip>

// x^3 + 7x^2 + 4x - 12 = 0

using namespace std;

double f(double x) {

return pow(x, 3) + 7 \* pow(x, 2) + 4 \* x - 12;

}

double f1(double x) {

return 3 \* pow(x, 2) + 14 \* x + 4;

}

double F(double x, double M) {

return x - f(x) / M;

}

void diapazons(int a, int b) {

cout << "общий промeжуток: ";

for (int i = a; i < b + 1; i++) {

cout << i << " ";

}

cout << endl;

cout << "диапазоны, в которых находится х: ";

for (int x = a; x < b + 1; x++) {

int y = f(x);

if (y == 0) {

cout << "[" << x - 1 << " ; " << x + 1 << "] ";

}

}

cout << endl;

}

void ff(double a, double b) {

double f\_a, f\_b, M, fi, e(0.0000001);

f\_a = fabs(f1(a));

f\_b = fabs(f1(b));

if (f\_a > f\_b) {

M = f\_a;

}

else {

M = f\_b;

}

fi = F(b, M);

double x1(0), x2(0);

x1 = b;

for (int i = 1; i < 11; i++)

{

cout << "Итерация №" << i << "\t";

x2 = F(x1, M);

cout << "Приближение: " << fabs(x2 - x1) << setprecision(6) << fixed <<"\t\t";

if (fabs(x2 - x1) < e) {

break;

}

else {

x1 = x2;

cout << "a = " << x1 << "\t\tb = " << x2 << "\t\tx = " << (x1+x2)/2 << endl;

}

}

}

int main() {

setlocale(LC\_ALL, "rus");

int a(-10), b(10);

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

diapazons(a, b);

cout << "diapazon" << endl;

cout << "[-7 ; -5]" << endl;

a = -7; b = -5;

ff(a, b);

cout << "[0 ; 2]" << endl;

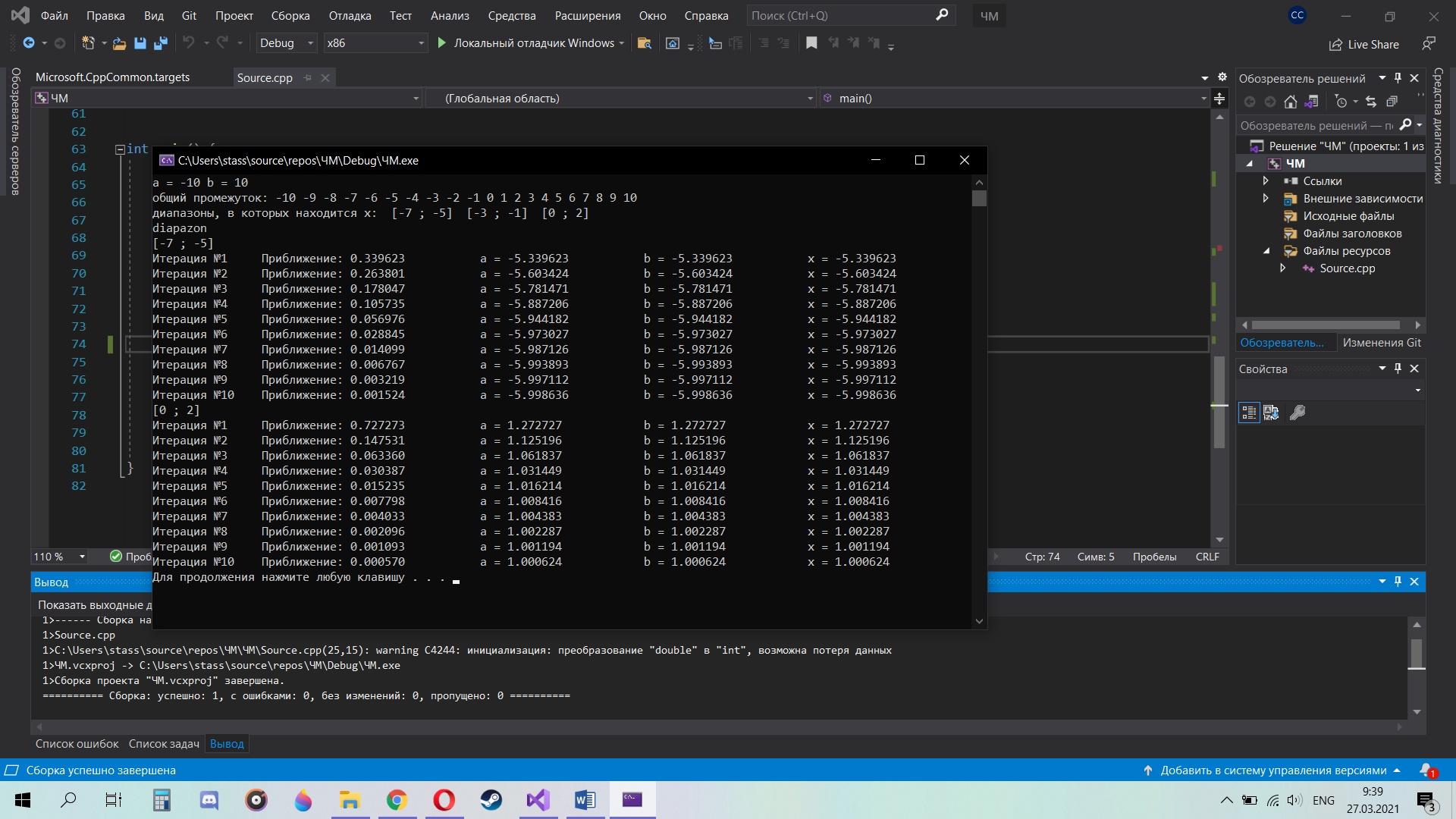
a = 0; b = 2;

ff(a, b);

system("pause");

return 0;

}



Висновок: за допомогою комбінованого метода та мотода ітерацій ми навчилися розв’язувати квадратні рівняння програмним шляхом.