

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ “КИЇВСЬКИЙ
ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО”
ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ
**Кафедра системного програмування і спеціалізованих
комп’ютерних систем**

Лабораторна робота №2

з дисципліни “Алгоритми та методи обчислень”

Тема: “Розв’язання рівнянь з одним невідомим”

Варіант 4

Виконала: студентка III курсу

КВ-72 Дорош Карина

Перевірів: Зорін Ю. М.

Київ 2019

Завдання

Варіант	Рівняння	Метод уточнення коренів рівняння
4	$0.3e^x - x^2 + 1 = 0$	І,Д

Код програми

calculation.cpp

```
#include "Header.h"
```

```
double resFunc(double x) {
    return 0.3 * exp(x) - x * x + 1;
}
```

```
double derative1(double x) {
    return 0.3 * exp(x) - 2 * x;
}
```

```
double derative2(double x) {
    return 0.3 * exp(x) - 2;
}
```

```
range* rangeSelection(int *length) {
    range* temp = (range*)malloc(sizeof(range));
    *length = 0;
    for(double step = a; step <= b-0.01; step += 0.01) {
        if ((resFunc(step) * resFunc(step + 0.01) < 0) && (derative1(step)*derative1(step+0.01) >
0)){
            *length += 1;
            temp = (range*)realloc(temp, (*length) * sizeof(range));
            temp[(*length) - 1].a1 = step;
            temp[(*length) - 1].b1 = step + 0.01;
        }
    }
    return temp;
}
```

```
double iterationMethod(double a1, double b1, double eps, double* error, int maxit, bool& done,
vector<tab> &iterNumb1) {
    double M1, m1, lambda, q, xn;
    double x0 = (a1 + b1) / 2;
    done = true;
    if (fabs(derative1(a1)) > fabs(derative1(b1))) {
        M1 = derative1(a1);
        m1 = derative1(b1);
    }
    else {
        M1 = derative1(b1);
        m1 = derative1(a1);
    }
    lambda = 1 / M1;
    q = 1 - fabs(m1 / M1);
    xn = x0;
```

```

    int count = 0;
    do {
        x0 = xn;
        xn = x0 - lambda * resFunc(x0);
        count++;
        if (count > maxit) {
            done = false;
            break;
        }
    } while (fabs(xn - x0) > (1 - q) / q * eps);
    *error = (fabs(xn - x0)) * q / (1 - q);
    if (done) {
        iterNumb1.push_back(make(xn, eps, count));
    }
    return xn;
}

tab make(double xn, double eps, int iter) {
    tab temp;
    temp.xn = xn;
    temp.eps = eps;
    temp.iter = iter;
    return temp;
}

double tangentMethod(double a1, double b1, double eps, double* error, int maxit, bool& done,
vector<tab> &iterNumb2) {
    double m1, x0, xn;

    m1 = fabs(derative1(a1));

    for (double i = a1; i < b1; i += 0.001)
        if (fabs(derative1(i)) < m1) m1 = fabs(derative1(i));

    (resFunc(a1) * derative2(a1) > 0)? x0 = a1 : x0 = b1;
    xn = x0;
    int count = 0;
    do {
        xn = xn - resFunc(xn) / derative1(xn);
        count++;
        if (count > maxit) {
            done = false;
            break;
        }
    } while (fabs(resFunc(xn) / m1) > eps);
    *error = fabs(resFunc(xn)) / m1;
    if (done) {
        iterNumb2.push_back(make(xn, eps, count));
    }
    return xn;
}

void tableIter(range* t, int *length, vector<tab> &iterNumber1) {
    double error = 0, xn;
    bool done = true;
    cout << "\tIteration method:\n";
    cout << "-----" << endl;
    cout << "| Eps | Xn | Error |" << endl;
    cout << "-----" << endl;
    for (int i = 0; i < (*length); i++) {
        for (double eps = 1e-2; eps >= 1e-14; eps *= 1e-3) {
            xn = iterationMethod(t[i].a1, t[i].b1, eps, &error, 150, done, iterNumber1);
            if (done) {
                cout << "|" << setw(7) << eps << "|" << setprecision(11) << setw(15) << xn
                << "|" << setprecision(11) << setw(17) << error << "|" << endl;
            }
            else {

```

```

        cout << "|" << setw(7) << eps << "|" << setprecision(11) << setw(16) << "
not found |" << setprecision(11) << setw(18) << " not found |" << endl;
    }
    }
    cout << "-----" << endl;
}
cout << endl;
}

void tableTangent(range* t, int* length, vector<tab> &iterNumb2) {
    double error = 0, xn;
    bool done = true;
    cout << "\tTangent method:\n";
    cout << "-----" << endl;
    cout << "|   Eps   |      Xn      |      Error      |" << endl;
    cout << "-----" << endl;

    for (int i = 0; i < (*length); i++) {
        for (double eps = 1e-2; eps >= 1e-14; eps *= 1e-3) {
            xn = tangentMethod(t[i].a1, t[i].b1, eps, &error, 150, done, iterNumb2);
            if (done) {
                cout << "|" << setw(7) << eps << "|" << setprecision(11) << setw(15) << xn
<< "|" << setprecision(11) << setw(17) << error << "|" << endl;
            }
            else {
                cout << "|" << setw(7) << eps << "|" << setprecision(11) << setw(16) << "
not found |" << setprecision(11) << setw(18) << " not found |" << endl;
            }
        }
        cout << "-----" << endl;
    }
    cout << endl;
}

void tableCompare(vector<tab>& iterNumber1, vector<tab>& iterNumber2) {
    cout << "\tNumber of iterations:\n";
    cout << "-----" << endl;
    cout << "|      Xn      |   Eps   |Iteration method|Tangent method|" << endl;
    cout << "-----" << endl;
    int temp;
    (iterNumber1.size() < iterNumber2.size()) ? temp = iterNumber1.size() : temp =
iterNumber2.size();
    for (auto i = 0; i < temp; i++)
        cout << "|" << setw(13) << iterNumber1[i].xn << "|" << setw(6) << iterNumber1[i].eps <<
"| " << setw(6) << iterNumber1[i].iter << setw(12) << "|" << setw(6) << iterNumber2[i].iter <<
setw(6) << "|" << endl;
    cout << "-----" << endl;
}

```

Header.h

```

#pragma once
#include <iostream>
#include <cmath>
#include <windows.h>
#include <iomanip>
#include <vector>
#define a -5
#define b 5

using namespace std;

typedef struct {
    double a1;
    double b1;
}

```

```

}range;

struct tab {
    double xn;
    double eps;
    int iter;
};

double resFunc(double x);
double derative1(double x);
double derative2(double x);

range* rangeSelection(int * length);

double iterationMethod(double a1, double b1, double eps, double* error, int maxit, bool& done, vector
<tab> &iterNumber1);
double tangentMethod( double a1, double b1, double eps, double* error, int maxit, bool& done, vector
<tab> &iterNumber2);
void tableIter(range* temp, int* length, vector<tab> &iterNumb1);
void tableTangent(range * temp, int* length, vector<tab>& iterNumb2);
void tableCompare(vector<tab>& iterNumber1, vector<tab>& iterNumber2);
tab make(double xn, double eps, int iter);

```

main.cpp

```

#include "header.h"

int main() {
    int* length = (int*)malloc(sizeof(int));
    vector<tab> iterNumb1;
    vector<tab> iterNumb2;

    range* temp = rangeSelection(length);
    if ((*length) > 0) {
        tableIter(temp, length, iterNumb1);
        tableTangent(temp, length, iterNumb2);
        tableCompare(iterNumb1, iterNumb2);
    }
    system("pause");
}

```



Результати роботи програми

Iteration method:			
Eps	Xn	Error	
0.01	-1.0511476999	3.3116815933e-05	
1e-05	-1.0511249564	1.9551769993e-07	
1e-08	-1.0511247845	1.4771490174e-09	
1e-11	-1.0511247832	8.4532815319e-14	
1e-14	-1.0511247832	6.3946015879e-16	
0.01	1.5564777984	5.0774216406e-06	
1e-05	1.5564777984	5.0774216406e-06	
1e-08	1.5564799395	7.3563076552e-09	
1e-11	1.5564799421	8.7986522716e-12	
1e-14	1.5564799421	1.2969319014e-17	
0.01	3.805075942	1.5001385072e-06	
1e-05	3.805075942	1.5001385072e-06	
1e-08	3.8050766811	1.3834556958e-10	
1e-11	3.8050766811	1.3233382494e-12	
1e-14	3.8050766811	1.228139865e-16	

Tangent method:		
Eps	Xn	Error
0.01	-1.051158356	3.3605753127e-05
1e-05	-1.0511247837	4.8436092503e-10
1e-08	-1.0511247837	4.8436092503e-10
1e-11	-1.0511247832	5.0350676995e-17
1e-14	-1.0511247832	5.0350676995e-17
0.01	1.5564820435	2.0988992977e-06
1e-05	1.5564820435	2.0988992977e-06
1e-08	1.5564799421	7.5324569106e-13
1e-11	1.5564799421	7.5324569106e-13
1e-14	1.5564799421	2.6240922873e-16
0.01	3.8051002507	2.3805755007e-05
1e-05	3.8050766817	5.4871425315e-10
1e-08	3.8050766817	5.4871425315e-10
1e-11	3.8050766811	3.0572258201e-16
1e-14	3.8050766811	3.0572258201e-16

Number of iterations:			
Xn	Eps	Iteration method	Tangent method
-1.0511476999	0.01	1	1
-1.0511249564	1e-05	2	2
-1.0511247845	1e-08	3	2
-1.0511247832	1e-11	5	3
-1.0511247832	1e-14	6	3
1.5564777984	0.01	1	1
1.5564777984	1e-05	1	1
1.5564799395	1e-08	2	2
1.5564799421	1e-11	3	2
1.5564799421	1e-14	5	3
3.805075942	0.01	1	1
3.805075942	1e-05	1	2
3.8050766811	1e-08	3	2
3.8050766811	1e-11	4	3
3.8050766811	1e-14	6	3

Результати з wolframalpa

Solutions:	
$x \approx -1.05112$	
$x \approx 1.55648$	
$x \approx 3.80508$	