Міністерство освіти і науки України

Львівський національний університет імені Івана Франка

Факультет електроніки та комп’ютерних технологій

Звіт

про виконання лабораторної роботи № 9

з курсу “Алгоритмізація та програмування”

«Віконний проект розв'язання нелінійних рівнянь у VC++»

Виконав:

ст. гр. ФЕІ-11

Стасів Петро

Перевірив:

доц. Хвищун І.О.

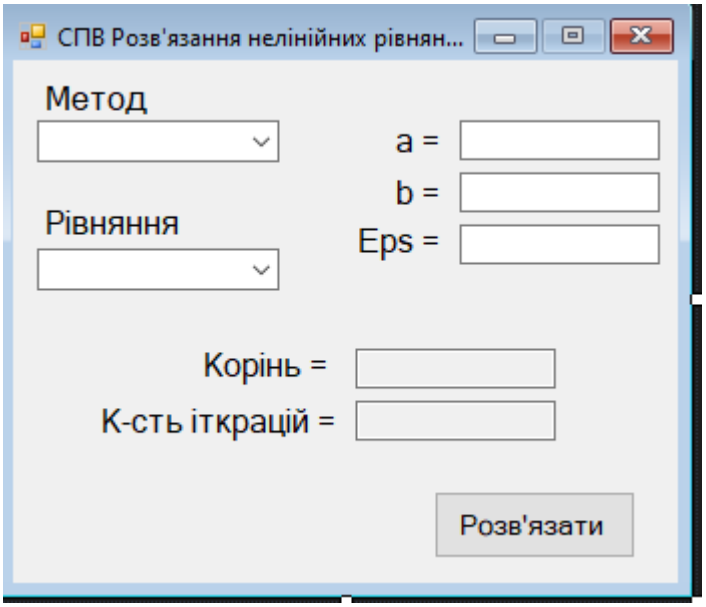
Львів-2021

Звіт

Мета: вивчити процес створення C++ проектів в середовищі Visual C++ 2019 на прикладі програми для розв’язання нелінійних рівнянь.

Структура програми:

1) Створюємо графічний інтерфейс:



2) Функції які реалізують два методи розв’язання МДН і МН:

```
MethodRes BisectionMethod(const float a, const float b, const float eps,
                                                    const PolyFunc& polyFunc)
{
        MethodRes res;

        if (std::abs(polyFunc(a)) <= eps)
        {
                res.Root = a;
                res.Iterations = 1;

                return res;
        }
        else if (std::abs(polyFunc(a)) <= eps)
        {
```

```
                res.Root = b;
                res.Iterations = 1;

                return res;
        }

        float c = 0.0f;
        uint16_t iterationsCount = 0;

        float mA = a;
        float mB = b;

        while (1)
        {
                c = (mA + mB) / 2.0f;

                if (std::abs(polyFunc(c)) <= eps
                        || iterationsCount >= IterationsLimit)
                {
                        res.Root = c;
                        res.Iterations = iterationsCount;

                        break;
                }

                if (polyFunc(mA) * polyFunc(mB) < 0.0f)
                        mB = c;
                else
                        mA = c;

                ++iterationsCount;
        }

        return res;
}

MethodRes NewtonMethod(const float a, const float b, const float eps,
                                        const PolyFunc& polyFunc)
{
        float x = a;

        if (polyFunc(x) * GetDerivative2(x, eps, polyFunc) < 0.0f)
        {
                x = b;

                if (polyFunc(x) * GetDerivative2(x, eps, polyFunc) < 0.0f)
                        std::cout << "For the specified interval result is not
guaranteed!\n";
```

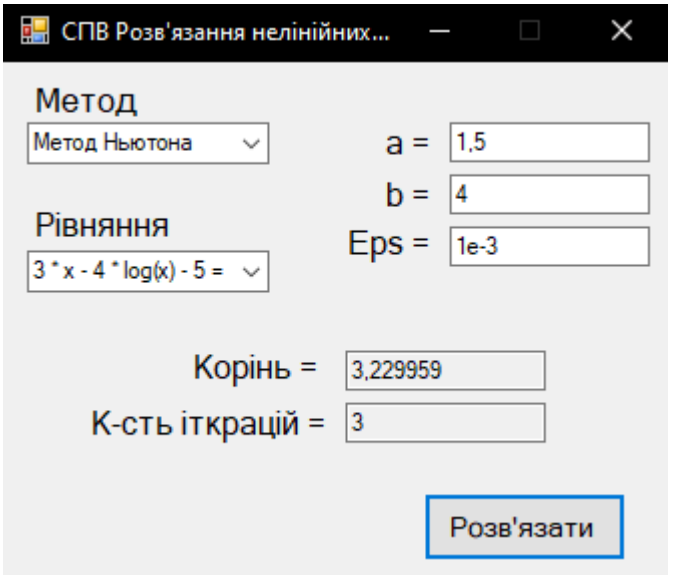
```
        }

        uint16_t iterations = 0;

        while (1)
        {
                if (iterations >= IterationsLimit)
                        break;

                float d = polyFunc(x) / GetDerivative(x, eps, polyFunc);
                x = x - d;

                ++iterations;

                if (std::abs(d) <= eps)
                        break;
        }

        MethodRes res;
        res.Root = x;
        res.Iterations = iterations;

        return res;
}
```

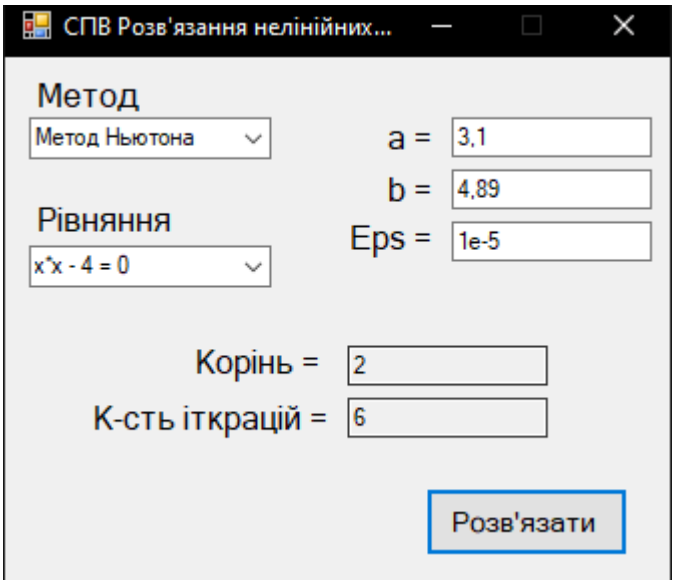
3) Код функції яка викликається при натисканні кнопки “Розв’язати”:

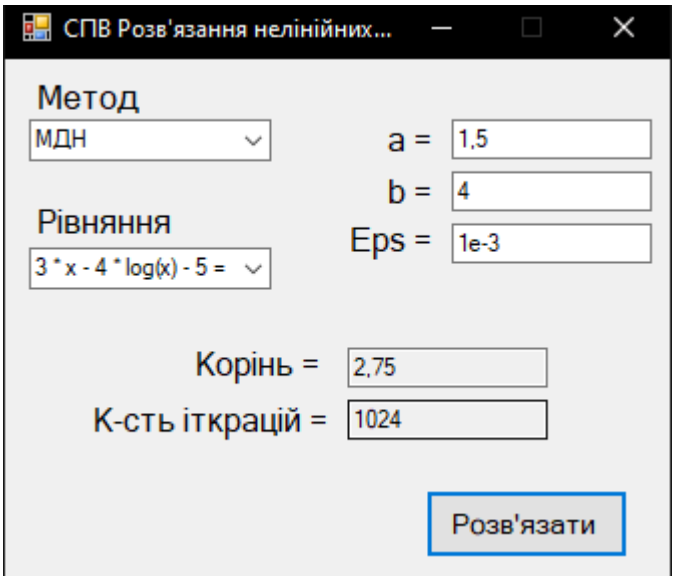
```
private: System::Void SolveButton_Click(System::Object^ sender,
System::EventArgs^ e)

{

        PolyFunc polyFunc;

        MethodFunc methodFunc;



        switch (PolynomialComboBox->SelectedIndex)

        {
```

```
            case 0: polyFunc = Polynomial1; break;
            case 1: polyFunc = Polynomial2; break;
            }

            switch (MethodComboBox->SelectedIndex)
            {
            case 0: methodFunc = BisectionMethod; break;
            case 1: methodFunc = NewtonMethod; break;
            }

            const float a = Convert::ToDouble(aValueTB->Text);
            const float b = Convert::ToDouble(bValueTB->Text);
            const float eps = Convert::ToDouble(epsValueTB->Text);

            MethodRes res = methodFunc(a, b, eps, polyFunc);

            ResultTB->Text = Convert::ToString(res.Root);
            IterationsTB->Text = Convert::ToString(res.Iterations);
}
```

Тестування:


СПВ Розв'язання нелінійних...
Метод
Метод Ньютона
Рівняння
3 * x - 4 * log(x) - 5 =
a = 1,5
b = 4
Eps = 1e-3
Корінь = 3,229959
К-сть іткрацій = 3
Розв'язати


СПВ Розв'язання нелінійних...
Метод
Метод Ньютона
Рівняння
x*x - 4 = 0
a = 3,1
b = 4,89
Eps = 1e-5
Корінь = 2
К-сть іткрацій = 6
Розв'язати


СПВ Розв'язання нелінійних...
Метод
МДН
Рівняння
3 * x - 4 * log(x) - 5 =
a = 1,5
b = 4
Eps = 1e-3
Корінь = 2,75
К-сть іткрацій = 1024
Розв'язати

Текст програми:

```
//MyForm.h

private: System::Void SolveButton_Click(System::Object^ sender,
System::EventArgs^ e)

{

	PolyFunc polyFunc;

	MethodFunc methodFunc;


	switch (PolynomialComboBox->SelectedIndex)

	{

	case 0: polyFunc = Polynomial1; break;

	case 1: polyFunc = Polynomial2; break;

	}


	switch (MethodComboBox->SelectedIndex)

	{

	case 0: methodFunc = BisectionMethod; break;

	case 1: methodFunc = NewtonMethod; break;

	}


	const float a = Convert::ToDouble(aValueTB->Text);

	const float b = Convert::ToDouble(bValueTB->Text);
```

```
	const float eps = Convert::ToDouble(epsValueTB->Text);

	MethodRes res = methodFunc(a, b, eps, polyFunc);

	ResultTB->Text = Convert::ToString(res.Root);
	IterationsTB->Text = Convert::ToString(res.Iterations);
}

//App.h
#include <cmath>
#include <cstdint>
#include <iostream>

constexpr uint16_t IterationsLimit = 1024;

struct MethodRes
{
	float Root;
	uint16_t Iterations;
};

using PolyFunc = float(*)(const float);
using MethodFunc = MethodRes(*)(float, float, float, const
PolyFunc&);

inline float Polynomial1(const float x)
{
```

```
	return x * x - 4;
}

inline float Polynomial2(const float x)
{
	return 3 * x - 4 * std::log(x) - 5;
}

inline float GetDerivative(const float x, const float eps, const
PolyFunc& polyFunc)
{
	return ((polyFunc(x + eps) - polyFunc(x)) / eps);
}

inline float GetDerivative2(const float x, const float eps, const
PolyFunc& polyFunc)
{
	return ((GetDerivative(x + eps, eps, polyFunc) -
GetDerivative(x, eps, polyFunc)) / eps);
}

MethodRes BisectionMethod(const float a, const float b, const float
eps,
                          const PolyFunc& polyFunc);

MethodRes NewtonMethod(const float a, const float b, const float
eps,
                       const PolyFunc& polyFunc);
```

```
//App.cpp

MethodRes BisectionMethod(const float a, const float b, const float
eps,
                                    const PolyFunc& polyFunc)
{
	MethodRes res;


	if (std::abs(polyFunc(a)) <= eps)
	{
		res.Root = a;
		res.Iterations = 1;


		return res;
	}
	else if (std::abs(polyFunc(a)) <= eps)
	{
		res.Root = b;
		res.Iterations = 1;


		return res;
	}


	float c = 0.0f;
	uint16_t iterationsCount = 0;
```

```
	float mA = a;
	float mB = b;

	while (1)
	{
		c = (mA + mB) / 2.0f;

		if (std::abs(polyFunc(c)) <= eps
			|| iterationsCount >= IterationsLimit)
		{
			res.Root = c;
			res.Iterations = iterationsCount;

			break;
		}

		if (polyFunc(mA) * polyFunc(mB) < 0.0f)
			mB = c;
		else
			mA = c;

		++iterationsCount;
	}

	return res;
}
```

```
MethodRes NewtonMethod(const float a, const float b, const float
eps,
                                const PolyFunc& polyFunc)
{
      float x = a;

      if (polyFunc(x) * GetDerivative2(x, eps, polyFunc) < 0.0f)
      {
            x = b;

            if (polyFunc(x) * GetDerivative2(x, eps, polyFunc) <
0.0f)
                  std::cout << "For the specified interval result is
not guaranteed!\n";
      }

      uint16_t iterations = 0;

      while (1)
      {
            if (iterations >= IterationsLimit)
                  break;

            float d = polyFunc(x) / GetDerivative(x, eps, polyFunc);
            x = x - d;

            ++iterations;
```

```
            if (std::abs(d) <= eps)
                break;
        }

        MethodRes res;
        res.Root = x;
        res.Iterations = iterations;

        return res;
}
```

Код лабораторної: https://github.com/ptrstasiv/Lab9.git

Висновок: при виконанні даної лабораторної роботи ми ознайомилися з базовими принципами розробки C++ програм з графічним інтерфейсом в середовищі Visual C++ 2019.