КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

ЗВІТ

з лабораторної роботи № 1

на тему:

«**Розв'язування рівнянь чисельними методами**»

Виконала: студентка 3-го курсу групи ТТП-3

спеціальності «Комп’ютерні науки»

Факультету комп'ютерних наук та кібернетики

Потієнко Ірина

Київ

29 березня 2021 року

Зміст

[**1.** **Опис завдання** 3](#_Toc68215228)

[**2.** **Теоретичні відомості** 4](#_Toc68215229)

[**3.** **Опис алгоритмів, які застосовуються для обчислень** 7](#_Toc68215230)

[**4.** **Робота з програмою** 8](#_Toc68215231)

[**5.** **Висновки** 12](#_Toc68215232)

[**6.** **Список використаних джерел** 13](#_Toc68215233)

[**Додатки** 14](#_Toc68215234)

[**Додаток A. Основний код програми** 14](#_Toc68215235)

[**Додаток B. Програмний код для виконання розрахунків** 24](#_Toc68215236)

[**Додаток C. Програмний код для візуалізації (побудови графіка функції на проміжку)** 30](#_Toc68215237)

# **Опис завдання**

Мета лабораторної роботи полягає у тому, щоб розробити мобільний застосунок, який розв’язує задане рівняння f(x) = 0 різними чисельними методами.

Вхідні дані: обраний тип функції для рівняння f(x) = 0 з-поміж 3 заданих, значення коефіцієнтів цього рівняння, точності та кінців проміжку для знаходження його розв'язку.

Реалізувати можливість для вхідних даних бути зчитаними з файлу.

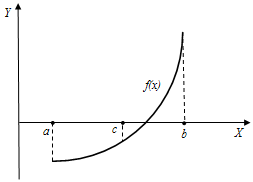
Вихідні дані: результати розв'язання рівняння 3 різними методами, пораховані із заданою точністю, та кількість ітерацій, що знадобилася для розв'язку.

Зберегти у файлі результати табуляції функції на введеному проміжку та на основі цих даних побудувати графік функції на цьому проміжку.

# **Теоретичні відомості**

**Чисельні методи** — методи наближеного або точного розв'язування задач чистої або прикладної математики, які ґрунтуються на побудові послідовності дій над скінченною множиною чисел. Дана наука вивчає алгоритми, які застосовують числову апроксимацію (на відміну від загальних символьних обчислень) для розв'язування задач математичного аналізу (чим відрізняється від дискретної математики). Основні вимоги до числових методів, щоб вони були стійкими та збіжними.

Нехай задано рівняння , яке на відрізку *[a; b]* має єдиний розв’язок, при чому, функція  на даному відрізку є неперервною.



*Метод дихотомії (ділення відрізку навпіл)*

Для знаходження шуканого розв’язку розділимо відрізок навпіл точкою . Якщо значення функції в даній точці відмінне від нуля (), то можливі два випадки:

1. Функція  змінює знак на відрізку
2. Функція  змінює знак на відрізку

Вибираючи той відрізок, на якому функція змінює знак і продовжуючи процес половинного ділення далі, отримаємо як завгодно малий відрізок, який буде містити корінь рівняння .

*Метод Ньютона*

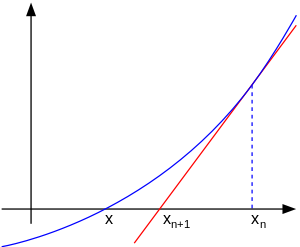
Метод Ньютона (також метод дотичних) — метод наближеного знаходження кореня дійсного рівняння: , де - диференційована функція.

Основна ідея методу полягає у виділенні з вихідних рівнянь лінійних частин, які є головними при малих змінах аргументів. Це дозволяє звести вихідну задачу до вирішення послідовності лінійних систем.

Пошук рішення здійснюється шляхом побудови послідовних наближень і заснований на принципах простої ітерації. Метод має квадратичну збіжність.

Можна також дати геометричну інтерпретацію. Основна ідея методу полягає в наступному: задається початкове наближення поблизу кореня, після чого будується дотична до досліджуваної функції в точці наближення, для якої знаходиться перетин з віссю абсцис. Точка перетину і береться як наступне наближення. І так далі, поки не буде досягнута необхідна точність. Формула наближення може бути виведена таким чином:

де — кут нахилу дотичної в точці



Отже, шуканий вираз для має вигляд:

*Модифікований метод Ньютона*

Модифікований метод Ньютона є модифікацією метода Ньютона. Він дозволяє не обчислювати похідну на кожній ітерації, а отже і позбутися можливого ділення на нуль. Однак цей алгоритм має тільки лінійну збіжність.

Шуканий вираз для має вигляд:

# **Опис алгоритмів, які застосовуються для обчислень**

*Алгоритм методу дихотомії*

Крок 1. Знаходиться середина відрізка .

Крок 2. Перевіряються нижчевикладені умови.

1. Якщо корінь знайдено.
2. Якщо - корінь на , тому .
3. Якщо - корінь на , тому .

Крок 3. Перевіряються умова . Якщо вона виконувться, то корінь знайдено. В цьому випадку він дорівнює . Інакше повертаються до кроку 1.

*Алгоритм методу Ньютона*

Крок 1. Знаходиться початкове наближення

Крок 2. Запам’ятовується останнє наближення .

Крок 3. Знаходиться нове наближення .

Крок 4. Перевіряються умова Якщо вона виконується, то корінь знайдено. В цьому випадку він приймається рівним . Інакше пepexiд на крок 2.

*Алгоритм модифікованого методу Ньютона*

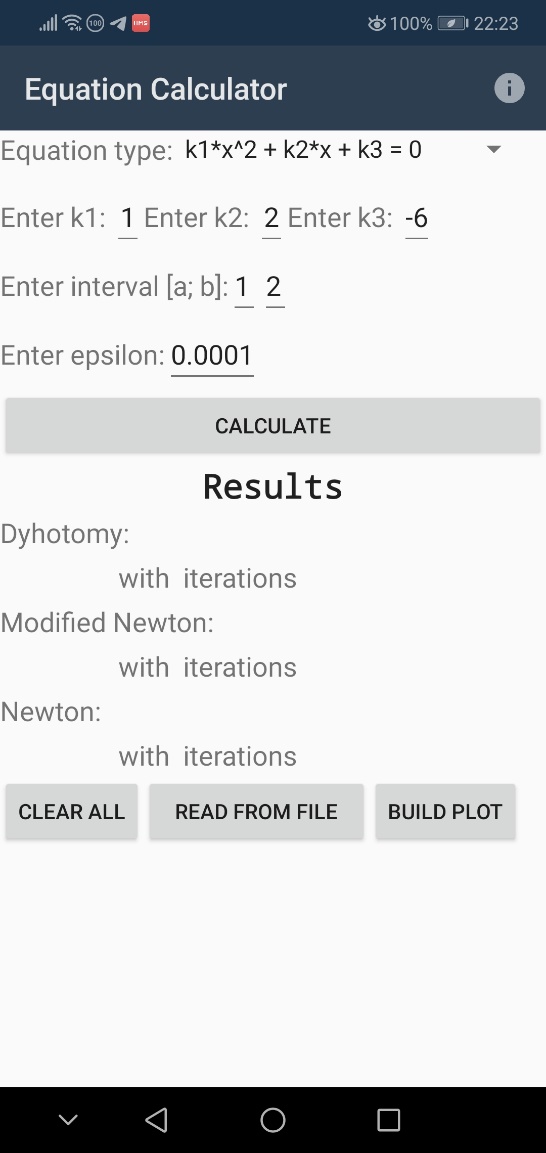
Крок 1. Знаходиться початкове наближення

Крок 2. Запам’ятовується останнє наближення .

Крок 3. Знаходиться нове наближення .

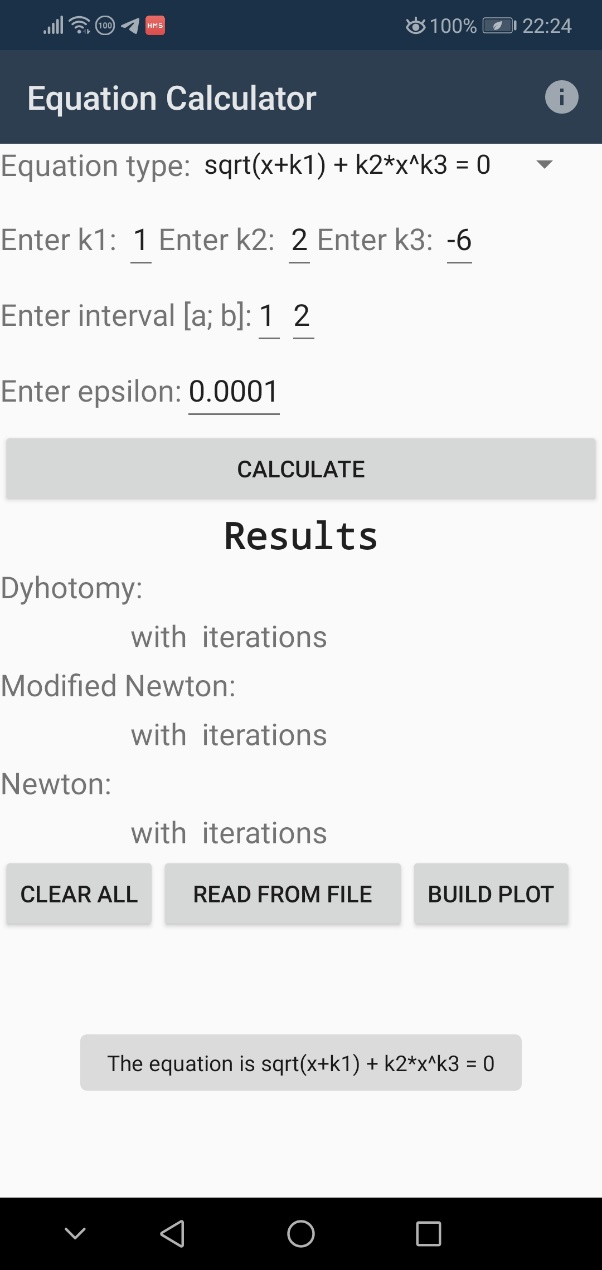
Крок 4. Перевіряються умова Якщо вона виконується, то корінь знайдено. В цьому випадку він приймається рівним . Інакше пepexiд на крок 2.

# **Робота з програмою**



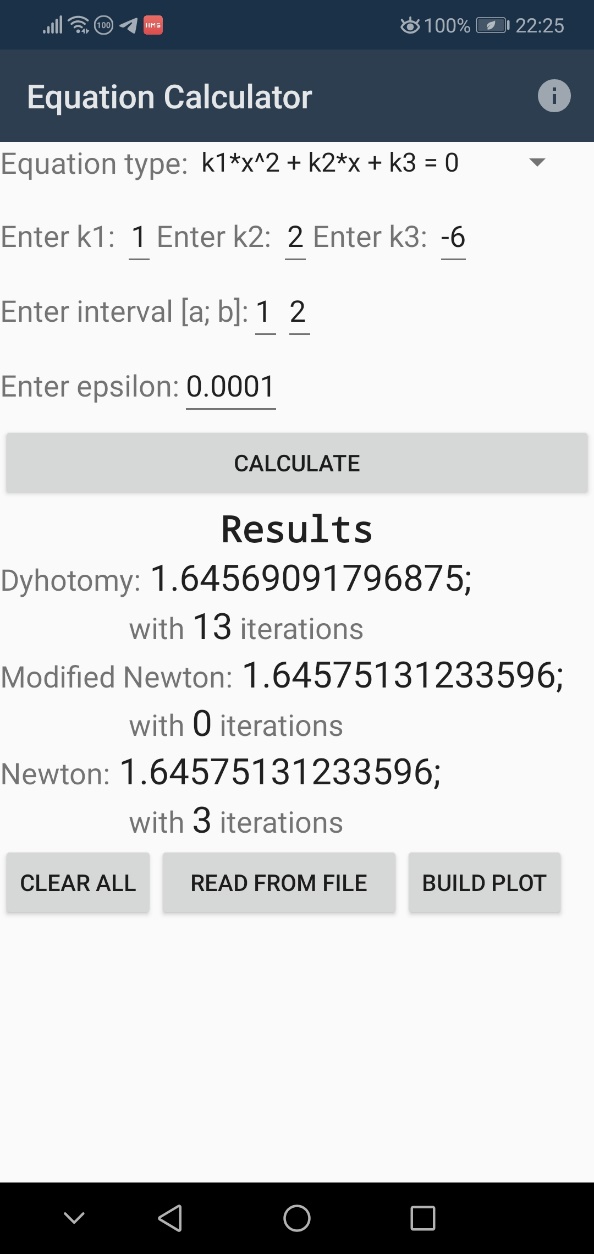
Інтерфейс програми реалізовано англійською мовою.

При вході у мобільний додаток ми бачимо такий інтерфейс із вхідними даними, заданими за замовченням:

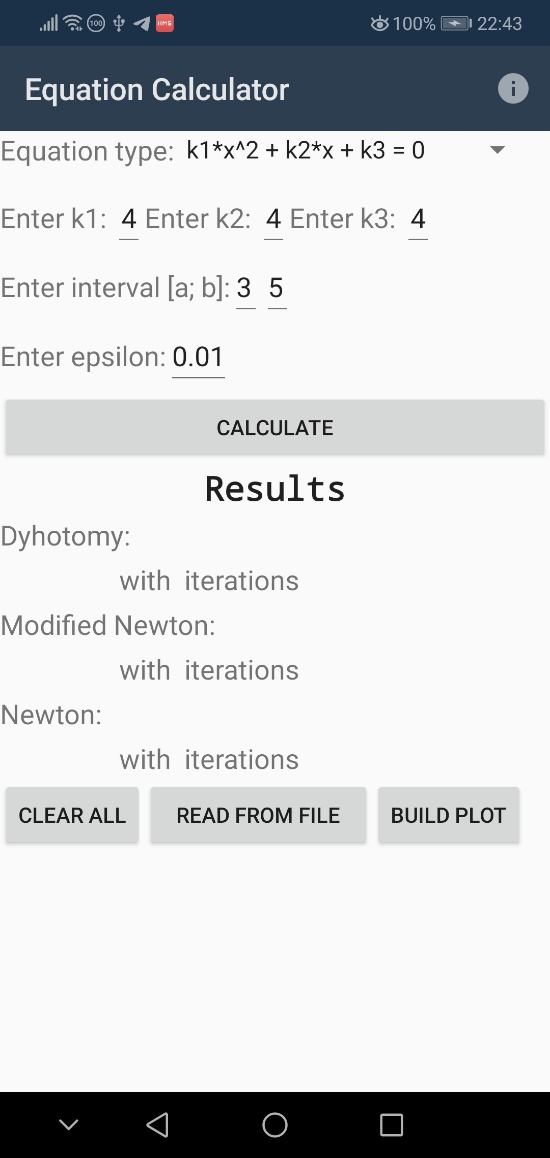


За допомогою перемикача можна змінити тип функції для рівняння f(x)=0.

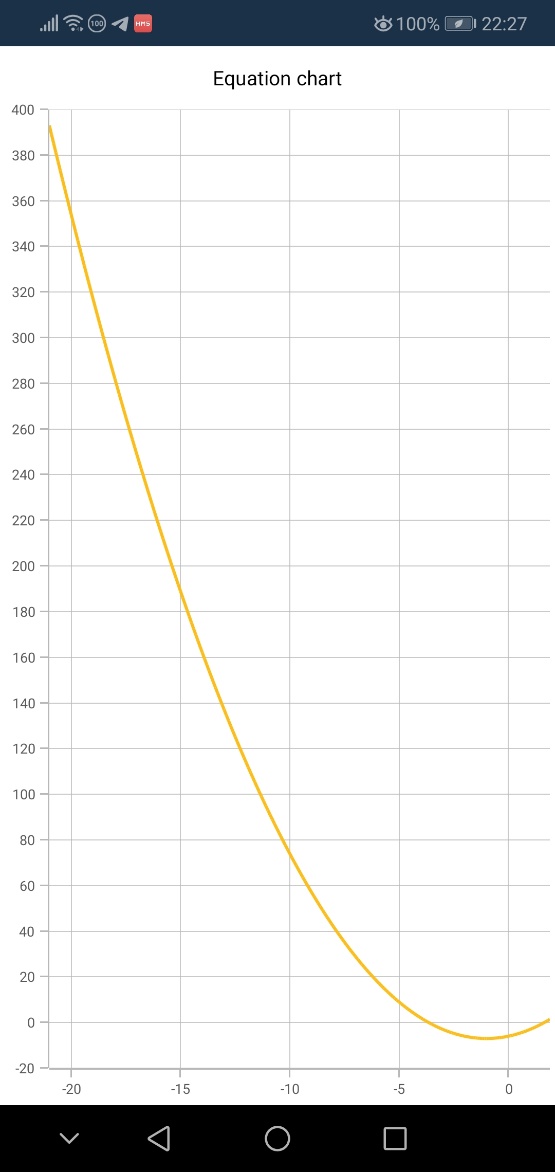
Також можемо ввести (або змінити) значення коефіцієнтів рівняння k1, k2, k3, кінців проміжку a, b та точності epsilon.



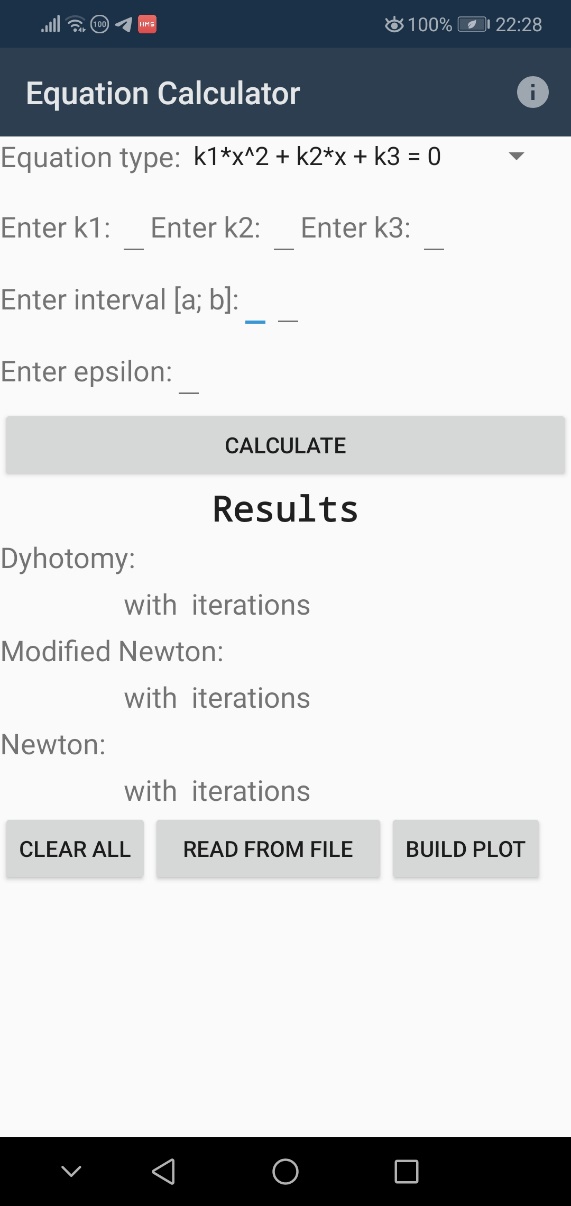
Натиснувши кнопку Calculate, отримаємо результати розв'язків заданого рівняння методами дихотомії, Ньютона та модифікованим Ньютона, виконаними за вказану кількість ітерацій:



Натиснувши кнопку Read from file, отримаємо вхідні значення коефіцієнтів, кінців проміжку та точності, прочитані з текстового файлу:



При натисканні на кнопку Build plot проведеться табуляція заданої функції, результати якої запишуться у текстовий файл. Потім на основі даних, прочитаних з цього файлу, побудується графік заданої функції.



При натисненні на кнопку Clear All всі поля для воду вхідних значень та виведення результатів стануть пустими:

Натиснувши іконку info у правому верхньому куті, можемо побачити інформацію про розробника:



# **Висновки**

Я сформувала базові знання та уміння для розробки додатків для мобільного пристрою з операційною системою Android.

Даний додаток був створений за допомогою Xamarin - фреймворку для кросплатформенної розробки мобільних додатків (iOS, Android, Windows Phone) з використанням мови C #.

В результаті маємо додаток для найпопулярнішої операційної системи для мобільних застосунків, який знаходить корені заданого рівняння, використовуючи чисельні методи.

# **Список використаних джерел**

1. Конспект лекцій з курсу Чисельні методи
2. Вікіпедія [Електронний ресурс] : <https://uk.wikipedia.org/>.
3. Loginom [Електронний ресурс] : <https://wiki.loginom.ru/>

# **Додатки**

# **Додаток A. Основний код програми**

MainActivity.cs

using Android.App;

using Android.OS;

using Android.Support.V7.App;

using Android.Runtime;

using Android.Widget;

using System;

using System.IO;

using Android;

using Android.Content.PM;

using Android.Content;

using Android.Views;

namespace EquitationsCalculator

{

[Activity(Label = "@string/app\_name", Theme = "@style/AppTheme", MainLauncher = true)]

public class MainActivity : AppCompatActivity

{

string equationType;

Equation equation;

TextView dyhotomyResult, modNewtonResult, newtonResult, errorInterval;

TextView dyhotomyIterationsNumber, modNewtonIterationsNumber, newtonIterationsNumber;

EditText aNumb, bNumb, epsilon, k1Coef, k2Coef, k3Coef;

private void spinner\_ItemSelected(object sender, AdapterView.ItemSelectedEventArgs e)

{

Spinner spinner = (Spinner)sender;

string toast = string.Format("The equation is {0}", spinner.GetItemAtPosition(e.Position));

equationType = spinner.GetItemAtPosition(e.Position).ToString();

Toast.MakeText(this, toast, ToastLength.Long).Show();

}

public override bool OnCreateOptionsMenu(IMenu menu)

{

MenuInflater.Inflate(Resource.Menu.top\_menus, menu);

return base.OnCreateOptionsMenu(menu);

}

public override bool OnOptionsItemSelected(IMenuItem item)

{

Toast.MakeText(this, "Action selected: " + item.TitleFormatted,

ToastLength.Short).Show();

return base.OnOptionsItemSelected(item);

}

public void Info()

{

var intent = new Intent(this, typeof(AboutMe));

StartActivity(intent);

}

protected override void OnCreate(Bundle savedInstanceState)

{

base.OnCreate(savedInstanceState);

Xamarin.Essentials.Platform.Init(this, savedInstanceState);

// Set our view from the "main" layout resource

SetContentView(Resource.Layout.activity\_main);

Toolbar toolbar = FindViewById<Toolbar>(Resource.Id.toolbar);

toolbar.Title = "Equation Calculator";

toolbar.InflateMenu(Resource.Menu.top\_menus);

toolbar.MenuItemClick += (sender, e) => Info();

Spinner spinner = FindViewById<Spinner>(Resource.Id.spinner);

spinner.ItemSelected += new EventHandler<AdapterView.ItemSelectedEventArgs>(spinner\_ItemSelected);

var adapter = ArrayAdapter.CreateFromResource(

this, Resource.Array.planets\_array, Android.Resource.Layout.SimpleSpinnerItem);

adapter.SetDropDownViewResource(Android.Resource.Layout.SimpleSpinnerDropDownItem);

spinner.Adapter = adapter;

// Get our UI controls from the loaded layout

aNumb = FindViewById<EditText>(Resource.Id.CoefficientA);

bNumb = FindViewById<EditText>(Resource.Id.CoefficientB);

epsilon = FindViewById<EditText>(Resource.Id.Epsilon);

k1Coef = FindViewById<EditText>(Resource.Id.Coef1);

k2Coef = FindViewById<EditText>(Resource.Id.Coef2);

k3Coef = FindViewById<EditText>(Resource.Id.Coef3);

errorInterval = FindViewById<TextView>(Resource.Id.ErrorInterval);

Button calculateButton = FindViewById<Button>(Resource.Id.CalculateButton);

Button readButton = FindViewById<Button>(Resource.Id.ReadButton);

Button plotButton = FindViewById<Button>(Resource.Id.PlotButton);

Button clearAllButton = FindViewById<Button>(Resource.Id.ClearAllButton);

dyhotomyResult = FindViewById<TextView>(Resource.Id.DyhotomyResult);

dyhotomyIterationsNumber = FindViewById<TextView>(Resource.Id.DyhotomyIterationsNumber);

modNewtonResult = FindViewById<TextView>(Resource.Id.Method2);

modNewtonIterationsNumber = FindViewById<TextView>(Resource.Id.ModNewtonIterationsNumber);

newtonResult = FindViewById<TextView>(Resource.Id.Method3);

newtonIterationsNumber = FindViewById<TextView>(Resource.Id.NewtonIterationsNumber);

calculateButton.Click += (sender, e) => Calculate();

readButton.Click += (sender, e) => ReadFile();

clearAllButton.Click += (sender, e) => ClearAll();

plotButton.Click += async (sender, e) =>

{

if (EquationInitialize())

{

await TabulateAsync();

var intent = new Intent(this, typeof(Visualization));

StartActivity(intent);

}

};

}

public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum] Android.Content.PM.Permission[] grantResults)

{

Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

base.OnRequestPermissionsResult(requestCode, permissions, grantResults);

}

public void Calculate()

{

ClearResults();

if (EquationInitialize())

{

Methods methods = new Methods();

methods.GetResult(equation);

dyhotomyResult.Text = Methods.StringResult(methods.DyhotomyRoots);

dyhotomyIterationsNumber.Text += methods.DyhotomyIterations;

modNewtonResult.Text = Methods.StringResult(methods.ModNewtonRoots);

modNewtonIterationsNumber.Text += methods.ModNewtonIterations;

newtonResult.Text = Methods.StringResult(methods.NewtonRoots);

newtonIterationsNumber.Text += methods.NewtonIterations;

}

}

public bool EquationInitialize()

{

Double a, b, k1, k2, k3;

Methods.epsilon = Convert.ToDouble(epsilon.Text);

a = Convert.ToDouble(aNumb.Text);

b = Convert.ToDouble(bNumb.Text);

k1 = Convert.ToDouble(k1Coef.Text);

k2 = Convert.ToDouble(k2Coef.Text);

k3 = Convert.ToDouble(k3Coef.Text);

if (a >= b)

{

errorInterval.Text = "Incorrect interval!";

return false;

}

else

{

if (equationType == "sqrt(x+k1) + k2\*x^k3 = 0") equation = new SqrtEquation(k1, k2, k3, a, b);

else if (equationType == "k1\*exp(x\*k2) + k3 = 0") equation = new ExpEquation(k1, k2, k3, a, b);

else equation = new SquareEquation(k1, k2, k3, a, b);

return true;

}

}

public async System.Threading.Tasks.Task TabulateAsync()

{

bool isWriteable = Android.OS.Environment.MediaMounted.Equals(Android.OS.Environment.ExternalStorageState);

if (isWriteable)

{

Double x = equation.a, y;

var backingFile = Path.Combine(Android.OS.Environment.GetExternalStoragePublicDirectory(Android.OS.Environment.DirectoryDocuments).AbsolutePath, "results.txt");

using (var writer = File.CreateText(backingFile))

{

do

{

y = equation.f(x);

await writer.WriteLineAsync(x + " " + y);

x = x + 0.1;

} while (x <= equation.b);

}

Toast.MakeText(this, "Saved to file", ToastLength.Short).Show();

}

else Toast.MakeText(this, "No access to External Memory", ToastLength.Short).Show();

}

public void ReadFile()

{

bool isReadonly = Android.OS.Environment.MediaMountedReadOnly.Equals(Android.OS.Environment.ExternalStorageState);

bool isWriteable = Android.OS.Environment.MediaMounted.Equals(Android.OS.Environment.ExternalStorageState);

if (isReadonly || isWriteable)

{

var backingFile = Path.Combine(Android.OS.Environment.GetExternalStoragePublicDirectory(Android.OS.Environment.DirectoryDocuments).AbsolutePath, "startData.txt");

if (backingFile == null || !File.Exists(backingFile))

{

Toast.MakeText(this, "File not exist", ToastLength.Long).Show();

}

else

{

ClearAll();

Toast.MakeText(this, "Information from file", ToastLength.Short).Show();

using (var reader = new StreamReader(backingFile, true))

{

aNumb.Text = reader.ReadLine();

bNumb.Text = reader.ReadLine();

epsilon.Text = reader.ReadLine();

k1Coef.Text = reader.ReadLine();

k2Coef.Text = reader.ReadLine();

k3Coef.Text = reader.ReadLine();

}

}

}

else Toast.MakeText(this, "No access to External Memory", ToastLength.Short).Show();

}

public void ClearResults()

{

dyhotomyResult.Text = string.Empty;

modNewtonResult.Text = string.Empty;

newtonResult.Text = string.Empty;

dyhotomyIterationsNumber.Text = string.Empty;

modNewtonIterationsNumber.Text = string.Empty;

newtonIterationsNumber.Text = string.Empty;

errorInterval.Text = string.Empty;

}

public void ClearAll()

{

aNumb.Text = string.Empty;

bNumb.Text = string.Empty;

epsilon.Text = string.Empty;

k1Coef.Text = string.Empty;

k2Coef.Text = string.Empty;

k3Coef.Text = string.Empty;

ClearResults();

}

}

}

activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:chart="clr-namespace:Syncfusion.SfChart;assembly=Syncfusion.SfChart"

android:orientation="vertical"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/linearLayout1">

<include

android:id="@+id/toolbar"

layout="@layout/toolbar" />

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "wrap\_content"

android:layout\_height= "match\_parent"

>

<TextView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginTop="10dip"

android:text="@string/planet\_prompt"

android:layout\_gravity="center\_horizontal|right"

android:textAppearance="?android:attr/textAppearanceMedium"

/>

<Spinner

android:id="@+id/spinner"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:prompt="@string/planet\_prompt"

android:layout\_gravity="center\_horizontal"

/>

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "match\_parent"

>

<TextView

android:text="Enter k1: "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/C1"

android:layout\_marginRight="0.0dp"

android:gravity= "center\_horizontal" />

<EditText

android:inputType="number|numberSigned|numberDecimal"

android:id="@+id/Coef1"

android:layout\_width="wrap\_content"

android:text="1"

android:layout\_height="wrap\_content"

android:gravity= "center\_horizontal"/>

<TextView

android:text="Enter k2: "

android:textAppearance="?android:attr/textAppearanceMedium"

android:id="@+id/C2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:gravity= "center\_horizontal" />

<EditText

android:inputType="number|numberSigned|numberDecimal"

android:id="@+id/Coef2"

android:layout\_width="wrap\_content"

android:text="2"

android:layout\_height="wrap\_content"

android:gravity= "center\_horizontal"

/>

<TextView

android:text="Enter k3: "

android:textAppearance="?android:attr/textAppearanceMedium"

android:id="@+id/C3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:gravity= "center\_horizontal"

/>

<EditText

android:inputType="number|numberSigned|numberDecimal"

android:id="@+id/Coef3"

android:text="-6"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:gravity= "center\_horizontal"

/>

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<TextView

android:text="Enter interval [a; b]:"

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textView2" />

<EditText

android:inputType="number|numberDecimal|numberSigned"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/CoefficientA"

android:text="1" />

<EditText

android:inputType="number|numberSigned|numberDecimal"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/CoefficientB"

android:text="2" />

<TextView

android:textAppearance="?android:attr/textAppearanceSmall"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/ErrorInterval"

android:textColor="#ffeb3030" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<TextView

android:text="Enter epsilon:"

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textView1" />

<EditText

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/Epsilon"

android:text="0.0001"

android:inputType="number|numberDecimal" />

</LinearLayout>

<Button

android:text="Calculate"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/CalculateButton"/>

<TextView

android:text="Results"

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/textView4"

android:textStyle="bold"

android:typeface="monospace"

android:gravity="center\_horizontal|center\_vertical" />

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<TextView

android:text="Dyhotomy:"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textView3"

android:textAppearance="?android:attr/textAppearanceMedium"/>

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/DyhotomyResult" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "match\_parent"

>

<TextView

android:text=" with "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/wIt1" />

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="wrap\_content"

android:id="@+id/DyhotomyIterationsNumber"

android:layout\_height="wrap\_content" />

<TextView

android:text=" iterations "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/It1" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<TextView

android:text="Modified Newton:"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewM2"

android:textAppearance="?android:attr/textAppearanceMedium" />

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/Method2" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "match\_parent"

>

<TextView

android:text=" with "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/wIt2" />

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="wrap\_content"

android:id="@+id/ModNewtonIterationsNumber"

android:layout\_height="wrap\_content" />

<TextView

android:text=" iterations "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/It2" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<TextView

android:text="Newton:"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/textViewM3"

android:textAppearance="?android:attr/textAppearanceMedium" />

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/Method3" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "match\_parent"

>

<TextView

android:text=" with "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/wIt3" />

<TextView

android:textAppearance="?android:attr/textAppearanceLarge"

android:layout\_width="wrap\_content"

android:id="@+id/NewtonIterationsNumber"

android:layout\_height="wrap\_content" />

<TextView

android:text=" iterations "

android:textAppearance="?android:attr/textAppearanceMedium"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/It3" />

</LinearLayout>

<LinearLayout

android:orientation= "horizontal"

android:layout\_width= "match\_parent"

android:layout\_height= "wrap\_content"

>

<Button

android:text="Clear All"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/ClearAllButton"

android:layout\_marginRight="0.0dp"

android:layout\_marginLeft="0.0dp" />

<Button

android:text="Read from file"

android:layout\_width="149.0dp"

android:layout\_height="wrap\_content"

android:id="@+id/ReadButton"

android:layout\_marginRight="0.0dp" />

<Button

android:text="@string/Visualization"

android:layout\_width="wrap\_content"

android:layout\_height="match\_parent"

android:id="@+id/PlotButton" />

</LinearLayout>

</LinearLayout >

# **Додаток B. Програмний код для виконання розрахунків**

Equation.cs

using …

namespace EquitationsCalculator

{

public abstract class Equation

{

public Equation(double k1, double k2, double k3, double a, double b)

{

this.k1 = k1;

this.k2 = k2;

this.k3 = k3;

this.a = a;

this.b = b;

this.rootsNumber = RootsAmountCheck();

CreateOneRootIntervals();

}

public Double k1 { get; set; }

public Double k2 { get; set; }

public Double k3 { get; set; }

public Double a { get; set; }

public Double b { get; set; }

public int rootsNumber { get; set; }

public List<List<double>> OneRootIntervals;

public abstract double f(double x);

public abstract double f1Deriv(double x);

public abstract double f2Deriv(double x);

public virtual int RootsAmountCheck()

{

double fa = f(a);

double fb = f(b);

if (fa \*fb <= 0) return 1;

else return 0;

}

public virtual void CreateOneRootIntervals()

{

var intervals = new List<List<double>>();

if (rootsNumber == 1)

intervals.Add(new List<double> { a, b });

//if (rootsNumber == 0)

//intervals.Add(new List<double> { });

OneRootIntervals = intervals;

}

}

}

SqrtEquation.cs

using …

namespace EquitationsCalculator

{

class SqrtEquation : Equation

{

public SqrtEquation(double k1, double k2, double k3, double a, double b) : base(k1, k2, k3, a, b) { }

private double sqrt(double x)

{

return Math.Sqrt(x);

}

public override double f(double x)

{

double ans = sqrt(x + k1) + k2 \* Math.Pow(x, k3);

return ans;

}

public override double f1Deriv(double x)

{

double ans = 1 / (2 \* sqrt(x)) + k2 \* k3 \* Math.Pow(x, k3 - 1);

return ans;

}

public override double f2Deriv(double x)

{

return -1 / (4 \* Math.Pow(x, 3 / 2)) + k2 \* k3 \* (k3 - 1) \* Math.Pow(x, k3 - 2);

}

public override int RootsAmountCheck()

{

if (a < k1) a = k1;

if (a > b) return 0;

return base.RootsAmountCheck();

}

}

}

SquareEquation.cs

using …

namespace EquitationsCalculator

{

public class SquareEquation: Equation

{

public SquareEquation(double k1, double k2, double k3, double a, double b) : base(k1, k2, k3, a, b) { }

public override double f(double x)

{

double ans = k1 \* x \* x + k2 \* x + k3;

return ans;

}

public override double f1Deriv(double x)

{

double ans = k1 \* 2 \* x + k2;

return ans;

}

public override double f2Deriv(double x)

{

return k1\*2;

}

public int NumberOfRoots()

{

int number;

double d = D();

if (d > 0) number = 2;

else if (d == 0) number = 1;

else number = 0;

return number;

}

public double D()

{

return Math.Pow(k2, 2) - 4 \* k1 \* k3;

}

public double[] ApexCoordinates()

{

double[] apex = new double[2];

apex[0] = -k2 / (2 \* k1);

apex[1] = -D() / (4 \* k1);

return apex;

}

public override int RootsAmountCheck()

{

if (NumberOfRoots() == 0) return 0;

int baseNumb = base.RootsAmountCheck();

if (baseNumb == 1) return 1;

else if (k1 \* f(a) > 0)

if ((a < ApexCoordinates()[0] && b < ApexCoordinates()[0]) || (a > ApexCoordinates()[0] && b > ApexCoordinates()[0])) return 0;

else return 2;

else return 0;

}

public override void CreateOneRootIntervals()

{

double x;

var intervals = new List<List<double>>();

if (rootsNumber == 2)

{

x = ApexCoordinates()[0];

intervals.Add(new List<double> { a, x });

intervals.Add(new List<double> { x, b });

OneRootIntervals = intervals;

}

else base.CreateOneRootIntervals();

}

}

}

ExpEquation.cs

using …

namespace EquitationsCalculator

{

public class ExpEquation : Equation

{

public ExpEquation(double k1, double k2, double k3, double a, double b):base(k1, k2, k3, a, b) { }

private double exp(double x)

{

return Math.Exp(x);

}

public override double f(double x)

{

double ans = k1\*exp (x\*k2) +k3;

return ans;

}

public override double f1Deriv(double x)

{

double ans = k1 \* k2 \* exp(x \* k2);

return ans;

}

public override double f2Deriv(double x)

{

return k2 \* f1Deriv(x);

}

}

}

Methods.cs

using …

namespace EquitationsCalculator

{

public class Methods

{

public static Double epsilon { get; set; }

public int DyhotomyIterations;

public int ModNewtonIterations;

public int NewtonIterations;

public List<Double> DyhotomyRoots = new List<double>();

public List<Double> ModNewtonRoots = new List<double>();

public List<Double> NewtonRoots = new List<double>();

public void GetResult(Equation equation)

{

//if (equation.rootsNumber == 0) return;

foreach (var interval in equation.OneRootIntervals)

{

for (int i = 0; i <= 1; i++) {

if (equation.f(interval[i]) == 0)

{

DyhotomyRoots.Add(interval[i]);

ModNewtonRoots.Add(interval[i]);

NewtonRoots.Add(interval[i]);

return;

}

}

DyhotomyRoots.Add(Dyhotomy(interval[0], interval[1], equation));

ModNewtonRoots.Add(ModNewton(interval[0], interval[1], equation));

NewtonRoots.Add(Newton(interval[0], interval[1], equation));

}

}

public static string StringResult(List<double> roots)

{

string res="";

foreach(var root in roots)

{

res += " " + root.ToString() + ";";

}

if (res == "") res = " no roots";

return res;

}

public double Dyhotomy(double a, double b, Equation equation)

{

double x;

int i = 0;

//double apriori = Math.Log((b - a) / epsilon) + 1;

while (b - a > 2 \* epsilon)

{

x = (a + b) / 2;

if (equation.f(x) != 0)

{

if (equation.f(a) \* equation.f(x) < 0)

b = x;

else

a = x;

i++;

}

else

{

DyhotomyIterations = i;

return x;

}

}

DyhotomyIterations = i;

return (a + b) / 2;

}

public double ModNewton(double a, double b, Equation equation)

{

int i = 1;

//double h;

double x0 = equation.f(a) \* equation.f2Deriv(a) > 0 ? a : b;

double xn = x0 - equation.f(x0) / equation.f1Deriv(x0);

//h = Math.Abs(xn - x0);

while (Math.Abs(xn - x0) > epsilon)

{

x0 = xn;

xn = x0 - equation.f(xn) / equation.f1Deriv(x0);

i++;

//h = Math.Abs(xn - x0);

}

NewtonIterations = i;

return xn;

}

public double Newton(double a, double b, Equation equation)

{

int i = 1;

double h;

double x0 = equation.f(a) \* equation.f2Deriv(a) > 0 ? a : b;

double xn = x0 - equation.f(x0) / equation.f1Deriv(x0);

h = Math.Abs(xn - x0);

while (Math.Abs(xn - x0) > epsilon)

{

x0 = xn;

xn = x0 - equation.f(xn) / equation.f1Deriv(xn);

i++;

//h = Math.Abs(xn - x0);

}

NewtonIterations = i;

return xn;

}

}

}

# **Додаток C. Програмний код для візуалізації (побудови графіка функції на проміжку)**

Visualization.cs

using Android.App;

using Android.OS;

using Android.Support.V7.App;

using Android.Runtime;

using Android.Widget;

using System;

using System.IO;

using Android;

using Android.Content.PM;

using Com.Syncfusion.Charts;

using Android.Content;

using System.Collections.ObjectModel;

using Android.Graphics;

using System.Collections.Generic;

namespace EquitationsCalculator

{

[Activity(Label = "Visualization")]

public class Visualization : Activity

{

protected override void OnCreate(Bundle savedInstanceState)

{

Syncfusion.Licensing.SyncfusionLicenseProvider.RegisterLicense("NDE1OTgzQDMxMzgyZTM0MmUzMFovQUh1STN4UVBoc0lsQkUvSGVEbmFXaFBUdVJHUzY5Yy83bnphSXgvK009");

base.OnCreate(savedInstanceState);

SfChart chart = new SfChart(this);

chart.Title.Text = "Equation chart";

chart.SetBackgroundColor(Color.White);

//Initializing primary axis

NumericalAxis primaryAxis = new NumericalAxis();

chart.PrimaryAxis = primaryAxis;

//Initializing secondary Axis

NumericalAxis secondaryAxis = new NumericalAxis();

chart.SecondaryAxis = secondaryAxis;

Collection<ChartData> dt = ReadTabulation();

ObservableCollection<ChartData> data = new ObservableCollection<ChartData>();

foreach (var item in dt)

{

data.Add(new ChartData (item.X, item.Y) );

}

//Initializing column series

LineSeries series = new LineSeries();

series.ItemsSource = data;

series.XBindingPath = "X";

series.YBindingPath = "Y";

series.TooltipEnabled = true;

chart.Series.Add(series);

SetContentView(chart);

}

public Collection<ChartData> ReadTabulation()

{

Collection<ChartData> dataList = new Collection<ChartData>();

ChartData data;

double x, y;

var backingFile = System.IO.Path.Combine(Android.OS.Environment.GetExternalStoragePublicDirectory(Android.OS.Environment.DirectoryDocuments).AbsolutePath, "results.txt");

using (var reader = new StreamReader(backingFile, true))

{

string line;

while ((line = reader.ReadLine()) != null)

{

string[] splitLine = line.Split(' ');

x = Convert.ToDouble(splitLine[0]);

y = Convert.ToDouble(splitLine[1]);

data = new ChartData(x, y);

dataList.Add(data);

}

}

return dataList;

}

}

}

ChartData.cs

namespace EquitationsCalculator

{

public class ChartData

{

public ChartData(double x, double y)

{

this.X = x;

this.Y = y;

}

public double X { get; set; }

public double Y { get; set; }

}

}