გრაფიკების აგება IMSL Chart ბიბლიოთეკის გამოყენებით

ლაშა იაშვილი, მიხეილ ნინუა, მარიამ თურმანიძე
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

კომპიუტერული ტექნოლოგიების ეპოქაში სისტემატურად ხორციელდება პროგრამული პაკეტების დახვეწა და განვითარება, ფართოვდება და იქმნება ახალი პროგრამული ბიბლიოთეკები, რომლებიც უფრო მარტივად წყვეტს საინჟინრო და სამეცნიერო სფეროს სხვადასხვა ამოცანებს. არსებული პროგრამული უზრუნველყოფის პირობებშიც გვიწევს დამატებითი ბიბლიოთეკების გამოყენება. შემოთავაზებულია დაპროგრამების მეთოდები, თუ როგორ შეიძლება გადაიჭრას კონკრეტული ამოცანა ან რომელი ბიბლიოთეკის გამოყენება არის შესაძლებელი მის გადასაწყვეტად. C# პროგრამული ენის გამოყენებით და ნებისმიერი Framework ვერსის პირობებში სხვადასხვა ორგანზომილებიანი გრაფიკების აგება შესაძლებელია Microsoft Visual Studio გარემოში, მაგრამ იგივე გარემოში ასევე შესაძლებელია ანალოგიური განზომილებიანი გრაფიკის აგება, უფრო მარტივად, IMSL Chart ბიბლიოთეკის გამოყენებით.

საკვანძო სიტყვები: C# . IMSL Chart. კომპიუტერული ტექნოლოგიები.

1. შესავალი - ბიბლიოთეკის დაყენება

ავაგოთ C# და ენის გამოყენებით გრაფიკები IMSL Chart ბიბლიოთეკის გამოყენებით. სანამ მთავარზე გადავიდოდეთ საჭიროა გავეცნოთ ბიბლიოთეკის ინსტალირებას და კონფიგურაციას. დაყენებისას საჭიროა ვიცოდეთ თუ რომელ დირექტორიაში დავაყენეთ. ჩვენ შემთხვევაში ესაა Program Files-ს VNI დირექტორია. ფაილის გადმოწერისას დავინახავთ, რომ მასში, საინსტალაციოს გარდა, არის სასწავლო დოკუმენტაცია, რომელიც მოყვება VNI კატალოგში.

2. ძირითადი ნაწილი

გამოვიყენოთ Microsoft Visual Studio 2015 ვერსია, ავირჩიოთ Windows Forms Application ფორმა. რადგანაც კონსოლურ აპლიკაციაში სახელსივრცეების დამატება მოგვიწევს, რაც თავიდანვე ისედაც დამატებულია. ჩართვისას, Solution Explorer-ში References-ზე მაუსის მარჯვენა ღილაკით ვირჩევთ Add Reference-s, შემდეგ Browse-ში და მოვძებნით IMSLCS.dll, რომელიც წესით უნდა იყოს VNI\imsl\imslcs500\bin დირექტორიაში. დავამატოთ სახელსივრცეში:

```
using Imsl.Chart2D;
```

ჩვენი კოდი გვიწერია კლასის კონსტრუქტორში, რომელშიც Chart და AxisXY კლასების ობიექტები გვაქვს გამოყენებული. აუცილებლად უნდა იყოს FrameChart კლასის მემკვიდრე. გარდა ამისა კლასები, რომლებშიც ფუნქციები იწერება აუცილებლად უნდა იყოს ChartFunction ინტერფეისის მემკვიდრეები. ეს ინტერფეისი მოითხოვს, რომ ფუნქციის ტიპი იყოს double.

```
class kvadratuli : ChartFunction
  {
    int a = 2, b = 2, c = 1;
    public double F(double x)
```

```
    {
      return a * x * x + b * x + c;
    }}
```

დავწეროთ კოდი და შევქმნათ რაიმე კლასი. ჩვენ შემთხვევაში ეს იქნება SampleProject. ასევე შევქმნათ უპარამეტრო კონსტრუქტორი.

```
public class SampleProject : FrameChart
  {
  }
public SampleProject()
    {
    }
```

ჩვენი ძირითადი კოდი არის სწორედ ამ კონსტრუქტორში. ჩვენ გვჭირდება Chart, AxisXY და Data კლასის ობიექტები. ერთს ერქმევა x ღერძი, ხოლო მეორეს - y. გამოგვაქვს ორი გრაფიკი, ერთი კვადრატული ფუნქციის და მეორე სინუსის. შესაბამისად დაგვჭირდება 2 Data კლასის ობიექტი (მონაცემები). ამ შემთხვევაში data-ს აქვს 4 პარამეტრი: Axsi, ფუნქცია, x-ის საწყისი და საბოლოო მნიშვნელობები. ამრიგად ჩვენი კონსტრუქტორი თავისი კოდით ასე გამოიყურება:

```
  public SampleProject()
    {
      Chart chart = this.Chart;
      AxisXY axis = new AxisXY(chart);
      axis.AxisX.SetTitle("x ღერძი");
      axis.AxisY.SetTitle("y ღერძი");

      chart.Legend.IsVisible = true;

      Data data1 = new Data(axis, new kvadratuli(), -2, 1);
      data1.LineColor = Color.Blue;
      data1.SetTitle("კვ.გრაფიკი");

      Data data2 = new Data(axis, new sinusi(), -10, 10);
      data2.LineColor = Color.Red;
      data2.SetTitle("სინუსი");
    }
```

თუ ამ კოდს პირდაპირ დააკოპირებთ, გექნებათ ერორი, რადგან კვადრატული და სინუსი კლასები ჯერ არ გაქვთ, ჯერ განვიხილოთ კვადრატული ფუნქცია:

```
class kvadratuli : ChartFunction
    {
        int a = 2, b = 2, c = 1;
        public double F(double x)
```

```
        {
            return a * x * x + b * x + c;
        }
    }
```

შემდეგ კი სინუსი:

```
class sinusi : ChartFunction
    {
        public double F(double x)
        {
            if (x == 0.0)
                return 1.0;

            return Math.Sin(Math.PI * x) / (Math.PI * x);
        }
    }
```

აქ, 0.0-ზე ერთი იმიტომ გავხადეთ, რომ NaN არ დაგვიწეროს, საბოლოოდ ჩემი მთლიანი კოდი ასე გამოიყურება:

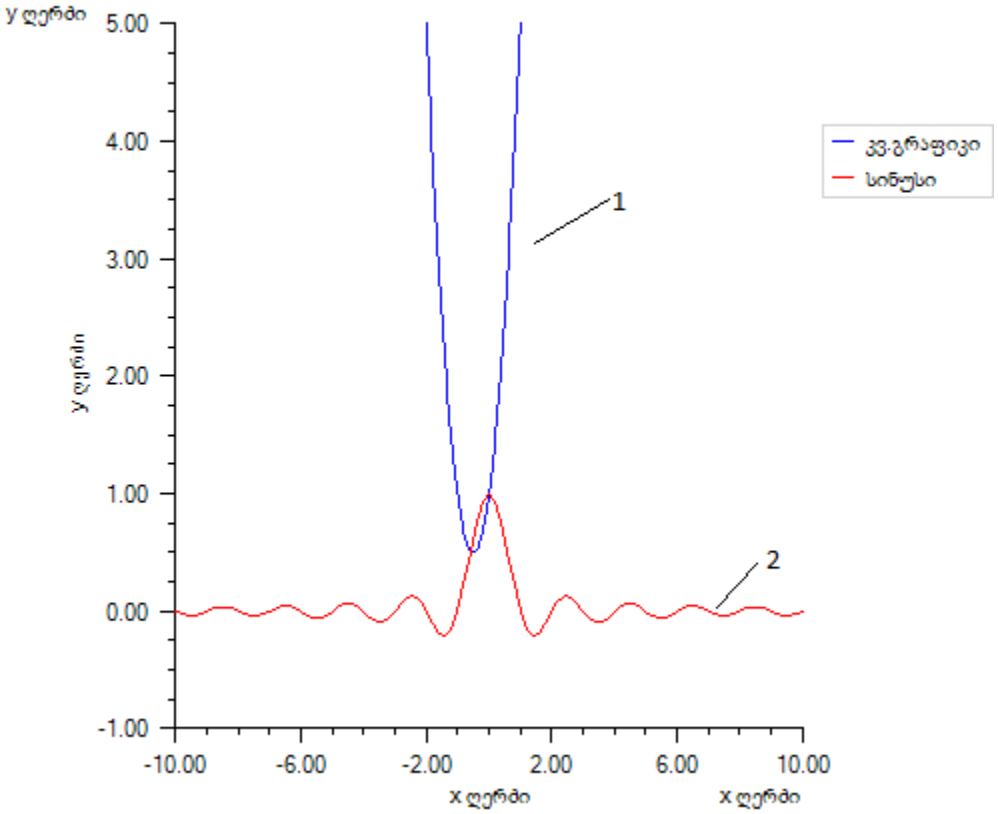
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Imsl.Chart2D;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
    }

    public class SampleProject : FrameChart
    {

        public SampleProject()
        {
            Chart chart = this.Chart;
            AxisXY axis = new AxisXY(chart);
```

```
            axis.AxisX.SetTitle("x ღერძი");
            axis.AxisY.SetTitle("y ღერძი");

            chart.Legend.IsVisible = true;

            Data data1 = new Data(axis, new kvadratuli(), -2, 1);
            data1.LineColor = Color.Blue;
            data1.SetTitle("კვ.გრაფიკი");

            Data data2 = new Data(axis, new sinusi(), -10, 10);
            data2.LineColor = Color.Red;
            data2.SetTitle("სინუსი");
        }
    }

    class kvadratuli : ChartFunction
    {
        int a = 2, b = 2, c = 1;
        public double F(double x)
        {
            return a * x * x + b * x + c;
        }
    }

    class sinusi : ChartFunction
    {
        public double F(double x)
        {
            if (x == 0.0)
                return 1.0;

            return Math.Sin(Math.PI * x) / (Math.PI * x);
        }
    }
}
```

კოდი რომ გაიშვას Program.cs-ში გამშვები კოდი უნდა შეიცვალოს

```
Application.Run(new SampleProject());
```

ეს არის ჩვენი მთავარი კლასის სახელი, ასე რომ თქვენს კლასს თუ სხვა რამე დაარქვით, მაშინ ის სახელი მიუთითეთ. ჩვენი კოდის გამოყენების შემდეგ მიიღებთ იმ შედეგს რომელიც სურ.1 არის ნაჩვენები.



ნახ.1. გრაფიკი, 1-კვადრატული ფუნქცია, 2-სინუსის ფუნქცია

3. დასკვანა:

გამოვიყენეთ Microsoft Visual Studio, ავირჩიეთ Windows Forms Application-ის ფორმა, სადაც C_sharp ობექტური ენის და ახალი IMSL Chart ბიბლიოთიკის გამოყენებით ავაგეთ ორგანზომილებიანი გრაფიკი. შესაძლებელია სამეცნიერო ამოცანების ამოხსნა და შედეგების ნახვა გრაფიკის სახით. მოცემული ერთ-ერთი მეთოდი შესაძლებას იძლევა ახალი ბიბლიოთიკის დასამატებლად და უფრო მარტივად გამოყენება ვიდრე Visual Studio გვთავაზობს. ასევე განხილულია ვიზუალიზაციისთვის როგორც ცალკელ კლასები, ასევე მთლიანი კოდი მარტივ მაგალითზე და მოცემულია შედეგები გრაფიკის სახით.

ლიტერატურა:

1. იაშვილი ლ., ნინუა მ. (2015). IMSL Chart. www.scripts.ge

2. სმიტი თ. (2015). IMSL Chart დოკუმენტაცია. http://www.roguewave.com/help-support-/documentation/imsl-numerical-libraries

BUILT CHARTS USING IMSL CHART LIBRARY

Iashvili Lasha, Ninua Mikheil, Turmanidze Mariam
Georgian Technical University

Summary

Every day takes place in the era of computer technology to refine and develop software packages, set up new software packages and libraries, which are more easily decide the different types of tasks. We have to use additional libraries in the conditions of the software, is offered a variety of programs, methods on how to solve a specific task, or that the library may be used for a given task. The programming language C #, using a variety of two-dimensional graphs, and any framework miles conditions can be constructed with Microsoft Visual Studio environment, but the environment as well as possible a similar dimensional graph, more simply, IMSL Chart library.

ПОСТРОЕНИЕ ГРАФИКОВ С ИСПОЛЬЗОВАНИЕМ IMSL CHART БИБЛИОТЕКИ

Иашвили Л., Нинуа М., Турманидзе М.
Грузинский Технический Университет

Резюме

Компьютерные технологии в эпоху систематического совершенствования и развития программных пакетов, расширяются и создают новые программные библиотеки, которые более легко решают различные задачи инженерных и научных сфер. Мы должны использовать дополнительные библиотеки в условиях существующего программного обеспечения. Предлагаемые методы программирования позволяют как решить определенную задачу, так и определить библиотеку, которая может быть использована для ее решения. Использование языка программирования C# позволяет строить различные двумерные графики в среде Microsoft Visual Studio. В этой же среде возможно построение также аналового графика с использованием IMSL Chart библиотеки.