**O‘ZBEKISTON RESPUBLIKASI  
MIRZO ULUG’BEK NOMIDAGI  
O‘ZBEKISTON MILLIY UNIVERSITETI**

**AMALIY MATEMATIKA YO‘NALISHI**

**TMI-2303-MAXSUS GURUH TALABASI  
Paritdinov Samariddin Muzaffar o‘g‘lining**

**“ALGORITMIK TILLAR VA DASTURLASH”** **fanidan**

**“Turli sanoq tizimlarida ishlovchi universal kalkulyator dasturini ishlab chiqish” mavzusida yozilgan**

**KURS ISHI**

**Topshirdi: PARITDINOV SAMARIDDIN**

**Kurs ishi rahbari: XAJIYEV IKROM**

**Topshirilgan sana: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

TOSHKENT – 2025

**MUNDARIJA**

KIRISH ............................................................................................................... 2   
1-BOB. HISOBLASH TIZIMLARI VA DASTURIY YECHIMLAR   
1.1. Sanoq sistemalari va ularning qo‘llanilishi ................................................. 3  
1.2. Ilmiy funksiyalarni kompyuterda hisoblash nazariyasi ............................... 4  
1.3. C# Windows Forms texnologiyasi asoslari ................................................. 5

2-BOB. UNIVERSAL KALKULYATOR DASTURINI ISHLAB CHIQISH   
2.1. Texnik topshiriq .......................................................................................... 7  
2.2. Tizim arxitekturasi va modullari ................................................................. 8  
2.3. Asosiy interfeys (Form1) dizayni va funksiyalari ....................................... 9  
2.4. Hisoblash algoritmlari va kod tuzilmasi .................................................... 13  
2.5. Ilmiy funksiyalarni qo‘llash (Sin, Cos, Log, Sqrt, Pow, va b.) ................. 15  
2.6. Sanoq sistemalari bilan ishlash funksiyalari ............................................. 20  
2.7. Klaviatura orqali kiritmalarni qayta ishlash (KeyDown) .......................... 24  
2.8. Tarixni faylga yozish va uni ko‘rish (Form2) ........................................... 27  
2.9. Dasturdan foydalanish bo‘yicha misollar ................................................. 29  
2.10. Rivojlantirish yo‘nalishlari ..................................................................... 32

XULOSA ......................................................................................................... 33  
GLOSSARIY ................................................................................................... 34  
FOYDALANILGAN ADABIYOTLAR ......................................................... 36

**KIRISH**

Bugungi raqamli dunyoda murakkab matematik ifodalarni hisoblash, sonlar ustida sanoq sistemalari asosida amallar bajarish, trigonometrik va logarifmik funksiyalarni hisoblash zarurati kundalik hayotda va ilmiy-tadqiqot sohalarida juda katta ahamiyatga ega. Shu boisdan bunday funksiyalarni avtomatik bajaruvchi, foydalanuvchiga qulay interfeysga ega, zamonaviy grafik interfeys asosida ishlab chiqilgan dasturiy ta'minotga ehtiyoj kundan-kunga ortib bormoqda.

Mazkur kurs ishida universal ko‘rinishdagi, ya'ni bir vaqtning o‘zida turli sanoq sistemalarida (ikkilik, sakkizlik, o‘nlik, o‘n oltilik va umumiy holatda ixtiyoriy sanoq sistemalarida) ishlay oluvchi, ilmiy funksiyalarni hisoblab bera oladigan, foydalanuvchi kiritgan ifodalarni tarixga yozib boruvchi, klaviatura yordamida boshqarish imkoniyatiga ega kalkulyator dasturi ishlab chiqilgan.

Kalkulyatorning asosiy imkoniyatlari:

* Har xil sanoq sistemalarida ishlash;
* Trigonometrik (sin, cos, tg, ctg) va logarifmik (log, ln, lg) funksiyalarni hisoblash;
* Modul, ildiz, daraja kabi matematik amallarni bajarish;
* Klaviatura tugmalari orqali boshqarish (0-9, A-F, +, -, \*, /, Enter, Backspace);
* Har bir ifoda va natijani history.txt fayliga yozish;
* Tarixni maxsus Form2 orqali ko‘rish va tozalash.

Dastur C# tili va Windows Forms texnologiyasi asosida ishlab chiqilgan bo‘lib, sodda, tushunarli va qulay interfeysga ega. Bunda foydalanuvchi hech qanday texnik bilimga ega bo‘lmagan holda ham dasturdan samarali foydalanishi mumkin.

Kurs ishining maqsadi – foydalanuvchiga matematik hisoblashlarni imkon qadar tez va aniq bajarish imkoniyatini beruvchi, ko‘p funksiyali va turli sanoq sistemalarida ishlovchi kalkulyator dasturini ishlab chiqish.

**1-BOB. HISOBLASH TIZIMLARI VA DASTURIY YECHIMLAR**

**1.1. Sanoq sistemalari va ularning qo‘llanilishi**

Sanoq sistemasi — bu sonlarni belgilangan tartibda raqamlar orqali ifodalash usulidir. Har bir sanoq sistemasining o‘ziga xos asos (baza)si bo‘ladi va shu asosga ko‘ra sonlar shakllantiriladi. Asos deb sistemada nechta turli raqam ishlatilishini tushunamiz. Masalan, o‘nlik sanoq sistemasida 10 ta raqam (0 dan 9 gacha) mavjud bo‘lsa, ikkilik sanoq sistemasida atigi 2 ta raqam (0 va 1) mavjud bo‘ladi.  
  
Kompyuter texnologiyalarida turli xil sanoq sistemalari qo‘llaniladi, chunki kompyuterlarning ishlash mexanizmi inson miyasidan farq qiladi. Quyida asosiy sanoq sistemalari va ularning dasturlash hamda texnik sohalardagi qo‘llanilishi keltirilgan:

**Ikkilik sanoq sistemasi (Binary, baza 2)**

- Faqat 0 va 1 raqamlaridan iborat.  
- Kompyuterlarning barcha ma’lumotlarni qayta ishlash jarayoni aynan ikkilik sistema asosida amalga oshiriladi.  
- Har bir bit (binary digit) ikkilik sistemadagi bitta raqamni ifodalaydi.  
- Masalan, 1101 ikkilik soni o‘nlik sistemasida 13 ga teng:  
 1×2³ + 1×2² + 0×2¹ + 1×2⁰ = 8 + 4 + 0 + 1 = 13

**Sakkizlik sanoq sistemasi (Octal, baza 8)**

- 0 dan 7 gacha bo‘lgan raqamlardan iborat.  
- Dasturiy ta’minot va apparat darajasida ba’zan qisqaroq ifoda uchun ishlatiladi.  
- Ikkilik sonlarni 3 bitli guruhga bo‘lib o‘qish orqali sakkizlikka o‘tkazish mumkin.  
- Masalan, ikkilik 110111 soni sakkizlikda 67 bo‘ladi.

**O‘nlik sanoq sistemasi (Decimal, baza 10)**

- Insonlar eng ko‘p foydalanadigan sanoq tizimi.  
- 0 dan 9 gacha raqamlar ishlatiladi.  
- Barcha kundalik hisob-kitoblar va arifmetik amallar aynan shu sistemada bajariladi.

**O‘n oltilik sanoq sistemasi (Hexadecimal, baza 16)**

- 0–9 raqamlari va A–F harflari ishlatiladi (A = 10, B = 11, ..., F = 15).  
- Kompyuter xotirasi, rang kodlari, assembly tilidagi manzillar va boshqa past darajadagi dasturlash sohalarida keng qo‘llaniladi.  
- Har bir 4 bitli ikkilik son bitta o‘n oltilik raqamga to‘g‘ri keladi.  
- Masalan, 2F (16 lik) o‘nlikda:  
 2×16 + 15 = 47

**Sanoq sistemalarining o‘zaro konvertatsiyasi**

Turli sanoq sistemalarida sonlarni ifodalash imkoniyati foydalanuvchiga moslashuvchanlik yaratadi. Masalan:  
- 1101 (ikkilik) = 13 (o‘nlik)  
- 15 (o‘nlik) = F (16 lik)  
- 77 (sakkizlik) = 63 (o‘nlik)

**Kalkulyator dasturidagi qo‘llanilishi**

Tuzilgan dastur foydalanuvchiga quyidagi imkoniyatlarni beradi:  
- Foydalanuvchi istalgan sanoq sistemasini tanlaydi (2, 8, 10, 16).  
- Kiritingan sonlar tanlangan sistema asosida kiritiladi.  
- Dastur barcha hisob-kitoblarni ichki tarzda o‘nlik sanoq sistemasida bajaradi.  
- Hisoblangan natija foydalanuvchining tanlagan sanoq sistemasiga avtomatik tarzda o‘tkazilib, natija shu ko‘rinishda chiqariladi.  
  
Bu yondashuv foydalanuvchiga interfeysni oddiy va tushunarli tutish imkonini beradi, ammo dastur ichki ishlov berishda murakkab konvertatsiya jarayonlarini bajaradi. Aynan shu holat dasturiy yechimda modulli, mustaqil konvertatsiya funksiyalarining mavjudligini talab qiladi.

**1.2.** Ilmiy funksiyalarni kompyuterda hisoblash nazariyasi

Matematik funksiyalarni kompyuterda hisoblash uchun ko‘plab algoritmlar ishlab chiqilgan. Standart .NET kutubxonasida quyidagi funksiyalar mavjud:

* Math.Sin(x) – sinus qiymatini hisoblaydi
* Math.Cos(x) – kosinus
* Math.Tan(x) – tangens
* Math.Sqrt(x) – ildiz
* Math.Abs(x) – modul
* Math.Pow(a,b) – darajaga oshirish
* Math.Log(x) – natural logarifm (ln)
* Math.Log10(x) – o‘n asosli logarifm (lg)
* Math.Log(a,b) – a asosli logarifm ()

Bularning barchasi kalkulyator dasturida mos ravishda btnSin, btnCos, btnLog, btnPow kabi tugmalar orqali chaqiriladi. Dasturda radian va gradus rejimlari ham mavjud. Agar foydalanuvchi gradus rejimida ishlayotgan bo‘lsa, kiritilgan qiymat avtomatik radianlarga o‘tkaziladi:

x (rad) = x (gradus) \* PI / 180

Shuningdek, foydalanuvchi daraja va ildiz kabi kompleks amallarni ham bajarishi mumkin. Bularning barchasi oldindan yozilgan funksiyalar orqali natijaga aylantiriladi va ifodalar NCalc kutubxonasi yordamida baholanadi.

**1.3. C# Windows Forms texnologiyasi asoslari**

C# Windows Forms (WinForms) — bu Microsoft .NET Framework asosida ishlab chiqilgan va foydalanuvchi interfeysi (UI) yaratish uchun mo‘ljallangan texnologiyalardan biridir. U ish stoli (desktop) ilovalari ishlab chiqishda keng qo‘llaniladi va o‘zida obyektga yo‘naltirilgan dasturlash (OOP) prinsiplarini mujassam etadi.

Windows Forms texnologiyasi yordamida foydalanuvchi interfeysining quyidagi elementlarini yaratish mumkin:

* **Formalar** – asosiy va qo‘shimcha oynalar
* **Tugmalar (Button)** – foydalanuvchi harakatlarini qabul qiluvchi komponentlar
* **Label, TextBox** – matn ko‘rsatish va kiritish elementlari
* **RadioButton, CheckBox** – tanlov variantlari
* **Chart, PictureBox** – grafik va rasm ko‘rinishidagi komponentlar

C# Windows Forms ilovasi Visual Studio muhitida yaratiladi. Dizayn oynasi orqali foydalanuvchi interfeysini yaratish osonlashadi, barcha hodisalar (eventlar) esa avtomatik tarzda kod bilan bog‘lanadi. Misol uchun, Click hodisasi yordamida foydalanuvchi tugma bosganda bajarilishi kerak bo‘lgan kod yoziladi:

private void btnEqual\_Click(object sender, EventArgs e)

{

// hisoblash funksiyasi shu yerda bajariladi

}

WinForms texnologiyasining afzalliklari:

* **Sodda va tez ishlab chiqish**: Drag-and-drop interfeys orqali dizaynni yaratish mumkin
* **Boy komponentlar to‘plami**: Label, ComboBox, DatePicker, DataGridView va boshqalar
* **Hodisa asosidagi dasturlash**: Har bir foydalanuvchi harakatiga mos hodisa bilan javob beriladi
* **Ma’lumotlar bazasi bilan integratsiya qilish oson**: ADO.NET, Entity Framework orqali

Kamchiliklari:

* Zamonaviy veb interfeyslarga nisbatan cheklangan dizayn imkoniyatlari
* Platformaga bog‘langan (faqat Windows)

Kalkulyator dasturimizda aynan Windows Forms texnologiyasi tanlandi, chunki u:

* Talaba va o‘rta darajadagi foydalanuvchi uchun yetarlicha qulay;
* Kodni modullashtirish (ajratish) imkonini beradi;
* Har bir tugmani alohida funksiya bilan bog‘lash mumkin;
* Klaviatura hodisalarini tutish va qayta ishlashni qo‘llab-quvvatlaydi.

Shu sababli, universal kalkulyator loyihasini ishlab chiqishda Windows Forms C# texnologiyasi eng maqbul va amaliy tanlov bo‘ldi.

**2-BOB. UNIVERSAL KALKULYATOR DASTURINI ISHLAB CHIQISH**

**2.1. Texnik topshiriq**

Dastur nomi: Universal Ilmiy Kalkulyator

Ishlab chiquvchi: PARITDINOV SAMARIDDIN

Texnologiya: C# tilida, Windows Forms muhitida

Kutubxonalar: .NET Framework 4.0, NCalc (matematik ifodalarni hisoblash uchun)

Platforma: Windows operatsion tizimi

Foydalanuvchi qurilmasi: klaviatura + sichqoncha

Dastur quyidagi asosiy funksiyalarni bajarishi kerak:

* Raqamli (0-9), harfiy (A-F) va kasrli sonlar bilan ishlash
* 2, 8, 10, 16 (va ixiyoriy boshqa) sanoq sistemalari o‘rtasida konvertatsiya
* Trigonometrik funksiyalar: sin, cos, tg, ctg
* Logarifmik funksiyalar: log, ln, lg
* Matematik amallar: ildiz, modul, daraja, e va π qiymatlari
* Klaviatura yordamida kiritma qabul qilish
* Foydalanuvchi ifodalarini faylga yozish
* Tarixni alohida formda ko‘rish (Form2)
* Tugma bosilganda tegishli amalni bajarish (masalan: +, -, \*, /)

**2.2. Tizim arxitekturasi va modullari**

Kalkulyator dasturi quyidagi qatlamlardan iborat:

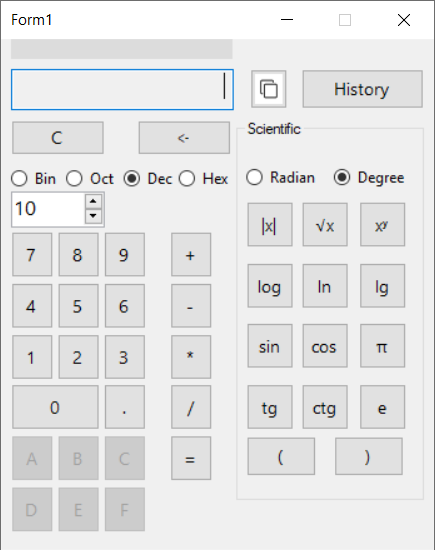
1. **Interfeys (UI) qatlami**:
   * Form1 va Form2 oynalari
   * Tugmalar (Button), matn oynalari (TextBox, Label)
2. **Hisoblash logikasi**:
   * NCalc kutubxonasi orqali ifodalarni tahlil qilish va hisoblash
   * Matematik funksiyalar uchun maxsus metodlar: AddScientificFunc, ConvertToDecimal, ConvertBase
3. **Tarix bilan ishlash moduli**:
   * history.txt faylga yozish
   * Form2 da o‘qish va tozalash
4. **Klaviatura kiritmalarini qayta ishlovchi modul**:
   * Form1\_KeyDown metodi orqali barcha kerakli tugmalarni ushlash
5. **Sanoq sistemasini boshqarish**:
   * numericBase va radioButtonlar orqali tanlash
   * Har bir tizimda amal qiladigan tugmalarni faollashtirish yoki o‘chirish

Tizim oddiy, ammo kengaytiriladigan ko‘rinishda ishlab chiqilgan. Har bir komponent alohida metodlar orqali modular tuzilgan bo‘lib, test qilish va qayta ishlash qulay.

**2.3. Asosiy interfeys (Form1) dizayni va funksiyalari**

Form1 kalkulyatorning asosiy oynasi bo‘lib, unda quyidagi komponentlar mavjud:

* txtResult – hozirgi kiritilayotgan sonni ko‘rsatish uchun matn oynasi
* lblHistory – hozirgi amal ifodasini ko‘rsatish uchun label
* btnEqual, btnClear, btnBackspace – standart tugmalar
* Ilmiy funksiyalar: btnSin, btnCos, btnSqrt, btnLog, btnPow va boshqalar
* Sanoq sistemalari uchun: rbtBin, rbtOct, rbtDec, rbtHex radio tugmalar
* numericBase – hozirgi sanoq sistemasini o‘zgartiruvchi raqamli maydon



Form1 o‘zida barcha hisoblash, interfeys boshqaruvi, va klaviatura hodisalarini birlashtiradi. Foydalanuvchi harakatlariga mos ravishda txtResult va lblHistory qiymatlari yangilanadi, hisob natijasi esa btnEqual\_Click orqali chiqariladi.

Dizaynda asosiy e'tibor qulaylik, soddalik va funksionallikka qaratilgan. Har bir tugma Tag orqali aniqlanib, hozirgi sanoq sistemasiga mos bo‘lsa, Enabled = true qilinadi. Bu esa noto‘g‘ri amal va raqam kirishining oldini oladi.

using System;

using System.Text;

using System.Windows.Forms;

using NCalc;

using System.IO;

namespace Kurs\_Ishi\_4\_sem

{

public partial class Form1 : Form

{

bool isNewEntry = true;

bool lastInputWasFunction = false;

bool isDegreeMode = true;

int currentBase = 10;

string userExpression = "";

string calcExpression = "";

string pendingFunction = "";

string pendingArgument = "";

string historyFilePath = @"C:\Users\user\Desktop\Paritdinov\_Samariddin\_Kurs\_ishi\_2-kurs\_4-sem\history.txt";

public Form1()

{

InitializeComponent();

}

private void AddInput(string value)

{

if (isNewEntry || txtResult.Text == "0")

txtResult.Text = value;

else

txtResult.Text += value;

userExpression += value;

lblHistory.Text = userExpression;

isNewEntry = false;

}

private void Operator\_Click(object sender, EventArgs e)

{

Button btn = sender as Button;

string opSymbol = btn.Text;

if (pendingFunction != "")

{

string secondVal = txtResult.Text;

userExpression += ")";

calcExpression += ConvertBaseToDecimal(secondVal, currentBase).ToString() + ")";

pendingFunction = "";

pendingArgument = "";

txtResult.Text = "0";

isNewEntry = true;

lastInputWasFunction = true; // Bu yer muhim!

}

else if (!isNewEntry && !lastInputWasFunction)

{

string lastInput = txtResult.Text;

string decimalVal = ConvertBaseToDecimal(lastInput, currentBase).ToString();

calcExpression += decimalVal;

}

// Ketma-ket operatorlarni tekshirish va almashtirish

if (userExpression.Length > 0)

{

char lastChar = userExpression[userExpression.Length - 1];

// Agar oxirgi belgilar operator bo‘lsa uni yangisiga almashtiramiz

if ("+-\*/".IndexOf(lastChar) >= 0)

{

userExpression = userExpression.Substring(0, userExpression.Length - 1);

calcExpression = calcExpression.Substring(0, calcExpression.Length - 1);

}

}

userExpression += opSymbol;

calcExpression += opSymbol;

lblHistory.Text = userExpression;

txtResult.Text = "0";

isNewEntry = true;

lastInputWasFunction = false;

}

private void btnEqual\_Click(object sender, EventArgs e)

{

// Agar pendingFunction bo'lsa, ikkinchi argumentni yakunlab qo'shamiz

if (pendingFunction != "")

{

string secondVal = txtResult.Text;

userExpression += secondVal + ")";

calcExpression += ConvertBaseToDecimal(secondVal, currentBase).ToString() + ")";

pendingFunction = "";

pendingArgument = "";

}

try

{

if (!isNewEntry && !lastInputWasFunction)

{

string lastInput = txtResult.Text;

string decimalVal = ConvertBaseToDecimal(lastInput, currentBase).ToString();

calcExpression += decimalVal;

}

Expression exp = new Expression(calcExpression);

object result = exp.Evaluate();

string finalResult = result.ToString();

if (currentBase != 10)

finalResult = ConvertBase(finalResult, 10, currentBase);

string historyLine = userExpression + " = " + finalResult + " ; Base=" + currentBase;

File.AppendAllText(historyFilePath, historyLine + Environment.NewLine);

txtResult.Text = finalResult;

listBox1.Items.Add(calcExpression);

userExpression = "";

calcExpression = "";

}

catch (Exception ex)

{

MessageBox.Show("Hisoblashda xatolik: " + ex.Message);

}

isNewEntry = true;

}

**2.4. Hisoblash algoritmlari va kod tuzilmasi**

Kalkulyatorning markaziy hisoblash mexanizmi btnEqual\_Click metodida joylashgan bo‘lib, foydalanuvchi kiritgan ifoda NCalc.Expression obyektiga uzatiladi va .Evaluate() orqali hisoblanadi.

Misol:

Expression exp = new Expression(calcExpression);

object result = exp.Evaluate();

Hisoblashdan oldin, calcExpression nomli o‘zgaruvchi orqali barcha amal va raqamlar (o‘nlikka o‘tkazilgan ko‘rinishda) jamlanadi. Foydalanuvchi interfeysda A+2 kiritsa, dastur uni 10+2 deb baholaydi.

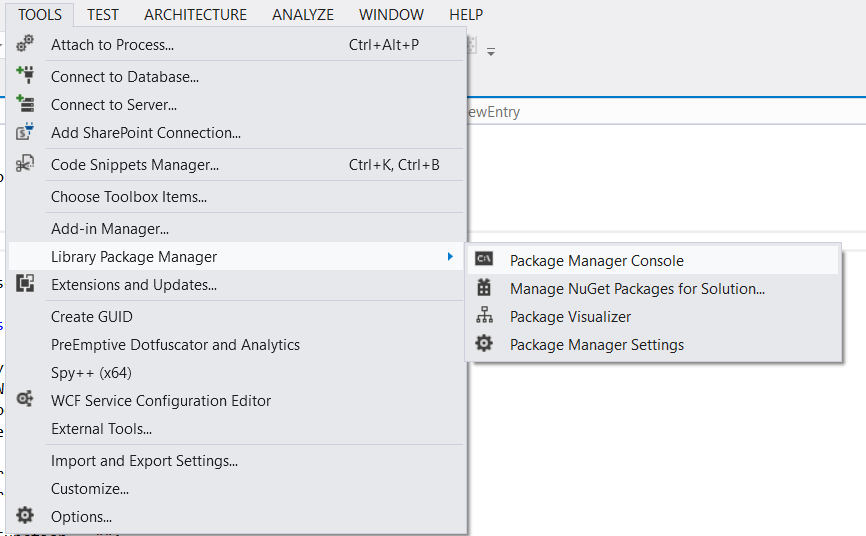
Natijada olingan qiymat txtResult oynasiga chiqariladi, va userExpression = ... = natija ko‘rinishida history.txt fayliga yoziladi:

**Nega aynan NCalc ishlatildi?**  
Kalkulyatorda arifmetik ifodalarni to‘g‘ri hisoblash muhim vazifadir. Masalan, foydalanuvchi 2 + 15 \* 2 deb kiritsa, to‘g‘ri natija 32 bo‘lishi kerak (15\*2 = 30, keyin 2 + 30). Oddiy Convert.ToDouble() yoki DataTable.Compute() kabi metodlar bu ifodani ketma-ketlikda hisoblab, xatolikka yo‘l qo‘yadi.

Ammo **NCalc** kutubxonasi matematik ifodalarni **to‘liq sintaktik daraxt asosida** baholaydi va **ustuvorlik (priority)** qoidalariga to‘liq amal qiladi. Shuningdek, Sin, Cos, Log, Pow, Abs kabi funksiyalarni ham qo‘llab-quvvatlaydi.

Bu kutubxona string ko‘rinishidagi ifodalarni analiz qiladi va to‘g‘ridan-to‘g‘ri natija qaytaradi. Shu sababli NCalc - aynan ilmiy kalkulyator uchun eng maqbul yechimdir.

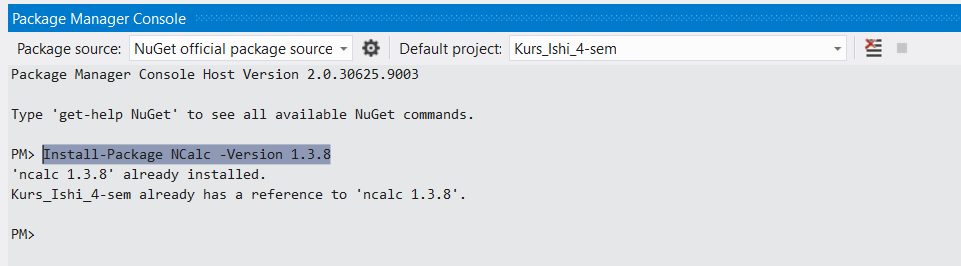
Ncalc ni yuklash:



Ochilgan oynada shunday yozamiz:

Install-Package NCalc -Version 1.3.8

Quyidagicha:



Endi yuqoriga

using NCalc;

Kabi yozib qo’ysak yetarli

**2.5. Ilmiy funksiyalarni qo‘llash (Sin, Cos, Log, Sqrt, Pow, va b.)**

Ilmiy funksiyalarni hisoblash uchun umumiy AddScientificFunc metodi yozilgan. Bu metodga funksiyaning nomi va hisoblovchi delegate uzatiladi:

private void AddScientificFunc(string name, Func<double, double> mathFunc)

{

string input = txtResult.Text;

string inputDecimal = ConvertToDecimal(input);

double inputVal = Convert.ToDouble(inputDecimal);

double computed = mathFunc(inputVal);

...

txtResult.Text = computed.ToString();

}

Agar foydalanuvchi daraja (Pow) yoki logarifm (Log) funksiyasini tanlasa, ular ikki argumentli bo‘lgani uchun pendingFunction va pendingArgument orqali vaqtinchalik saqlanadi:

pendingFunction = "Pow";

pendingArgument = txtResult.Text;

Ikkinchi qiymat kiritilgach, btnEqual tugmasi bosilganda bu funksiya yakunlanadi.

private void AddScientificFunc(string name, Func<double, double> mathFunc)

{

if (!isNewEntry)

{

string input = txtResult.Text;

double inputVal = ConvertBaseToDecimal(input, currentBase);

string degreeSuffix = "";

if (isDegreeMode && (name == "Sin" || name == "Cos" || name == "Tan"))

{

inputVal = inputVal \* Math.PI / 180;

degreeSuffix = "°";

}

double computed = mathFunc(inputVal);

// agar juda kichik qiymat chiqsa uni 0 deb olish maqsad sin(180) dan juda kichik son chiqadi shuni 0 deb olib ketish

if (Math.Abs(computed) < 1e-6) computed = 0;

// Oldingi sonni olib tashlash

if (userExpression.EndsWith(input))

userExpression = userExpression.Substring(0, userExpression.Length - input.Length);

userExpression += name + "(" + input + degreeSuffix + ")";

calcExpression += computed;

txtResult.Text = computed.ToString();

lblHistory.Text = userExpression;

isNewEntry = true;

lastInputWasFunction = true;

}

}

private void btnSin\_Click(object sender, EventArgs e)

{

AddScientificFunc("Sin", Math.Sin);

}

private void btnCos\_Click(object sender, EventArgs e)

{

AddScientificFunc("Cos", Math.Cos);

}

private void btnTan\_Click(object sender, EventArgs e)

{

AddScientificFunc("Tan", Math.Tan);

}

private void btnCtg\_Click(object sender, EventArgs e)

{

string input = txtResult.Text;

double angle = ConvertBaseToDecimal(input, currentBase);

if (isDegreeMode) angle = angle \* Math.PI / 180;

if (Math.Tan(angle) == 0)

{

MessageBox.Show("ctg aniqlanmagan");

return;

}

double val = 1 / Math.Tan(angle);

userExpression += "ctg(" + input + (isDegreeMode ? "°" : "") + ")";

calcExpression += val;

txtResult.Text = val.ToString();

lblHistory.Text = userExpression;

isNewEntry = true;

lastInputWasFunction = true;

}

private void btnSqrt\_Click(object sender, EventArgs e)

{

AddScientificFunc("Sqrt", Math.Sqrt);

}

private void btnAbs\_Click(object sender, EventArgs e)

{

AddScientificFunc("Abs", Math.Abs);

}

private void btnLg\_Click(object sender, EventArgs e)

{

AddScientificFunc("Log10", Math.Log10);

}

private void btnLog\_Click(object sender, EventArgs e)

{

pendingFunction = "Log";

pendingArgument = txtResult.Text;

string inputDecimal = ConvertBaseToDecimal(pendingArgument, currentBase).ToString();

if (userExpression.EndsWith(pendingArgument))

userExpression = userExpression.Substring(0, userExpression.Length - pendingArgument.Length);

if (calcExpression.EndsWith(inputDecimal))

calcExpression = calcExpression.Substring(0, calcExpression.Length - inputDecimal.Length);

userExpression += "Log(" + pendingArgument + ",";

calcExpression += "Log(" + inputDecimal + ",";

txtResult.Text = "0";

lblHistory.Text = userExpression;

isNewEntry = true;

lastInputWasFunction = true;

}

private void btnLn\_Click(object sender, EventArgs e)

{

AddScientificFunc("Log", Math.Log);

}

private void btnPow\_Click(object sender, EventArgs e)

{

pendingFunction = "Pow";

pendingArgument = txtResult.Text;

string inputDecimal = ConvertBaseToDecimal(pendingArgument, currentBase).ToString();

// eski sonni o'chirish

if (userExpression.EndsWith(pendingArgument))

userExpression = userExpression.Substring(0, userExpression.Length - pendingArgument.Length);

if (calcExpression.EndsWith(inputDecimal))

calcExpression = calcExpression.Substring(0, calcExpression.Length - inputDecimal.Length);

userExpression += "Pow(" + pendingArgument + ",";

calcExpression += "Pow(" + inputDecimal + ",";

txtResult.Text = "0";

lblHistory.Text = userExpression;

isNewEntry = true;

lastInputWasFunction = true;

}

private void btnE\_Click(object sender, EventArgs e)

{

string val = Math.E.ToString();

userExpression += "e";

calcExpression += val;

txtResult.Text = val;

lblHistory.Text = userExpression;

isNewEntry = true;

}

private void btnPi\_Click(object sender, EventArgs e)

{

string val = Math.PI.ToString();

userExpression += "π";

calcExpression += val;

txtResult.Text = val;

lblHistory.Text = userExpression;

isNewEntry = true;

}

private void rbtfordagree\_CheckedChanged(object sender, EventArgs e)

{

isDegreeMode = rbtDegree.Checked;

}

**2.6. Sanoq sistemalari bilan ishlash funksiyalari**

Dastur istalgan sanoq sistemasida ishlay oladi. Asosiy currentBase o‘zgaruvchisi tanlangan sistemani ifodalaydi. ConvertBaseToDecimal metodida istalgan sistema raqami o‘nlikka o‘tkaziladi:

Hisobdan so‘ng, natijani foydalanuvchiga tanlangan sistemada ko‘rsatish uchun ConvertBase funksiyasi ishlatiladi. Bu metod butun va kasr qismlarini ajratib, har birini yangi sistema asosida qayta hisoblaydi.

numericBase.ValueChanged hodisasi orqali interfeysdagi tugmalar avtomatik Enabled = true/false qilinadi. Bu foydalanuvchining noto‘g‘ri kiritmalar oldini oladi.

Hisob kitobda xato ketmasligi uchun ikkita o’zgaruvchi yaratilgan userexpression va calcexpression maqsad shunda ediki birinchisiga foydalanuvchi ko’rishi uchun saqlanadi, ikkinchisiga esa Ncalc hisoblashi uchun 10 lik sanoq sistemaga o’tkazib saqlab boriladi hisob kitob uchun natija chiqgandan keyin uni yana user tanlagan sanoq sistemaga o’tkaziladi

private void CheckedChanged(object sender, EventArgs e)

{

if (rbtBin.Checked) numericBase.Value = 2;

if (rbtOct.Checked) numericBase.Value = 8;

if (rbtDec.Checked) numericBase.Value = 10;

if (rbtHex.Checked) numericBase.Value = 16;

}

private void numericBase\_ValueChanged(object sender, EventArgs e)

{

int oldbase = currentBase;

currentBase = (int)numericBase.Value;

userExpression = "";

calcExpression = "";

string txt = txtResult.Text;

if (txt != "" && txt != "0")

{

string natija = ConvertBase(txt.ToString(), oldbase, currentBase);

txtResult.Text = natija;

userExpression = natija;

lblHistory.Text = "";

isNewEntry = false;

}

if (currentBase == 2) rbtBin.Checked = true;

if (currentBase == 8) rbtOct.Checked = true;

if (currentBase == 10) rbtDec.Checked = true;

if (currentBase == 16) rbtHex.Checked = true;

foreach (Control ctrl in this.Controls)

{

Button btn = ctrl as Button;

if (btn != null && btn.Tag != null)

{

string tag = btn.Tag.ToString();

int value;

if (int.TryParse(tag, out value)) // Raqamlar

{

btn.Enabled = value < currentBase;

}

else if (tag.Length == 1 && char.IsLetter(tag[0])) // Harflar (A-F)

{

value = 10 + (char.ToUpper(tag[0]) - 'A');

btn.Enabled = value < currentBase;

}

}

}

}

public static string ConvertBase(string input, int fromBase, int toBase)

{

input = input.ToUpper().Trim();

string[] parts = input.Split('.');

// Butun qismini o'qib olish

long integerPart = ParseFromBase(parts[0], fromBase);

string resultInt = ConvertIntegerPart(integerPart, toBase);

// Kasr qismi yo‘q bo‘lsa

if (parts.Length == 1)

return resultInt;

// Kasr qismi

string frac = parts[1];

double fractional = 0;

for (int i = 0; i < frac.Length; i++)

{

int digitValue = GetDigitValue(frac[i]);

if (digitValue >= fromBase)

throw new ArgumentException("Noto‘g‘ri raqam kasr qismida");

fractional += digitValue / Math.Pow(fromBase, i + 1);

}

// Kasrni yangi sanoq tizimiga o‘tkazish

StringBuilder resultFrac = new StringBuilder();

int maxDigits = 12;

while (fractional > 0 && resultFrac.Length < maxDigits)

{

fractional \*= toBase;

int digit = (int)fractional;

resultFrac.Append(GetCharFromDigit(digit));

fractional -= digit;

}

return resultFrac.Length > 0 ? resultInt + "." + resultFrac.ToString() : resultInt;

}

private static int GetDigitValue(char c)

{

if (char.IsDigit(c))

return c - '0';

return char.ToUpper(c) - 'A' + 10;

}

private static char GetCharFromDigit(int digit)

{

return digit < 10 ? (char)(digit + '0') : (char)(digit - 10 + 'A');

}

private static long ParseFromBase(string input, int fromBase)

{

if (fromBase < 2 || fromBase > 16)

throw new ArgumentException("Sanoq tizimi 2-16 oralig‘ida bo‘lishi kerak.");

bool isNegative = input.StartsWith("-");

if (isNegative)

input = input.Substring(1); // '-' ni olib tashlash

input = input.ToUpper();

string chars = "0123456789ABCDEF";

long result = 0;

foreach (char c in input)

{

int digit = chars.IndexOf(c);

if (digit < 0 || digit >= fromBase)

throw new ArgumentException("Noto‘g‘ri raqam butun qismda");

result = result \* fromBase + digit;

}

return isNegative ? -result : result;

}

private static string ConvertIntegerPart(long value, int toBase)

{

if (toBase < 2 || toBase > 36)

throw new ArgumentException("Sanoq tizimi 2–36 oralig‘ida bo‘lishi kerak.");

string chars = "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ";

StringBuilder result = new StringBuilder();

do

{

int remainder = (int)(value % toBase);

result.Insert(0, chars[remainder]);

value /= toBase;

} while (value > 0);

return result.ToString();

}

private static double ConvertBaseToDecimal(string input, int fromBase)

{

input = input.ToUpper().Trim();

string[] parts = input.Split('.');

double result = Convert.ToInt64(parts[0], fromBase);

if (parts.Length == 1)

return result;

string frac = parts[1];

for (int i = 0; i < frac.Length; i++)

{

int digitValue = char.IsDigit(frac[i]) ? frac[i] - '0' : 10 + (frac[i] - 'A');

result += digitValue / Math.Pow(fromBase, i + 1);

}

return result;

}

**2.7. Klaviatura orqali kiritmalarni qayta ishlash (KeyDown)**

Foydalanuvchi Form1 oynasida klaviatura orqali quyidagi amallarni bajara oladi:

* 0-9, A-F — raqamlar kiritish
* +, -, \*, / — amallarni bajarish (btnAdd, btnSub, btnMul, btnDiv)
* Enter — hisoblash (btnEqual)
* Backspace — bitta belgini o‘chirish (btnBackspace)
* Escape — tozalash (btnClear)

Kod:

if (e.KeyCode == Keys.Enter) btnEqual.PerformClick();

if (e.KeyCode == Keys.OemPlus && e.Shift) btnAdd.PerformClick();

Bu orqali klaviaturada ishlash qulayligi ta'minlanadi va kalkulyatorga mobil ilova singari ergonomika xosil qilinadi.

private void Number\_Click(object sender, EventArgs e)

{

Button btn = sender as Button;

AddInput(btn.Text.ToUpper());

}

private void btnClear\_Click(object sender, EventArgs e)

{

txtResult.Text = "0";

userExpression = "";

calcExpression = "";

lblHistory.Text = "";

isNewEntry = true;

}

private void btnBackspace\_Click(object sender, EventArgs e)

{

if (txtResult.Text.Length > 1)

txtResult.Text = txtResult.Text.Substring(0, txtResult.Text.Length - 1);

else

txtResult.Text = "0";

if (userExpression.Length > 0)

userExpression = userExpression.Substring(0, userExpression.Length - 1);

if (calcExpression.Length > 0)

calcExpression = calcExpression.Substring(0, calcExpression.Length - 1);

lblHistory.Text = userExpression;

}

private void btnDot\_Click(object sender, EventArgs e)

{

if (!txtResult.Text.Contains("."))

{

txtResult.Text += ".";

userExpression += ".";

lblHistory.Text = userExpression;

}

}

private void btnLeftPar\_Click(object sender, EventArgs e)

{

string input = txtResult.Text;

if (!string.IsNullOrEmpty(input) && input != "0")

{

string inputDecimal = ConvertBaseToDecimal(input, currentBase).ToString();

calcExpression += inputDecimal + "\*";

userExpression += input + "\*";

}

userExpression += "(";

calcExpression += "(";

txtResult.Text = "0";

isNewEntry = true;

lastInputWasFunction = false;

lblHistory.Text = userExpression;

}

private void btnRightPar\_Click(object sender, EventArgs e)

{

string input = txtResult.Text;

if (pendingFunction != "")

{

userExpression += ")";

calcExpression += ConvertBaseToDecimal(input, currentBase) + ")";

pendingFunction = "";

pendingArgument = "";

}

else

{

if (!string.IsNullOrEmpty(input) && input != "0")

{

string inputDecimal = ConvertBaseToDecimal(input, currentBase).ToString();

calcExpression += inputDecimal;

}

userExpression += ")";

calcExpression += ")";

}

txtResult.Text = "0";

isNewEntry = true;

lastInputWasFunction = true;

lblHistory.Text = userExpression;

}

private void btnCopy\_Click(object sender, EventArgs e)

{

Clipboard.SetText(txtResult.Text);

MessageBox.Show("Natija nusxalandi: " + txtResult.Text);

}

private void Form1\_KeyDown(object sender, KeyEventArgs e)

{

// Faqat raqamlar uchun

if (e.KeyCode >= Keys.D0 && e.KeyCode <= Keys.D9)

{

int key = e.KeyCode - Keys.D0;

if (key < currentBase)

AddInput(key.ToString());

return;

}

// NumPad raqamlar

if (e.KeyCode >= Keys.NumPad0 && e.KeyCode <= Keys.NumPad9)

{

int key = e.KeyCode - Keys.NumPad0;

if (key < currentBase)

AddInput(key.ToString());

return;

}

// A-F harflar (faqat 11 dan katta sanoq tizimi uchun)

if (e.KeyCode >= Keys.A && e.KeyCode <= Keys.F && currentBase > 10)

{

int digit = 10 + (e.KeyCode - Keys.A);

if (digit < currentBase)

{

char letter = (char)('A' + (e.KeyCode - Keys.A));

AddInput(letter.ToString());

}

return;

}

// Asosiy klavishlar: orqaga o‘chirish, natija, tozalash

if (e.KeyCode == Keys.Back) btnBackspace.PerformClick();

if (e.KeyCode == Keys.Enter) btnEqual.PerformClick();

if (e.KeyCode == Keys.Escape) btnClear.PerformClick();

// Nuqta va arifmetik amallar

if (e.KeyCode == Keys.OemPeriod || e.KeyCode == Keys.Decimal) btnDot.PerformClick();

if (e.KeyCode == Keys.Add) btnAdd.PerformClick();

if (e.KeyCode == Keys.Subtract) btnSub.PerformClick();

if (e.KeyCode == Keys.Multiply) btnMul.PerformClick();

if (e.KeyCode == Keys.Divide) btnDiv.PerformClick();

}

**2.8. Tarixni faylga yozish va uni ko‘rish (Form2)**

Kalkulyatorda har bir = tugmasi bosilganda ifoda va natija history.txt fayliga yoziladi (Faylga bu ifoda qaysi sanoq sistemasida ishlanganini ham saqlab boradi):

File.AppendAllText("history.txt", userExpression + " = " + finalResult + Environment.NewLine);

Form2 oynasi ochilganda, bu fayl o‘qiladi va textBox1 oynasiga butun tarix matni ReadOnly rejimida ko‘rsatiladi:

textBox1.ReadOnly = true;

textBox1.Text = File.ReadAllText("history.txt");

Clear tugmasi bosilganda fayl bo‘shatiladi:

File.WriteAllText("history.txt", "");

textBox1.Text = "";

Bu funksiya foydalanuvchiga barcha avvalgi ifodalarni ko‘rish, tahlil qilish va nazorat qilish imkonini beradi.

private void btnHistory\_Click(object sender, EventArgs e)

{

Form2 f2 = new Form2();

f2.ShowDialog();

}

Form2.cs:

using System;

using System.Windows.Forms;

using System.IO;

namespace Kurs\_Ishi\_4\_sem

{

public partial class Form2 : Form

{

string historyFilePath = @"C:\Users\user\Desktop\Paritdinov\_Samariddin\_Kurs\_ishi\_2-kurs\_4-sem\history.txt";

public Form2()

{

InitializeComponent();

if (File.Exists(historyFilePath))

{

textBox1.Text = File.ReadAllText(historyFilePath);

}

else

{

textBox1.Text = "Tarix topilmadi.";

}

}

private void Clear\_Click(object sender, EventArgs e)

{

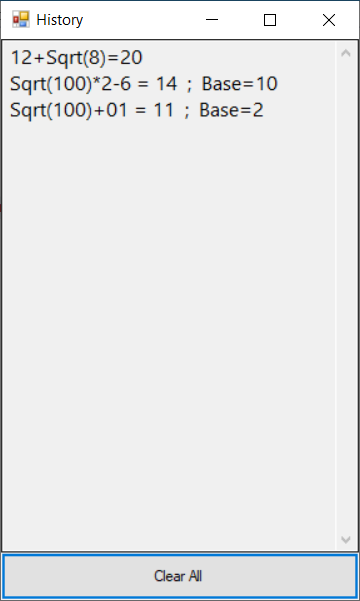
File.WriteAllText(historyFilePath, "");

textBox1.Text = "";

}

}

}

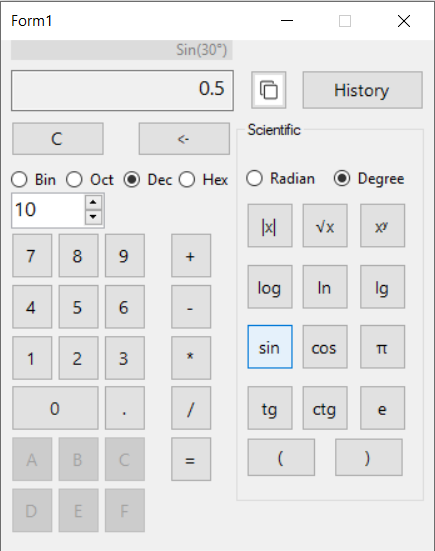


**2.9. Dasturdan foydalanish bo‘yicha misollar**

Quyida foydalanuvchining real vaqt rejimida qanday ishlashi ko‘rsatiladi:

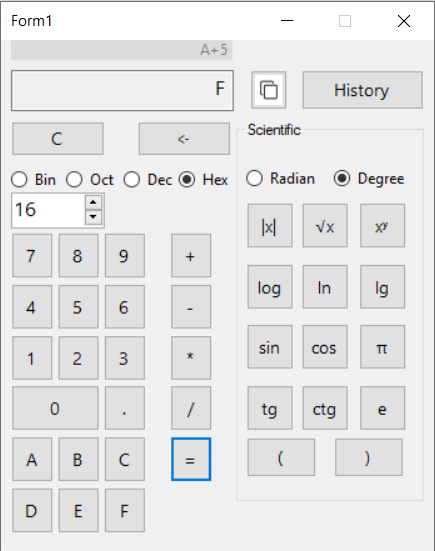
**Misol 1: Trigonometrik funksiya hisoblash**

* Foydalanuvchi 30 sonini kiritadi
* Sin tugmasini bosadi
* Sin(30°) ifodasi hisoblanadi va natija 0.5 ko‘rinishida chiqadi



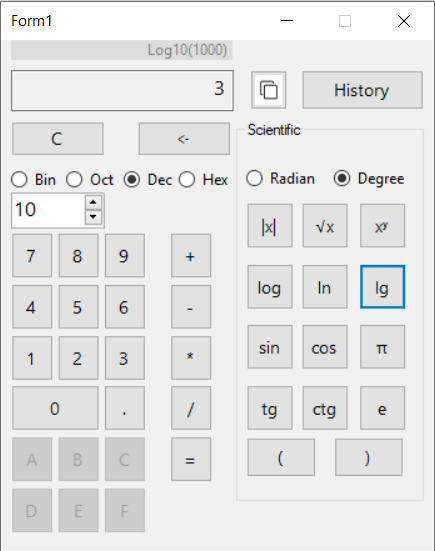
**Misol 2: Sanoq sistemasida hisoblash**

* Foydalanuvchi HEX (16 lik) sistemasini tanlaydi
* A + 5 kiritadi
* = bosiladi, natija F chiqadi (10 + 5 = 15)

****

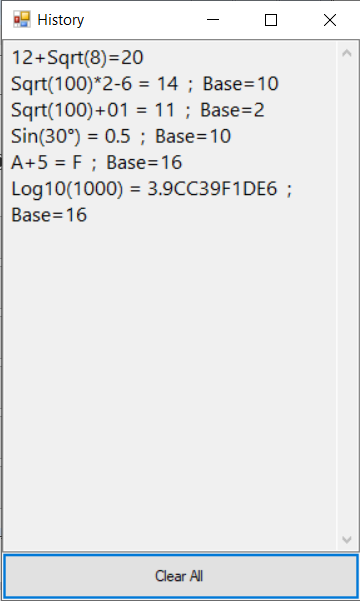
**Misol 3: Log funksiyasi (ikkita argument bilan)**

* Foydalanuvchi 1000 ni kiritadi
* Log tugmasini bosadi
* Keyin 10 ni kiritadi
* = bosilganda natija 3 chiqadi (log10(1000) = 3)

****

**Misol 4: Tarixga yozish va o‘qish**

* Foydalanuvchi 4 \* 5 amalini bajaradi
* 20 natija chiqadi
* history.txt faylga 4 \* 5 = 20 yoziladi
* btnHistory bosiladi, Form2 ochiladi va bu yozuv ko‘rinadi

****

**2.10. Rivojlantirish yo‘nalishlari**

Quyidagi imkoniyatlar orqali dastur keyinchalik yanada rivojlantirilishi mumkin:

1. **Mobil versiya**
   * Android/iOS qurilmalarda ishlaydigan variant ishlab chiqish
   * Xamarin yoki MAUI texnologiyalari orqali
2. **Foydalanuvchi interfeysini yaxshilash**
   * Qorong‘i (dark) va yorug‘ (light) rejimlar qo‘shish
   * Tugmalarning rangli dizayni va animatsiyalar
3. **Graflar va interaktiv vizualizatsiya**
   * y = sin(x), y = log(x) kabi funksiyalarni grafik chizish
   * Zoom va interaktiv o‘qlar bilan ko‘rish imkoniyati
4. **Avtozaxira (backup) va eksport**
   * history.txt faylni PDF yoki Excel ko‘rinishiga eksport qilish
   * Yangi oynada chop etish (print) funksiyasi
5. **Ko‘p foydalanuvchi rejimi**
   * Har bir foydalanuvchi alohida history\_{username}.txt faylga yozishi
6. **Internet orqali funksiyalarni kengaytirish**
   * Onlayn API orqali valyuta konvertatsiyasi yoki maxsus matematik modul ulash

**XULOSA**

Mazkur kurs ishida C# va Windows Forms texnologiyasi asosida universal, ilmiy va ko‘p bazali kalkulyator dasturi ishlab chiqildi. Dastur foydalanuvchiga istalgan sanoq sistemasida ishlash, ilmiy hisoblashlarni bajarish, natijalarni saqlash va boshqarish imkonini beradi.

Kalkulyatorning asosiy afzalliklari:

* Foydalanishdagi qulay interfeys
* Matematikaning keng sohalarini qamrab oluvchi funksiyalar
* Tarixni saqlash va qayta ko‘rish
* Sanoq sistemalari o‘rtasida konvertatsiya
* Modullar bo‘yicha yaxshi tuzilgan kod arxitekturasi

Dastur hozirgi shaklida talaba, muhandis yoki kundalik foydalanuvchi uchun qulay hisoblash vositasi bo‘lib xizmat qiladi. Kelgusida uning mobil versiyasini yaratish, grafik funksiyalarni qo‘shish, va onlayn xizmatlar bilan integratsiyalash orqali yana ham kuchaytirish mumkin.

Kodni to’liq ko’rish uchun github ga ilova:

<https://github.com/Samariddin-Paritdinov/Universal-Calculator.git>

**GLOSSARIY**

| **Atama** | **Izoh** |
| --- | --- |
| **Kalkulyator** | Raqamlar va matematik ifodalar ustida amallar bajaruvchi dasturiy ta'minot. |
| **Sanoq sistemasi** | Sonlarni belgilash va ifodalash uchun ishlatiladigan tizim; masalan, 2-lik, 8-lik, 10-lik, 16-lik. |
| **Binary (ikkilik)** | 0 va 1 raqamlaridan iborat sanoq tizimi; kompyuterlar ichki ishlovda shu sistemadan foydalanadi. |
| **Decimal (o‘nlik)** | 0 dan 9 gacha bo‘lgan raqamlardan iborat, insonlar tomonidan eng ko‘p ishlatiladigan sistemadir. |
| **Octal (sakkizlik)** | 0–7 raqamlardan tashkil topgan sanoq tizimi, qisqa ifodalash uchun ishlatiladi. |
| **Hexadecimal (o‘n oltilik)** | 0–9 raqamlari va A–F harflaridan tashkil topgan tizim, kompyuter xotirasi va rang kodlarida keng qo‘llaniladi. |
| **Windows Forms** | Microsoft .NET texnologiyasi asosidagi grafik interfeys yaratish tizimi; C# tilida desktop ilovalar yaratish uchun ishlatiladi. |
| **C#** | Microsoft tomonidan ishlab chiqilgan obyektga yo‘naltirilgan dasturlash tili. |
| **NCalc** | .NET platformasi uchun mo‘ljallangan matematik ifodalarni tahlil qilish va hisoblash kutubxonasi. |
| **Expression** | NCalc kutubxonasida matematik ifodalarni ifodalovchi obyekt. Masalan: new Expression("2+2"). |
| **ConvertBase** | Sonlarni bir sanoq tizimidan boshqasiga o‘tkazuvchi funksiya. Masalan: ConvertBase("1010", 2, 10) = 10. |
| **ConvertBaseToDecimal** | Har qanday sanoq sistemasidagi sonni o‘nlik tizimga aylantiruvchi funksiya. |
| **ParseFromBase** | String ko‘rinishdagi sonni tanlangan sanoq sistemasida tahlil qilib, o‘nlik son sifatida qaytaruvchi funksiya. |
| **txtResult** | Foydalanuvchi kiritayotgan yoki natijani ko‘rsatuvchi TextBox komponenti. |
| **lblHistory** | Kiritilgan ifoda va amallar tarixini ko‘rsatuvchi Label komponenti. |
| **Form1** | Kalkulyator dasturining asosiy interfeys oynasi. |
| **Form2** | Tarixni ko‘rsatish uchun alohida ochiladigan oynalar. |
| **KeyDown** | Klaviaturadan tugma bosilishi bilan bog‘liq hodisa (event); foydalanuvchi tugma bosganda ishga tushadi. |
| **PendingFunction** | Ikkita argument talab qiluvchi funksiyalarni (masalan: Pow, Log) vaqtinchalik saqlash uchun foydalaniladi. |
| **currentBase** | Foydalanuvchi tanlagan joriy sanoq tizimini ifodalovchi butun son (int). |
| **Tag (Button.Tag)** | Har bir tugmaga ulanadigan atribut; tugmadagi qiymatni ifodalash uchun ishlatiladi. |
| **Func<T, TResult>** | C# tilida funksiyalarni parametr sifatida uzatish uchun foydalaniladigan delegate. |
| **Math.Sin, Math.Cos, Math.Log** | .NET kutubxonasidagi matematik funksiyalar; sinus, kosinus, logarifm va h.k. hisoblash uchun ishlatiladi. |
| **Kasr qismi (fractional part)** | Sonning verguldan keyingi (yoki nuqtadan keyingi) qismi, masalan 12.**5** da 5. |
| **History.txt** | Dasturda bajarilgan amallar va natijalar yoziladigan tarixiy fayl. |
| **Clipboard** | Kompyuter xotirasining vaqtincha nusxalash (Copy/Paste) uchun ishlatiladigan qismi. |
| **RadioButton** | Foydalanuvchi tanlovi uchun ishlatiladigan tugma, odatda biror variantni tanlash uchun ishlatiladi. |
| **NumericUpDown (numericBase)** | Foydalanuvchiga raqamli qiymat tanlash imkonini beruvchi forma elementi. |

**FOYDALANILGAN ADABIYOTLAR**

1. Jo‘rayev A.A. – Algoritmlar va ma’lumotlar tuzilmasi. – Toshkent: O‘zbekiston Milliy Universiteti, 2019
2. Qodirov Sh.A., Abdurahmonov Z.T. – Dasturlash asoslari (C# tili misolida). – Toshkent: Fan va texnologiya, 2020
3. Microsoft Documentation – <https://learn.microsoft.com/>
4. Metanit C# qo‘llanmasi – <https://metanit.com/sharp/windowsforms/>
5. StackOverflow – [https://stackoverflow.com](https://stackoverflow.com/)
6. NCalc GitHub sahifasi – <https://github.com/sklose/NCalc2>