TOSHKENT TEMIR YO’L MUHANDISLARI INSTITUTI

TEMIR YO’L TRANSPORTIDA AXBOROT TIZIMLARI KAFEDRASI

M.M. Rasulmuxamedov, A.X. Boltayev

« C# tilida dastur »

FANIDAN

1-qism

5330200 – “Infarmatika va axborot texnalogiyalari 2,3 – bosqish bakalavr talabalari va professor o’qituvchilar uchun

O’QUV- QO’LLANMA

TOSHKENT-2014 y

TOSHKENT TEMIR YO’L MUHANDISLARI INSTITUTI

“Nashirga ruhsat etaman”
O’quv ishlari bo’yicha prorektor
dosent F.F. Karimova
201__ yil “_____” _____

M.M. Rasulmuxamedov, A.X. Boltayev

« C# tilida dastur »

5330200 – “Infarmatika va axborot texnalogiyalari 2,3 – bosqish bakalavr talabalari va professor o’qituvchilar uchun

O’QUV- QO’LLANMA

Toshkent-2014 y.

« C# tilida dastur »

O’QUV- QO’LLANMA

UDK 681.31

Ushbu uslubiy laboratiriya praktikumi “C# tilida dasturlash” fani bo‘yicha reja va 5 ta laboratoriya ishidan iborat bo‘lib, unda axborot tizimlari bo‘yicha barcha muhim tushuncha va ularni ishlab chiqish hamda yaratish tamoyillari keltirilgan. Laboratoriya ishlariga mos ravishda nazariy qism, topshiriq va vazifalar berilgan. Talabalarning Visual Studio muhitda ishlashi, C# tilida dastur yaratish qoidalari rasmiylashtirilgan. Visual Studio.NET ishlashga zarur bo‘lgan to‘liq ma'lumotlarni, hujjatlardan yoki [8], [16] kitoblardan olish mumkin.

Rasm 12 ta, shakl 8 ta.

Institutining Ilmiy-uslubiy kengashi qarori bilan nashrga tavsiya etilgan.

Tuzuvchilar : M.M. Rasulmuxamedov — f.-m.f.n., dots.;
A.X. Boltayev — ass.

Taqrizchilar : R.I.Ibragimov – t.f.n., dots.(ToshDMI);
T.R.Nurmuhamedov – t.f.d., prof.

Laboratoriya ishlarini bajarishga qo‘yilgan talablar

1. Laboratoriya ishi mavzularini uslubiy ko‘rsatmada bayon qilingan tartibda o‘zlashtirish.
2. Uslubiy ko‘rsatmada keltirilgan vazifalarni bajarish.
3. Ishning natijalarini shaxsiy papkada saqlash.
4. Ishning natijalarini o’qituvchiga ko‘rsatish va uni himoya qilish.
5. Loyiha fayllarini semestr yakunlangunicha shaxsiy papkada saqlash (disketda yoki diskda).
6. Hisobot:

- laboratoriya ishining nomi, maqsadi va masalasi;
- nazorat savollariga javoblar va vazifalar;
- yakun va xulosa.

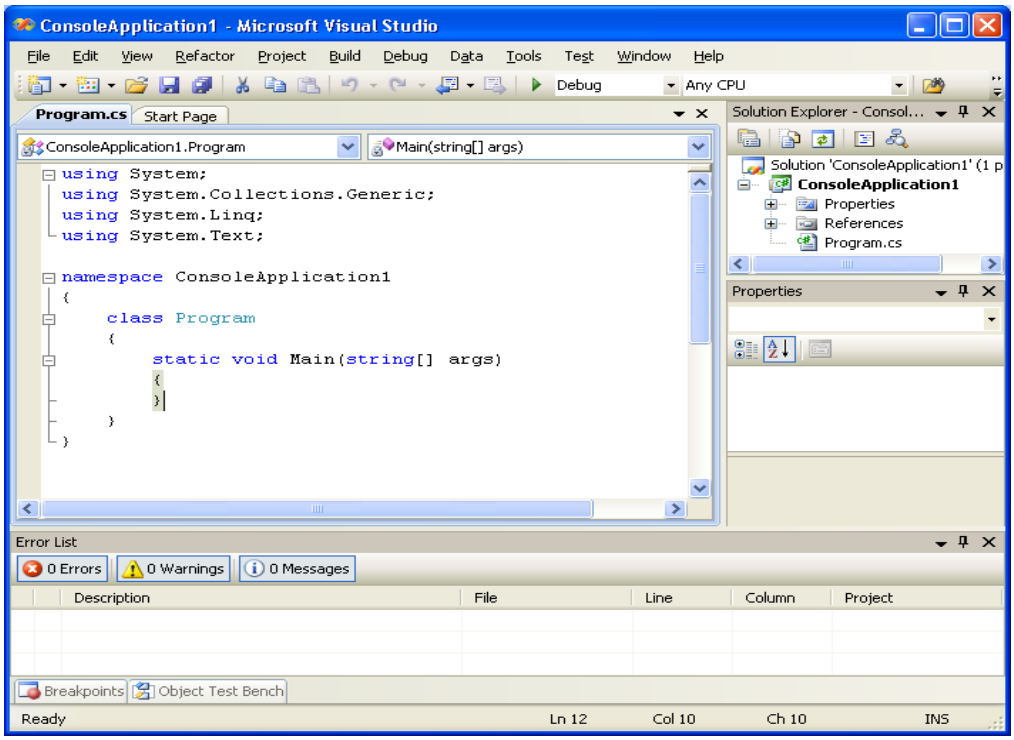
Kirish

Konsol ilovalar

Visual Studio.NET muhiti Windows platformasida ishlaydi hamda Windows va Web – ilovalar yaratishga mo‘ljallangan, lekin ishlab chiquvchilar konsol ilovalar yaratishni ham nazarda tutishgan. Konsol ilovani ishga tushirganda operatsion tizim konsol oynani yaratadi, uning yordamida hamma kiritish-chiqarish olib boriladi. Tashqaridan qaraganda u operatsion tizimning komanda satri rejimi ishini esga soladi, qachonki kiritish-chiqarish simvollar oqimini namoyish etadi. Konsol ilovalar tilni o‘rganishga eng yaxshi uslub, chunki unda grafik interfeysni hosil qilish uchun ko‘plab standart obyektlar ishlatilmaydi. Kursning birinchi qismida faqat konsol ilovalar yaratishdan maqsad, C# tilining asosiy xossalarida fikrni jamlash. Qolgan qismlarida muhitdagi eng sodda amallarni ko‘ramiz: C# da konsol ilovani yaratish va ishga tushirish. Visual Studio.NET ishlashga zarur bo‘lgan to‘liq ma'lumotni, hujjatlardan yoki [8], [9] adabiyotlardan olish mumkin.

Loyihani yaratish. Muhitning asosiy oynasi

Loyiha yaratish uchun Visual Studio.NET ni ishga tushirgandan so‘ng asosiy menyuda File • New • Project komandasini tanlash zarur. Ochilgan muloqot oynasining chap tomonidagi Visual C# Projecs punktni tanlash zarur, o‘ngda esa Console Application punktini. Name maydonida loyiha nomini kiriting, agarda kompyuterda ko‘rsatilgan joy sizni qanoatlantirmasa Location maydonida esa uni diskda saqlash joyini ko‘rsating. OK knopkasini bosganingizdan so‘ng muhit yechim va loyihani siz ko‘rsatgan nomda yaratadi. Ekranning taxminiy ko‘rinishi 1-rasmda keltirilgan. Ekranning yuqori qismida asosiy menyu ((File, Edit, Viyew va hokazo bo‘limlari bilan) va (toolbars) uskunalar paneli joylashgan. Muhitda uskunalar paneli juda ko‘pdir, agarda ularni hammasini (Viyew • Toolbars...) ishga tushirsa, ular ekranning yarmini egallab oladi.



1-rasm. Konsol ilova yaratilgandan so‘ng ekranning taxminiy ko‘rinishi

Ekranning yuqori qismining chap tomonida Solution Explorer loyihani boshqarish oynasi joylashgan (agar u aks ettirilgan bo‘lsa, asosiy menyuning Viyew • Solution Explorer komandasidan foydalaning). Loyihaga kiruvchi hamma resurslar sanab o‘tildi: kutubxonaga murojaat (System, System.Data, System.XML), fayl yorlig‘iga (App.ico), (Classl.cs) klass dastlabki matn fayl va axborotni yig‘uvchi fayl (Assemblylnfo.cs).

Bu oynada o‘zga axborotni ham ko‘rish mumkin, agarda Class Viyew bayonchasiga o‘tilsa, yorliq oynaning quyi qismida joylashgan. Class Viyew bayonchasida ilovaga kiritilgan barcha klasslar ro‘yxati keltirilgan, ularning elementlari va uning avlodi. Oynaning chap quyi qismida Propertiyes xossasi oynasi joylashgan (agarda u akslantirilmagan bo‘lsa, asosiy menyuning Viyew • Propertiyes komandasidan foydalaning). Oyna xossasida oldingi bo‘limda aytilgandek, yig‘ma kompilyator ishining natijasi hisoblanadi va u oraliq til kodini hamda maxsus ma'lumotni o‘z ichiga oladi, ajratilgan elementning asosiy tavsifini akslantiradi. Misol, fayl nomini o‘zgartirish uchun, unda Classl sinfi saqlanadi, bu faylni loyihaning boshqarish oynasida ajratib belgilash kerak va FileName xossasiga yangi qiymat xossalar oynasi beriladi (Enter tugmasini bosish bilan ma'lumot kiritish yakunlanadi).

Ekranning asosiy maydonini oyna tahrirlagichi egallaydi, unda dastur matni joylashadi, muhit yordamida avtomatik ravishda tuzilgan. Matn karkasni namoyish qiladi, unga dasturchi zaruratga qarab kod qo‘shadi. Kalit (zaxiradagi) so‘zlar ko‘k rang bilan, eslatmalar turli tiplarniki – kul

rang va to‘q-yashil rang, qolgan matnlar qora rang bilan akslanadi. Matn chap tomonida simvollar tuzilmasi turadi: ixtiyoriy minus kvadratiga chertib shu blokka kirgan kodlarni yashirish mumkin. Unda minus belgi plyusga aylanadi, unga yana chertib, bu blokka tegishli kodni ekranga chiqarish mumkin. Bu muhit yaxshi vizual tuzilmali koddir va fikrni kerakli fragmentlarga fokuslash imkonini beradi.

Konsol dasturining qolipi

Dastur qolipi har bir satrini ko‘rib chiqamiz (0.1-varaqcha). Birdaniga mutloq unda yozilganlarni hammasini tushunishga harakat qilmang. Hozircha bizning maqsadimiz - qobiq prinsiplarini o‘rganish, dasturni butunligicha keyinroq o‘rganamiz.

0.1-varaqcha. Konsol dasturning qolipi

```
using System;
namespace ConsoleApplicationl
{
   /// <summary>
   /// Summary description for Classl.
   /// </summary>
   class Classl
   {
      /// <summary>
      /// The main entry point for the application.
      /// </summary>
      [STAThread]
      static void Main(string[] args)
      {
         //
         // TODO: Add code to start application here
         //
      }
   }
}
```

Kalit so‘zlar - bu asosiy so‘zlar bo‘lib, ular kompilyator uchun maxsus qiymatga ega. Ular qanday aniqlangan bo‘lsa, shu ma'noda foydalanish mumkin. Eslatmalar dasturchilar uchun mo‘ljallangan va hujjatlashtirilgan hujjatlarni ta’minlaydi.

Using System direktivasi System fazosi nomlaridan mos ravishda standart klasslar nomlarini foydalanishiga ruxsat beradi (fazoni nomini ko‘rsatmasdan turib).

Kalit so‘z namespace loyiha (проект) uchun o‘zining xususiy nomlar fazosini yaratadi, odat bo‘yicha ConsoleApplicationl deb nomlangan. Bu shuning uchun qilinganki, dastur obyektlariga nomlar berishda o‘zga fazo nomlari bilan mos kelishligini o‘ylamasdan nomlash mumkinligiga imkon yaratilgan. Satrlar agarda ikkita qiyshiq chiziq bilan boshlangan bo‘lsa u holda eslatma berish yoki dastur matnlarini hujjatlash uchun mo‘ljallangan bo‘ladi.

C# - obyektga mo‘ljalangan tildir, shuning uchun unda yozilgan dasturlar o‘zaro klasslarning birgalikdagi harakati to‘plamini namoyish qiladi. Ushbu zogatovkamizda faqat bitta klass-unga odat bo‘yicha Class1 nomi berilgan. Klass bayoni class kalit so‘zidan boshlanadi, undan so‘ng uning nomi va katta qavs ichida klass elementlari ro‘yxati (uning usul deb nomlanadigan ma'lumotlari va funksiyalari). Bu holda klass ichida bitta element – Main usuli. Har bir ilova Main usulini o‘zida saqlashi shart, undan dastur bajarilishi boshlanadi. Hamma usullar bir xil qonuniyat bilan bayon qilinadi.

Usulning soddalashtirilgan sintaksisi:
[spetsifikatorlar] usul_nomi turi ([parametrlar])
{
Usul jismi: harakat, usul tomonidan bajariladi.
}

Shunday qilib, har qanday usul tur, nom va jismidan iborat, qolgan qismlari bayon qilinishi shart emas, shuning uchun ulardan foydalanmaymiz. Usulni to‘laqonli ravishda «Usullar» bo‘limida ko‘rib chiqamiz.

Muhit Main usuli ichiga yordam tariqasida eslatmalarni joylashtirgan:

// TODO: Add code to start application here

Bu degani: «Ilovani ishga tushirganingizda, bu yeriga kod tering». Biz maslahatga amal qilamiz va eslatma satridan keyingi satrda quyidagi satrni teramiz:

Console.WriteLine{"Barakalla, dasturimiz ishga tushdi!");

Bu yerda Console - bu System nomli fazoning standart nomli klassi. Bu WriteLine usuli qo‘shtirnoqqa olingan matnni ekranga chiqaradi. Ko‘rib turibsizki, usulga murojaat qilish uchun klass_nomi.usul_nomi konstruksiyasi ishlatiladi.

Agar siz birinchi so‘zda xatoga yo‘l qo‘ymasangiz, klaviatura yordamida kiritgan Console so‘zidan so‘ng nuqta qo‘ysangiz muhit ro‘yxatni tasvirlab beradi, unda Console klassiga tegishli elementlar ro‘yxatini beradi. Kerakli nom sichqon bilan yoki klaviaturaning boshqaruv kursorida, yoki nomning bir necha simvolini kiritish yordamida tanlanadi. Enter tugmasi bosilganda tanlangan ism dastur matnida paydo bo‘ladi. Dastur 0.2-varaqchada ko‘rsatilgan ko‘rinishni olishi shart (dastur tuzilmasida siz fikringizni jamlashingiz uchun, undan hamma eslatmalar va ortiqcha narsalar olib tashlangan). Bu varaqchada murakkab hech narsa yo‘q. Quyidagiga e'tiboringizni qarating, o‘zgartirish kiritilgandan so‘ng fayl ismi oldidagi yorliqda * simvoli paydo bo‘lsa - bu matnda yoki saqlangan diskda taqdim etilgan matn mos kelmaydi. Faylni saqlash uchun asosiy menyuning File • Save komandasidagi asboblar panelidan Save knopkasidan foydalaning (matn kursori bu vaqtda tahrirlash oynasida joylashgan bo‘lishi shart). Aytgandek, dasturni ishga tushirganda dastlabki matn provardida mustaqil saqlanadi.

0.2-varaqcha. C# tilida birinchi dastur

```
using System;
namespace ConsoleApplicationl
{
   class Classl
   {
      static void Main()
      {
         Console.WriteLine("Ofarin! Ishga tushadi! Do‘stim ");
      }
   }
}
```

Dasturni ishga tushirish

Dasturni ishga tushurishni eng sodda usuli F5 tugmani bosish (yoki Debug • Start komandasini tanlash). Agar dastur xatosiz yozilgan bo‘lsa, u holda «Ofarin! Ishga tushadi! Do‘stim» co‘zlari konsol oynasida ko‘rinadi va ekran yopiladi. Bu yaxshi natija, lekin tinchkina ko‘rish uchun Ctrl+F5 tugmalarini bosish kerak (yoki komandalar menyusidan Debug • Start Without Debugging ni tanlash kerak).

Dastur matniga o‘zgartirish kiritilgandan so‘ng kompilyator sintaksis xatolarni topishi mumkin. Ekranning quyi qismida joylashgan oyna orqali ma'lumot berishi mumkin.

Keling dasturga sintaksis xato kiritaylik, birinchi marotaba ko‘rish uchun, keyinchalik bunday xatoning ming marotaba ko‘rish nasib qiladi. Console.WriteLine() operatoridan so‘ng nuqta vergulni olib tashlang va dasturni ishga tushiring. Siz muloqot oynasida axborotni ko‘rasiz, unda ilovani qurishda xatolikka yo‘l qo‘yildi, bu holda u «davom ettiraymi» degan savol bilan murojaat qiladi (There were build errors. Continuye?). Sovuq javob berib, xatolar oynasidagi izohga e'tiborni qaratamiz.

Axborotlar satrida ikki karra chertish “error CS1002” xatoni ko‘rsatadi; expected (demak, bu yerda nuqta vergul nazarda tutilgan edi) dasturning bu yerida xato satr borligini ko‘rsatadi. Qo‘shimcha tushintirishlar olish uchun F1 tugmasini bosish kerak, bu holda ma'lumot tizimining mos sahifasi bilan yangi bayoncha hosil bo‘ladi. Kod klassiga o‘tish uchun tahrirlagich yuqori satridagi fayl nomi yorlig‘i ustiga chertish kerak.

Ma'lumot olishning o‘zga yo‘li - Dynamic Help oynasidan foydalanish, u xossalar oynasi Propertiyes yonida joylashgan (uning yorlig‘i oynaning quyi oynasida joylashgan). Bu oynaning mazmuni qansi elementni ajratib belgilanishiga qarab dinamik ravishda o‘zgaradi. Ma'lumot olish uchun mos uzatishga chertish kerak va ma‘lumot tizimi sahifasida u alohida tahrirlagich oynasida akslanadi (Visual C# 2008 Express Edition da ma'lumotlar axboroti alohida oynada paydo bo‘ladi).

Endi, siz .NET platformasi to‘g‘risida umumiy tushunchalar oldingiz, bundan buyon C# tilini o‘rganishga rejali ravishda kirishsak bo‘ladi.

1 - LABORATORIYA ISHI
Mavzu: Chiziqli dasturlar

Ishdan maqsad: Chiziqli dasturlar bilan tanishish.

Barcha operatorlar berilgan ketma ketlikda bajarilsa - chiziqli dastur deyiladi. Chiziqli dastur dasturning sodda ko‘rinishi bo‘lib, berilgan formula bo‘yicha hisoblashni amalga oshiradi. U uch etapdan iborat: dastlabki ma'lumotni kiritish, formula bo‘yicha hisoblash va natijani chiqarish. Bu dasturni tuzish uchun bizda hozircha C# tilida kiritish va chiqarishni tashkil etish bo‘yicha bilimlarimiz kamlik qiladi. Bu yerda faqat minimal ma'lumotlar keltiriladi.

Oddiy kiritish-chiqarish

Har qanday dastur dastlabki ma'lumotlarni kiritishda va natijani chiqarishda tashqi qurilmalar bilan o‘zaro xarakatdadir. Kiritish va chiqarish qurilmalarining standart jamlanmasi, ya'ni klaviatura va ekran konsol deb nomlanadi.

C# dasturlash tilida o‘zga tillardagi kabi, kiritish va chiqarish operatori yo‘q. Ularni o‘rniga tashqi qurilmalar bilan axborot almashish uchun standart obyektlar qo‘llaniladi. Konsolda ishlash uchun C# da System nomlar fazosida aniqlangan Console klassi qo‘llaniladi. Bu klassning Write i WriteLine usullarini avvalgi misollarda qo‘llaganmiz. Bular haqida biz to‘liq gaplashamiz, buning uchun 0.1-varaqchaga o‘zgartirish kiritamiz. Bu kiritilgan o‘zgarishlar 1.1-varaqchada keltirilgan.

1.1-varaqcha. Chiqarish usullari

```
using System;
namespace ConsoleApplicationl
{
   class Classl
   {
      static void Main()
      {
         int i = 3;
         double u = 4.12;
         decimal d = 600m;
         string s = "Vosiq";
         Console.WriteLine("i = " + i);  //  1
         Console.WriteLine("y = {0} \nd = {1}", y, d);  //  2
         Console.WriteLine("s = " + s);  //  3
         Console.ReadKey();
      }
   }
}
```

Dastur ishi natijasi:

```
i = 3
u = 4,12
d = 600
s = Vosiq
```

Shu vaqtgacha o‘zgaruvchilar qiymatlari va turli o‘rnatilgan literal turlarini chiqarishda WriteLine usulidan foydalandik. Bu imkoniyat Console klassida turli qiymatlarni chiqaruvchi Write va WriteLine usullarning mavjudligi tufaydi amalga oshirilmoqda. Usullar bir xil nomli

bo‘lib, lekin turli parametrli bo‘lsa ular o‘ta yuklangan deyiladi. Kompilyator qansi bir usul chaqirilganligini o‘tkazilayotgan qiymatlar turiga qarab aniqlaydi. Console klassi chiqarish usullari hamma o‘rnatilgan turlar uchun o‘ta yuklangan, undan tashqari 1.1-varaqchada usulni chiqarishning ikkita keng foydalaniladigan varianti mavjud. Avval e'tiborni 1 va 3 satrda o‘zgaruvchilarga izohni chiqarishning uslubiga qarating. Izohlar satrli literallarning tushuntirishdan iborat. Agar WriteLine usuli bir parametr bilan chaqirilgan bo‘lsa, u o‘rnatilgan ixtiyoriy tur bo‘lishi mumkin, misol tariqasida u son, simvol yoki satr. Biz uchun shu talab qilinadi. Bir satrda bitta emas, ikkita qiymat chiqariladigan bo‘lsa: matnli izoh va o‘zgaruvchi qiymatini chiqarishga uzatishdan avval ularni + amali yordamida «ulash» talab qilinadi.

Satrni son bilan birlashtirishdan avval soni simvollar ketma-ketligi bo‘lgan ichki shaklga aylantirish kerak bo‘ladi, ya'ni satrga. Satrga aylantirish C# ning hamma klasslarida aniqlangan, buning uchun ToString() usuli xizmat qiladi. Bu holda u yaqqol bajarilmaydi, lekin biz uni oshkora chaqirishimiz mumkin:

Console.WriteLine("i = " + i.ToString());

Operator 2 illyustrativ formatli chiqarishni namoyish qiladi. Bu holda WriteLine ning o‘zga variantlaridan foydalaniladi, ular birdan ortiq parametrni o‘z ichiga oladi. Usulning birinchi parametrga satrli literal uzatiladi, odatdagidan farqli simvollar, konsolda chiqarishga mo‘ljallangan bo‘lib, parametrlar katta qavs ichida, shuningdek boshqaruv ketma-ketligida bajariladi.

Parametrlar noldan boshlab nomerlanadi, chiqarishdan oldin ular mos ro‘yxatdagi o‘zgaruvchi qiymatlari bilan almashinadi: nolinchi parametr birinchi o‘zgaruvchi qiymati bilan almashinadi (bu misolda - u), birinchi parametr - ikkinchi o‘zgaruvchi (bu misolda - d) va hokazo. Boshqaruv ketma-ketligidan ko‘pincha satrni ko‘chirish (\n) va gorizontal tabulyatsiya (\t) ko‘proq ishlatiladi.

Klaviaturadan chiqarishning sodda uslublarini ko‘ramiz. Console klassida sartni va alohida simvolni kiritish usullari aniqlangan, lekin alohida soni klaviaturadan o‘qiydigan usul yo‘qdir. Sonli ma'lumotni kiritadigan usul ikki etapdan iborat:

1. Sonni tasvirlovchi simvollar, klaviatura orqali satrli o‘zgaruvchi sifatida kiritiladi.
2. Satrni mos o‘zgaruvchi turiga almashtirish bajariladi.

O‘zgartirish maxsus klass Convert yordamida amalga oshiriladi, System nomlar fazosida aniqlangan, yoki Parse usuli yordamida, u arifmetik klass standartida hammasida mavjud. 1.2-varaqchada ikkala uslub ham foydalaniladi.

1.2-varaqcha. Chiqarish usuli

```
using System;
namespace ConsoleApplicationl
{
   class Classl
   {
      static void Main()
      {
         Console.WriteLine("Satrni kiriting ");
         string s = Console.ReadLine(); // 1
         Console.WriteLine("s = " + s);
         Console.WriteLine("simvolni kiriting");
         char s = (char)Console.Read(); // 2
         Console.ReadLine(); // 3
         Console.WriteLine("c = " + s);
         string buf; // bufer sonni kiritig uchun – satr
         Console.WriteLine("Butun son kiriting");
         buf = Console.ReadLine();
         int i = Convert.ToInt32(buf); //4
         Console.WriteLine(i);
         Console.WriteLine("Haqiqiy son kiriting");
         buf = Console.ReadLine();
         double x = Convert.ToDouble(buf);  // 5
         Console.WriteLine(x);
         Console.WriteLine("Haqiqiy son kiriting");
         buf = Console.ReadLine();
         double y = double.Parse(buf); // 6
         Console.WriteLine(y);
         Console.WriteLine("Haqiqiy son kiriting");
         buf = Console.ReadLine();
         decimal z = decimal.Parse(buf); // 7
         Console.WriteLine(z);
      }
   }
}
```

Bu dasturga bir necha izoh beriladi. Satrni kiritish 1 - operatorda bajariladi. Satr uzunligi cheklanmagan, kiritish - satrni ko‘chirish simvoligacha bajariladi.

Simvolni kiritish Read usuli yordamida amalga oshiriladi, u kirish oqimidan bitta simvolni o‘qiydi (2 - operator). Usul int turli qiymatini qantaradi, agarda simvol kodi - 1 yoki kiritish oqimida yo‘q bo’lsa (misol uchun foydalanuvchi Enter tugmasini bosadi). Bizga int turi emas, balki char turi kerak, bu yerda esa int dan char ga o‘tkazishning yaqqol amali yo‘q, o‘zga turga o‘zgartirishning yaqqol amalini qo‘llashga to‘g‘ri keladi, uni «turning yaqqol ozgartirilishi» bo‘limida o‘qiymiz.

2 - operatordan so‘ng operator 3 yozilgan, u satrning qoldig‘ini hisoblaydi va hech qayerga uzatmaydi, u ma'lumotni kiritish bufer orqali bajariladi - tezkor xotiraning maxsus sohasida. Ma'lumotlar avval buferda o‘qiladi, so‘ngra kiritish protseduralari orqali o‘qiladi. Buferga kiritish uning kodi bilan birgalikda Enter tugmasini bosish orqali bajariladi. Read usuli, ReadLine dan farqli ravishda, buferni tozalamaydi, shuning uchun bundan keyingi kiritish oldingisi qayerda tugagan bo‘lsa shu yerdan boshlanadi.

Operatorlar 4 va 5 da Convert klassi usullari ishlatiladi, operatorlar 6 va 7 - .NET kutubxonasining Double va Decimal klassini Parse usullari, ular bu yerda C# double va decimal nomli turlar orqali ishlatiladi.

O‘zlashtirish uchun savollar

1. "Konsol", "konsol ilova" o‘zi nima? U qanday yaratiladi?
2. Konsol kiritish-chiqarishga misol keltiring .
3. Asosiy C# dasturi ilovasi strukturasi va alohida elementlarini bayon qiling.
4. C# dasturi ilovasi loyihasida modulning vazifasi nimadan iborat? Modulning alohida qismlari uchun vazifalarini bayon qiling.
5. Visual Studio ilovalarni yaratish integrallashgan muhitiga qanday komponentalar kiradi?
6. Visual Studio muhiti oynasi asosiy komponentalarini sanab o‘ting va vazifalarini ko‘rsating.
7. Visual Studio da oyna elementlari va loyihalashtirilayotgan grafik interfeys komponentalari bo‘yicha qanday qilib ma'lumotni olish mumkin?
8. Propertiyes va Evens sahifalarining vazifalari nimalardan iborat?
9. Formada komponentalar qanday joylashtiriladi?
10. Qanday uslub bilan komponenta xossasini o‘zgartirish mumkin? Misollar keltiring.
11. Visual Studio loyihasi tarkibini sanab o‘ting. Turli fayllar vazifasini bayon qiling.
12. Xossalarga ishlov beruvchilar qanday vazifa bajaradi? Qanday qilib xossalarga ishlov beruvchi-protseduralarni yaratishni nomlash mumkin?

2 – LABORATORIYA ISHI
Mavzu: Boshqaruv tuzilmalari. Tarmoqlanuvchi tuzilmalar algoritmlarini dasturlash

Ishdan maqsad: Tarmoqlanuvchi dasturlar bilan tanishish.

Shartli ifoda—bu shunday ifodaki, unda ikki qiymat Truye (rost) yoki False (yolg‘on) larning birini taqdim etadi. Oddiy mantiqiy ifodalar munosabat amallarini (solishtirish amallarini) o‘z ichiga oladi:): == (teng), > (katta), < (kichik), != (teng emas), >= (katta yoki teng), <= (kichik yoki teng). Murakkab mantiqiy ifodalar ularni qo‘llagan oddiy mantiqiy ifoda yoki mantiqiy amallardan tarkib topadi.

Mantiqiy amallarning mantiqiy ifodalarda bajarilish tartibi:

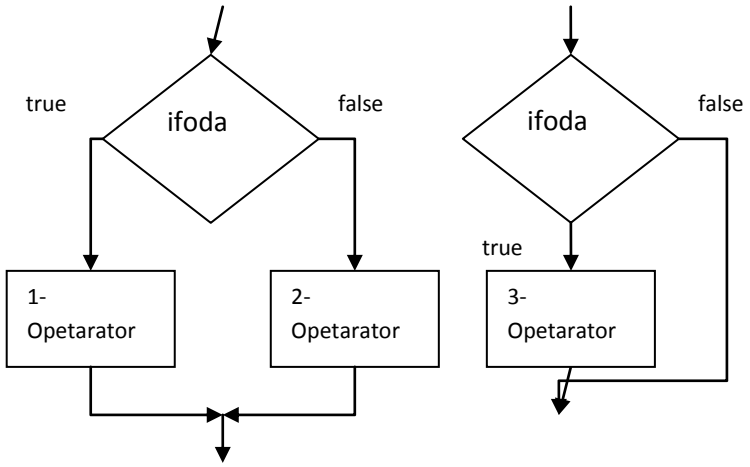
1. Inkor (!)
2. Mantiqiy ko‘paytiruv (&)
3. Mantiqiy yig‘indi (|)

Qavslar amallar bajarilishining tartibini o‘zgartiradi.

Shartli operatorlar berilgan shartga qarab mumkin bo‘lgan amallar (operatorlar) dan birini bajarishga mo‘ljallangan, unda biror bittasi mavdud bo‘lmasligi mumkin.

Amallarni tanlash shartini bajarilishiga qarab IF operatori yordamida amalga oshirish mumkin.

Shartli operator if



2.1-rasm. Shartli operator tuzilma sxemasi

Operator shakli:

if (mantiqiy ifoda) operator_1; [else operator_2;]

Ikki xil ko‘rinishdagi shartli operatorda mavjud: If(…){…} va If(…){…}Else{…}.

If(…){…} konstruksiya qandaydir amallarni bajarish zarur bo‘lgan holda, berilgan shart qiymati «rost» bo‘lsa qo‘llaniladi.

If(…){…}operator sintaksisi

If (<shart>) { < rost shartida bajariladigan operatorlar>}

Avval shartning qiymati hisoblanadi (mantiqiy turdagi ifoda), agar shart qiymati «rost» bo‘lsa qavs {} ichidagi qo‘llanma bajariladi. Tarkibiy operator ichida o‘zga tarkibiy operator joylashishi mumkin.

If(…){…}Else{…} operator sintaksisi:

```
If (shart)
{/* < rost shartida bajariladigan ifodalar>}*/}
Else
{/* yolg‘on shartida bajariladigan ifodalar*/
}
```

If(…){…}Else{…} operatori quyidagi tartibda bajariladi:

1. Avval shartning qiymati hisoblanadi (mantiqiy turdagi ifoda).
2. Agar shart qiymati «rost» bo‘lsa qavs {} ichidagi qo‘llanma bajariladi . Agar shart qiymati «yolg‘on» bo‘lsa Else dan keyin kelgan qavs {} ichidagi qo‘llanma bajariladi.

Misollar:

1. Uchta sondan eng kattasini topish dasturi:

```
Double a=double.Parse(textBox1.Text);
Double b=double.Parse(textBox2.Text);
Double c=double.Parse(textBox3.Text);
Double max=a;
if(a>=b){
if(a>=c){max=a;}else{max=c;}
}else{
if(b>=c){max=b;}else{max=c;}
}
textBox4.Text=max.ToString();
if(b!=0)      //bo‘linuvchini noldan farqligi tekshiriladi
{
c=a/b;   //haqiqiy songa bo‘lish amali bajariladi
//haqiqiy sonni matnli satrga aylantiriladi
TextBox3.Text=c.ToString();
}else{
textBox3.ForeColor=Color.Red; //shrift rangini qizilga o‘zgartirish
```

```
textBox3.Width=130;   //matnni darchani kengligini o‘zgartirish
textBox3.Text=”Nolga bo’lish mumkin emas";   //Axborotni chiqarish
}
```

Switch operatori to‘plamdan belgilanganlarni tanlovini amalga oshirishga imkon beradi. Ifodaning qiymatiga qarab biror bir tarmog‘iga o‘tish amalga oshiriladi (tanlov selektori).

Switch operator ikki xil variantda mavjud:

```
switch ( ifoda )
{
case o‘zgarmas_ifoda_1: [ operatorlar_1_ro‘yxati ]
case o‘zgarmas_ifoda_2: [ operatorlar_2_ro‘yxati ]
case o‘zgarmas_ifoda_n: [ operatorlar_ro‘yxati_n ]
[ default: operatorlar ]
/*ko‘rsatma default, agar ifoda qiymati bir-biriga ustma-ust tushmasa hodisa bajariladi */
}
```

Bu yerda

ifoda - selektor, buning qiymatidan dasturning qadamlari bog‘liq bo‘ladi, u faqat oddiy tartibli tur bo‘lishi mumkin (butun, simvolli, mantiqiy);

o‘zgarmas_ifoda - o‘zgarmaslar, xuddi o‘sha tur, selektor kabi, tarmoq metkasi rolini bajaradi. Agar o‘zgarmaslar oraliq sonini bildirsa, u holda oraliqning ro‘yxatdagi birinchi va oxirgi o‘zgarmasini ikki nuqta bilan ajratib ko‘rsatishi mumkin.

Operatorning bajarilishi k ifodani hisoblashdan boshlanadi, hosil bo‘lgan qiymati o‘zgarmaslar bilan solishtiriladi va mos operator bajariladi.

Misol:

Berilgan ikkita teng bo‘lmagan haqiqiy sondan eng kattasini topish uchun quyidagi operatordan foydalaniladi:

```
double a= double.Parse(Console.ReadLine());
double b= double.Parse(Console.ReadLine());
switch(a>b)
{
case truye: max=a;
break;
case false: max=b;
break;
}
```

```
Console.WriteLine("max="+max);
```

Agar tanlov natijasida 2, 3 yoki undan ko‘p yo‘l-yo‘riq bajariladigan bo‘lsa u holda tarkibiy operatordan foydalanish maqsadga muvofiq.

O‘zlashtirish uchun savollar:

1. Qanday operator yordamida «Tarmoqlanish» algoritmik tuzilmasi amalga oshiriladi? Uning blok-sxemasini chizing.
2. Qanday operator yordamida «Tanlov» algoritmik tuzilmasi amalga oshiriladi? Uning blok-sxemasini chizing.
3. Asosiy mantiqiy amallarning solishtirish belgilarini sanab o‘ting.
4. Shartli operator qachon qo‘llanadi?
5. Ikki xil shartli operatorni sanab o‘ting.
6. Tarkibiy operator nimalardan iborat?
7. Tanlov operatori qanday ishlarni bajaradi?
8. Selektor nima?
9. SHartli operatordan foydalanishga misol keltiring.
10. Tanlov operatoridan foydalanishga misol keltiring.
11. Bog‘liq bo‘lmagan fiksatsiya qiluvchi knopka nima uchun xizmat qiladi?
12. Javob olish uchun Label obyektining qanday xossasidan foydalanish mumkin?

3- LABORATORIYA ISHI

Mavzu: Takrorlanuvchi algoritmlarni dasturlash

Ishdan maqsad: Takrorlanuvchi algoritmlarni dasturlashni o‘rganish.

Siklli konstruksiyalar sikl jismi deb nomlanuvchi ketma - ketlikni ko‘p marotaba qantarilib bajarilishini ta’minlaydi. Elementarda ikki ko‘rinishli tuzilma mavjud:

- parametrli sikl;
- iteratsiyali sikli yoki shart bilan sikl.

Parametrli sikllardagi avvaldan sikl jsmi qantarilish soni ma'lum bo‘lsa ulardan foydalanadi.

Amaliyotda shart bilan sikldan asosan ikki holatda foydalanadi:

• qantarilish soni avvaldan ma'lum (misol uchun, natijani berilgan kerakli aniqlikka erishgancha siklni davom ettirish).

• takrolanishlar soni avvaldan ma'lum bo‘lsa.

Avvaldan shartli while sikli

Operatorning shakli sodda:

while (ifoda) operator

Ifoda mantiqiy turli bo‘lishi shart. Misol uchun, bu solishtirish yoki oddiy mantiqiy o‘zgaruvchi amal bo‘lishi mumkin. Agar hisoblash natijasi truye ga teng bo‘lsa, oddiy yoki tarkibiy operator (blok) bajariladi. Bu harakat ifodaning qiymati false davom ettiriladigan momentgacha bajariladi. Takrorlanish tugaganidan so‘ng boshqaruv keyin kelgan operatorga uzatiladi. Ifoda har bir iteratsiya siklida hisoblanadi. Agar birinchi tekshiruvdayoq ifoda qiymati false bo‘lsa, sikl bir marotaba ham takrorlanmaydi va bajarilmaydi.

Misol. Berilgan sondagi raqamlar soni hisoblab topilsin.

Berilgan algoritmni amalga oshiruvchi dasturning ko‘rinishi quyidagichadir:

```
using System;
namespace ConsoleArrIication1
{
   class Classl
   {
      static void Main()
      {
         int a = int.Parse(Console.ReadLine());
         int k = 0;
         while (a != 0)
         {
            k++;
            a /= 10;
         }
         Console.WriteLine("k={0}", k);
      }
   }
}
```

So‘ngra shartli do sikli

So‘ngra shartli sikl quyidagi tuzilmali sxema asosida amalga oshiriladi, uning ko‘rinishi quyidagicha

do operator while ifoda;

Sikl jismini tashkil etgan oddiy yoki tarkibiy operator avval bajariladi, so‘ngra mantiqiy ifoda hisoblanadi (u bool turli bo‘lishi shart). Agar ifoda qiymati rost bo‘lsa, sikl jismi yana bir marotaba takrorlanadi va tekshruv davom etadi. Ifodaning qiymati false teng bo‘lsa yoki sikl jismida boshqaruvni uzatish operatori bajarilsa, sikl tugallanadi.

Sikl jismi hech bo‘lmasa bir marotaba bajarilishi shart bo‘lsa bu sikl ko‘rinishi qo‘llaniladi, misol uchun, ma'lumot kiritiladi va tekshiruv bajariladi. Agar bunga zarurat tug‘ilmasa, u holda avvaldan shartli sikldan foydalanish foydaliroq.

Misol. Cheksiz qatorni berilgan eps aniqlikda hisoblansin, (ya'ni ketma - ketlikning hamma hadlari yig‘indisi hisoblansin, hadlari eps berilgan sondan kichik bo‘magan holda).

Berilgan algoritmni amalga oshiruvchi dasturning ko‘rinishi quyidagichadir:

```
using System;
namespace ConsoleArrIication1
{
   class Classl
   {
      static void Main()
      {
         double e = double.Parse(Console.ReadLine());
         int k = 1;
         double s = 0, y;
         do
         {//Ketma-ketlikning keyingi hadini hisoblash
            y = (k + 0.3) / (3 * k * k + 5);
            s += y;    // Ketma-ketlikning yig‘indisini hisoblash
            k++;    // Ketma-ketlikning hadlar sonini hisoblash
         } while (y > e);
         //Hosil bo‘lgan yig‘indi va hadlar soni ekranga chiqarish
         Console.WriteLine("s= {0}    k={1}", s, k - 1);
      }
   }
}
```

for parametrli sikl

Parametrli sikl quyidagi ko‘rinishga ega:

for (initsiallash; ifoda; o‘zgartirish kiritish) operator;

Initsiallash siklda foydalaniladigan o‘zgaruvchini bayon qilishga kerak va unga boshlang’ich qiymat taqdim qilamiz. Bu qismida bir necha operatorni vergul bilan ajratib yozish mumkin.

Misollar:

1. y=sin x berilgan [a, b] oraliqda h qadam bilan funksiyani tabulyatsiya qilish (funksiyaning qiymatini topish).

Berilgan algoritmni amalga oshiruvchi dasturning ko‘rinishi quyidagicha:

```
using System;
namespace ConsoleArrIication1
{
 class Classl
 {
  static void Main()
   {
     double a = double.Parse(Console.ReadLine()); //Orlaliq boshlanishi
     double b = double.Parse(Console.ReadLine()); //Oraliq oxiri
     double h = double.Parse(Console.ReadLine()); //Tabulatsiya qadami
     int n = (int)Math.Truncate((b - a) / h) + 1; //nuqtalar sonini hisobi
     double x = a,y;
     for (int i = 0; i < n; i++)
     {
     y = Math.Sin(x); //Funksiyaning qiymatini hisoblash
     x += h; // Tabulyatsiya keyingi nuqtasini hisoblash
     Console.WriteLine("x= {0}  y={1}", x, y);
     }
}}}
```

2. Berilgan chekli qatorning yig‘indisi hisoblansin, (ya'ni (k=1,2,3..,n). ketma - ketlikning n ta hadlari yig‘indisi hisoblansin).

Berilgan algoritmni amalga oshiruvchi dasturning ko‘rinishi quyidagichadir:

```
using System;
namespace ConsoleArrIication1{
```

```
class Classl {
static void Main(){
int n=int.Parse(Console.ReadLine());
double s=0,y;
//Ketma – ketlikning hadlari soniga qarab siklni tashkil etish
for (int k=1; k<=n; k++)
{
// Ketma – ketlikning keyingi hadini hisoblash
y=(k+0.3)/(3*k*k+5);
s+=y; //Ketma-ketlikning yig‘indisi hisoblash
}
Console.WriteLine("s= {0}", s );     //Hosil bo‘lgan yig‘indini ekranga chiqarish
}}}
```

Label (metka) komponentasi. Oddiy matnni aksettira oladi.

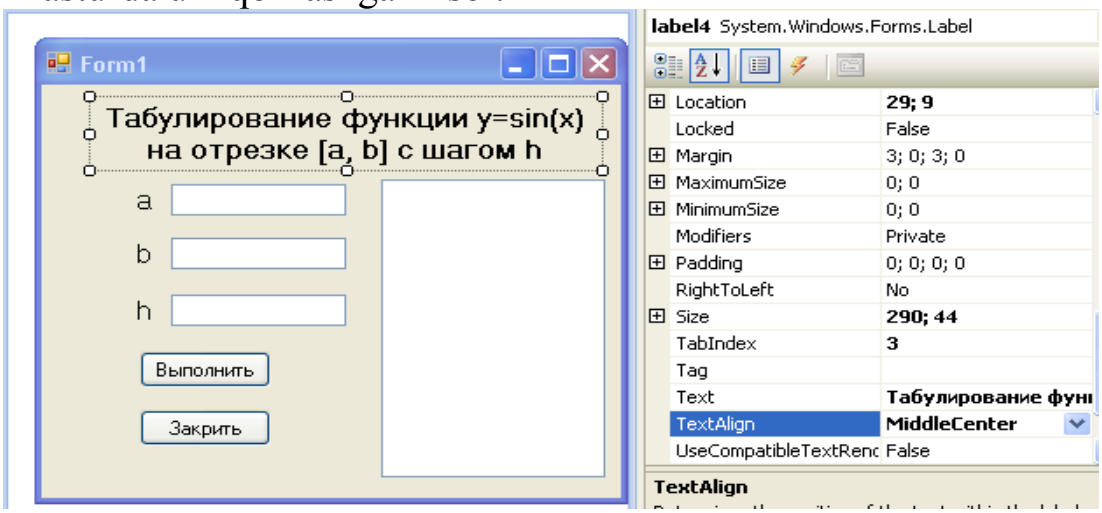
Text Matn beriladi.

TextAlign Komponenta ichida matnni joylanishini tekislash rejimi beriladi.

AutoSize Agar truye bo‘lsa, avtomatik ravishda komponenta o‘lchovi o‘zgaradi, agar false qiymatga ega bo‘lsa komponenta o‘lchovini qo‘lda o‘zgartirish mumkin.

Font Matn shriftini o‘zgartirish.

Dasturda uni qo‘llashga misol:



TextBox (ma'lumot kiritish maydoni) komponentasi. Matn kiritish, akslantirish va ko‘p satrli matnni tahrirlashda foydalanadi.

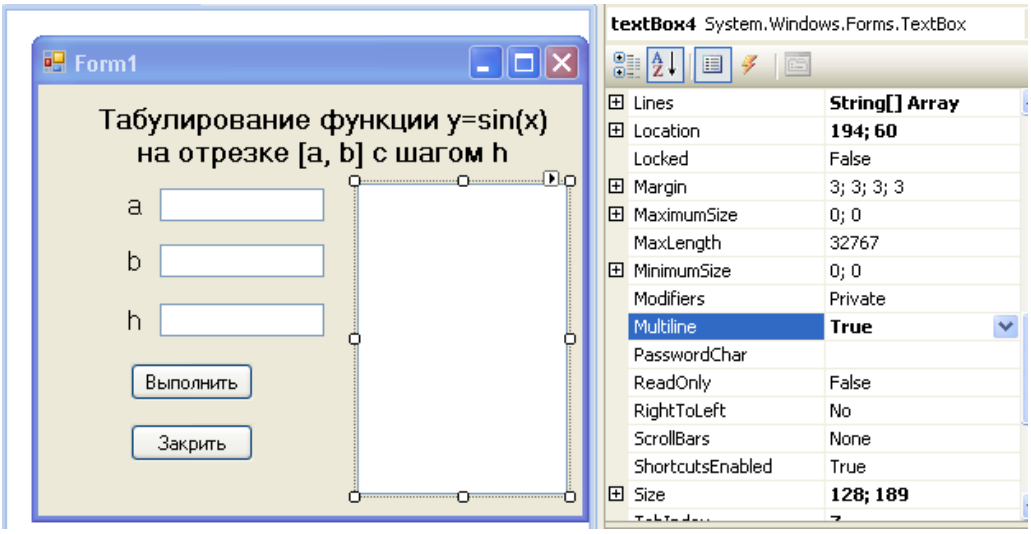
TextAlign Komponenta ichida matnni joylanishini tekislash rejimi beriladi.

Font Matn shriftini o‘zgartirish.

Text TextBox elementi sirtidagi matnni hosil qilishda foydalanadi.

Multiline Agarda qiymati truye bo‘lsa, komponenta ko‘p satrli matnni yozishi mumkin.

Dasturda uni qo‘llashga misol:



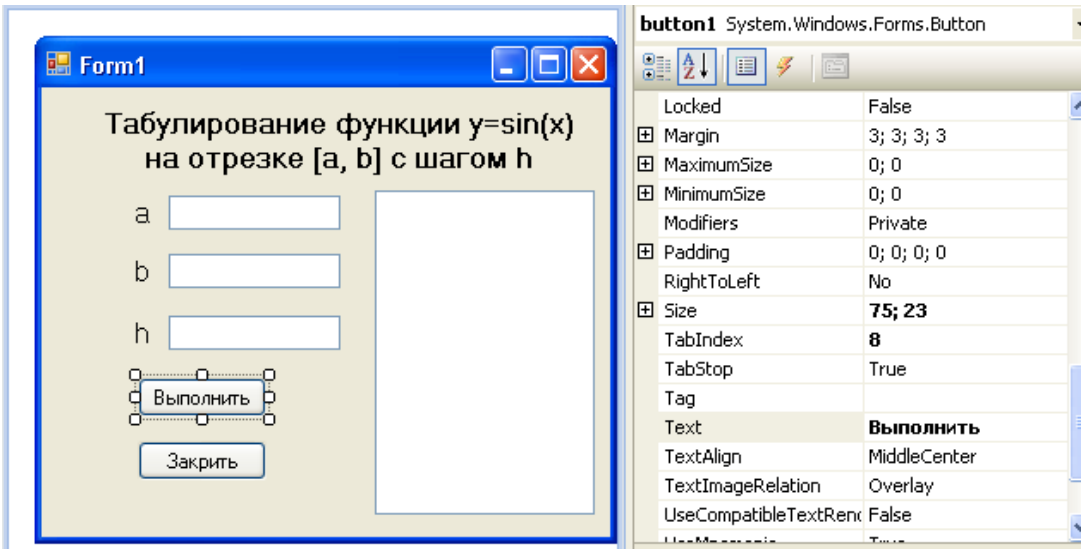
Button (knopka) komponenti. Oddiy knopka sifatida foydalaniladi.

Text Knopka elementining sirtidagi matnni hosil qilishda foydalanadi.

TextAlign Komponenta ichida matnni joylanishini tekislash rejimi beriladi.

Font Matn shriftini o‘zgartirish.

Dasturda undan foydalanishga misol:



TextBox ni to‘ldirish, Text xossasidan foydalanib, funksiyaning tabulyatsiyasi misolida ko‘rsatilgan:

```
double a = double.Parse(textBox1.Text);
double b = double.Parse(textBox2.Text);
double h = double.Parse(textBox3.Text);
double x = a, y;
textBox4.Clear();
string[] s = new string[100];
s[0] = " x    |    y ";
s[1] = "------|------";int i=2;
while (x <= b)
{
   y = Math.Sin(x);
   s[i++] = x.ToString("###.##") + "    |   " + y.ToString("###.##");
   x+=h;
}
textBox4.Lines = s;
```

O‘zlashtirish uchun savollar:

1. Siklik operatorlarning vazifasi?
2. Siklik operatorlarning qansi birini bilasiz?
3. For parametrli sikl operatori qachon ishlatiladi?
4. Avval va so‘ngra shartli sikl operatorlaridan qachon foydalanadi?
5. For parametrli sikl operatorida boshqaruv o‘zgaruvchisi qanday o‘zgaradi?
6. Qachon siklda tarkibiy operator qo‘llanadi?

7. Avval shartli sikl While (Hozircha) operatorida sikl jismi qachon bajariladi?
8. Shartli Do While sikl operatorida sikl jismi qachon bajariladi?
9. Shartli Do While sikl operatoridan chiqish qachon amalga oshiriladi?
10. Avval shartli While sikl operatorida chiqish qachon amalga oshiriladi?
11. While va Do While sillari bir - biridan nima bilan farq qiladi?
12. Button obyekti qanday vazifani bajaradi?
13. Image boshqaruv elementi nima uchun xizmat qiladi?
14. Label qanday formatdan foydalanadi?
15. TextBox obyekti qanday vazifani bajaradi?
16. TextBox komponentining Text xossasidan qanday foydala-niladi?

4-LABORATORIYA ISHI
Mavzu: Massivlar

Ishdan maqsad: Bir o‘lchovli massivlar bilan ishlashni o‘rganish.

Kompyuter yordamida yechiladigan ko‘pgina masalalar yagona matematik mazmun yoki mohiyatan o‘zaro bir-biri bilan bog‘liq bo‘lgan ma'lumotlar yig‘indisini ifoda etuvchi katta hajmdagi axborotni qanta ishlash bilan bog‘liq. Bunday ma'lumotlarni chiziqli yoki to‘g‘ri burchakli jadvallarda ifoda etish qulay sanaladi.

Chiziqli jadvalning har bir elementiga muvofiq o‘zining tartib raqami mavjud. To‘g‘ri burchakli jadvalning elementi uchun esa ikkita raqam ko‘rsatilishi zarur: vertikal yo‘nalishdagi raqam (satr raqami) va gorizontal yo‘nalishdagi raqam (ustun raqami).

Oliy matematikada jadval kattaliklarini vektorlar va matritsalar deb ataydilar.

Massivning tarkibiy qismlarini kompyuter xotirasiga yozish uchun massivning o‘lchami (hajmi) orqali aniqlanadigan, ularni saqlash uchun kerakli bo‘lgan xotira katakchalarini ajratib olish zarur. Shuningdek, boshlang‘ich bazaga ega bo‘lgan raqamlash ham qo‘llanadi, ya'ni massivning elementlari 0 dan boshlab raqamlanadi.

Dasturda har bir massiv uchun o‘z parametr ko‘rsatkichlari belgilangan bo‘lishi lozim: nomi, o‘lchamligi va o‘lchovlari. Ushbu ma'lumot raqamli qiymatlarni saqlash maqsadida zaruriy xotira hajmini rezervatsiya (band)

qilish uchun kerak; u massivlarni tavsiflab beruvchi maxsus operator yordamida belgilanadi.

Masalan, 10 ta butun sondan iborat bo‘lgan massiv va 100 ta satrdan iborat bo‘lgan massivni yaratishga mo‘ljallangan operatorlar quyidagicha ko‘rinishga ega:

```
int[] w = new int[10];
string[] z = new string[100];
```

Bir xil turdagi massivlarni o‘zaro bir-biriga qo‘llash mumkin. Bunda elementlarni emas, balki iqtiboslarni (ssilka) o‘zlashtirish jarayoni ro‘y beradi, masalan:

```
int[] a = new int [10];
int[] b = a; // b va a bitta massivni belgilaydi (ko‘rsatadi).
```

C# dagi barcha massivlar System nomlar bo‘shlig‘ida aniqlangan Array deb ataluvchi umumiy baza turkumiga ega. Unda massivlar bilan ishlashni yengillashtiruvchi bir necha foydali usullar mavjud, masalan, o‘lchamlik, saralash va qidiruv usullari. C# da massivlarning 3 ta turi mavjud: bir o‘lchamli, to‘g‘ri burchakli va pog‘onali (tekislanmagan).

Bir o‘lchamli massivlar

Dasturlarda bir o‘lchamli massivlardan ko‘p foydalaniladi. Massivlarni tavsiflash variantlari (turlari):

```
tur[] nom:
tur[] nom = new tur [o‘lchamlik];
tur[] nom = {initsializatorlar ro‘yxati };
tur[] nom = new tur [] {initsializatorlar ro‘yxati};
tur[] nom = new tur [o‘lchamlik] {initsializatorlar ro‘yxati}
```

Tavsif namunalari (har bir tavsif varianti uchun bitta namuna):

```
int[] a // 1 elementlar yo‘q
int[] b= new int[4];                    // 2  elementlar teng O
int[] s = { 61, 2, 5, -9 };        //3   new anglatadi
int[] d = new int[] { 61,2, 5,-9 };     // 4  o‘lchamligi aniqlanadi
int[] e = new int[4] { 61,2, 5, -9 };   // 5  ortiqcha tavsif
```

Bu yerda beshta massiv tavsiflangan. Birinchi operatorning boshqalaridan farqi shundaki, unda faqatgina massivning iqtibosi

tavsiflangan, massivning elementlari uchun xotira esa ajratilmagan. Agar inizializatsiya ro‘yxati belgilanmagan bo‘lsa, o‘lchamlik nafaqat konstanta, balki butunlikka olib keluvchi turning ko‘rinishi ham bo‘lishi mumkin.

Random turkumi

Massivlardan foydalanadigan dasturlarni sozlashda, tasodifan belgilangan dastlabki ma'lumotlarni yuzaga chiqarish bir qancha qulayliklar tug‘diradi. C# kutubxonasida bu kabi vaziyatlar uchun System nomlar bo‘shlig‘ida aniqlangan Random turkumi mavjud. Tasodifiy bo‘lmagan raqamlar ketma-ketligini aniqlash uchun avvalambor, konstruktor yordamida turkumning nusxasini yaratish zarur, masalan:

```
Random a = new Random(); //1
Random b = new Random(1 ); //2
```

Konstruktorning ikki turi mavjud: parametrlarga ega bo‘lmagan konstruktor

(1-operator) generatorning joriy vaqt oralig‘ida aniqlangan dastlabki vazifasidan foydalanadi. Bunday vaziyatda har gal betakror ketma-ketlik vujudga keladi. Int turidagi parametrga ega bo‘lgan konstruktor (2-operator) sonlarning bir xil ketma-ketligini olish imkonini beruvchi generatorning dastlabki vazifasini belgilaydi.

Galdagi qiymatlarni aniqlash uchun jadvalda keltirilgan usullardan foydalaniladi:

4.1-jadval

System turkumilagi asosiy usullslar. Random	
Номи	Tavsifi
Next()	int turidagi musbat oraliq butun musbat raqamni qaytaradi
Next(макс)	Oraliqda butun musbat raqamni qaytaradi [0, макс]
Next(min, макс)	Oraliqda butun musbat raqamni qaytaradi [min, макс]
NextBytes(massiv)	Oraliqda raqamlar massivini qaytaradi[0, 255]
NextDouble ()	Oraliqda moddiy musbat raqamni qaytaradi [0,1]

4.1-varaqcha. Tasodifiy bo‘lmagan raqamlar generatori bilan ishlash.

```
using System;
namespace ConsoleApplication1{
class Classl
{static void Main()
```

```
{
Random a = new Random();
Random b = new Random( 1 );
const int n = 10;
Console.WriteLine( "\n Oraliq [0, 1]:" );
for ( int i = 0; i < n; ++i )
Console.Write("{0,6:0.##}", a.NextDouble() );
Console.WriteLine("An Oraliq [0, 1000]:" );
for ( int i = 0; i < n; ++i )
Console.Write(" " + b.Next( 1000 ) );
Console.WriteLine( "\n Oraliq [-10, 10]:" );
Console.ReadKey();
}
}}
for ( int i = 0; i < n; ++i )
Console.Write( " " + a.Next(-10, 10) );
Console.WriteLine( "\n Massiv [0, 255]:" );
byte[] mas = new byte[n];
a.NextBytes(mas);
for (int i = 0; i < n; ++i) Console.Write( " " + mas[i] );
```

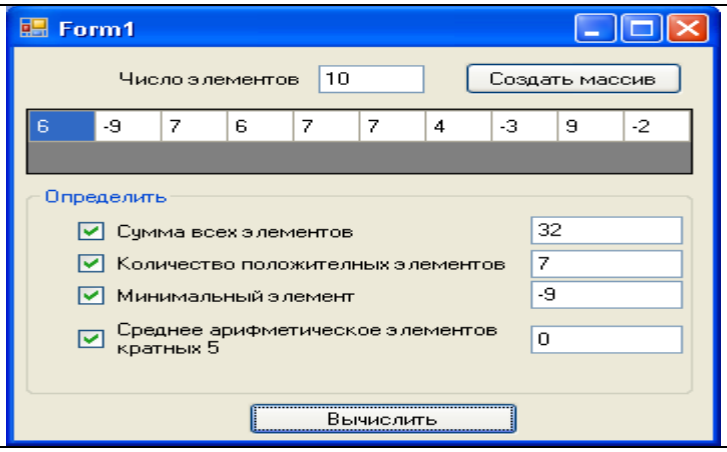
Dastur ishining natijasi:

```
Oraliq [0, 1]:
0,02 0,4 0,24 0,55 0,92 0,84
Oraliq [0, 1000]:
248 110 467 771 657 432
Oraliq [-10, 10]:
-8 9 -6 -10 7 4 9 -5
Massiv [0, 255]:
181 105 60 50 70 77 9 28 133 150
0,9 0,78 0,78 0,74
354 943 101 642
-2 -1
```

Massivning shakllanishi, elementlar soni, yig‘indisi, ko‘paytmasi, massiv elementlarining o‘rtacha arifmetikasi, massivning maksimal va minimal elementini hisoblash

Namuna. Ilova foydalanuvchiga Chiziqli massivning hajmini belgilash imkonini beradi, massivni 10 dan -10 gacha bo‘lgan oraliqdagi tasodifiy butun raqamlar bilan to‘ldiradi, massiv elementlarining ro‘yxatini

chiqaradi, so‘ng foydalanuvchining tanloviga ko‘ra quyidagilarni aniqlaydi: massivning barcha elementlari yig‘indisi, ko‘paytmasi, massivning musbat elementlari soni, massivning eng kichik elementi va besh karrali massiv elementlarining o‘rtacha arifmetik soni.

Ilovaning tarkibiy qismlari	Ilova shaklining oynasi (oknasi)
TextBox Label CheckBox RadioButton Button GroupBox	

GroupBox tarkibiy qismlariga qisqacha tavsifnoma

GroupBox paneli – bu RadioButton ulagichlari, CheckBox bayroqchalari va boshqalar kabi boshqaruv organlari bilan bog‘liq bo‘lgan guruhni birlashtiruvchi ramka va yozuvga ega bo‘lgan konteynerdir.

GroupBox panelining xususiyatlari

Text Birlashgan tarkibiy qismlar guruhining ramkasi uchun yozuvni belgilab beradi

Ilovaning dasturiy kodi

E'tibor bering, ilovaning barcha modullari uchun qulay bo‘lgan turlar, konstantalar, o‘zgaruvchilar, funksiyalar va bajariladigan ishlarni e'lon qilish bo‘limida N va M massivlarining hajmlari ifodalangan. Chunki bu o‘zgaruvchilardan turli xil vaziyatlarda, masalan, massivlarni tasodifiy raqamlar bilan to‘ldirish jarayonida private void button1_Click(object sender, EventArgs e) va

massivni qanta ishlash jarayonida private void button2_Click(object sender, EventArgs e) foydalaniladi.

Barcha kichik masalalar massivning elementlarini bir ko‘rib chiqishda yechiladi.

Masalan, massivda minimal elementni qidirish kerak. Avvalambor, massivning boshlang‘ich elementi bo‘yicha joriy minimum belgilanadi min=M[0]. So‘ng massiv elementlari ko‘rib chiqiladi: galdagi M[i]elementi tanlanadi va min bilan taqqoslanadi. Agar M[i] elementi joriy min dan kichik bo‘lsa, min=M[i] qanta o‘zlashtiriladi.

```
public Form1()
{
InitializeComponent();
}
int N;
int[] M;
Random rnd = new Random();
private void button1_Click(object sender, EventArgs e)
{
N = int.Parse(textBox1.Text);
dataGridViyew1.ColumnHeadersVisible = false;
dataGridViyew1.RowHeadersVisible = false;
dataGridViyew1.AutoSizeColumnsMode =
DataGridViyewAutoSizeColumnsMode.Fill;
dataGridViyew1.RowCount = 1;
dataGridViyew1.ColumnCount = N;
M = new int[N];
for (int i = 0; i < N; i++)
{
M[i]=rnd.Next(-10, 10);
dataGridViyew1.Rows[0].Cells[i].Valuye = M[i].ToString();
}
}
private void button2_Click(object sender, EventArgs e)
{
int Sum, Min, CountP, Kol; double Sum1;
Min = M[0];
textBox2.Text = textBox3.Text = textBox4.Text = textBox5.Text = "";
Sum = CountP = Kol = 0; Sum1 = 0;
for (int i = 0; i < N; i++)
{
if (checkBox1.Checked) Sum += M[i];
if (checkBox2.Checked) if (M[i] >= 0) CountP++;
```

```
if (checkBox3.Checked) if (Min > M[i]) Min = M[i];
if (checkBox1.Checked) if ((M[i] % 5 == 0) && (M[i] != 0)) {
Sum1 += M[i]; Kol++; }
}
if (checkBox1.Checked) textBox2.Text = Sum.ToString();
if (checkBox2.Checked) textBox3.Text = CountP.ToString();
if (checkBox3.Checked) textBox4.Text = Min.ToString();
if (checkBox4.Checked) { if (Kol != 0)textBox5.Text = (
Sum1 / Kol).ToString(); else textBox5.Text = "0"; }
}}}
```

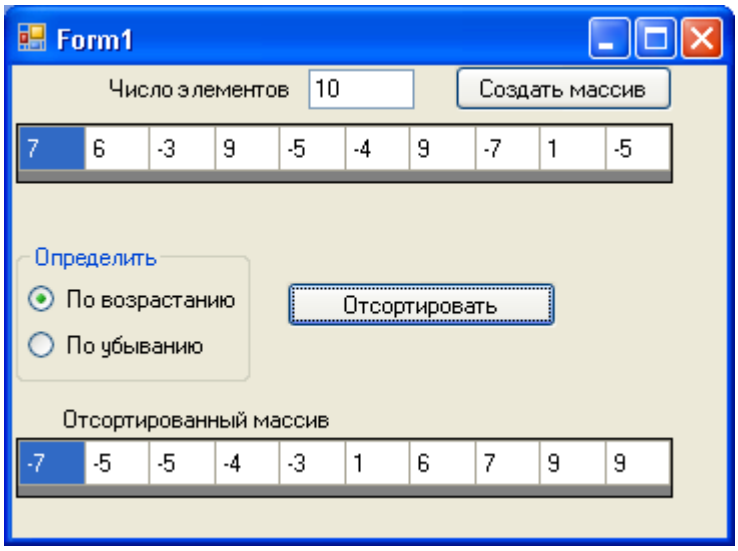
Tanlov asosida saralash

Ushbu usulning mohiyati juda oddiy, u quyidagicha ta'riflanishi mumkin:

1. n elementlar ketma-ketligida eng kichik (eng katta) element tanlab olinadi;
2. birinchi element bilan o‘rin almashadi;
3. toki eng katta (kichik) element qolmaguncha, avval qolgan n-1 elementlari bilan, keyin esa qolgan n-2 elementlari bilan va h. bu jarayon takrorlanadi.

Mazkur algoritmni amalga oshirish uchun For parametriga ega bo‘lgan o‘rnatilgan ikkita sikldan foydalanish zarur. Tashqi sikl (i bo‘yicha) massiv elementlarini ketma-ket tarzda qand qilish uchun mo‘ljallangan, ichki sikl (j bo‘yicha) – massivning saralanmagan qismida minimal (maksimal) elementni va uning joylashuvini aniqlaydi. Ichki sikldan chiqib bo‘lingach, elementlar qanta joylashadi. Tashqi sikldagi so‘nggi element ko‘rib chiqilmaydi: u o‘z-o‘zidan joyiga turib qoladi.

Ilova shaklining oynasi (oknasi)



Tanlov asosidagi saralash jarayonining dasturiy kodi

```
public Form1()
{
InitializeComponent();
}
int N;
int[] M;
Random rnd = new Random();
private void button1_Click(object sender, EventArgs e)
{
N=int.Parse(textBox1.Text);
dataGridViyew1.ColumnHeadersVisible = false;
dataGridViyew1.RowHeadersVisible = false;
dataGridViyew1.AutoSizeColumnsMode =
DataGridViyewAutoSizeColumnsMode.Fill;
dataGridViyew1.RowCount = 1;
dataGridViyew1.ColumnCount = N;
M = new int[N];
for (int i = 0; i < N; i++)
{
M[i]=rnd.Next(-10, 10);
dataGridViyew1.Rows[0].Cells[i].Valuye = M[i].ToString();
}
}
private void button2_Click(object sender, EventArgs e)
{
dataGridViyew2.ColumnHeadersVisible = false;
dataGridViyew2.RowHeadersVisible = false;
```

```
dataGridViyew2.AutoSizeColumnsMode=
DataGridViyewAutoSizeColumnsMode.Fill;
dataGridViyew2.RowCount = 1;
dataGridViyew2.ColumnCount = N;
if (radioButton1.Checked)Array.Sort(M);
if (radioButton2.Checked)
{
for (int i = 0; i < N - 1; i++)
for (int j = i+1; j < N; j++)
if (M[j] > M[i])
{
int p=M[i];
M[i]= M[j];
M[j]=p;
}
}
for (int i = 0; i < N; i++)
{
dataGridViyew2.Rows[0].Cells[i].Valuye = M[i].ToString();
}
}
```

Pufakcha usuli yordamida saralash

Mazkur usul qo‘shni elementlarning o‘zaro taqqoslanishiga asoslanadi. Bir-biriga nisbatan “noto‘g‘ri” joylashgan elementlar joy almashadilar. O‘rnatilgan sikllarda massivning juft qo‘shni elementlari navbatma-navbat qand qilinadi. Natijada, minimal ko‘rsatkichli element massivning birinchi pozitsiyasiga (pog‘onasiga) chiqadi.

Pufakcha usuli bilan saralashning dasturiy kodi fragmenti (parchasi):

```
for (int i = 0; i < n - 1; i++)
   for (int j = n - 1; j > i + 2; j--)
     if (a[j - 1] < a[j])
       {
        p = a[j - 1]; a[j - 1] = a[j]; a[j] = p;
       }
```

Massivning zichlanishi – uning tarkibidan u yoki bu talablarga javob beruvchi elementlarni o‘chirib tashlashdir. Vujudga kelgan bo‘shliqlar qolgan elementlarning surilishi hisobiga to‘ldiriladi. Massiv kichrayishi

sababli uni qanta ishlash jarayonida parametrli sikldan emas, talab (shart) siklidan foydalanish zarur.

Namuna. Shakllangan massivdan uch karrali sonlarni o‘chirib tashlang.

Massivning zichlanishi dasturiy kodining fragmenti (E'tibor bering: o‘chirib tashlangan i elementining o‘rniga i+1 elementi yoziladi, i+1 elementi o‘rniga esa i+2 elementi yoziladi va hokazo):

```
int i = 0;
while (i <= N)
  {
   if ((a[i] % 3 != 0) || (a[i] == 0))
   {      //  Elementlarning 3 karraligini tekshirish
   i++;  // Sikl parametrining kattalashishi
   }
   else{          // Agar element 3 karrali bo‘lmasa
   for (int j = i; i < n - 1; i++) a[j] = a[j + 1];
   n--;   // Massiv elementlari sonining 1 ga kamayishi
   }
 }
```

Massiv tarkibiga elementni joylashtirish (qo‘shish) - oldingi masalaga nisbatan butunlay teskari jarayon. Bir xil usuldan foydalanadi, ya'ni elementlar guruhi bir pozitsiyaga suriladi. Zichlanish jarayonida elementlar chap tomonga suriladi, elementlarni joylashtirishda esa - o‘nga. Elementlarni joylashtirish (qo‘shish) jarayonida so‘nggi elementlarni nima qilish kerak degan muammo tug‘iladi. Agar massiv bilan ishlash jarayonida faqatgina belgilangan elementlargina ishtirok etsa, qoldiq elementlarni (so‘nggi elementlarni) siqib chiqarishga to‘g‘ri keladi, bunda oxirgi qiymatlar yo‘qolib ketadi. Aks holda, qo‘shilgan elementlar hisobiga boshlang‘ich massivning o‘lchamligidan kattaroq bo‘lgan qo‘shimcha massivni yaratish zarur. Har bir aniq vaziyatga qarab, massiv bilan ishlash siklining turi tanlanadi.

Namuna. Belgilangan va elementlari kichikdan kattaga qarab tartibga solingan massivga uning tartibini buzmagan holda belgilangan raqamni qo‘shing. Oxirgi elementni siqib chiqaring.

Bunda parametrli sikldan foydalanish mumkin, chunki oxirgi elementlar siqib chiqariladi va shu bilan, massivdagi elementlar o‘zgarmaydi.

Dasturiy kod fragmenti (parchasi):

```
// Massiv elementlarini saralash sikli - yangi k qiymati uchun joy qidirish
for (int i = 0; i < n; i++)
  {
   if (k < a[i])
     {
      // Joy topildi, elementlarning bir pog‘ona o‘ngga surilishi sikli
      for (int j = n - 1; j >= i + 1; j--) a[j] = a[j - 1];
      a[i] = k;  // Bo‘shatilgan joyga yangi qiymatni joylashtirish
      break;      // Operator siklidan chiqish
      }
  }
```

Bunday masalalarni yechishda o‘zgarayotgan indekslar oralig‘ining chegaralarini nazorat qilish muhim sanaladi: ular butun bo‘lishi, oraliq chegarasidan chiqib ketmasligi, shuningdek, oraliqning pastki chegarasi ustki chegaradan kichikroq bo‘lishi zarur.

Namuna. n elementlaridan tashkil topgan bir o‘lchamli massivda qiymatlar ketma-ketligi tartibini teskari tomonga – n1 pozitsiyasidan n 2 pozitsiyasiga o‘zgartiring (n1<n2<n).

Dasturiy kod fragmenti (keltirilgan dastur fragmentida sikl oraliqning yarmigacha bajariladi (n1 dan (n1+n2) div 2 gacha), aks holda massivda hech narsa o‘zgarmaydi):

```
// element indeksi 0 dan boshlanadi!
        n1 = n1 - 1;
        n2 = n2 - 1;
        int k = (int)((n1 + n2) / 2);
        // elementlar ketma-ketligi tartibining o‘zgarishi sikli
        for (int i = n1; i <= k; i++)
        {
           int p = a[i];       // joylarni alamashtiring
           a[i] = a[n1 + n2 - i];
           a[n1 + n2 - i] = p;
```

```
}
```

Aylanali surilish – massivning o‘ng yoki chap tomonga surilishi bo‘lib, siqib chiqarilgan elementlar massivning qarama-qarshi tomonlarida, ya'ni surilish natijasida bo‘sh qolgan joylarini band qiladilar. Bunda massiv aylana shakliga kiradi (birinchi va oxirgi elementlar qo‘shilib ketadi). Elementlarning ketma-ketlik tartibi esa saqlanib qoladi.

Namuna n elementlaridan tashkil topgan bir o‘lchamli massivda elementlarning k pozitsiyalarga aylanali surilishini amalga oshiring. k qiymati belgilanadi, u musbat yoki manfiy bo‘lishi mumkin, ammo butun bo‘lib, oraliqda joylashgan bo‘lishi zarur: - (n-1) < k < n-1

Dasturiy kod fragmenti (parchasi):

```
/*k raqami tahlili, k 0 dan farqli bo‘lsagina, surilish amalga oshiriladi*/
if (k != 0){
    if (k > 0) sdving = k; else sdving = n + k;
    for (int i = 1; i <= sdving; i++)
       {
        tmp = a[n - 1];
        for (int j = n - 2; j >= 0; j--) a[j + 1] = a[j];
        a[0] = tmp;  }}
```

O‘zlashtirish uchun savollar:

1. Ma'lumotlar massivi nima?
2. Massivlar dasturiy kodning qansi bo‘limida va qan tarzda tavsiflanadi?
3. Qanday qilib massivdagi elementning joylashuvini aniqlash mumkin?
4. Indeks nima? U qanday talablarga javob berishi kerak?
5. Massiv elementlari bilan ishlash qan tarzda amalga oshiriladi?
6. Dinamik massiv nima?
7. Statik va dinamik massivlarning farqini tushuntiring. Dinamik massivlarni tavsiflash va ulardan foydalanish tartibi qanday?

5-LABORATORIYA ISHI
Mavzu: Ikki o’lchamli massivlar

Ishdan maqsad: Ikki o‘lchovli mvssivlar bilan ishlashni o‘rganish.

To‘g‘ri burchakli massivlar

To‘g‘ri burchakli massiv 1 dan ortiq o‘lchamga ega. Dasturlarda ikki o‘lchamli massivlardan ko‘p foydalaniladi. Ikki o‘lchamli massivga beriladigan tavsifning variantlari:

tur[,] nom;

tur[,] nom = new tur [o‘lcham_1, o‘lcham_2];

tur[,]nom= { initsializatorlar ro‘yxati };

tur[,] nom = new tur [,] { initsializatorlar ro‘yxati };

tur[,] nom = new tur [o‘lcham_1, o‘lcham_2] { initsializatorlar ro‘yxati };

Tavsif namunalari (har bir tavsif varianti uchun bitta namuna):

```
int[,] a; //1       elementlar yo‘q

int[,] b = new int[2, 3]; //2         elementlar teng O
int[,] s = {{1, 2, 3},{4, 5, 6}}; // 3      new anglatadi
int[,] s = new int[,] {{1, 2, 3}, {4, 5, 6}}; // 4       o‘lchamligi
aniqlanadi
int[,] d = new int[2,3] {{1, 2, 3}, {4, 5, 6}};// 5    ortiqcha tavsif
```

Agar initsializatsiya ro‘yxati berilmagan bo‘lsa, o‘lchamlik konstanta va butunlikka olib keluvchi turning ifodasi bo‘lishi mumkin. Ikki o‘lchamli massivning elementi bilan ishlashda, ushbu element joylashgan satr va ustun raqami ko‘rsatiladi, masalan:

```
a[1, 4]          b[i, j]              b[j, i]
```

Kompilyator dasturdagi indeksning qanday ifodalanishidan qat'iy nazar, satrning raqami sifatida qabul qilishini esda tutish zarur.

Pog‘onali massivlar

Pog‘onali massivlarda turli satrlardagi elementlar soni bilan farqlanishi mumkin. Pog‘onali massiv to‘g‘ri burchakli massivga nisbatan xotirada biroz boshqacha – har biri o‘z o‘lchamiga ega bo‘lgan bir necha ichki massivlar ko‘rinishida saqlanadi. Bundan tashqari, har bir ichki massivning iqtibosi uchun alohida xotira sohasi ajratiladi.

Pog‘onali massivning tavsifi:

Tur [][] nom;

Pog‘onali massivni tashkil etuvchi har bir massiv tagiga aniq tarzda xotira ajratish zarur, masalan:

int[][] a = new int[3][]; // iqtiboslar uchun uchta satr ko‘rinishida xotira ajratish

a[0] = new int[5]; // 0-satr uchun xotira ajratish (5 ta element)

a[1] = new int[3]; // 1-satr uchun xotira ajratish (3 ta element)

a[2] = new int[4]; // 2-satr uchun xotira ajratish (4 ta element)

Bunda a[0], a[1] i a[2] — alohida massivlar. Xotira ajratishning boshqa usuli:

int[][] a = { new int[5], new int[3], new int[4] };

Pog‘onali massivning elementi bilan ishlashda, uning o‘lchamligi to‘rtburchak qavslarda ko‘rsatiladi, masalan:

a=[i][2] a[i][j] a[j][i]

Qolgan vaziyatlarda pog‘onali massivlardan foydalanish to‘g‘ri burchakli massivlardan foydalanish jarayoni bilan bir xil.

To‘g‘rilanmagan (tekislanmagan) massivlardan katta hajmdagi uchburchak matritsalar bilan ishlashda foydalanadi.

System.Array turkumi

C# da massivlar dasturchi uchun kerakli (foydali) bo‘lgan xususiyat va usullardan tashkil topgan Array baza turkumi asosida qurilishi aytib o‘tildi. Ularning bir qismi 5.1-jadvalda ko‘rsatilgan.

5.1-jadval

Array turkumining asosiy elementlari		
Element	**Tur**	**Tafsifi**
Length	Xususiyat	Massiv elementlarining soni (barcha o’lchamliklar bo’yicha).
Rank	Xususiyat	Massiv o’lchamliklarining soni
BinarySearch	Statik usul	Saralangan massivda ikkilik qidiruvi (poisk)
Clear	Statik usul	Dastur talabiga ko‘ra qiymatlarning massiv elementlari tomonidan o‘zlashtirilishi
Copy	Statik usul	Bir massiv elementlarining oralig‘ini ikkinchi massivga nusxa ko‘chirish
CopyTo	Usul	Joriy bir o‘lchamli massiv elementlarini ikkinchi bir

		o‘lchamli massivga nusxa ko‘chirish
GetValue	Usul	Massiv elementining qiymatini qabul qilish
IndexOf	Statik usul	Bir o‘lchamli massivga dastlabki kiritilgan elementni qidirib topish
LastlndexOf	Statik usul	Bir o‘lchamli massivga oxirgi kiritilgan elementni qidirib topish
Reverse	Statik usul	Elementlar ketma-ketligi tartibining teskari tomonga o‘zgarishi
SetValue	Usul	Massiv elementining qiymatini o‘rnatish
Sort	Statik usul	Bir o‘lchamli massiv elementlarini tartibga solish

Length xususiyati turli xil uzunlikka ega bo‘lgan massivlar (masalan, pog‘onali massiv) bilan ish olib boruvchi algoritmlarni yo‘lga qo‘yish imkonini beradi. Ushbu xususiyatni aniq belgilangan o‘lchamlik o‘rniga qo‘llash indeksning massiv chegarasidan tashqarisiga chiqib ketmasligini ta'minlaydi.

6.3-varaqchada bir o‘lchamli massivning Array turkumi bilan ishlash jarayonida elementlarning qo‘llanilishi ko‘rsatilgan.

6.3-varaqcha. Array turkumi elementlarining bir o‘lchamli massiv bilan qo‘llanilishi.

```
using System;
namespace ConsoleApplication1
{
   class Classl
   {
      static void Main()
      {
         int[] a = { 24, 50, 18, 3, 16, -7, 9, -1 };
         PrintArray("Isxodniy massiv:", a);
         Console.WriteLine(Array.IndexOf(a, 18));
         Array.Sort(a);
         PrintArray("Uporyadochenniy massiv:", a);
         Console.WriteLine(Array.BinarySearch(a, 18));
         Console.ReadKey();
```

```
        }
        public static void PrintArray(string header, int[] a)
        {
           Console.WriteLine(header);
           for (int i = 0; i < a.Length; ++i)
              Console.Write("\t" + a[i]);
           Console.WriteLine();
        }
     }
}
```

Sort, IndexOf va BinarySearch usullari statik usullar qatoriga kiradi, shuning uchun ular bilan ishlashda nusxa nomi emas, turkum nomidan foydalanadi hamda ular orqali massivning nomi uzatiladi. Ikkilik qidiruvini faqatgina tartibga solingan massivlarda qo‘llash mumkin. Bu qidiruv turi IndexOf usulida amalga oshirilgan chiziqli qidiruvga (lineyniy poisk) nisbatan tezroq ishlaydi. Varaqchada 18 qiymatiga ega bo‘lgan elementni qidirishda, ushbu qidiruv usullarining ikkalovidan ham foydalanadi.

Classl turkumida massivning ekranga uzatilishini ta'minlovchi PrintArray yordamchi statik usuli tavsiflangan. Unga ikkita parametr uzatiladi: header sarlavhasining satri va massivi. Massiv elementlarining soni ushbu usulning o‘zida Length xususiyati yordamida aniqlanadi. Istalgan butun sonli bir o‘lchamli massivlarni chiqarishda ushbu usuldan foydalanadi.

Dastur ishining natijasi:
Dastlabki belgilangan massiv:
24 50 18 3 16 -7 9 -1
Tartibga solingan massiv:
-7 -1 3 9 16 18 24 50

Boshqa turdagi elementlardan tashkil topgan massivlarda PrintArray usulini qo‘llashda, uning ikkinchi parametrini Array sifatida tavsiflash mumkin. Bunda massiv elementining qiymati GetValuye usuli yordamida olinadi, chunki indeks orqali dasturga kirish imkoniyati Array turkumida ko‘zda tutilmagan.

DataGridViyew komponentining (tarkibiy qismining) qisqacha tavsifi quyidagicha:

DataGridViyew komponenti satrlardan tashkil topgan jadvalda ifodalanadi. Jadvalda prokrutka chiziqlari bo‘lishi mumkin, bunda jadvaldagi birinchi satr va ustunlarning belgilangan raqami bo‘lib, prokrutkaga bo‘ysunmaydi. Shunday qilib, komponent oynasida doimiy

mavjud bo‘lgan ustun va satr sarlavhalarini belgilash mumkin. Jadvalning har bir katakchasiga muvofiq holda biron bir obyekt qo‘yilishi mumkin.

DataGridViyew komponenti xususiyatlari

Rows[i].Cells[j] Ushbu xususiyatda jadvalning barcha elementlari saqlanadi.

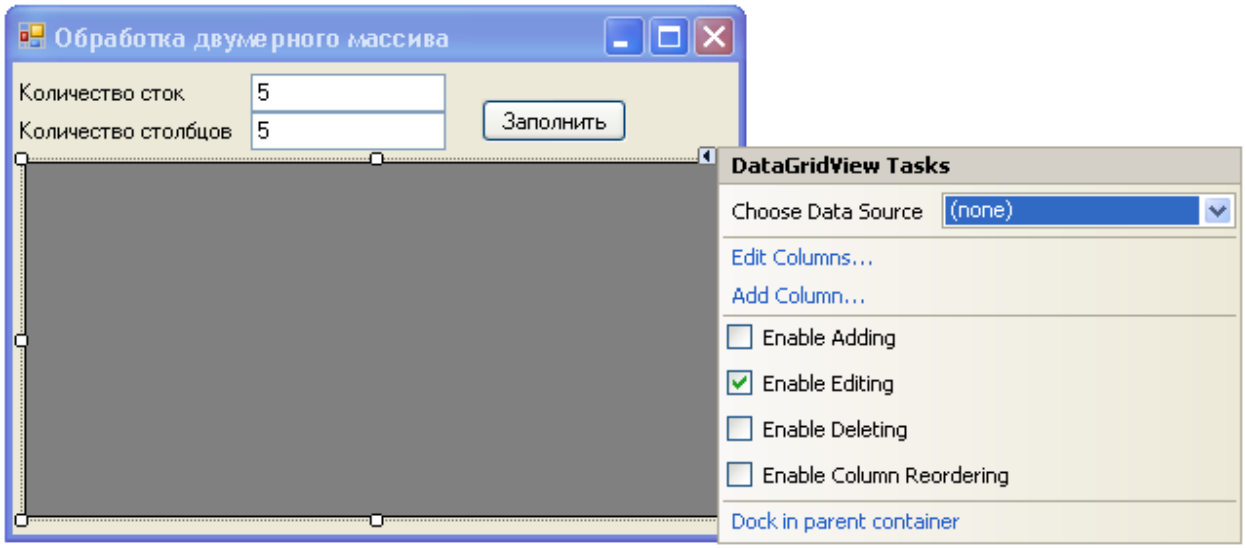
ColumnHeadersVisible jadvalda ustunlarning qand qilingan sonini belgilaydi.

RowHeadersVisible jadvalda satrlarning qand qilingan sonini belgilaydi.

Enable Editing dastur talabiga ko‘ra ma'lumotlarni jadvalga kiritish mumkin emas. Ushbu xususiyatda mazkur taqiqni yo‘qotish uchun Truye ni ishga solish zarur.

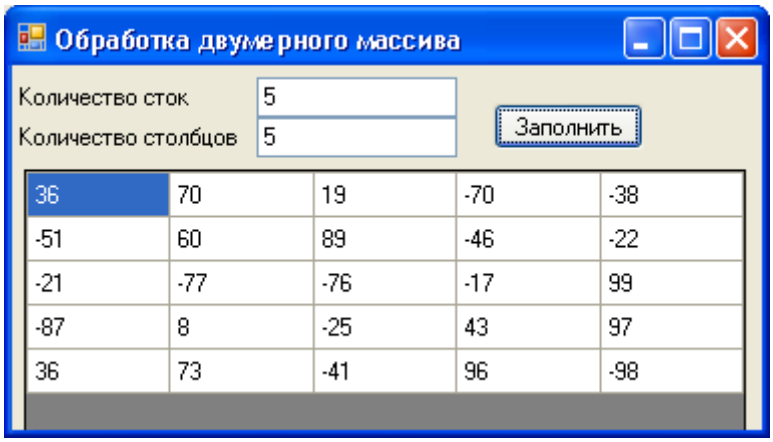
ColumnCount. jadvaldagi ustunlarning umumiy sonini belgilaydi.

RowCount. jadvaldagi satrlarning umumiy sonini belgilaydi.



Ikki o‘lchamli massivni chiqarishda DataGridViyew komponentidan foydalanish

Ilova shaklining oynasi



Ilovaning dasturiy kodi

```
private void button1_Click(object sender, EventArgs e)
{
 Random rnd=new Random();
 int n = int.Parse(textBox1.Text);
 int m = int.Parse(textBox2.Text);
 int [,] A=new int[n,m];
 dataGridViyew1.ColumnHeadersVisible = false;
 dataGridViyew1.RowHeadersVisible = false;
 dataGridViyew1.AutoSizeColumnsMode =
DataGridViyewAutoSizeColumnsMode.Fill;
 dataGridViyew1.RowCount = n;
 dataGridViyew1.ColumnCount = m;
    for (int i = 0; i < n; i++)
     for (int j = 0; j < m; j++)
       {
        A[i, j] = rnd.Next(-100, 100);
        dataGridViyew1.Rows[i].Cells[j].Valuye = A[i, j];
       }
}
```

Ikki o‘lchamli massivlarning tipovoy (ishchi) masalalariga har bir satr va ustun elementlarining hamda belgilangan satr va ustunlarning yig‘indisi, soni, o‘rta arifmetik qiymati, maksimumi, minimumini hisoblash kabilar kiritiladi.

Bunday turdagi hisoblash ishlari standart amallar yordamida amalga oshiriladi. Ularning asosiy xususiyati – massivlarni qanta ishlash jarayonida o‘rnatilgan sikllarni tashkillashtirishda ifodalanadi.

O‘rnatilgan parametrli siklning sxemasi

```
for (int i = a1; i < a2; i+=h1)
{
 for (int j = b1; j < b2; j+=h2)
      {
           ...
      }
}
```

Tarkibida boshqa sikl mavjud bo‘lgan sikl – tashqi, boshqa sikl tanasida mavjud bo‘lgan sikl – ichki yikl deyiladi. Ichki sikldagi barcha operatorlar tashqi sikl tanasida joylashishi zarur.

Ichki sikl tugallanganda, tashqi sikl o‘z schyotchigini (hisoblagich) 1-siklga oshiradi, so‘ng ichki sikl qanta bajariladi.

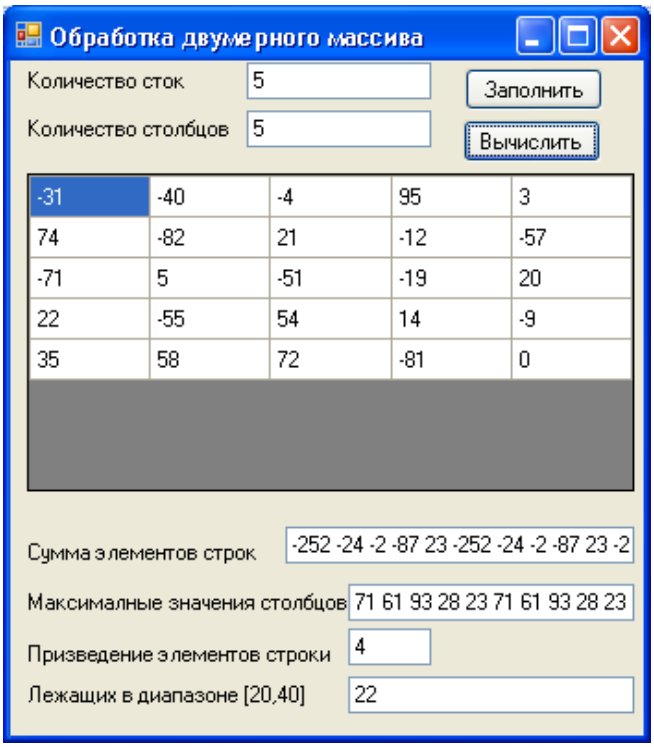
Shuning uchun tashqi sikl schyotchigi o‘rniga satr indeksi, ichki sikl schetchigi o‘rniga esa ustun raqami olinsa, ikki o‘lchamli massivni qanta ishlash jarayoni satr bo‘ylab kechadi, agar teskari bo‘lsa, ustun bo‘ylab kechadi.

Namuna.

M * N ikki o‘lchamli massivni [-40,40] oralig‘idan olingan tasodifiy butun sonlar bilan to‘ldiring. Aniqlang:

1. Har bir satr elementlarining yig‘indisi;
2. Har bir satr uchun maksimal qiymatlar;
3. 20 dan 40 gacha oraliqda qiymatlari joylashgan k satrining elementlarini ishlab chiqish

Ilova shaklining oynasi



Dasturiy kod fragmenti

```
int n, m;
int[,] A;
private void button1_Click(object sender, EventArgs e)
{
  Random rnd = new Random();
  n = int.Parse(textBox1.Text);
  m = int.Parse(textBox2.Text);
  A = new int[n, m];
  dataGridViyew1.ColumnHeadersVisible = false;
  dataGridViyew1.RowHeadersVisible = false;
  dataGridViyew1.AutoSizeColumnsMode=
DataGridViyewAutoSizeColumnsMode.Fill;
  dataGridViyew1.RowCount = n;
  dataGridViyew1.ColumnCount = m;
  for (int i = 0; i < n; i++)
    for (int j = 0; j < m; j++)
     {
      A[i, j] = rnd.Next(-100, 100);
      dataGridViyew1.Rows[i].Cells[j].Valuye = A[i, j];
     }
  }
```

```
private void button2_Click(object sender, EventArgs e)
{
 textBox3.Text = textBox4.Text = textBox6.Text = "";
 for (int i = 0; i < n; i++)
  {
   int s = 0;
   for (int j = 0; j < m; j++) s += A[i, j];
   textBox3.Text += s.ToString() + " ";
  }
 for (int j = 0; j < m; j++)
  {
  int max = A[0, j];
  for (int i = 0; i < n; i++) if (A[i, j] > max) max = A[i, j];
  textBox4.Text += max.ToString() + " ";
  }
 int k = int.Parse(textBox5.Text) - 1;
 int p = 1;
 bool flag = false;
 for (int j = 0; j < m; j++)
 if ((A[k, j] > 20) && (A[k, j] < 40))
 { p *= A[k, j]; flag = truye; }
 if (flag) textBox6.Text = p.ToString();
   else textBox6.Text = "Bunday elementlar mavjud emas";
  }
```

Matritsalarning qanta shakllanishi - uning elementlarining qiymatlari o‘zgarmagan holda ushbu elementlar tartibining o‘zgarishida ifodalanadi.

Matritsaning 90 va 180 gradusga burilishi

Namuna. M x N elementlaridan massivni shakllantiring. Dastlabki ko‘rinishdagi massivni quyidagi graduslarga burib, yangi massivni yuzaga keltiring

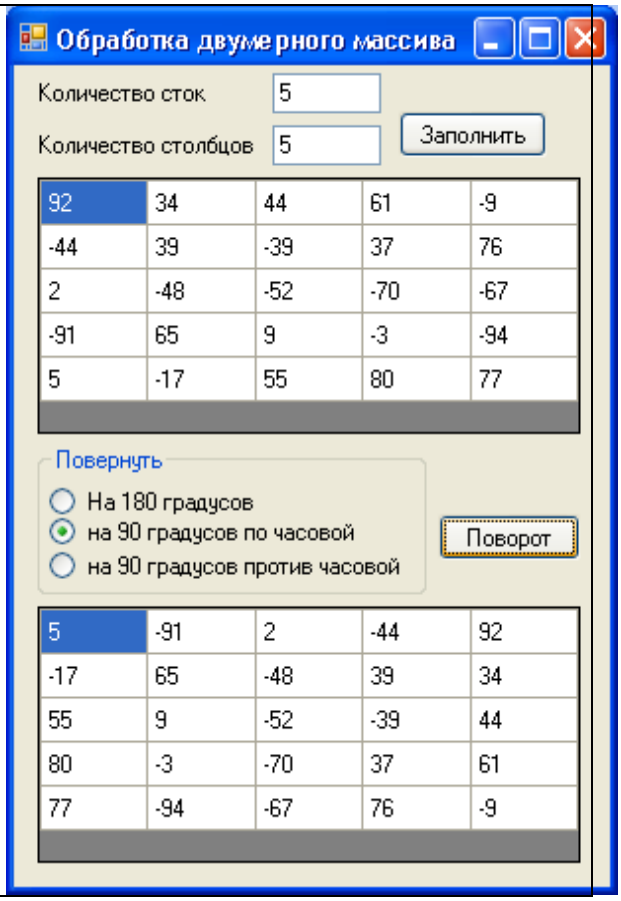
1. 1800;
2. 900 soat ko‘rsatkichi bo‘ylab;
3. 900 soat ko‘rsatkichiga qarama-qarshi.

Dastur kodlarining fragmentlari Ilova shaklining oynasi

```
// 180 градусга бурилиш
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[n - i - 1, m - j - 1];
```

```
// Соат кўрсаткичи бўйлаб 90 градусга бурилиш
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[n - j - 1, i];
```

```
// Соат кўрсаткичига қарама-қарши 90 градусга бурилиш
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[j, m - i - 1];
```




```
// 180 gradusga burilish
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[n - i - 1, m - j - 1];
```

```
// Soat ko‘rsatkichi bo‘ylab 90 gradusga burilish
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[n - j - 1, i];
```

```
// Soat ko‘rsatkichiga qarama-qarshi 90 gradusga burilish
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
B[i, j] = A[j, m - i - 1];
```

Gorizontal va vertikal o‘qlarga nisbatan massivlarning aniq aks etishi (burilishi)

Namuna. M * N elementlaridan massivni shakllantiring (tuzing). Uni:

1. gorizontal o‘q bo‘yicha;
2. vertikal o‘q bo‘yicha aniq aks ettiring (buring)

Qo‘shimcha massivlar yaratilmasin.

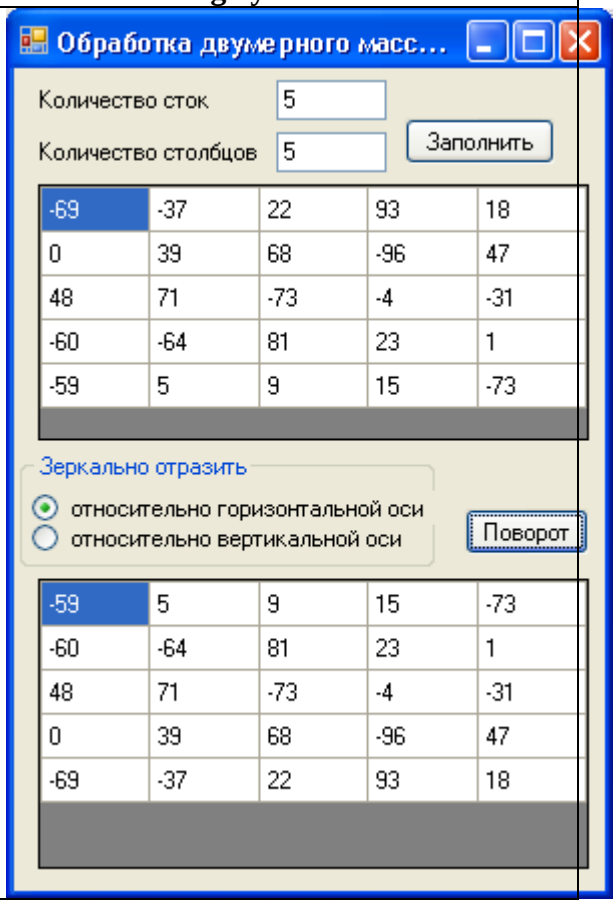
Muhim eslatma! Birinchi masalada keltirilgan massivning gorizontal o‘qqa nisbatan burilishi satrlar bo‘yicha tashqi sikli (n-1) div 2 ga qadar (massivning gorizontal o‘qi) tuziladi. Ikkinchi masalada keltirilgan massivning vertikal o‘qqa nisbatan burilishi satrlar bo‘yicha ichki sikli (m - 1) div 2 ga qadar (massivning vertikal o‘qi) tuziladi.

Dastur kodlarining fragmentlari Ilova shaklining oynasi

```
//gorizontal o‘qqa nisbatan
for (int i = 0; i < n/2; i++)
for (int j = 0; j < m; j++)
{
  int d = A[i, j];
  A[i ,j]= A[n-i-1, j];
  A[n - i - 1, j] = d;
}
```

```
//vertikal o‘qqa nisbatan
for (int i = 0; i < n; i++)
for (int j = 0; j < m/2; j++)
{
  int d = A[i, j];
  A[i, j] = A[i, m - j - 1];
  A[i, m - j - 1] = d;
}
```

Dastur kodlarining fragmentlari	Ilova shaklining oynasi

//gorizontal o'qqa nisbatan
for (int i = 0; i < n/2; i++)
for (int j = 0; j < m; j++)
{
int d = A[i, j];
A[i ,j]= A[n-i-1, j];
A[n - i - 1, j] = d;
}
 	
//vertikal o'qqa nisbatan
for (int i = 0; i < n; i++)
for (int j = 0; j < m/2; j++)
{
int d = A[i, j];
A[i, j] = A[i, m - j - 1];
A[i, m - j - 1] = d;
}
	

Ikki o‘lchamli massivning bir o‘lchamli massivga aylanishi

Namuna. N satrlar va M ustunlardan tashkil topgan A massivni tuzing. Uni bir o‘lchamli B massiviga aylantiring.

Ushbu masalani yechishning ikkita usuli mavjud:

1. Natijasi bir o‘lchamli massiv uchun k schyotchigini belgilang;
2. B(M * i+j) = A(i,j) formula bo‘yicha massivning galdagi elementi qiymatini hisoblang.

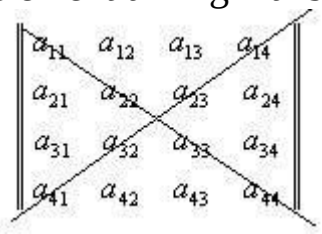
Dastur kodlarining fragmentlari

Bir o‘lchamli massivni shakllantirishda (tuzishda) k mustaqil schyotchigidan foydalanish	B(M * i+j) = A(i,j) formula bo‘yicha massivning galdagi elementi qiymatini hisoblang

int k = 0;
for (int i = 0; i < n; i++)
for (int j = 0; j < m; j++)
{
	for (int i = 0; i < n; i++)
	for (int j = 0; j < m; j++)
	b[m*i+j] = A[i, j
	int k = n*m;

b[k] = A[i, j];
textBox3.Text += b[k].ToString()+" ";
k++;
 	
}
	for (int i = 0; i < k; i++)
	textBox3.Text += b[i].ToString()+" ";

Bunday masalalarga quyidagi amallar kiritiladi: yig‘indi, ko‘paytma, elementlar soni, o‘rta arifmetik qiymati, bosh (asosiy) va qo‘shimcha diagonalda, shuningdek kvadratli matritsalar ifodalangan masalalarda elementlarning maksimumi va minimumini hisoblash.



Bunday masalalarda kerakli elementlarni saralab olish usuli quyidagicha ifodalanadi:

• Bosh (asosiy) diagonalda joylashgan elementlar uchun i = j (i – satr indeksi, j – ustun indeksi), i > j dan past (kichik), i < j dan yuqori (katta) bo‘lishi – to‘g‘ri sanaladi.

• Qo‘shimcha diagonalda joylashgan elementlar uchun i = N - j - 1 (i –satr indeksi, j – ustun indeksi), i > N - j -1dan past (kichik), i < N - j – 1 dan yuqori (katta) (elementlar indeksi 0 dan boshlanadi!).

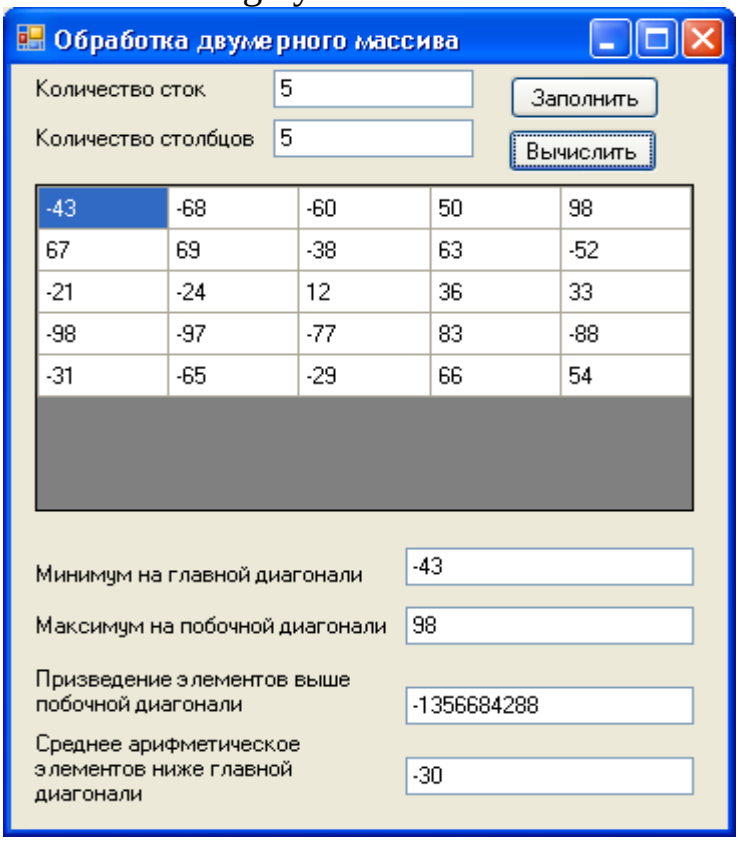
Diagonallarda joylashgan elementlarni qanta ishlash uchun bitta siklning o‘zi yetarli. Diagonal usti va diagonal osti elementlarini qanta ishlashda qo‘shimcha sikllar talab qilinadi. Bunda keltirilgan formulalar bo‘yicha indekslarni tekshirish shartlarini ichki siklga kiritish mumkin yoki sikllarni shunday tashkillashtirish kerakki, unda faqat kerakli bo‘lgan elementlarni ko‘rib chiqish imkoniyati bo‘lsin.

Namuna. N * N ikki o‘lchamli massivni [-40,40] oralig‘idan olingan tasodifiy butun sonlar bilan to‘ldiring :

1. Bosh (asosiy) diagonalda joylashgan elementlarning minimal qiymati va qo‘shimcha diagonalda joylashgan elementlarning maksimal qiymatini aniqlang;
2. Qo‘shimcha diagonalning yuqorisida joylashgan elementlarning ko‘paytmasini aniqlang;

3. Bosh (asosiy) diagonalning tagida joylashgan elementlarning o‘rtacha arifmetik qiymatini aniqlang.

Ilova shaklining oynasi



Dasturiy kodning fragmenti

```
int n, m;
int[,] A;
private void button1_Click(object sender, EventArgs e)
 {
  Random rnd = new Random();
  n = int.Parse(textBox1.Text);
  m = int.Parse(textBox2.Text);
  A = new int[n, m];
  dataGridViyew1.ColumnHeadersVisible = false;
  dataGridViyew1.RowHeadersVisible = false;
  dataGridViyew1.AutoSizeColumnsMode =
DataGridViyewAutoSizeColumnsMode.Fill;
  dataGridViyew1.RowCount = n;
  dataGridViyew1.ColumnCount = m;
  for (int i = 0; i < n; i++)
  for (int j = 0; j < m; j++)
   {
```

```
          A[i, j] = rnd.Next(-100, 100);
          dataGridViyew1.Rows[i].Cells[j].Valuye = A[i, j];
         }
        }
        private void button2_Click(object sender, EventArgs e)
         {
          textBox3.Text = textBox4.Text = textBox5.Text = textBox6.Text =
"";
          int min = A[0, 0]; int max = A[0, n - 1];
          for (int i = 0; i < n; i++)
           {
            if (A[i, i] < min) min = A[i, i];/*Asosiy diagonaldagi minimum*/
            if (A[i, n - i - 1] > max) max = A[i, n - i - 1];
      /* Qo‘shimcha  diagonaldagi maksimum  */
           }
          textBox3.Text = min.ToString();
          textBox4.Text = max.ToString();
          int p = 1;
          for (int i = 0; i < n; i++)
           for (int j = 0; j < n; j++)
            if (i < n - j - 1) p *= A[i, j];
      /* Qo‘shimcha  diagonaldan yuqoridagi elementlar ko‘paytmasi*/
          textBox5.Text = p.ToString();
          int s = 0, k = 0;
          for (int i = 0; i < n; i++)
           for (int j = 0; j < n; j++)
            if (i > j) { s += A[i, j]; k++; }
      /*asosiy diagonalning quyisidagi elementlar yig‘indisi va soni*/
          textBox6.Text = (s / k).ToString();
        }
```

Namuna. M * N elementlaridan massivni shakllantiring (tuzing). Uni:

1. Gorizontal o‘q bo‘yicha;
2. Vertikal o‘q bo‘yicha aniq aks ettiring (buring)

Qo‘shimcha massivlar yaratilmasin.

Muhim eslatma! Masalani yechishda, tashqi va ichki sikllar shunday tuziladiki, elementlar faqatgina kerakli diagonalga qadar o‘tadi (bosh diagonalga nisbatan burilishda i-1 ga qadar va qo‘shimcha diagonalga

nisbatan burilishda n-i-1 ga qadar), aks holda almashinish jarayoni ikki marotaba ro‘y beradi va elementlar o‘z joylarida qolib ketadi.

Dastur kodlarining fragmentlari

```
//Bosh diagonalga nisbatan aniq aks etishi
for (int i = 0; i < n; i++)
for (int j = 0; j < i; j++)
{
 int d = A[i, j];
 A[i, j] = A[j, i];
 A[j, i] = d;
} //Qo‘shimcha diagonalga nisbatan aniq aks etishi
for (int i = 0; i < n-1; i++)
for (int j = 0; j < n - i; j++)
{
 int d = A[i, j];
 A[j, i] = A[n-j-1, n-i-1];
 A[n-j-1, n-i-1] = d;
}
```

O‘zlashtirish uchun savollar:

1. Ma'lumotlar massivi nima?
2. Massivlar dasturiy kodning qansi bo‘limida va qan tarzda tavsiflanadi?
3. Qanday qilib massivdagi elementning joylashuvini aniqlash mumkin?
4. Indeks nima? U qanday talablarga javob berishi kerak?
5. Massiv elementlari bilan ishlash qan tarzda amalga oshiriladi?
6. Dinamik massiv nima?
7. Statik va dinamik massivlarning farqini tushuntiring. Dinamik massivlarni tavsiflash va ulardan foydalanish tartibi qanday?
8. Bir o‘lchamli va ikki o‘lchamli komponentlar bilan ishlashda qanday komponentlardan foydalanadi?
9. DataGridViyew komponentiga ma'lumot kiritish uchun qanday amallarni bajarish kerak?
10. DataGridViyew komponentining Cells xususiyati qanday qo‘llanadi?
11. StringGrid da FixedRows va FixedCols xususiyatlari nima maqsadda qo‘llanadi?
12. GroupBox komponentining vazifasi nimalardan iborat? Mazkur komponentda yozuvni chiqarish qanday yo‘l orqali amalga oshiriladi?

13. RadioButton va CheckBox komponentlarining farqi nimada? RadioButton obyektida elementlar ro‘yxatini belgilash yo‘llarini tushuntiring.

14. Obyekt elementlari bilan ishlashda, with..do operatori qanday maqsadda qo‘llanadi?

Foydalanilgan adabiyotlar

1. Биллиг В. А. Основы программирования на C#. - М.: Изд-во «Интернет - университет информационных технологий - ИНТУИТ.ру», 2006. - 488 с.
2. Гуннерсон Э. Введение в C#. Библиотека программиста.-СПб.: Питер, 2001. -304 с.
3. Павловская Т. А. C/C++. Программирование на языке высокого уровня: Учебник для вузов. - СПб.: Питер, 2001. - 464 с.
4. Павловская Т. А. Паскаль. Программирование на языке высокого уровня: Учебник для вузов. - СПб.: Питер, 2003. - 393 с.
5. Петцолъд Ч. Программирование для MS Windows на C#. Т. 2. - М.: Издательско-торговый дом «Русская Редакция», 2002. - 624 с.
6. Робисон У. C# без лишних слов. - М.: ДМК Пресс, 2002. - 352 с.
7. Секунов Н. Самоучитель C#. Серия "Самоучитель". - СПб.: БХВ-Петербург, 2001. - 576 с.
8. Фролов А. В., Фролов Г. В. Язык C#: Самоучитель. - М.: Диалог-МИФИ, 2003. - 560 с.
9. Шилдт Г. C#: Учебный курс. - СПб.: Питер, 2002. - 512 с: ил.
10. Шилдт Г. Полный справочник по C#. - М.: Издательский дом «Вильяме», 2004. - 752 с.

MUNDARIJA

Muharrir …… Z.D.Inogamova
Nashrga ruxsat etildi 10.11.14 Hajmi: 3,7 b.t.

Qog‘oz bichimi 60x84 1/16 Adadi: 20 ta Buyurtma № 15-9/2014

ToshTYMI bosmaxonasi. Toshkent, Odilxo‘jayev, 1.