Z.T.Məhərrəmov, D.R.Nəsibov

VİSUAL C# DİLİNDƏ LABORATORİYA İŞLƏRİ

(Windows əlavələri)

Z.T. Məhərrəmov, D.R. Nəsibov

VISUAL C# DİLİNDƏ LABORATORİYA İŞLƏRİ (Windows əlavələri)

Ali məktəb tələbələri üçün tədris-metodiki vəsait

BAKI – 2020

Elmi redaktor:
Azərbaycan Texniki Universitetinin
“İnformasiya texnologiyaları və proqramlaşdırma”
kafedrasının müdiri, tex.f.d., dosent **Z.Ə. Cəfərov**

Rəy verənlər:

Azərbaycan Texniki Universitetinin
“Mühəndis riyaziyyatı”
kafedrasının dosenti, f.-r.e.n, dosent **H.P.Vəliyev**

Azərbaycan Texniki Universitetinin
“İnformasiya texnologiyaları və proqramlaşdırma”
kafedrasının dosenti, f.-r.e.n, dosent **T.İ.Qəhrəmanova**

Visual C# dilində laboratoriya işləri (Windows əlavələri): Ali məktəb tələbələri üçün tədris-metodiki vəsait/ Məhərrəmov Z.T., Nəsibov D.R. – Bakı: ADPU-nəşriyyat, 2020. –149 s.

Ali məktəb tələbələri üçün nəzərdə tutulmuş bu tədris-metodiki vəsait Visual C# dilinə aid laboratoriya işlərindən ibarətdir. Əhatə edilmiş materiallar tələbələrə tədris prosesində Visual C# dilinin Windows əlavələrinin yaradılması üçün zəruri olan əsas konstruksiyaların öyrənilməsinə imkan verir.

Bütün laboratoriya işləri onların yerinə yetirilməsi üçün minimal nəzəri materiallarla və çoxlu praktiki misallarla müşayiət olunmuşdur.

Mündəricat

GİRİŞ

Bu tədris-metodiki vəsait C# dilində Konsol əlavələrinin yaradılması üçün nəzərdə tutulmuş və 2020-ci ildə nəşr edilmiş laboratoriya işlərinin davamıdır (bax: Məhərrəmov Z.T., Abidov Ç.C., Həşimov R.H., Məmmədzadə N.F. C# dilində laboratoriya işləri (Konsol əlavələri). Bakı: ADPU-nəşriyyat, 2020. –144 s.). Bu vəsaitdə də 10 laboratoriya işi təqdim edilmişdir. Həmin işlərdə Windows-əlavələrinin yaradılması xüsusiyyətləri şərh edilmişdir. Bu işləri yerinə yetirən tələbə gələcəkdə C# dilində kifayət qədər mürəkkəb Windows-əlavələrinin yaradılması üçün zəruri bilik və vərdişlər əldə edə bilər.

Vəsait AzTU-nun bütün ixtisaslarında tədris edilən "İnformatika" fənni, "Kompüter elmləri" ixtisasında tədris edilən "Alqoritmik dillər", "Müasir proqramlaşdırma dilləri", Proqramlaşdırmanın əsasları" fənlərinin və "İnformasiya texnologiyaları" və "Sistem mühəndisliyi" ixtisaslarında tədris edilən "Proqramlaşdırmaya giriş", "Kompüter proqramlaşdırmasının elementləri", "Obyektyönlü proqramlaşdırma" və s. fənlərinin laboratoriya məşğələlərinin yerinə yetirilməsində istifadə edilmək üçün nəzərdə tutulmuşdur.

Bütün laboratoriya işləri minimal zəruri nəzəri materiallarla və həll edilmiş çoxlu misallarla müşayiət edilmişdir. Misallar Microsoft Visual Studio Community 2019 mühitində yerinə yetirilmişdir.

Kitab haqqında rəy və təkliflərinizi, iradlarınızı bu ünvana göndərə bilərsiniz:. Bakı ş. H.Cavid prospekti. AzTU, "İnformasiya texnologiyaları və proqramlaşdırma" kafedrası.

LABORATORİYA İŞİ № 1.
MICROSOFT VISUAL STUDIO COMMUNITY İNTEQRALLAŞDIRILMIŞ İŞ MÜHİTİ

İşin məqsədi C# proqramlaşdırma dilində Windows əlavələrini layihələndirmək məqsədi ilə MICROSOFT VISUAL STUDIO COMMUNITY inteqrallaşdırılmış iş mühitinin öyrənilməsindən ibarətdir.

Nəzəri məlumat
Əsas menyu və alətlər paneli

Visual Studio mühitinin əsas menyusunun strukturunu aşağıdakı menyu kateqoriyalarına bölmək olar:

Əsas menyu:

- File (Файл);
- Edit (Правка);
- View (Вид);
- Command (Команда);
- Validation (Тест);
- Analyze (Анализ);
- Tools (Сервис);
- Window (Окно);
- Help (Справка);

Layihədən asılı menyu:

- Project (Проект);
- Assembly (Сборка);
- Debug (Отладка);

Konkret sənədin menyusu:

- Format (Формат);
- Table (Таблица).

Proqram işə salındıqda ilkin olaraq menyuda göstərilən elementlərdən yalnız bəziləri olur. Qalan elementlər yeni pəncərələr açıldıqda menyuya əlavə edilir.

Əsas menyunun Project (Проект) bəndi layihəyə forma, proqram modulu, sinif kimi elementləri əlavə etməyə imkan verən əmrlərdən və habelə qoşulan kitabxanalara istinad əlavə etməyə imkan verən əmrlərdən ibarətdir. Project (Проект) bəndinin ən sonuncu əmri Properties: (Свойства:) əmridir ki, onun köməyi ilə layihənin xassələri pəncərəsi açılır.

Əsas menyunun Assembly (Сборка) bəndi həlli və ya layihəni yığmağa imkan verən əmrlərdən ibarətdir.

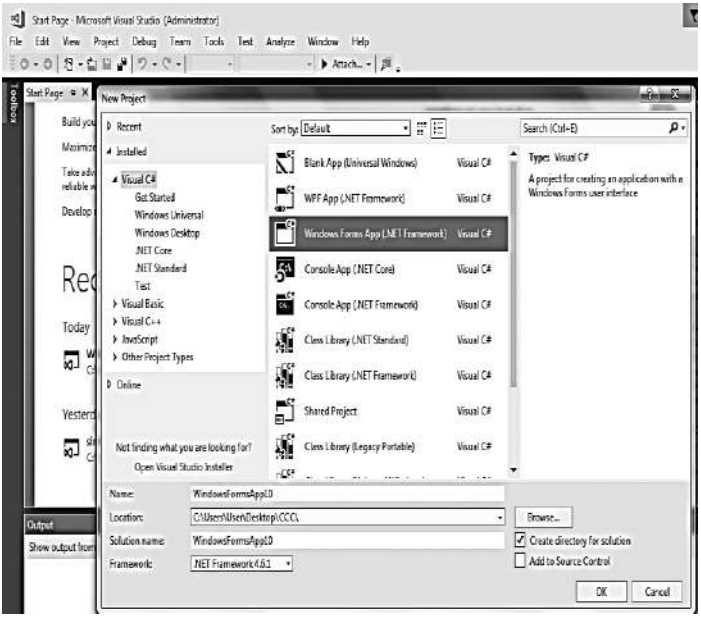
Əsas menyunun Debug (Отладка) bəndi əlavəni kompilyasiya etməyə və işə buraxmağa imkan verən əmrlərdən ibarətdir. Bu menyunun əmrlərinin köməyi ilə yerinə yetirilmək üçün əlavəni işə buraxmaq, proqramın dayanma nöqtəsini təyin etmək, əlavəni addım-addım icra etmək, proqramın yoxlanması üçün xüsusi pəncərəni açmaq olar.

Yeni layihə yaratmaq üçün File/New/Project… (Файл/Создать/Проект…) (və ya Start Page pəncərəsindən Create New Project... (Начальная страница/Создать проект…)) əmrini icra etmək lazımdır. Bu zaman qarşınızda New Project (Создать проект) pəncərəsi açılacaqdır (şəkil 1.1). Burada, sol tərəfdə Visual C#, sağ tərəfdən isə layihənin tipini seçmək lazımdır. Biz, bu laboratoriya işindən başlayaraq gələcəkdə yalnız Windows əlavələri yaradacağıq, ona görə də bu pəncərədə Windows Forms App (.NET Framework) (Приложение Windows Forms(.NET Framework)) seçirik .

Yeni layihənin yaradılması pəncərəsinin aşağısında iki daxiletmə sahəsi var:

- Name: (Имя:) - burada gələcək layihənin adı göstərilir;
- Location: (Расположение:) – burada layihənin yerləşəcəyi qovluğun adı göstərilir.

Layihə faylı `.csproj` genişlənməsinə malikdir. Burada layihənin fayllarına daxil olan bütün kökləmələr və onların təsvirləri yerləşir.

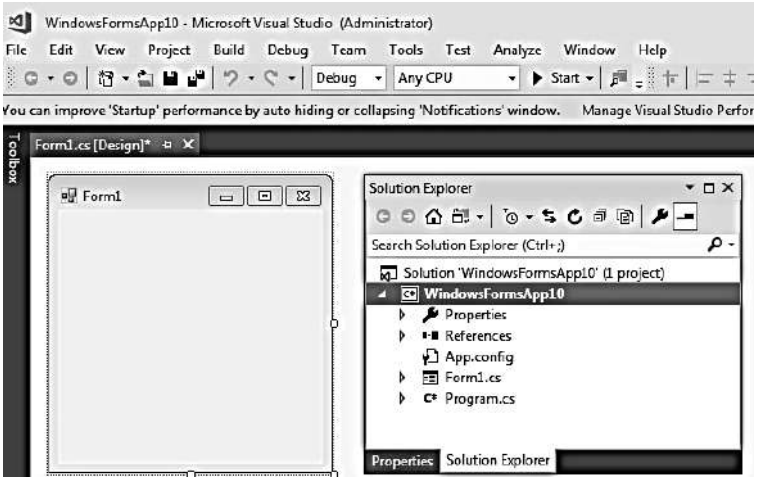


Şəkil 1.1. Yeni layihənin yaradılması pəncərəsi

Həllər müşahidəçisi

İndi isə boş C# layihəsi yaradaq. İlkin açılmış pəncərədə `Visual C#` və `Windows Forms App (.NET Framework)` (`Приложение Windows Forms(.NET Framework)`) seçib, layihəyə ad verib onun yerləşəcəyi qovluğu göstərək (şəkil 1.1). Peyda olan (ekranda görünən, zühur edən) pəncərədə, menyu sətrindən aşağıda, `Form1.cs[Design]` (`Form1.cs [Конструктор]`) adlı pəncərə, iş mühitinin oblastında isə `Form1` adlı "boş"

forma görünəcəkdir (şəkil 1.2). Bu forma üzərində gələcək əlavə layihələndiriləcəkdir.



Şəkil 1.2. Layihənin yaradılması pəncərəsi

Bu formaya uyğun kodun məzmununa baxaq. Bunun üçün `View` (`Вид`) menyusuna daxil olaraq `Solution Explorer` (`Обозреватель решений`) pəncərəsini ekranda təsvir edək. Bu pəncərəni işçi oblastın istənilən hissəsində yerləşdirmək olar.

Bu pəncərədə bütün obyektlər ağac şəklində təsvir edilmişdir. Ağacın başında həllin adı dayanır. Çox zaman o layihənin adı ilə eyni olur və qalın şriftlə yazılır. Həllin adını dəyişmək üçün ağacda onun adının üzərində mausun sağ düyməsini basmaqla kontekst menyudan `Rename` (`Переименование`) əmrini icra etmək lazımdır.

Həllə layihə əlavə etmək üçün ağacda həllin adı üzərində mausun sağ düyməsini basmaqla kontekst menyudan `Add` (`Добавить`) seçmək lazımdır. Bu menyudan aşağıdakı əməliyyatları icra etmək olar:

- New Item… (Создать элемент...) – açılan pəncərədən seçməklə yeni elementi layihəyə əlavə edir, məsələn, Форма Windows Forms seçilərsə, layihəyə ikinci forma əlavə ediləcəkdir;
- Existind Item… (Существующий элемент…) – layihəyə hazır layihə əlavə ediləcəkdir;
- Windows Form... (Форма Windows...) - layihəyə yeni forma əlavə ediləcəkdir;
- Component... (Компонент…)- layihəyə yeni komponent əlavə ediləcəkdir;
- Class... (Класс…) - layihəyə yeni sinif əlavə ediləcəkdir və s.

Həllə daxil olan bütün layihələr üçün icra edilən fayl almaq lazım gələrsə, onda onun adının üzərində mausun sağ düyməsini basmaqla kontekst menyudan Build Solution (Собрать) əmrini icra etmək lazımdır. Bu zaman yalnız dəyişdirilmiş fayllar kompilyasiya ediləcəkdir. Bütün faylları kompilyasiya etmək üçün Rebuild Solution (Перестроить) əmri icra edilməlidir. Həmin bu əməliyyatları əsas menyunun Build (Сборка) bəndində yerləşən Build Solution (Собрать решение) və Rebuild Solution (Пересобрать решение) əmrləri ilə də icra etmək olar.

Layihənin adı üzərində kontekst menyudan Rename (Переименовать) и Remove (Удалить) əmrlərini də icra etmək olar ki, bununla da, müvafiq olaraq, layihənin adı dəyişdiriləcək və ya pozulacaqdır.

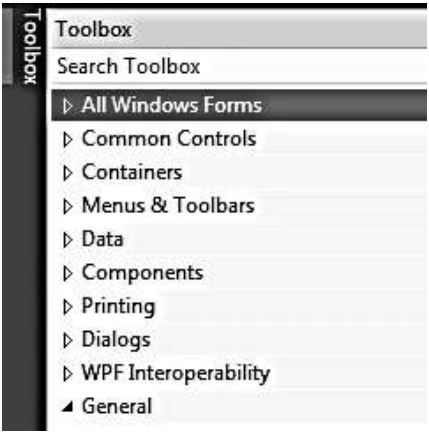
Layihəni dərhal yerinə yetirmək üçün *F5* klavişini basmaq və ya Debug/Start Debugging (Отладка/Начать отладку) əmrini icra etmək lazımdır. Layihənin yerinə yetirilməsinin digər variantı *Ctrl+F5* klavişlər kombinasiyası və ya Debug/Start Without Debugging

(Отладка/Запуск без отладки) əmri ilə icra oluna bilər.

Hətta ən sadə layihələr bir neçə fayllardan ibarət olur, ona görə də hər layihəni ayrıca bir qovluqda yadda saxlamaq lazımdır. Visual Studio-da açılacaq əsas fayl genişlənməsi `.csproj` olan fayldır. Həmin faylın adı layihəyə verdiyiniz adla eyni olur. Bu faylı Notepad (Блокнот) redaktoru ilə də açmaq olar. Visual Studio mühitinin File/Open/Project/Solution… (Файл/Открыть/Решение или проект…) əmri ilə faylı açdıqda iş mühiti bütün zəruri parametrləri ilə birlikdə bu faylı açır.

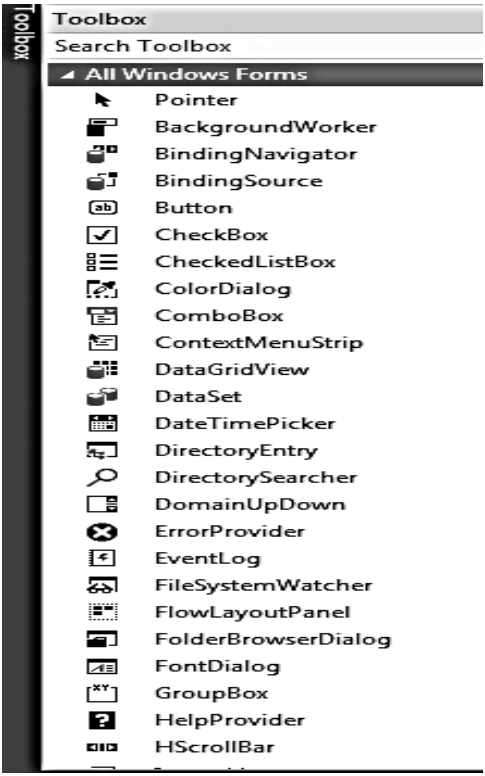
Elementlər paneli

Bu paneldə bölmələr üzrə elementlər **(**komponentlər) yerləşir, bu elementlər formada yerləşdirilməklə layihə konstruksiya ediləcəkdir. Bu panel View/Toolbox (Вид/Панель элементов) əmri ilə və ya *Ctrl+Alt+X* klavişlər kombinasiyası ilə ekranda təsvir etdirilir (şəkil 1.3).



Şəkil 1.3. Toolbox elementlər paneli

Layihənin tipindən asılı olaraq paneldə komponentlərin sayı və görünüşü fərqlənə bilər. Sadə `Application Windows Forms` (`Приложение Windows Forms`) əlavəsi üçün əsas komponentlər `All Windows Forms` (`Все формы Windows Forms`) bölməsində yerləşir (şəkil 1.4).



Şəkil 1.4. Elementlər panelinin `All Windows Forms` bölməsi

Kodlaşdırma prosesində biz elementlərə onların unikal adları vasitəsi ilə müraciət edəcəyik. Onların adlarında heç bir təhrifə yol vermək olmaz. Diqqət yetirin ki, elementlərin adları həmişə kiçik hərflə başlayır, məsələn: `textBox`,

pictureBox və s. Elementlərin bu adlarını Siz öz istədiyiniz adla dəyişdirə bilərsiniz.

Elementlərin forma üzərində yerləşdirilməsi üsulları

Elementləri forma üzərində bir neçə üsulla yerləşdirmək olar:

l. Mausun sol düyməsini basılı saxlayaraq komponenti formaya dartıb aparmaqla. Bu halda komponent onu atdığınız nöqtədə yaradılacaq, ölçüləri isə susmaya görə təyin ediləcək.

2. Komponentin üzərində mausun düyməsini iki dəfə basmaqla. Bu halda komponent ixtiyarı nöqtədə yaradılacaq, ölçüləri isə susmaya görə təyin ediləcək.

3. Lazım olan komponent üzərində mausun düyməsini basaraq onu seçmək və formada mausun düyməsini basaraq komponenti yerləşdirməklə. Bu halda komponent mausla göstərilən nöqtədə yaradılacaq, ölçüləri isə susmaya görə təyin ediləcək.

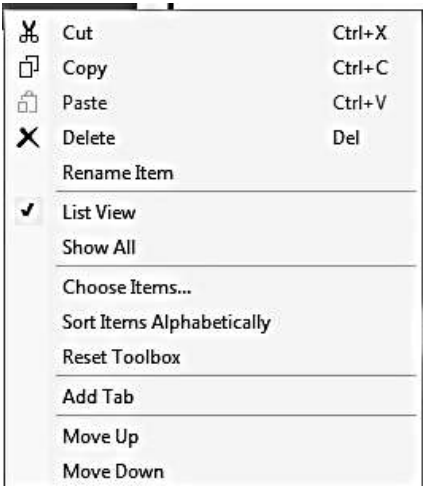
4. Lazım olan komponent üzərində mausun düyməsini basaraq onu seçmək lazımdır. Formada mausun sol düyməsini basmaq və komponenti yerləşdirmək üçün formada lazım olan ölçüləri əldə edənədək kursoru dartmaq. Bu halda komponent mausla göstərilən nöqtədə yaradılacaq, ölçüləri isə formada çəkilmiş düzbucaqlıya uyğun olaraq təyin ediləcək.

Formadan elementi pozmaq üçün onu seçib *Delete* klavişini basmaq lazımdır.

Elementlər panelini sazlamaq da olar. Bunun üçün panelin boş sahəsində və ya hər hansı element üzərində mausun sağ düyməsini basaraq açılan kontekst menyu əmrlərini icra etmək lazımdır. Bu əmrlər şəkil 1.5-də göstərilmişdir.

Burada daha çox maraq kəsb edən Choose Items... (Выбрать элементы...) əmridir. Onu icra etdikdə Choose Toolbox Items (Выбор элементов панели элементов) pəncərəsi ekranda təsvir olunacaqdır

ki, bu pəncərədə elementləri yaratmaq və ya pozmaq olar (şəkil 1.6). Yeni element əlavə etmək üçün siyahıda onun qarşısında bayraq işarəsi qoymaq kifayətdir. Həm .NET-komponentlərini, həm də COM-komponentlərini əlavə etmək olar.

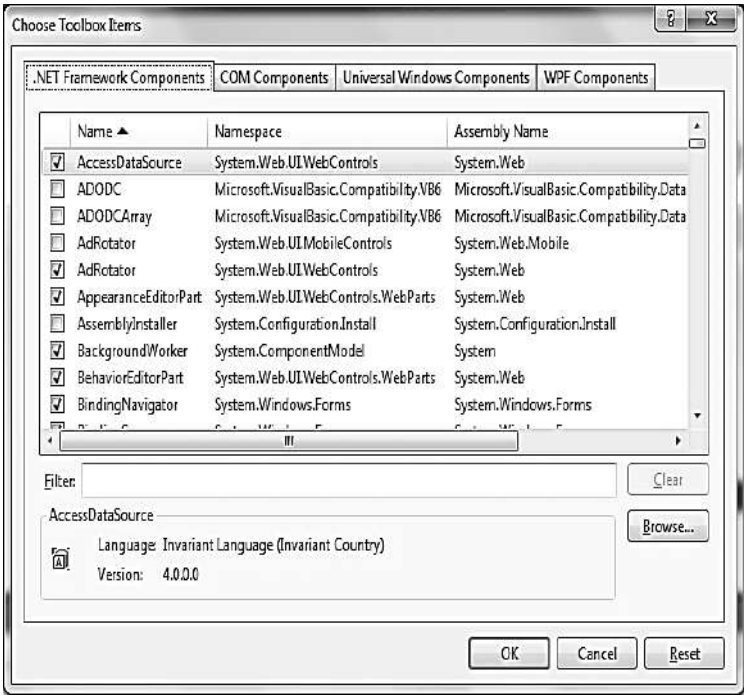


Şəkil 1.5. Elementlər panelini sazlamaq üçün kontekst menyu əmrləri

Xassələr pəncərəsi

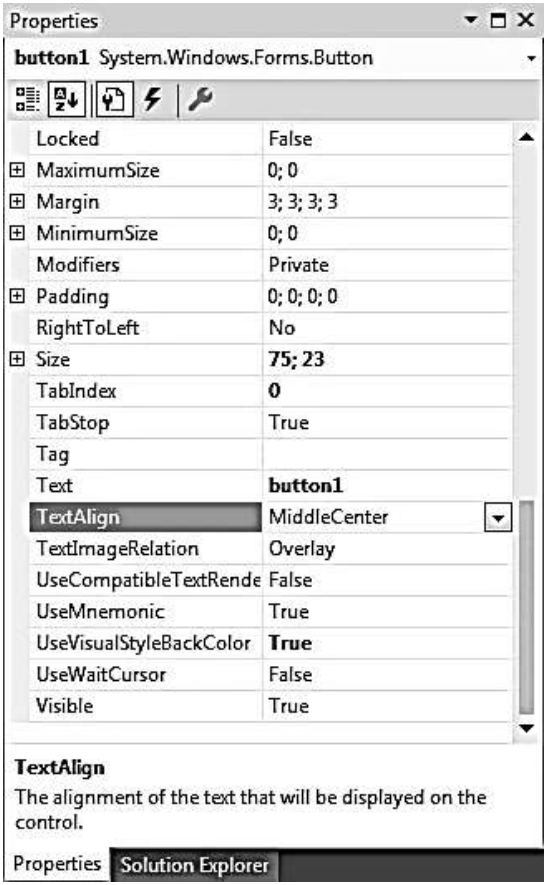
`Properties (Окно свойств)` – Xassələr pəncərəsi forma və onun üzərində yerləşdirilmiş komponentlərin xassə və hadisələrini müəyyən etmək üçündür (şəkil 1.7). Onu ekranda təsvir etmək üçün *F4* klavişini basmaq və ya `View/Properties Window (Вид/Окно свойств)` əmrini icra etmək lazımdır. Bu pəncərə `Solution Explorer` pəncərəsi ilə birlikdə bir pəncərənin müxtəlif səhifələrində yerləşir. Bu pəncərədə formanın və ya onun üzərində seçilmiş hər hansı komponentin bütün xassələri yerləşir və bu xassələrin köməyi ilə komponentin rəngi, ölçüləri, koordinatları, şrifti və s. dəyişdirilə bilər. Pəncərədə

hər hansı bir xassəni seçdikdə səhifənin sonunda həmin xassənin qısa təyinatı təsvir edilir.



Şəkil 1.6. `Choose Toolbox Items` pəncərəsi

Forma üzərində komponent seçildikdə və o, markerlərlə əhatə olunduqda, xassələr pəncərəsinin birinci sətrində həmin komponentin adı (məsələn, `button1 System.Windows.Forms.Button`), növbəti sətirlərdə isə bu komponentə xas olan xassələr təsvir olunur. Layihəçi həmin xassələrin qarşısında onlara qiymət verir və ya bu qiymətləri təklif olunan variantlardan seçir. Formanın özünün xassələri də analoji qaydada müəyyənləşdirilir. Formanı seçmək üçün onun komponentlər olmayan boş sahəsində mausun sol düyməsini basmaq kifayətdir. Bu və ya digər komponenti xassələr



Şəkil 1.7. Xassələr pəncərəsi

pəncərəsinin birinci sətrində yerləşən açılan siyahıdan da seçmək və onun xassələrinə müraciət etmək olar. Belə seçmə xüsusən o vaxt zəruri olur ki, bir komponent o biriləri tərəfindən tam örtülmüş vəziyyətdə olur və görünmür.

Xassələr pəncərəsi əsas iki səhifədən ibarətdir:

- `Properties (Свойства-`*Xassələr*`);`
- `Events (События-`*Hadisələr*`).`

`Properties (Свойства)` səhifəsində forma üzərində seçilmiş komponentin *xassələri* haqqında informasiya təsvir

olunur və, qeyd etdiyimiz kimi, layihələndirmə zamanı bir sıra xassələri çox asanlıqla dəyişdirə bilərik.

`Events (События)` səhifəsi göstərilən *hadisə* baş verdikdə komponentin yerinə yetirəcəyi metodu müəyyən edir. Əgər bu və ya digər hadisə üçün metod müəyyənləşdirilmişdirsə, onda layihənin yerinə yetirilməsi prosesində bu hadisə baş verdikdə həmin metod avtomatik olaraq icra olunacaqdır. Belə hadisələrə misal olaraq mausun düyməsinin bir və ya iki dəfə basılması zamanı baş verəcək əməliyyatları (formanın bağlanması, sərlövhənin dəyişdirilməsi və s.) göstərmək olar.

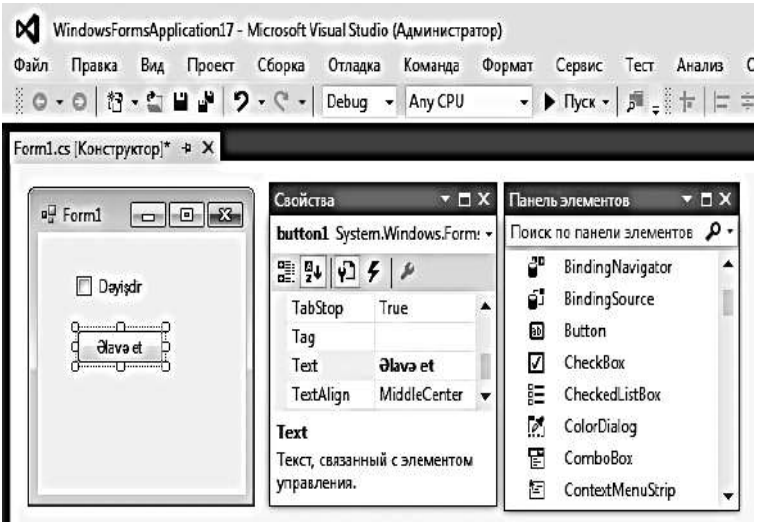
Komponentin və ya formanın xassələrinə müraciət etmək üçün xassələr pəncərəsinin `Properties (Свойства)` səhifəsinə keçmək lazımdır. Bunun üçün bu pəncərədə yerləşən  düyməni basmaq lazımdır (mausun göstəricisini bu düymə üzərində ləngitdikdə onun adı peyda olacaqdır.

Xassə – əlavə yerinə yetirildikdə komponentin əks olunması və fəaliyyətini müəyyənləşdirən atributlardan ibarətdir. Xassələr pəncərəsində komponentin xassəsini dəyişdikdə, bu dəyişiklik komponentin özündə əks olunur, yəni elə layihələndirmə prosesində dəyişikliklərin nəticəsi artıq görünür. Məsələn, düymənin `Text` (Mətn) xassəsinə hər hansı bir ad verdikdə (şəkil 1.8) həmin ad düymənin üzərində əks olunur. Xassəyə qiymət verdikdən və ya onu seçdikdən sonra, ya *Enter* klavişini basmaq, ya da sadəcə digər xassəyə keçmək lazımdır. Dəyişikliyi ləğv etmək üçün *Esc* klavişini basmaq kifayətdir.

Komponentlərin əksər xassələrinə, məsələn, `BackColor` (Rəng), `Text` (Mətn) və s. susmaya görə qiymətlər əvvəlcədən müəyyən edilmişdir (biz onları da dəyişdirə bilərik).

Komponentə onun `Name`-*Ad* xassəsi ilə müraciət olunur. Forma üzərində komponentin yerini və ya ölçülərini

dəyişdirdikdə bu parametrlərlə əlaqədar xassələrin (`X, Y, Width, Height`) də qiymətləri avtomatik olaraq dəyişir.



Şəkil 1.8. Komponentin xassəsinin dəyişdirilməsi

Komponent və ya forma ilə əlaqədar hər hansı bir hadisəyə uyğun metod yaratmaq üçün xassələr pəncərəsinin `Events` (`События`) səhifəsinə keçmək lazımdır. Bunun üçün bu pəncərədə yerləşən  düyməni basmaq lazımdır (mausun göstəricisini bu düymə üzərində ləngitdikdə onun adı peyda olacaqdır). `Events` səhifəsində komponentlərlə əlaqədar *hadisə emaledicilərini –metodları* yaratmaq üçün hadisələrin siyahısı yerləşir. Belə hadisələrə misal olaraq `Click, KeyUp, KeyDown, Keypress,Move, MouseClick` və s. göstərmək olar.

Xassələr pəncərəsində xassə və hadisələrin rahat təşkili üçün daha iki düymə vardır:

- `Categorized` (`По категориям`);

- Alphabetic (В алфавитном порядке).

 Categorized (По категориям) düyməsi (mausun göstəricisini bu düymə üzərində ləngitdikdə onun adı peyda olacaqdır) xassə və hadisələri müxtəlif məntiqi qruplara bölür. Əgər Siz obyektin bütün xassə və hadisələrini bilmirsinizsə və onları tiplərinə görə qruplaşdırmaq istəyirsinizsə, onda bu düyməni basın.

 Alphabetic (В алфавитном порядке) düyməsi (mausun göstəricisini bu düymə üzərində ləngitdikdə onun adı peyda olacaqdır) xassə və hadisələri əlifba sırası ilə düzür və onları bir neçə kateqoriyalarda birləşdirir.

Yoxlama sualları

1. Visual Studio 2017 (2019) mühitinin hansı menyuları vardir?
2. Visual Studio ilə yeni layihə necə yaradılır?
3. Layihə necə kompilyasiya edilir?
4. Genişlənmiş hissəsi `.csproj` olan fayl nə üçündür?
5. `Debug` ve `Release` qovluqları nə üçündür?
6. Həllər müşahidəçisi pəncərəsi ekrana necə çıxarılır və nə üçündür?
7. `Form1.cs` faylının məzmunu nədən ibarətdir?
8. Həllə yeni layihə, forma, elementlər və s. necə əlavə edilir?
9. Elementlər paneli pəncərəsi ekrana necə çıxarılır və nə üçündür?
10. Elementlər forma üzərinə necə yerləşdirilir?
11. Xassələr pəncərəsi ekrana necə çıxarılır və nə üçündür?
12. Xassələr pəncərəsinin hansı səhifələri var və onlar nə üçündür?

LABORATORİYA İŞİ № 2.
FORMA İLƏ İŞ

İşin məqsədi əlavənin əsasını təşkil edən formanın xassələrini (bu xassələrin əksəriyyəti digər elementlər üçün də doğrudur) öyrənmək və bu xassələrə qiymətlər vermək qaydalarını, həmçinin layihənin fayllarını öyrənməkdir.

Nəzəri məlumat
Formanın xassələri

Digər elementlər kimi, formanın da xassələri `Properties` (`Свойства`) pəncərəsinin köməyi ilə müəyyənləşdirilir. Bu xassələrin əksəriyyəti əsasən formanın görünüşünə təsir edir. Əsas xassələri nəzərdən keçirək:

- `Name` – formaya, daha dəqiq desək, `Form` sinfindən irsən alınmış sinfə ad verməkdir. Susmaya görə forma `Form1` adlanir;
- `BackColor` – formanın fonunun rəngini göstərir. Bu xassəni seçib onun qarşısındakı açılan siyahıdan təklif edilən rəngləri və ya rəng palitrasının siyahısından mövcud olan rəngləri seçə bilərik;
- `BackgroundImage` – formanın fonu üçün təsvir seçməyə imkan verir. Seçilmiş təsvirin ölçülərini idarə etmək üçün bu xassənin çoxlu alt xassələri mövcuddur. Həmin xassələrə müraciət etmək üçün `BackgroundImage` xassəsinin qarşısında, daxilində + simvolu yerləşən kiçik kvadrat üzərində mausun düyməsini basmaq lazımdır;
- `Cursor` – formada istifadə olunan kursorun tipini müəyyən edir. Forma ilə işlədikdə kursorun müxtəlif görünüşlü göstəricisini təsvir etmək üçün bu xassəyə 25-dən çox qiymət seçmək olar;

❖ `Enabled` – elementin aktivliyini müəyyən edir. Susmaya görə bu xassəyə `true` qiyməti verilmişdir. Ona görə də bütün elementlər aktiv olur. Bu xassəyə `false` qiyməti verdikdə element aktiv olmur, ona görə də onunla işləmək mümkün olmur;

❖ `Font` – bütün forma və onun üzərində yerləşdirilmiş bütün idarəetmə elementləri üçün şrifti müəyyən edir. Ancaq, bu xassə ilə formanın üzərində yerləşdirilmiş elementlərinin özləri üçün də şrift seçə bilərik;

❖ `ForeColor` – formada şriftin rəngini bildirir. Əgər `Font` xassəsinin alt xassələrinin siyahısına baxsaq görərik ki, orada şriftin rəngini müəyyən edən rəng xassəsi yoxdur. Şriftin rəngini `ForeColor` xassəsi ilə təyin etmək lazımdır;

❖ `FormBorderStyle` – formanın konturlarının və başlığının üslubunu göstərir. Bu xassəyə aşağıdakı qiymətlərdən birini seçmək olar:

- `None` – formanın konturları olmayacaq;
- `FixedSingle` – pəncərənin ölçülərini dəyişdirməyə imkan verməyən qeyd edilmiş kontur;
- `Fixed3D` – pəncərənin ölçülərini dəyişdirməyə imkan verməyən qeyd edilmiş kontur, lakin kontur üçölçülü olacaqdır;
- `FixedDialog` – `FixedSingle` xassəsinə oxşayır, lakin pəncərənin başlığında sistem menyusu düyməsi olmayacaqdır. Belə kontur, adətən, dialoq pəncərələri üçün istifadə edilir;
- `Sizable` – pəncərənin ölçülərini dəyişdirməyə imkan verən standart kontur;
- `FixedToolWindow` – pəncərənin ölçülərini dəyişdirməyə imkan verməyən çox nazik başlıqlı kontur;

- `SizableToolWindow` – `FixectToolWindow` ilə eynidir, fərq ondan ibarətdir ki, pəncərənin ölçülərini dəyişdirmək mümkündür.

❖ `HelpButton` – formanın kömək (məlumat) düyməsinin əks olunmasını göstərir (`false` qiyməti seçildikdə düymə təsvir edilmir);

❖ `Icon` – forma üçün sistem menyusunun nişanını (ikona) müəyyən etməyə imkan verir;

❖ `ShowIcon` – formanın başlığında sistem menyusu nişanının göstərilməsini müəyyən edir, əgər bu xassəyə `false` qiyməti verilərsə, sistem menyusu düyməsi olmayacaqdır;

❖ `Location` – ekranın sol yuxarı küncünə nəzərən pəncərənin vəziyyətini müəyyən edir (komponentlər üçün də belə xassə mövcuddur, lakin orada məsafə yerləşdiyi valideyn komponentin sol yuxarı küncünə görə hesablanır). Bu xassə iki qiymətdən ibarətdir: `X` və `Y`;

❖ `MaximizeBox` – formanın başlığında pəncərəni tam ekran boyu açma düyməsinin əlçatan olub-olmadığını müəyyən edir. Əgər bu xassəyə `false` qiyməti verilərsə, pəncərəni tam ekran boyunca açmaq mümkün olmayacaqdır (əl ilə onun ölçülərini dəyişdirmək mümkündür);

❖ `MinimizeBox` – pəncərəni bükmə düyməsidir, ona `false` qiyməti verilərsə, pəncərəni bükmə düyməsi əlçatan olmur;

❖ `MaximumSize` – formanın maksimal ölçüsünü müəyyən edir. Pəncərənin eni və hündürlüyünə verilmiş qiymətdən böyük ölçülərə formanı böyütmək mümkün olmayacaq;

❖ `MinimumSize` – `MaximumSize` xassəsinə analojidir. Formanın minimal ölçüsünü müəyyən edir. Pəncərənin eni və hündürlüyünə verilmiş qiymətdən kiçik ölçülərə formanı kiçiltmək mümkün olmayacaq;

❖ `Opacity` – formanın şəffaflığını müəyyən edir. Adi halda forma şəffaf deyil, bu xassənin qiyməti `100%`-dir. Formanın şəffaf olması üçün bu xassəyə `100`-dən kiçik qiymət vermək lazımdır;

❖ `Size` – formanın başlanğıc ölçüsünü (`Width` - eni və `Height` - hündürlüyünü) müəyyən edir;

❖ `StartPosition` – pəncərənin başlanğıc mövqeyini müəyyən edir. Bu xassə üçün aşağıdakı qiymətlər seçilə bilər:

- `WindowsDefaultLocation` – pəncərənin vəziyyəti ƏS tərəfindən təyin edilir;
- `WindowsDefaultBounds` – sistem nəinki pəncərənin vəziyyətini, həm də ölçülərini təyin edəcəkdir;
- `CenterParent` – pəncərə valideyn pəncərənin mərkəzində yerləşəcək. Bunu, adətən, bütün törəmə pəncərələr üçün tətbiq etmək yaxşı olar;
- `Manual` – pəncərənin vəziyyətini proqramçı özü `Location` xassəsinə qiymət verməklə müəyyən edəcəkdir;
- `CenterScreen` – pəncərə ekranın mərkəzində yerləşəcəkdir.

❖ `Text` – formanın başlığını müəyyən edir;

❖ `TopMost` – əgər bu xassəyə `true` qiyməti verilərsə, onda forma başqa pəncərələrin üstünü örtəcəkdir;

❖ `Visible` – formanın görünməsini müəyyən edir, əgər bu xassəyə `false` qiyməti versək, onda forma görünməyəcəkdir (gizlədiləcəkdir);

❖ `WindowState` – əlavə işə buraxıldıqda formanın hansı vəziyyətdə – normal, tam ekran boyu açılmış və ya bükülmüş – olacağını müəyyən edir. Bu xassəyə aşağıdakı qiymətləri seçmək olar:

- `Normal` – normal vəziyyət;

- `Maximized` – pəncərə tam ekran boyu açılmış vəziyyətdə olacaq;
- `Minimized` – pəncərə bükülmüş vəziyyətdə olacaq;

❖ `ShowInTaskBar` –məsələlər panelində pəncərənin təsvir etdirilməsini müəyyən edir, əgər bu xassəyə `false` qiyməti verilərsə, pəncərə məsələlər panelində təsvir edilməyəcək. Əsas pəncərə üçün bu xassəyə `true` qiyməti, törəmə pəncərələr üçün isə `false` qiyməti vermək məqsədəuyğundur.

Əksər hallarda layihələr bir neçə formadan ibarət olur. Layihəyə yeni forma əlavə etmək üçün `Solution Explorer` (`Обозреватель решений`) pəncərəsində layihənin adı üzərində mausun sağ düyməsini basıb kontekst menyudan `Add/Windows Form...` (`Добавить/Форма Windows...`) əmrini icra edərək açılan pəncərədə `Windows Form` (`Форма Windows Forms`) seçib, `Name` (`Имя`) sahəsinə ad daxil edib `Add` (`Добавить`) düyməsini basmaq lazımdır. Bundan sonra isə `Form1.cs` faylına `Form2 f2 = new Form2();` kodunu yazmaq lazımdır. Formanı göstərmək üçün `Show()`, gizlətmək üçün isə `Hide()` metodu tətbiq edilir.

Proqram yolu ilə xassələrə qiymətlərin müəyyənləşdirilməsi

Biz indiyədək elementlərə xassələrin qiymətlərini `Properties` (`Свойства`) pəncərəsinin köməyi ilə müəyyən etməyi öyrəndik. Bu qiymətləri proqram yolu ilə də dinamik müəyyənləşdirmək olar. Bu zaman elementlərə xassələrin qiymətləri dərhal deyil, proqram icra olunduqda mənimsədiləcəkdir. Məsələn, şriftin ölçüsünü (`Size` xassəsi) kodlar vasitəsi ilə belə yazmaq olar:

```
Control.Font.Size = 12;
```

Yeri gəlmişkən qeyd edək ki, `Size` yalnız şriftin deyil, formanın, elementlərin ölçüsünü müəyyən etmək üçündür. `Size` strukturunun iki sahəsi vardır: `Width` və `Height`. Bunlar müvafiq olaraq eni və hündürlüyü müəyyən edir. Strukturun qiymətini Siz dəyişdirə bilməzsiniz və aşağıdakı kodlar səhvə gətirəcəkdir:

```
Size.Height = 100;
```

Həmişə strukturun yeni nüsxəsini yaratmaq lazımdır. Məsələn, aşağıdakı kodla pəncərənin ölçüsü *200x300* kimi təyin edilir:

```
Size = new Size(200,300);
```

`Size` tipli verilənlər `MaximumSize`, `MinimunSize`, `AutoScrollMargin` və `AutoScrollMinSize` xassələri üçün də istifadə edilir.

İndi isə formanın bəzi xassələrini dəyişdirək. Bunun üçün `Form1.cs` kod faylına aşağıdakı əlavələri edək:

```
using System;
using System.Windows.Forms;

namespace WindowsFormsApplication24
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
Text = "İLK PROQRAM!";
BackColor = Color.Green;
Size = new Size(400, 300);
}}}
```

Bu kodlarla formanın başlığı, rəngi və ölçüləri dəyişdirilir. Həmin kodları aşağıdakı kodlarla da əvəz etmək olar:

```
Text = "İLK PROQRAM!";
this.BackColor = Color.Green;
```

```
this.Width = 400;
this.Height = 300;
```

Proqram yolu ilə xassələrə qiymətlər verdikdə yadda saxlamaq lazımdır ki, bütün kodlar `InitializeComponent();` metodundan sonra yazılmalıdır.

Formanın başlanğıc vəziyyətini `StartPosition` xassəsi müəyyən edir. Formanı ekranın mərkəzində yerləşdirmək üçün proqrama aşağıdakı kodları yazmaq lazımdır:

```
this.StartPosition =
          FormStartPosition.CenterScreen;
```

`Location` strukturu `Point` tiplidir və, əksər hallarda o, nöqtə verir. Bu tip komponentin və ya formanın vəziyyətini müəyyən etmək üçün tətbiq edilir. Bu strukturun iki vacib xassəsi var: `X` və `Y`. Onlar müvafiq koordinat oxlarına görə yerdəyişmələrin qiymətini özündə saxlayır. Məsələn, aşağıdakı kod cari formanı *10x10* mövqeyinə sürüşdürür:

```
this.Location = new Point (10,10);
```

Ölçü və vəziyyəti müəyyən edən strukturun hər iki xassələrinin qiymətləri `int` tam tiplidir (çünki, formanın ölçüsü və vəziyyəti kəsr ədəd ola bilməz).

Formanın tam ekran boyu açılması üçün kodu belə yaza bilərik:

```
this.WindowState =
     FormWindowState.Maximized;
```

Layihənin faylları

Qeyd etdiyimiz kimi, əlavə layihələndirildikdə çoxlu fayllar yaranır. Onlardan ən əsaslarına qısaca nəzər yetirək.

Resurslar faylı. Bu fayl `.resx` genişlənmiş hissəsinə malikdir. Burada kod faylının atributları saxlanır və iş mühiti onlarla işləyir. O, iş mühiti tərəfindən avtomatik olaraq yaradılır. Bu fayl `Properties` budağında yerləşən

`Resources.resx` faylıdır. Bu faylda sətirlər, təsvirlər və s. kimi resurslar yaradıb onları sonralar proqramda istifadə etmək olar.

`Program.cs` faylı. Bu layihəni icra edən kodlardan ibarət fayldır. Burada adlar fəzası ilə yanaşı `Main` metoduna müraciət edilir. `Main` metodu `Application` sinfinin üç metodundan ibarətdir.

`Form1.cs` faylı. `Bu` faylda sinfin daxilində yalnız konstruktor müəyyən edilmişdir ki, o, `InitializeComponent();` metodunu çağırır. Əslində biz kodlaşdırma işlərini məhz bu faylda yerinə yetiririk.

`Form1.Designer.cs` faylı. Forma üzərində vizual olaraq gördüyümüz işlərin reallaşdırılması üçün kodlar `Form1.Designer.cs` faylında saxlanır.

Yoxlama sualları

1. Forma nədir?
2. Formanın xassələri nədir?
3. Layihənın faylari hansılardır?
4. `this` parametri nəyə müraciəti göstərir?
5. Yaradılmış formanın arxa fonunun rəngini dəyişin.
6. `button` düyməsini basdıqda ikinci formanı görünməz edin.
7. `button` düyməsinin ölçülərini dəyişin.
8. Formanın ölçülərini dəyişin.
9. Formanin yerini ekranda dəyişin.
10. Forma üzərinə dörd düymə yerləşdirin. Düymələrin səthində “Qirmızı”, “Yaşıl”, “Göy” və “Sarı” mətnləri yazın. Düymələri basdıqda formanın rəngi müvafiq mətnə uyğun dəyişsin.

11. Forma yaradıldıqda hadisə emaledicisi yaradın ki, o baş verdikdə formanın rəngi dəyişsin və başlığında “İşin başlanması” mətni yazılsın.

12. Forma yaradıldıqda hadisə emaledicisi yaradın ki, o baş verdikdə forma ölçülərini dəyişsin və şəffaf olsun.

13. Forma üzərində iki düymə yerləşdirib, onların səthində “Salam” və “Xudahafiz” mətnləri yazın. Müvafiq düymələri basdıqda ekranda həmin mətnlərdən ibarət ismarıclar peyda olsun və formanın başlığında təsvir edilsin.

14. Elementin aktivliyini müəyyən edən xassə hansıdır?

15. Şriftin rəngini bildirən xassə hansıdır?

16. Formanın konturlarının və başlığının üslubunu müəyyən edən xassə hansıdır?

17. `Location` xassəsi nə üçündür?

18. `MaximizeBox` və `MinimizeBox` xassələri nə üçündür?

19. `StartPosition` xassəsi nə üçündür?

20. `WindowState` xassəsi nə üçündür?

LABORATORİYA İŞİ № 3.
YAZI KOMPONENTLƏRİ İLƏ İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən `label` idarəetmə elementinin xassələrini və əlavələrin layihələndirilməsində onun tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat

`label` idarəetmə elementi

`label` idarəetmə elementi digər idarəetmə elementlərinə imzaların yaradılması və ya birbaşa forma üzərinə məlumatların çıxardılması üçün nəzərdə tutulmuşdur.

`label` komponentinin aşağıdakı xassələri vardır:

❖ `Text` — bu xassə vasitəsilə biz komponentin səthində (üzərində) əks olunacaq mətni, yəni komponentin mətnini veririk;

❖ `AutoSize` — mətnin ölçüsünü avtomatik olaraq dəyişdirmək üçündür. Susmaya görə bu xassəyə `true` qiyməti verildiyi üçün komponent öz ölçüsünü mətnin uzunluğuna uyğun avtomatik olaraq dəyişdirir;

❖ `TextAlign` — komponent daxilində mətnin düzləndirilməsini idarə edir.

Bu xassə aşağıdakı qiymətlərdən birini ala bilər:

- `TopLeft` – mətn komponentin yuxarı sol tərəfində yerləşir;
- `TopRight` – mətn komponentin yuxarı sağ tərəfində yerləşir;

- `TopCenter` – mətn komponentin yuxarı mərkəzində yerləşir;
- `MiddleLeft` – mətn komponentin sol mərkəzində yerləşir;
- `MiddleRight` – mətn komponentin sağ mərkəzində yerləşir;
- `MiddleCenter` – mətn komponentin mərkəzində yerləşir;
- `BottomLeft` – mətn komponentin aşağı sol tərəfində yerləşir;
- `BottomRight` – mətn komponentin aşağı sağ tərəfində yerləşir;
- `BottomCenter` – mətn komponentin aşağı mərkəzində yerləşir.

Qeyd edək ki, `AutoSize` xassəsinə `true` qiyməti verildikdə `TextAlign` xassəsinin heç bir təsiri olmur.

❖ `AllowDrop` – komponentin sahəsində mətnin bir neçə sətirdə yerləşməsini müəyyən edir. Susmaya görə bu xassəyə `false` qiyməti verilmişdir, ona görə də mətn bir sətirdə yerləşir.

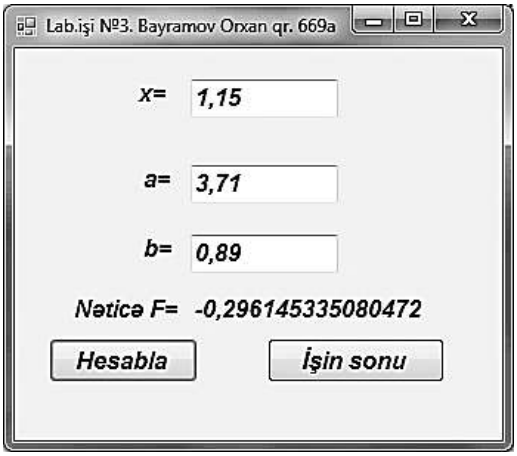
`Label` komponentində şəkil yerləşdirmək (`Image` xassəsi) və ona şəffaflıq vermək olar (`BackColor` xassəsinə `Transparent` qiyməti seçməklə). Bu komponent daxiletmə fokusu ala bilmir, ona görə də klaviaturadan daxil edilən məlumatı emal etmir.

Misal. Aşağıdakı funksiyanı hesablamaq üçün əlavə yaradaq:

$$F = x\left(b\left(\frac{x\cdot a}{x+a}\right)^{b}\right)^{x} - \frac{\lg a - 1}{\sqrt{a^2 - x^2 + b^2}} - \operatorname{ctg}\sqrt{x^2 - b}$$

Əlavə belə layihələndiriləcəkdir. Funksiyanı hesablamaq üçün ilkin verilənlər `textBox` komponentindən daxil ediləcək, "`Hesabla`" düyməsini basdıqda F funksiyası hesablanaraq `label` komponentində təsvir etdiriləcəkdir. "`İşin sonu`" düyməsini basdıqda isə proqramla iş sona yetəcəkdir.

Layihənin icrası üçün bizə beş ədəd `label` komponenti lazım olacaqdır. Birinci üç `label` komponentlərini şəkil 3.1-də göstərildiyi kimi `textBox` komponentlərinin qarşısında yerləşdirib, onların `Text` xassəsinə şəklə müvafiq mətnlər yazın. `Label4` komponentini isə `label5` komponentindən sonra yerləşdirin. `Label5` komponentində hesablamanın nəticəsi təsvir ediləcək.



Şəkil 3.1. 3 №-li laboratoriya işinin nəticəsi

Bu layihənin kodu belə olacaqdır:

```
using System;
using System.Windows.Forms;
namespace WindowsFormsApp13
```

```
    {
    public partial class Form1 : Form
    {
    public Form1()
    {
      InitializeComponent();
    }
    private void button1_Click(object
                sender, EventArgs e)
    {
      double a,b,x;
      x = Convert.ToDouble(textBox1.Text);
      a = Convert.ToDouble(textBox2.Text);
      b = Convert.ToDouble(textBox3.Text);
  double F=x*Math.Pow(b*(Math.Pow(((x*a)/
    (x+a)), b)), x)-(Math.Log10(a)-1)/
    Math.Sqrt(a*a-x*x+b*b)-1/Math.Tan
    (Math.Sqrt(x * x - b));
  label5.Text = F.ToString();
}

private void button2_Click(object sender,
                           EventArgs e)
{
  this.Close();
} } }
```

Proqramın nəticəsi şəkil 3.1-də göstərilmişdir.

Yoxlama sualları

1. Mətnlərin təsviri üçün nəzərdə tutulmuş elementlər hansılardır?

2. `label` elementinin ölçüsü susmaya görə necə təyin edilir?

3. `label` komponentinin məzmunu onun daxilində necə düzləndirilir?

4. `label` komponentinə mətn klaviaturadan necə daxil edilir?

5. `label` komponentinin məzmununa hansı xassə ilə müraciət edilir?

6. `label` komponentinin məzmununun rəngi hansı xassə ilə müəyyən edilir?

7. `label` komponentinin fonunun rəngi hansı xassə ilə müəyyən edilir?

8. `label` komponenti necə şəffaf edilir?

9. İstinad üçün nəzərdə tutulmuş komponent hansıdır?

10. `linkLabel` komponenti hansı xassələrə də malikdir?

Laboratoriya işlərinə aid tapşırıqların variantları

1-20-ci variantlarda verilmiş tapşırıqlar üçün şəkil 3.1-də göstərilmiş əlavəyə analoji əlavə hazırlayın.

1. $$t=\frac{2\cos\left(x-\frac{\pi}{6}\right)}{0.5+\sin^2 y}\left(1+\frac{z^2}{3-z^2/5}\right).$$

$x=14.26, y=-1.22, z=3.5\times10^{-2}$ $t=0.564849.$

2. $$u=\frac{\sqrt[3]{8+|x-y|^2+1}}{x^2+y^2+2}-e^{|x-y|}\left(\mathrm{tg}^2 z+1\right)^x.$$

$x=-4.5, y=0.75\times10^{-4}, z=0.845\times10^2$ $u=-55.6848.$

3. $$v=\frac{1+\sin^2(x+y)}{\left|x-\frac{2y}{1+x^2y^2}\right|}x^{|y|}+\cos^2\left(\mathrm{arctg}\frac{1}{z}\right).$$

$x=3.74\times10^{-2}, y=-0.825, z=0.16\times10^2,$ $v=1.0553.$

4. $w=\left|\cos x-\cos y\right|^{\left(1+2\sin^2 y\right)}\left(1+z+\frac{z^2}{2}+\frac{z^3}{3}+\frac{z^4}{4}\right).$

$x=0.4\times10^4, y=-0.875, z=-0.475\times10^{-3}$ $w=1.9873.$

5. $\alpha=\ln\left(y^{-\sqrt{|x|}}\right)\left(x-\frac{y}{2}\right)+\sin^2\operatorname{arctg}(z).$

$x=-15.246, y=4.642\times10^{-2}, z=20.001\times10^2$ $\alpha=-182.036.$

6. $\beta=\sqrt{10\left(\sqrt[3]{x}+x^{y+2}\right)}\left(\arcsin^2 z-\left|x-y\right|\right).$

$x=16.55\times10^{-3}, y=-2.75, z=0.15$ $\beta=-38.902.$

7. $\gamma=5\operatorname{arctg}(x)-\frac{1}{4}\arccos(x)\frac{x+3\left|x-y\right|+x^2}{\left|x-y\right|z+x^2}.$

$x=0.1722, y=6.33, z=3.25\times10^{-4}$ $\gamma=-172.025.$

8. $\varphi=\frac{e^{|x-y|}\left|x-y\right|^{x+y}}{\operatorname{arctg}(x)+\operatorname{arctg}(z)}+\sqrt[3]{x^6+\ln^2 y}.$

$x=-2.235\times10^{-2}, y=2.23, z=15.221$ $\varphi=39.374.$

9. $\psi=\left|x^{\frac{y}{x}}-\sqrt[3]{\frac{y}{x}}\right|+(y-x)\frac{\cos y-\frac{z}{(y-x)}}{1+(y-x)^2}.$

$x=1.825\times10^2, y=18.225, z=-3.298\times10^{-2}$ $\psi=1.2131.$

10. $a = 2^{-x}\sqrt{x + \sqrt[4]{|y|}}\sqrt[3]{e^{x-1/\sin z}}$.

$x = 3.981 \times 10^{-2}, y = -1.625 \times 10^{3}, z = 0.512$ $a = 1.26185$.

11. $b = y^{\sqrt[3]{|x|}} + \cos^3(y)\dfrac{|x-y|\left(1 + \dfrac{\sin^2 z}{\sqrt{x+y}}\right)}{e^{|x-y|} + \dfrac{x}{2}}$.

$x = 6.251, y = 0.827, z = 25.001$ $b = 0.7121$.

12. $c = 2^{(y^x)} + \left(3^x\right)^y - \dfrac{y\left(\operatorname{arctg} z - \dfrac{\pi}{6}\right)}{|x| + \dfrac{1}{y^2+1}}$.

$x = 3.251, y = 0.325, z = 0.466 \times 10^{-4}$ $c = 4.025$.

13. $f = \dfrac{\sqrt[4]{y + \sqrt[3]{x-1}}}{|x-y|\left(\sin^2 z + \operatorname{tg} z\right)}$.

$x = 17.421, y = 10.365 \times 10^{-3}, z = 0.828 \times 10^{5}$ $f = 0.33056$.

14. $g = \dfrac{y^{x+1}}{\sqrt[3]{|y-2|} + 3} + \dfrac{x + \dfrac{y}{2}}{2|x+y|}(x+1)^{-1/\sin z}$.

$x = 12.3 \times 10^{-1}, y = 15.4, z = 0.252 \times 10^{3}$ $g = 82.8257$.

15. $h=\dfrac{x^{y+1}+e^{y-1}}{1+x\left|y-\operatorname{tg}z\right|}\left(1+\left|y-x\right|\right)+\dfrac{\left|y-x\right|^2}{2}-\dfrac{\left|y-x\right|^3}{3}.$

$x=2.444, y=0.869\times10^{-2}, z=-0.13\times10^{3}$ $h=-0.49871.$

16. $y=\sqrt{cx}-2.7\dfrac{\left|c\right|+\left|x\right|}{c^2x^2}\cdot e^{cx}+\cos\dfrac{(a+b)^2}{cx-b}$

$a=3.7;\ b=0.07;\ c=1.5;\ x=5.75$

17. $y=4.5\dfrac{(a+b)^2}{(a-b)^2}-\sqrt{(a+b)(a-b)}+10^{-1}\dfrac{\ln(a-b)}{\ln(a+b)}\cdot e^{x^2}$

$a=7.5;\ b=1.2;\ x=0.5$

18. $y=2.4\left|\dfrac{x^2+b}{a}\right|+(a-b)\sin^2(a-b)+10^{-2}(x-b)$

$a=5.1;\ b=0.7;\ x=-0.05$

19. $y=\dfrac{ax-\sqrt{b}}{5.7(x^2+b^2)}-\dfrac{\left|x+b\right|-a^2}{x^2}\operatorname{tg}^2 b$

$a=0.1;\ b=2.4;\ x=-0.3$

20. $y=\sqrt{\dfrac{c-dx^2}{x}}+\dfrac{\ln(x^2+c)}{0.7x+ad}-\dfrac{10^{-2}}{c-dx^3}$

$a=4.5;\ c=7.4;\ d=-2.1;\ x=0.15$

21. Forma üzərinə `label`, `textBox` və `button` düymələri yerləşdirin. Düyməni basdıqda `textBox` komponentindən daxil edilmiş ədədin hesablanmış 10, 20, 30,

...,90 faizi `label` komponentində, faizlərin özləri isə düymənin səthində təsvir etdirilsin.

22. Forma üzərinə iki `label`, `textBox` və `button` düymələri yerləşdirin. Düyməni basdıqda `textBox` komponentindən daxil edilmiş tək ədəd `label1`, çüt ədəd isə `label2` komponentində təsvir edilsin.

23. `label1` komponentindən daxil edilmiş ədədin sinusunu hesablayıb `label2` komponentində təsvir edin. Əməliyyatı düyməbasma emaledicisi ilə icra edin.

24. `label1` və `textBox` komponentlərindən daxil edilmiş ədədlərin hasilini formanın başlığında təsvir edin. Əməliyyatı `load` hadisə emaledicisi ilə icra edin.

25. `button` düyməsini basdıqda `label` komponentində şəkil yerləşdirən əlavə layihələndirin.

26. Formanın fonunda yerləşən şəkil üzərində şəffaf mətn yazın.

27. `button` düyməsini basdıqda `label` komponenti daxilində mətn öz vəziyyətini dəyişsin. Mətnin vəziyyətinə uyğun ədədləri `textBox` komponentindən daxil edin və müxtəlif vəziyyətləri `switch` operatoru ilə idarə edin.

28. `button` düyməsini basdıqda `label` komponenti daxilində mətn öz rəngini dəyişsin. Rənglərə uyğun ədədləri `textBox` komponentindən daxil edin və müxtəlif rəngləri `switch` operatoru ilə idarə edin.

29. `button` düymələrini basdıqda mətn növbə ilə iki müxtəlif `label` komponentlərində təsvir edilsin (birində təsvir edildikdə digəri yox olsun).

30. `timer` komponentlərindən istifadə etməklə dörd `label` komponentlərində növbə ilə müxtəlif mətnlər təsvir edən (məsələn, AzTU, BDU, ADPU və ADNSU) əlavə yaradın.

LABORATORİYA İŞİ № 4.
MƏTN SAHƏSİ İLƏ İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən `textBox` idarəetmə elementinin xassələrini və əlavələrin layihələndirilməsində onun tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat

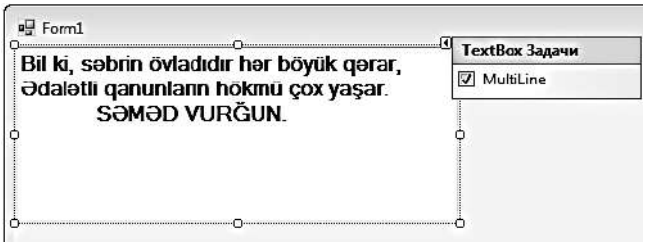
`textBox` komponentinə mətn sahəsi də deyirlər. Bu komponent oblastdan ibarətdir ki, istifadəçi buraya klaviaturadan mətn daxil edə bilər və ya bu oblasta mətn çıxarıla bilər. Əgər Sizə lazımdırsa ki, istifadəçi sətir, ədəd və ya digər nələrsə daxil etsin, onda formaya məhz bu komponenti yerləşdirməlisiniz. Bununla yanaşı bu komponent yalnız 64 Kb-a qədər mətnlə işləyə bilər. 64 Kb-dan böyük həcmli mətnlərlə işləmək üçün `richTextBox` komponentindən istifadə etmək lazımdır.

`textBox` komponentinin xarici görünüşü şəkil 4.1-də göstərilmişdir. Onun yuxarı sərhəddinin sağ tərəfində kiçik kvadrat daxilindəki oxun üzərində mausun düyməsini basdıqda `textBox Tasks` (`textBox Задачи`) menyusu açılır. Bu menyunun əmrləri ilə xassələrə daha tez müraciət etmək mümkün olur. Məsələn, şəkil 4.1-də `Multiline` xassəsini qoşmaqla, birsətirli redaktor çoxsətirli redaktora çevrilmişdir. Qeyd edək ki, bu menyu əksər idarəetmə elementlərində mövcuddur.

Bu komponentin xassələri aşağıdakılardır:

- `Text` – daxiletmə sahəsində təsvir olunan mətnin məzmunu;

- `Multiline` – əgər bu xassəyə `true` qiyməti verilərsə, onda komponent daxilində mətn bir neçə sətirdən ibarət olacaqdır. Susmaya görə ona `false` qiyməti verilmişdir;
- `UseSystemPassword` – əgər bu xassəyə `true` qiyməti verilərsə, onda mətn sahəsi şifrin daxil edilməsi rejimində işləyir və komponent daxilində mətn nöqtələrdən ibarət olur, yəni istifadəçinin daxil etdiyi informasiya gizlədilir;



Şəkil 4.1 `Multiline` xassəsi qoşulmuş birsətirli redaktor

- `ReadOnly` – əgər bu xassəyə `true` qiyməti verilərsə, onda komponent boz fona malik olur və onun məzmununa düzəlişlər etmək mümkün olmur;
- `SelectedText` – mətn sahəsində seçilmiş mətn fraqmentini alır və ya qaytarır;
- `SelectionStart` – seçilmiş fraqmentin başlanğıcını müəyyən edən simvolu alır və ya qaytarır;
- `SelectionLength` – seçilmiş fraqmentdə simvolların sayını müəyyən edən qiyməti alır və ya qaytarır.

Bu komponentin iki əsas hadisəsi var. Onlardan birincisi `TextChanged` hadisəsidir ki, o istifadəçi hər dəfə komponent daxilində mətni dəyişdirdikdə yaranır. Digər hadisə `Modified` hadisəsidir. İstifadəçi komponent daxilində mətnə dəyişikliklər etdikdə onun qiyməti avtomatik olaraq `true` olur.

Bu xassəyə `false` qiymətini yalnız proqram yolu ilə vermək olar. `Modified` metodunun təsirini misalda izah edək.

Misal. `Modified` metodunun izahı.

Forma üzərinə `textBox` və `label` komponentləri yerləşdirək. `textBox` komponentini seçib `ModifiedChanged` hadisəsinin qarşısında mausun düyməsini iki dəfə basıb aşağıdakı kodları yazaq:

```
private void textBox1_ModifiedChanged(
          object sender, EventArgs e)
{
 if (textBox1.Modified)
   label1.Text = "Mətn dəyişdirildi";
 else
   label1.Text = "Mətn dəyişdirilmədi";
}
```

Daxiletmə sahəsinə mətn daxil edən kimi `label` komponentində `"Mətn dəyişdirildi"` məlumatı təsvir ediləcək. İndi forma üzərinə `Button` düyməsi yerləşdirib onun üçün `Click` hadisəsini yaradaq:

```
private void button1_Click(object
              sender,EventArgs e)
{
  textBox1.Modified = false;
}
```

İndi `Button` düyməsini basdıqda `label` komponentində `"Mətn dəyişdirilmədi"` məlumatı təsvir ediləcəkdir.

Misal. Aşağıdakı funksiyanı hesablamaq üçün əlavə yaradaq:

$$U=\begin{cases} y\cdot\sin(x)+z, & z-x=0 \\ y\cdot e^{\sin(x)}-z, & z-x<0 \\ y\cdot\sin(\sin(x))+z, & z-x>0 \end{cases}$$

Əlavə belə layihələndiriləcəkdir. Funksiyanı hesablamaq üçün ilkin verilənlər olan x, y, z dəyişənləri textBox komponentindən daxil ediləcək, "Hesabla" düyməsini basdıqda U funksiyası hesablanaraq növbəti textBox komponentində təsvir etdiriləcəkdir. "Təmizlə" düyməsini basdıqda isə bütün textBox komponentinin məzmunu silinəcəkdir.

Layihənin icrası üçün bizə üç ədəd label və üç ədəd textBox komponentləri lazım olacaqdır. Bu label komponentlərini şəkil 4.2-də göstərildiyi kimi uyğun textBox komponentlərinin qarşısında yerləşdirib, onların Text xassəsinə şəklə müvafiq mətnlər yazın. Nəticəni göstərmək üçün dördüncü textBox komponenti yerləşdirək. Bu komponentdə nəticələr bir neçə sətirdə təsvir ediləcəkdir. Ona görə onun MultiLine xassəsinə true, ScrollBars xassəsinə isə Botch qiyməti vermək lazımdır. Bu iki əməliyyatdan sonra textBox komponenti çoxlu sətirlərdən və fırlatma zolaqlarından ibarət olacaqdır (şəkil 4.2). Aydınlıq üçün bu komponentin üstündə label4 komponenti yerləşdirib, onun Text xassəsinə "Proqramın işinin nəticəsi:" mətnini yazın.

textBox komponentinin hər bir sətri sətir tipli Lines sətirlər massivində yerləşir. Lakin redaktora yeni sətir əlavə etmək üçün hər bir sətrə birbaşa müraciət mümkün deyil. Yeni element əlavə etmək üçün Text xassəsinə yeni belə sətir əlavə etmək lazımdır:

```
textBox1.Text += Environment.NewLine +
                 "Yeni sətrin məzmunu";
```

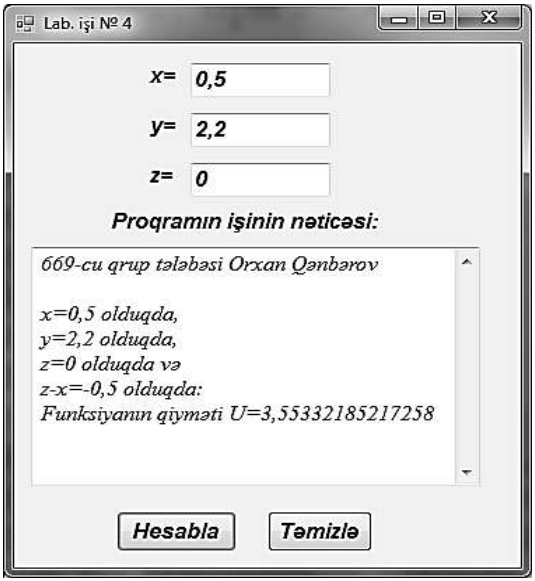
Bu kodla cari sətrə yeni sətrə keçid simvolu əlavə edilir. Bu simvol Environment sinfindəndir. Bundan sonra isə yeni sətir əlavə edilir.

AppendText metodunu istifadə etməklə bu kodun əvəzinə aşağıdakı kodu da yazmaq olar:

```
textBox1.AppendText("Yeni sətrin
                        məzmunu");
```

Bu məsələni belə proqramlaşdıra bilərik:

```
using System;
```



Şəkil 4.2. 4 №-li laboratoriya işinin nəticəsi

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Windows.Forms;

namespace WindowsFormsApp14
{
public partial class Form1 : Form
{
public Form1()
{
  InitializeComponent();
}
```

```
private void button1_Click(object
               sender,EventArgs e)
{

double x, y, z, U, t;
  x = Convert.ToDouble(textBox1.Text);
  y = Convert.ToDouble(textBox2.Text);
  z = Convert.ToDouble(textBox3.Text);
  t = z - x;
  if (t == 0)
    U = y * Math.Sin(x) + z;
  else
    if (t <= 0)
      U = y * Math.Exp(Math.Sin(x)) - z;
    else
     U = y * Math.Sin(Math.Sin(x)) + z;
  textBox4.Text += "669-cu qrup tələbəsi
   Orxan Qənbərov"+Environment.NewLine;
  textBox4.Text += Environment.NewLine +
      "x=" + x.ToString() + " olduqda,";
  textBox4.Text += Environment.NewLine +
      "y=" + y.ToString() + " olduqda,";
  textBox4.Text += Environment.NewLine +
    "z=" + z.ToString() + " olduqda və";
  textBox4.Text += Environment.NewLine +
    "z-x=" + t.ToString() + " olduqda:";
  textBox4.AppendText(""+"\n");
  textBox4.AppendText("Funksiyanın
        qiyməti  U=" + U.ToString());
}

private void button2_Click(object
                sender,EventArgs e)
{
```

```
        textBox1.Clear();
        textBox2.Clear();
        textBox3.Clear();
        textBox4.Clear();
    } } }
```

Proqramın nəticəsi şəkil 4.2-də göstərilmişdir.

Yoxlama sualları

1. Mətnləri təsvir etmək üçün hansı elementlər vardır?
2. `textBox` komponentinin hansı xassəsi onu çoxsətirli redaktora çevirir?
3. `textBox` komponentinin hansı xassəsi onun məzmununu bildirir?
4. `textBox` komponentindən hansı tip verilənlər daxil edilə bilər?
5. `textBox` komponentinin `ReadOnly` xassəsi nəyi müəyyən edir?
6. `textBox` komponenti ilə `richTextBox` komponentinin fərqi nədir?
7. Mətn sahəsində seçilmiş mətn fraqmentini müəyyən edən xassə hansıdır?
8. Seçilmiş fraqmentin başlanğıcını müəyyən edən simvolu alan və ya qaytaran xassə hansıdır?
9. Seçilmiş fraqmentdə simvolların sayını müəyyən edən xassə hansıdır?
10. `Modified` hadisəsi nədir?
11. İstifadəçi hər dəfə komponent daxilində mətni dəyişdirdikdə hansı hadisə yaranır?
12. `textBox` komponentin çoxsətirli variantında yeni sətirlər necə əlavə edilir?

Laboratoriya işlərinə aid tapşırıqların variantları

1-20-ci variantlarda *f(x)* funksiyası kimi $e^x, \sqrt{x}$ və x^2 funksiyalarından birini seçin və şəkil 4.2-də göstərilmiş əlavəyə oxşar əlavə hazırlayın.

1. $a=\begin{cases}(f(x)+y)^2-\sqrt{f(x)y}, & xy\rangle 0\\ (f(x)+y)^2+\sqrt{|f(x)\,y|}, & xy\langle 0\\ (f(x)+y)^2+1, & xy=0.\end{cases}$

2. $b=\begin{cases}\ln(f(x))+\left(f(x)^2+y\right)^3, & x/y\rangle 0\\ \ln|f(x)/y|+(f(x)+y)^3, & x/y\langle 0\\ \left(f(x)^2+y\right)^3, & x=0\\ 0, & y=0.\end{cases}$

3. $c=\begin{cases}f(x)^2+y^2+\sin(y), & x-y=0\\ (f(x)-y)^2+\cos(y), & x-y\rangle 0\\ (y-f(x))^2+\operatorname{tg}(y), & x-y\langle 0.\end{cases}$

4. $d=\begin{cases}(f(x)-y)^3+\operatorname{arctg}(f(x)), & x\rangle y\\ (y-f(x))^3+\operatorname{arctg}(f(x)), & y\rangle x\\ (y+f(x))^3+0.5, & y=x.\end{cases}$

5. $e=\begin{cases}i\sqrt{f(x)}, & i-\ təkdir,\ x\rangle 0\\ i/2\sqrt{|f(x)|}, & i-cütdür,\ x\langle 0\\ \sqrt{|if(x)|}, & əks\ halda.\end{cases}$

6. $g=\begin{cases}e^{f(x)-|b|}, & 0.5\langle xb\langle 10\\ \sqrt{|f(x)+b|}, & 0.1\langle xb\langle 0.5\\ 2f(x)^2, & əks\ halda.\end{cases}$

7. $s=\begin{cases}e^{f(x)}, & 1\langle xb\langle 10\\ \sqrt{|f(x)+4*b|}, & 12\langle xb\langle 40\\ bf(x)^2, & əks\ halda.\end{cases}$

8. $j=\begin{cases}\sin(5f(x)+3m|f(x)|), & -1\langle m\langle x\\ \cos(3f(x)+5m|f(x)|), & x\rangle m\\ (f(x)+m)^2, & x=m.\end{cases}$

9. $l=\begin{cases}2f(x)^3+3p^2, & x\rangle |p|\\ |f(x)-p|, & 3\langle x\langle |p|\\ (f(x)-p)^2, & x=|p|.\end{cases}$

10. $k=\begin{cases}\ln(|f(x)|+|q|), & |xq|\rangle 10\\ e^{f(x)+q}, & |xq|\langle 10\\ f(x)+q, & |xq|=10\end{cases}$

11. $m=\dfrac{\max(f(x),y,z)}{\min(f(x),y)}+5.$

12. $n=\dfrac{\min(f(x)+y,y-z)}{\max(f(x),y)}.$

13. $p=\dfrac{|\min(f(x),y)-\max(y,z)|}{2}.$

14. $q=\dfrac{\max(f(x)+y+z,xyz)}{\min(f(x)+y+z,xyz)}.$

15. $c=\begin{cases} f(\sin(x))^2+\sin(f(y)), & x-y=0 \\ (f(\cos(x)))+\cos(f(y)), & x-y\rangle 0 \\ (y-f(\operatorname{tg}(x)))^2+\operatorname{tg}(y), & x-y\langle 0. \end{cases}$

16. $a=\begin{cases} \dfrac{(\mathrm{ax}^2+2)}{(\mathrm{x}^2+1)}f(x), & 1<|\mathrm{x}|<3, \\ a^2+f(x), & |\mathrm{x}|\ \geq 3 \\ \mathrm{ax}\dfrac{\mathrm{f(x)}}{(\mathrm{x}+2)}, & |\mathrm{x}|\ \leq 1. \end{cases}$

17. $c=\begin{cases} f(x)^3-y^3\cdot\cos(x), & x+y=0 \\ (f(x)\cdot y)^2-\cos(y), & x+y\rangle 0 \\ (y\cdot f(x))^2+\pi, & x+y\langle 0. \end{cases}$

18. $k=\begin{cases} \ln\left(|f(x^2)|+|k|\right), & |x\cdot k|\rangle 10 \\ \pi^{f(x)+q}, & |x\cdot k|\langle 10 \\ f(x)-k, & |x\cdot k|=10 \end{cases}$

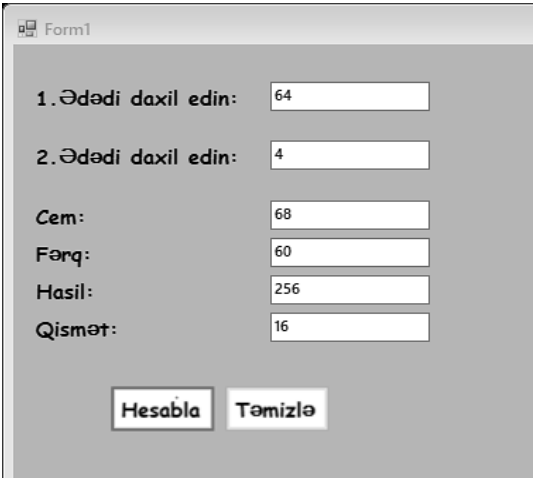
19. $c=\begin{cases} \sin(f(x))+\cos(f(y)), & x-y=0 \\ \operatorname{tg}(f(x+y), & x-y\rangle 0 \\ \sin^2(f(x))+\cos^2(f(y)), & x-y\langle 0. \end{cases}$

20. $r=\max(\min(f(x),y),z).$

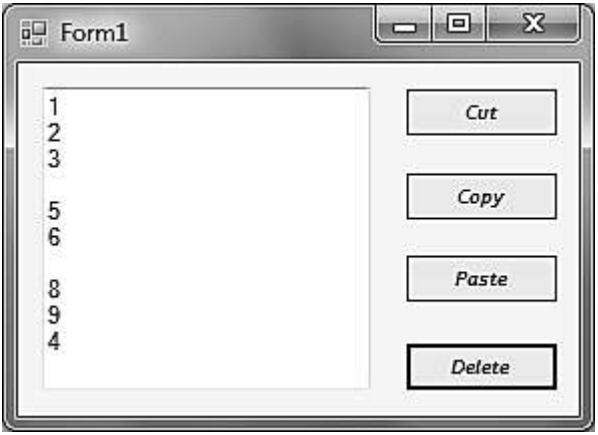
21. Valyuta mübadiləsini yerinə yetirən əlavə hazırlayın. Pulun manat ilə ifadəsini, çevriləcək valyuta növü və valyuta kursunu `textBox` komponentlərindən daxil edin. Düyməni basdıqda çevrilmiş valyutaya uyğun nəticə yeni `textBox` komponentində təsvir edilsin. Digər düyməni basdıqda `textBox` komponentlərinin məzmunu silinsin.

22. Şəkil 4.3-də göstərilmiş əlavəni hazırlayın.

23. Şəkil 4.4-də göstərilmiş əlavəni yaradın. Bu əlavədə `Cut` düyməsini basdıqda `textBox` komponentindən kəsilmiş mətn buferə köçürülsün, `Paste` düyməsini basdıqda isə kursorla göstərilən hissəyə əlavə edilsin. `Delete` düyməsi ilə seçilmiş mətn silinsin.



Şəkil 4.3. 22-ci variantın tapşırığı



Şəkil 4.4. 22-ci variantın tapşırığı

24. Elektrik enerjisinin sərfinə uyğun ödəniş etməyə imkan verən əlavə layihələndirin.

25. Suyun sərfinə uyğun ödəniş etməyə imkan verən əlavə layihələndirin.

26. Dörd hesab əməlini icra etməyə imkan verən kalkulyator yaradın. Ədədlər klaviaturadan deyil, kalkulyatorun düymələri ilə daxil edilsin.

27. `textBox` komponentindən daxil edilmiş ədəd təkdirsə, onu kvadrata yüksəldib, formanın başlığında, cüt olduqda isə onu onun onluq loqarifmini hesablayıb, vergüldən sonra dörd rəqəmə qədər yuvarlaqlaşdırıb digər `textBox` komponentində təsvir edin.

28. `textBox1` komponentinə bir neçə ədəd daxil edin. Bu komponentdən seçilmiş ədəd cütdürsə, onda həmin ədədin kubunu `label` komponentində göstərin, əks təqdirdə isə `textbox1` komponentinin bütün məzmununu digər `textBox2` komponentinə köçürün.

29. `textBox1` komponentinə bir neçə ədəd daxil edin. Bu komponentdən yalnız tək ədədləri `textBox2` komponentinə köçürün.

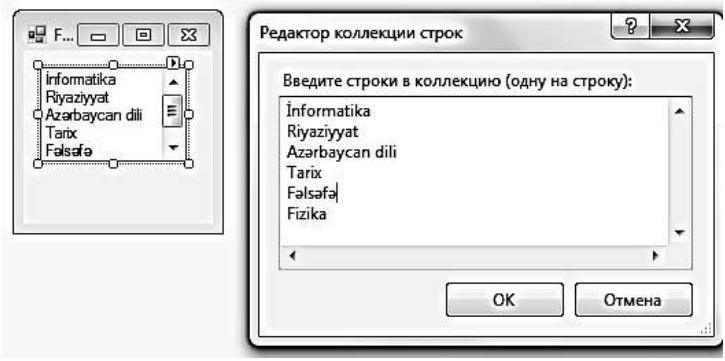
30. `textBox1` komponentinə bir neçə ədəd daxil edin. Bu komponentdən tək ədədləri `textBox2` komponentinə, cüt ədədləri isə `textBox3` komponentinə köçürün.

LABORATORİYA İŞİ № 5.
SİYAHILARLA İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən siyahı idarəetmə elementlərinin xassələrini və əlavələrin layihələndirilməsində onların tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat
`listBox` idarəetmə elementi

`listBox` komponenti siyahıdan ibarətdir ki, istifadəçi bu siyahıdan bir və ya bir neçə elementi seçə bilər. `listBox` siyahısına elementləri redaktor vasitəsi ilə daxil etmək üçün komponenti seçib `Items` xassəsi qarşısındakı üzərində üç nöqtə təsviri olan düyməni basmaq (və ya `listBox Tasks` (`listBox Задачи`) `menyusundan Edit Itens...` (`Изменить элементы...`) əmrini icra etmək) lazımdır. Bu zaman `String Collection Editor (Редактор коллекции строк)` pəncərəsi açılacaqdır (şəkil 5.1). Həmin pəncərədə siyahı tərtib edilir.



Şəkil 5.1. Siyahıya elementlərin daxil edilməsi

`SelectionMode` xassəsinin köməyi ilə istifadəçi siyahıdan bir və ya bir neçə elementi (sətri) seçə bilər. Bu xassə aşağıdakı qiymətləri ala bilər:

- `None` – element seçilə bilməz;
- `One` – susmaya görə məhz bu qiymət təyin edilmişdir, yəni siyahıdan yalnız bir element seçmək olar;
- `MultiSimple` – siyahıdan bir neçə element seçmək olar;
- `MultiExtended` – kursoru idarəetmə klavişləri, maus və *Ctrl* və *Shift* klavişlərinin köməyi ilə siyahıdan bir neçə element seçmək olar.

Hazırda hansı elementin seçildiyini bilmək üçün `SelectedItem` xassəsindən istifadə etmək olar. Əgər bir neçə elementin seçilməsinə icazə vermisinizsə, onda seçilmiş elementlərin siyahısını `SelectedItems` xassəsi ilə almaq olar. `MultiColumn` xassəsi bir neçə sütundan ibarət siyahı yaratmağa imkan verir. `SelectedIndex` xassəsi seçilmiş elementin indeksini qaytarır. Əgər siyahıda seçilmiş element yoxdursa, onda `-1` qiyməti qaytarır. `SelectedIndices` xassəsi isə bütün seçilmiş elementlərin indekslərini qaytarır. Siyahını əlifba sırası ilə nizamlamaq üçün onun `Sorted` xassəsinə `true` qiyməti vermək lazımdır. Siyahıda elementlər sıfırdan başlayaraq nömrələnir. Hər bir elementə `Items` xassəsi ilə müraciət etmək olar.

`listBox` komponentinin ən çox istifadə edilən hadisəsi `SelectedIndexChanged` hadisəsidir. Bu hadisə seçilmiş indeksli element dəyişdirildikdə baş verir. Komponent üzərində mausun düyməsini iki dəfə basdıqda avtomatik olaraq (susmaya görə) bu hadisə emaledicisi yaradılır.

`listBox` komponenti ilə işlədikdə aşağıdakı metodlardan istıfadə etmək lazım gələ bilər:

❖ `ClearSelected` – bütün elementlər üçün seçməni ləğv edir;

❖ `GetSelected` – əgər göstərilmiş indeksli element seçilərsə, `true` qiyməti qaytarır;

❖ `Add()` – arqument kimi verilmiş element siyahıya əlavə edilir;

❖ `Remove()`– arqumenti sətir kimi verilmiş element siyahıdan silinir;

❖ `RemoveAt()` – arqumenti indeks kimi verilmiş element siyahıdan silinir;

❖ `Insert()` – siyahının arqument ilə göstərilən hissəsinə mətnin əlavə edilməsi, məsələn, `Insert(1, "Riyaziyyat")` metodu ilə siyahıda 2-ci mövqeyə `"Riyaziyyat"` sətri əlavə edilir;

❖ `Count()` – siyahıda elementlərin sayını müəyyən edir. Bu metoda qiymət verilə bilməz;

❖ `Clear()` – siyahı silinir.

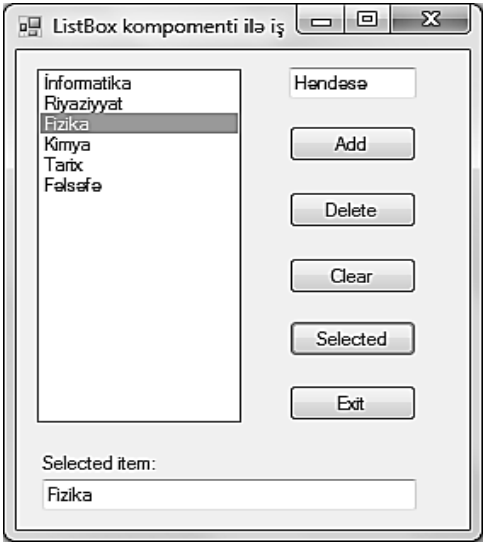
Misal. `textBox` komponentindən `listBox` komponentinə elementlərin əlavə edilməsi və onlar üzərində əməliyyatlar.

Forma üzərinə `listBox`, iki ədəd `textBox`, beş ədəd `button` və `label` komponentləri yerləşdirib onları şəkil 5.2–də göstərilən qaydada nizama salın.

Məsələnin mahiyyəti ondan ibarətdir ki, biz `textBox1` komponentində mətn yığacağıq və `button1` (`Add`) düyməsini basdıqda həmin mətn `listBox1` komponentinə yerləşdiriləcəkdir. `listBox1` komponentində seçilmiş element `button2` (`Delete`) düyməsi ilə silinəcək, `button5` (`Selected`) düyməsi ilə `textBox2` komponentində təsvir etdiriləcək, `button3` (`Clear`) düyməsini basdıqda `listBox1` komponentindən bütün

elementlər silinəcək və, nəhayət, `button4` düyməsini basdıqda proqramdan çıxış baş verəcəkdir.

`button1` düyməsi üçün `Click` hadisə emaledicisinin mətni belədir:



Şəkil 5.2. Forma üzərində komponentlərin yerləşdirilməsi

```
private void button1_Click(object
                sender, EventArgs e)
{
if (textBox1.Text != "")
   listBox1.Items.Add(textBox1.Text);
   textBox1.Clear();
   textBox1.Focus();
}
```

Burada, siyahıya boş sətrin daxil edilməməsi üçün onun boş olub-olmaması yoxlanır (`if (textBox1.Text != "")`. Sonra `Add`() metodu ilə `textBox1` komponentinin məzmunu

listBox1 komponentinə daxil edilir, textBox1 komponentinin məzmunu pozulur və bu komponent yenidən fokus alir.

button2 düyməsi üçün Click hadisə emaledicisinin mətni belədir:

```
private void button2_Click(object
                sender, EventArgs e)
{
if (listBox1.SelectedIndex != -1)
   listBox1.Items.RemoveAt(
   listBox1.SelectedIndex);
}
```

Bu kodlar vasitəsi ilə elementin seçilib-seçilməməsi yoxlanır (if(listBox1.SelectedIndex!=-1)) və element seçilmişdirsə, o RemoveAt() metodu ilə pozulur.

button3 və button4 düymələri üçün Click hadisə emaledicisinin mətni belədir:

```
private void button3_Click(object
                sender, EventArgs e)
{
listBox1.Items.Clear();
}

private void button4_Click(object
                sender, EventArgs e)
{
  Application.Exit();
}
```

Göründüyü kimi, burada müvafiq olaraq Clear() və Exit() metodları tətbiq edilmişdir.

button5 düyməsi üçün Click hadisə emaledicisinin mətni belədir:

```
private void button5_Click(object
```

```
                sender, EventArgs e)
{
if (listBox1.SelectedIndex != -1)
  textBox2.Text =
    listBox1.SelectedItem.ToString();
}
```

Burada isə, əgər siyahıda element seçilmişdirsə, o `textBox2` komponentinə yerləşdirilir. Bunun üçün

```
textBox2.Text =
     listBox1.SelectedItem.ToString();
```

kodundan istifadə edilmişdir.

comboBox idarəetmə elementi

`comboBox` komponenti mətn sahəsi və açılan siyahının kombinasiyasından ibarətdir. `comboBox` komponentində yalnız bir elementi seçmək olar. İstifadəçi siyahıdan element seçdikdə o mətn sahəsində təsvir olunur. `comboBox` elementinin üstün cəhəti ondan ibarətdir ki, forma üzərində çox az sahə tutur, belə ki, formada həmişə cəmi bir element təsvir olunur. Mənfi cəhəti isə ondan ibarətdir ki, bütün elementlərə baxmaq üçün, `listBox` komponentindən fərqli olaraq, bütün siyahı açılmalıdır.

Açılan siyahının elementləri, `listBox` komponentində olduğu kimi, `Items` xassəsində verilir. Siyahıya elementləri `Items` xassəsinin qarşısındakı üç nöqtə təsvirli düyməni basmaqla açılan `String Collection Editor` (`Редактор коллекции строк`) pəncərəsindən daxil etmək olar.

`comboBox` komponentinin əsas xassələri bunlardır:

❖ `DropDownStyle` – komponentin görünüşünü (tipini) müəyyən edir. Bu xassə aşağıdakı qiymətlərdən birini ala bilər:

- `Simple` — sadə tip, yəni daxiletmə sahəsi və elementlərin siyahısı eyni zamanda görünür;

- `DropDown` — aşağıya istiqamətlənmiş ox üzərində mausun düyməsini basmaqla siyahını açа bilərsiniz və ya klaviaturadan istənilən mətni daxil edə bilərsiniz (susmaya görə bu qiymət verilmişdir);
- `DropDownList` – əvvəlki tipdən fərqi ondadir ki, klaviaturadan mətni daxil etmək olmaz, yalnız açılan siyahıdan element seçmək olar;

❖ `MaxDropDownItems` — açılan siyahıda görünən maksimal elementlərin sayı;

❖ `Text` – redaktə sahəsində olan mətn (`DropDown` və `Simple` tipli komponentlər üçün);

❖ `Items` – siyahının elementləri – sətirlər kolleksiyası;

❖ `Count` – siyahıda elementlərin sayı;

❖ `Sorted` – yeni element daxil edildikdən sonra siyahının əlifba sırası ilə düzülməsi (`Sorted=true` olduqda siyahı əlifba sırası ilə yerləşir);

❖ `Size` – siyahı oblastını nəzərə almamaqla (`DropDown` və `DropDownList` tipli komponentlər üçün) və nəzərə almaqla (`Simple` tipli komponent üçün) komponentin ölçüsü;

❖ `Font` – daxiletmə sahəsi və siyahının elementlərini təsvir etmək üçün istifadə edilən şrift;

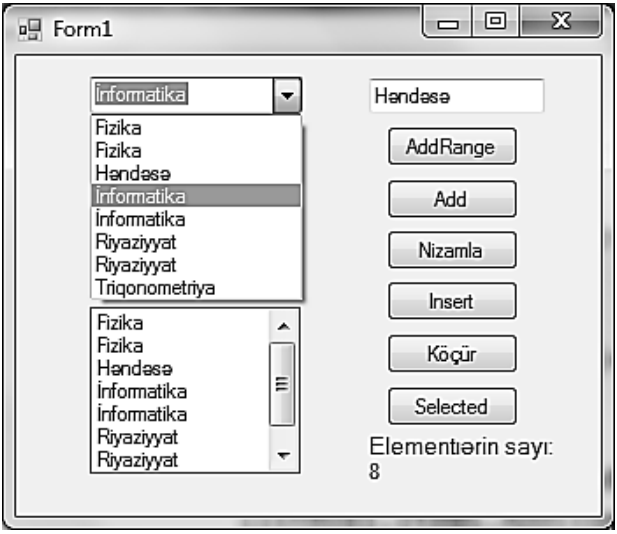
❖ `SelectedIndex` – siyahıda seçilən elementin nömrəsi. Əgər element seçilməmişdirsə, onda bu xassənin qiyməti `-1`-ə bərabər olur;

❖ `SelectedItem` – seçilmiş elementə istinad qaytarır. Bu xassə ilə işləməzdən əvvəl yaxşı olar ki, onun qiyməti yoxlanılsın. Əgər açılan siyahıda heç bir element seçilməzsə, onda bu xassənin qiyməti `null` olur.

`comboBox` komponentinin ən çox istifadə edilən hadisəsi `SelectedIndexChanged` hadisəsidir. Bu hadisə seçilmiş indeksi dəyişdirdikdə, məsələn, başqa elementi seçdikdə baş verir.

Misal. Siyahı elementləri üzərində əməliyyatlar.

Forma üzərinə altı ədəd `button` düymələri, iki ədəd `label` yazıları, `textBox`, `listBox` və `comboBox` komponentləri yerləşdirin (şəkil 5.3).



Şəkil 5.3. Siyahı elementləri üzərində əməliyyatlar

`label1` komponentinin `Text` xassəsinə "`Elementlərin sayı:`" mətnini yazın. `label2` yazısında isə `comboBox` siyahısındakı elementlərin sayı təsvir ediləcəkdir. `button1` (`AddRange`) düyməsini basdıqda elementlərdən massiv yaradılacaq və o dərhal `comboBox` elementinə yüklənəcəkdir. Bu kod belədir:

```
comboBox1.Items.AddRange(new String[]
        { "İnformatika", "Riyaziyyat",
                         "Fizika" });
```

Yəni, üç elementdən ibarət sətir tipli massiv yaradılır və sistem mötərizələri daxilində massivin elementləri sadalanır. `button1` (`AddRange`) düyməsini hər dəfə basdıqda siyahıya

üç eyni element əlavə ediləcəkdir. button2 (Add) düyməsi textBox komponentində yığılmış mətni comboBox siyahısına əlavə edir. button3 (Sorted) düyməsi ilə siyahıdakı elementlər əlifba sırası ilə düzülür. Bu aşağıdakı kodla yerinə yetirilir:

```
comboBox1.Sorted=true;
```

button4 (Insert) düyməsi ilə siyahının ikinci mövqeyinə "Triqonometriya" mətni əlavə edilir. Bunun üçün Insert() metodundan istifadə edilir:

```
comboBox1.Items.Insert(
          1,"Triqonometriya");
```

button5 (Köçür) düyməsini basdıqda comboBox siyahısının bütün elementləri listBox siyahısına köçürülür. Bu məqsədlə yazılmış kodlar belədir:

```
for (int i=0;
      i<=comboBox1.Items.Count-1;i++)
listBox1.Items.Add(
      comboBox1.Items[i].ToString());
```

Burada, listBox1 komponentinin Items nömrəli sətirlərinə for dövrü daxilində comboBox1 komponentinin i-ci sətirləri daxil edilir. Items ədəd tipli olduğuna görə o ToString() metodu ilə sətir tipinə çevrilir. for dövrü isə elementlərin sayı (Count) qədər təkrarlanır.

button6 (Selected) düyməsini basdıqda combBox1 komponentində seçilmiş element Show metodu ilə ekranda təsvir olunur. Nəhayət, sonuncu emaledici combBox1 komponentində elementlərin sayını label2 komponentində əks etdirir. Bu məqsədlə ayrıca bir hadisə emaledicisi yaradılmalıdır. Çünki, comboBox siyahısına müxtəlif düymələr vasitəsi ilə elementlər əlavə edilir. Əgər, biz elementlərin sayını göstərən kodları bu düymələrdən birinə uyğun hadisə emaledicilərı daxilində yazsaydıq, düzgün nəticə ala bilməzdik. Ona görə də biz comboBox komponenti üçün

`MouseClick` hadisə emaledicini yaratmışıq. Bu hadisə komponent daxilində mausun düyməsini basdıqda baş verdiyi üçün, komponent daxilində mausun düyməsini basdıqda `label2` komponentində elementlərin sayı (`Count` xassəsi) göstəriləcəkdir. Həmin kod belədir:

```
label2.Text = Convert.ToString(
              comboBox1.Items.Count);
```

Bu proqramın tam mətni belədir:

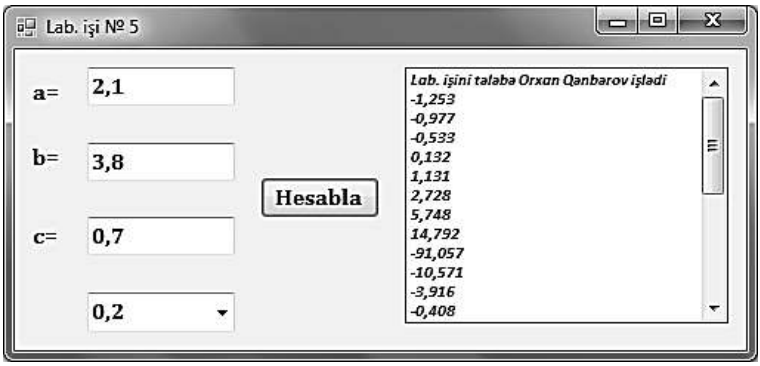
```
namespace comboBoxTest
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}

private void button1_Click(object
              sender, EventArgs e)
{
comboBox1.Items.AddRange(new String[]
   { "İnformatika", "Riyaziyyat",
                     "Fizika" });
}

private void button2_Click(object
              sender, EventArgs e)
{
comboBox1.Items.Add(textBox1.Text);
textBox1.Clear();
textBox1.Focus();
}

private void button3_Click(object
```

```
                sender, EventArgs e)
{
comboBox1.Sorted=true;
}

private void button4_Click(object
                sender, EventArgs e)
{
comboBox1.Items.Insert(1,
          "Triqonometriya");
}

private void comboBox1_MouseClick(object
                sender, MouseEventArgs e)
{
label2.Text =
     Convert.ToString(
          comboBox1.Items.Count);
}

private void button5_Click(object
                sender, EventArgs e)
{
listBox1.Items.Clear();
for (int i=0;
       i<=comboBox1.Items.Count-1; i++)
listBox1.Items.Add(
       comboBox1.Items[i].ToString());
}

private void button6_Click(object
                sender, EventArgs e)
{
if (comboBox1.SelectedItem != null)
```

```
    MessageBox.Show("Seçilmiş element: "+
     comboBox1.SelectedItem.ToString());
  else MessageBox.Show("Element
                  seçilməmişdir");
  }}}
```

Misal. *x* arqumenti *dx* addımı ilə x_0 qiymətindən x_k qiymətinə qədər dəyişdikdə $f(x) = \frac{bc}{\cos(x)} + (a-b)^3 e^x$ funksiyasının cədvəl qiymətlərini `listBox` komponentində təsvir etdirin. Burada, *a*, *b* və *c* sabitlərdir. Məsələni həll etdikdə x_0, x_k və *dx* qiymətlərini `comboBox` komponentindən, *a*, *b* və *c* sabitlərinin `textBox` komponentlərindən daxil etmək, nəticələri isə `listBox` komponentində təsvir etdirmək tələb olunur.

Əvvəlcə layihənin formasını şəkil 5.4-ə uyğun tərtib edək.



Şəkil 5.4. 5 №-li laboratoriya işinin nəticəsi

```
using System;
using System.Windows.Forms;

namespace lab5
{
public partial class Form1 : Form
{
```

```
public Form1()
{
  InitializeComponent();
}

private void button1_Click(object sender,
                           EventArgs e)
{
  double x, x0, xk, dx, a, b, c, f;
  a = Convert.ToDouble(textBox1.Text);
  b = Convert.ToDouble(textBox2.Text);
  c = Convert.ToDouble(textBox3.Text);
  x0= Convert.ToDouble(
              comboBox1.Items[0]);
  xk = Convert.ToDouble(
              comboBox1.Items[1]);
  dx = Convert.ToDouble(
              comboBox1.Items[2]);
  x = x0;
  listBox1.Items.Add("Lab. işini tələbə
              Orxan Qənbərov işlədi");

  while(x<=xk)
  {
    f = (b * c) / Math.Cos(x) +
         Math.Pow((a-b), 3)+Math.Exp(x);
    f = Math.Round(f, 3);
    listBox1.Items.Add(f.ToString());
    x += dx;
  }
} } }
```

Yoxlama sualları

1. Siyahılara elementlər necə daxil edilir?

2. `SelectionMode` xassəsi nə üçündür və hansı qiymətləri ala bilər?

3. `SelectedItem` xassəsi nə üçündür?

4. `SelectedItems` xassəsi nə üçündür?

5. `SelectedIndex` xassəsi nə üçündür?

6. `listBox` komponentinin ən çox istifadə edilən hadisəsi hansıdır?

7. `listBox` komponenti ilə işləmək üçün hansı metodları bilirsiniz?

8. Siyahıda elementlərin sayini necə tapırıq?

9. `comboBox` komponentinin `listBox` komponentindən fərqi nədədir?

10. Siyahıların sətirlər kolleksiyasına hansı xassə ilə müraciət edilir?

11. `DropDownStyle` xassəsi nə üçündür və hansı qiymətləri ala bilər?

12. Redaktə sahəsində olan mətnə hansı xassə ilə müraciət edilir?

13. `comboBox` komponentinin ən çox istifadə edilən hadisəsi hansıdır?

14. Siyahının elementləri əlifba sırası ilə hansı xassə ilə düzülür?

15. `comboBox` komponentində neçə element seçmək olar?

Laboratoriya işlərinə aid tapşırıqların variantları

1-20-ci variantlarda verilmiş tapşırıqlar üçün şəkil 5.4-də göstərilmiş əlavəyə oxşar əlavə hazırlayın.

1) $y = 10^{-2} bc / x + \cos\sqrt{a^3 x}$,
$x_0 = -1.5; x_k = 3.5; dx = 0.5;$
$a = -1.25; b = -1.5; c = 0.75;$

2) $y = 1.2(a-b)^3 e^{x^2} + x$,
$x_0 = -0.75; x_k = -1.5; dx = -0.05;$
$a = 1.5; b = 1.2;$

3) $y = 10^{-1} ax^3 tg(a - bx)$,
$x_0 = -0.5; x_k = 2.5; dx = 0.05;$
$a = 10.2; b = 1.25;$

4) $y = ax^3 + \cos^2(x^3 - b)$,
$x_0 = 5.3; x_k = 10.3; dx = 0.25;$
$a = 1.35; b = -6.25;$

5) $y = x^4 + \cos(2 + x^3 - d)$,
$x_0 = 4.6; x_k = 5.8; dx = 0.2;$
$d = 1.3;$

6) $y = x^2 + tg(5x + b/x)$,
$x_0 = -1.5; x_k = -2.5; dx = -0.5;$
$b = -0.8;$

7) $y = 9(x + 15\sqrt{x^3 + b^3})$,
$x_0 = -2.4; x_k = 1; dx = 0.2;$
$b = 2.5;$

8) $y = 9x^4 + \sin(57.2 + x)$,
$x_0 = -0.75; x_k = -2.05; dx = -0.2;$

9) $y = 0.0025bx^3 + \sqrt{x + e^{0.82}}$,
$x_0 = -1; x_k = 4; dx = 0.5;$
$b = 2.3;$

10) $y = x \cdot \sin(\sqrt{x + b - 0.0084})$,
$x_0 = -2.05; x_k = -3.05; dx = -0.2;$
$b = 3.4;$

11) $y = x + \sqrt{|x^3 + a - be^x|}$,
$x_0 = -4; x_k = -6.2; dx = -0.2;$
$a = 0.1;$

12) $y = 9(x^3 + b^3)tgx$,
$x_0 = 1; x_k = 2.2; dx = 0.2;$
$b = 3.2;$

13) $y = |x - b|^{1/2} / |b^3 - x^3|^{3/2} + \ln|x - b|$,
$x_0 = -0.73; x_k = -1.73; dx = -0.1;$
$b = -2;$

14) $y = (x^{5/2} - b)\ln(x^2 + 12.7)$,
$x_0 = 0.25; x_k = 5.2; dx = 0.3;$
$b = 0.8;$

15) $y = 10^{-3}|x|^{5/2} + \ln|x + b|$,
$x_0 = 1.75; x_k = -2.5; dx = -0.25;$
$b = 35.4;$

16) $y = 15.28|x|^{-3/2} + \cos(\ln|x| + b)$,
$x_0 = 1.23; x_k = -2.4; dx = -0.3;$
$b = 12.6;$

17) $y = 0.00084(\ln|x|^{5/4} + b)/(x^2 + 3.82)$,
$x_0 = -2.35; x_k = -2; dx = 0.05;$
$b = 74.2;$

18) $y = 0.8 \cdot 10^{-5}(x^3 + b^3)^{7/6}$,
$x_0 = -0.05; x_k = 0.15; dx = 0.01;$
$b = 6.74;$

19) $y = (\ln(\sin(x^3 + 0.0025)))^{3/2} + 0.8 \cdot 10^{-3}$,
$x_0 = 0.12; x_k = 0.64; dx = 0.2;$

20) $y = a + x^{2/3}\cos(x + e^x)$,
$x_0 = 5.62; x_k = 15.62; dx = 0.5;$
$a = 0.41$

21. Bir `comboBox` komponentinə dekanlıqların, digərinə isə kafedraların adlarını daxil edin. `Kafedra` adlı düyməni basdıqda seçilmiş kafedranın adı `listBox1` komponentinə,

Dekanlıq adlı düyməni basdıqda isə seçilmiş dekanlığın adı listBox2 komponentinə daxil edilsin.

22. comboBox komponentinə yeni element əlavə etməyə, mövcud elementi silməyə imkan verən əlavə layihələndirin. Hər əməliyyatdan sonra elementlərin sayını müəyyən edin.

23. İki listBox komponentinin birincisində tədris olunan fənlərin siyahısı verilmişdir, ikincisi isə bosdur. Fənləri təkrarlamadan birincidən ikinciyə əlavə edin. İkinci listBox-dan əlavə edilmiş fənləri silmək və yerdə qalan fənlərin sayını tapmaq mümkün olsun.

24. İki comboBox komponentlərinin hər birində 10 müxtəlif ədədlər var. Komponentlərin müvafiq elementlərinin cəmini hesablayıb textBox komponentində göstərin.

25. comboBox komponentində yerləşən ədədləri cəmləyib textBox komponentində göstərin.

26. listBox komponentində yerləşən siyahıdan seçilmiş ədədlərin hasilini textBox komponentində göstərin.

27. comboBox və listBox komponentlərindən seçilmiş ədədlərin hasilini label komponentində göstərin.

28. comboBox komponentində sətirlər sıfır və vahidlərdən ibarətdir. Seçilmiş sətirdəki sıfır və vahidlərin sayını tapın.

29. listBox komponentinə Toplama, Çıxma, Vurma, Bölmə, Qüvvətə yüksəltmə və Bölmədə qalıq adlı sətirlər daxil edin. Siyahıdan həmin sətirləri seçdikdə onların indekslərinə müvafiq olaraq textBox1 və textBox2 komponentlərindən daxil edilmiş ədədlər üzərində müvafiq əməliyyatlar icra edilərək textbox3 komponentində təsvir edilsin.

30. listBox komponentində yerləşən siyahı simvollar və ədədlərdən ibarətdir. Seçilmiş sətirdəki ədədləri comboBox komponentinə əlavə edin.

LABORATORİYA İŞİ № 6.
STANDART DÜYMƏ İLƏ İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən `button` düymə elementlərinin xassələrini və əlavələrin layihələndirilməsində onun tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat

`Button` düymə komponenti kompüter əlavələrinin yaradılmasında ən çox istifadə edilən komponentdir. Bütün komponentlərin, o cümlədən bu komponentin də adı `Name` xassəsi ilə müəyyənləşdirilir. Susmaya görə onun adı (`Name` xassəsinin qiyməti) `buttonN`-dir, burada `N` komponentin nömrəsidir (`button1`, `button2` və s.). İstifadəçi bu adı dəyişdirə bilər. Onun dıgər xassəsi `Text` xassəsidir. Bu xassə düymə üzərində təsvir olunan yazını müəyyən edir. `Button` komponentinin digər əsas xassələri aşağıdakılardır:

❖ `TextAlign` – düymənin üzərində təsvir olunan yazının vəziyyətini müəyyən edir. Yazının doqquz vəziyyəti mümkündür:

- `TopLeft` – mətn komponentin yuxarı sol tərəfində yerləşir;
- `TopRight` – mətn komponentin yuxarı sağ tərəfində yerləşir;
- `TopCenter` – mətn komponentin yuxarı mərkəzində yerləşir;
- `MiddleLeft` – mətn komponentin sol mərkəzində yerləşir;
- `MiddleRight` – mətn komponentin sağ mərkəzində yerləşir;

- `MiddleCenter` – mətn komponentin mərkəzində yerləşir;
- `BottomLeft` – mətn komponentin aşağı sol tərəfində yerləşir;
- `BottomRight` – mətn komponentin aşağı sağ tərəfində yerləşir;
- `BottomCenter` – mətn komponentin aşağı mərkəzində yerləşir;

❖ `FlatStyle` – düymənin görünüşünü müəyyən edir:

- `Standard` – standart görünüşlü;
- `Flat` – müstəvi formasında;
- `Popup` – "peyda" olan;
- `System` – sistem düyməsi;

❖ `Location` – forma üzərində düymənin vəziyyətini müəyyən edir. Bu xassənin `X` parametri düymənin sol kənarından formanın sol kənarına qədər olan məsafəni, `Y` parametri isə düymənin yuxarı kənarından formanın yuxarı kənarına (başlıqdan aşağı sərhəddə) qədər olan məsafəni müəyyən edir;

❖ `Size` – düymənin ölçüsünü müəyyən edir; `Size.Width` – düymənin enini, `Size.Height` isə hündürlüyünü təyin edir;

❖ `Font` – düymənin üzərində təsvir olunan yazının şriftini müəyyən edir;

❖ `ForeColor` – düymənin üzərində təsvir olunan yazının rəngini müəyyən edir;

❖ `Enabled` – düymənin aktivliyini müəyyən edir. Bu xassəyə `false` qiyməti verdikdə düymə solğun rəngdə olur və basıla bilmir;

❖ `Visible` – bu xassəyə `true` qiyməti verdikdə (susmaya görə bu belədir) düymə görünür, `false` qiyməti verdikdə isə düymə gizlədilir;

❖ Cursor – mausun göstəricisini düymə üzərinə gətirdikdə onun görünüşünü müəyyən edir. Bu xassəyə çoxlu qiymətlər vermək mümkündür, susmaya görə Default qiyməti təyin edilmişdir, yəni, mausun göstəricisi standart formadadır;

❖ Image – düymənin səthində şəkli müəyyən edir;

❖ ImageAlign – düymənin səthində şəklin vəziyyətini müəyyən edir. Bu xassənin qiymətləri TextAlign xassəsinin qiymətləri ilə eynidir.

❖ ToolTip – mausun göstəricisini düymə üzərinə gətirdikdə onun göstəricisinin yanında peyda olan mətni bildirir. Bu xassənin işləməsi üçün forma üzərinə ToolTip komponentini yerləşdirmək lazımdır.

Düymənin üzərinə şəklin yerləşdirilməsi qaydasını öyrənək. Bunun üçün forma üzərinə button komponenti yerləşdirib Properties (Свойства) pəncərəsində onun Image xassəsinin qarşısındakı üç nöqtə təsvirli düyməni basaq. Bu zaman ekrana Select Resource (Выбор ресурса) pəncərəsi çıxacaqdır. Bu pəncərədə iki dəyişdirici vardır: Local resource (Локальный ресурс) və Project resource file (Файл ресурсов проекта). Birinci dəyişdiricini seçdikdə düymə üzərində yerləşdiriləcək şəkil formanın resurslar faylında yerləşdiriləcəkdir. Bu o deməkdir ki, bu şəkli digər düymələrdə və ya menyularda istifadə etmək istəsək, biz onu təkrarən yükləməliyik, bu isə proqramın həcminin artmasına gətirir. Project resource file (Файл ресурсов проекта) dəyişdiricisini qoşduqda isə şəkil Properties qovluğunun saxlandığı yerdə yerləşən resurslar faylında yerləşəcək və biz onu təkrarən yükləmədən digər düymə və menyularda istifadə edə bilərik.

Düymə üzərində şəkil susmaya görə yazının altında yerləşdirilir. Şəklin vəziyyətini TextImageRelation

xassəsi idarə edir. Bu xassə üçün aşağıdakı qiymətlərdən birini seçmək olar:

- `Overlay` – mətn şəkildən yuxarıda yerləşəcək;
- `ImageAboveText` – mətn şəklin altında yerləşəcək;
- `TextAboveImage` – mətn şəklin üstündə yerləşəcək;
- `ImageBeforeText` – şəkil mətndən solda yerləşəcək;
- `TextBeforeImage` – mətn şəkildən solda yerləşəcək.

Düymələr üçün tətbiq edilən ən məşhur hadisə `Click` hadisəsidir.

Misal. Mausdan qaçan düymə.

Biz elə proqram yazacağıq ki, mausu düyməyə yönəltdikdə o, mausdan qaçsın. Bunun üçün forma üzərində yeganə komponent – `button1` yerləşdirərək, onun üçün `MouseMove` hadisəsini yaradaq. Bu məsələnin proqramı aşağıdakı kodlardan ibarət olacaqdır:

```
namespace QACHAN_DUYME
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}
private void button1_MouseMove(object
             sender, MouseEventArgs e)
{
Random r = new Random();
int index = r.Next(1, 5);

switch (index)
{
  case 1:
  button1.Left = button1.Left +
```

```
            button1.Width;  break;
    case 2:
    button1.Left = button1.Left -
           button1.Width;  break;
    case 3:
    button1.Top = button1.Top +
           button1.Height;  break;
    case 4:
    button1.Top = button1.Top -
            button1.Height;  break;
  }
  if (button1.Left < 0) button1.Left = 0;

  if (button1.Left + button1.Width >
       this.Width) button1.Left =
       this.Width -  button1.Width;

  if (button1.Top < 0) button1.Top = 0;

  if (button1.Top + button1.Height >
       this.Height) button1.Top =
       this.Height - button1.Height;
  }}}
```

Misal. Mausun sol və sağ düymələrinin basılması.

Forma üzərinə `label` və `button` komponentləri yerləşdirək. Elə layihə yaradaq ki, mausun sol düyməsini basdıqda `label` komponentində hər hansı bir mətn, sağ düyməsini basdıqda isə digər mətn təsvir edilsin. Mausun hansı düyməsinin basıldığını bilmək üçün `Control` sinfinin `MouseButtons` statik xassəsindən istifadə edilir. Məsələni həll etmək üçün `button1` düyməsi üçün `MouseDown` hadisə emaledicisini yaradaq. `label` komponentinin xassələrini kodlarla təyin edək. Proqramın kodları aşağıda göstərilmişdir:

```
namespace MAUSUN_DUYMESI_2
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
label1.TextAlign =
       ContentAlignment.MiddleRight;

label1.Font = new Font("Tahoma", 20.0F);

label1.AutoSize = false;
label1.AllowDrop = true;
}

private void button1_MouseDown(object
            sender, MouseEventArgs e)
{
if (Control.MouseButtons ==
          MouseButtons.Left)
{
label1.ForeColor = Color.Red;
label1.Text = "AZƏRBAYCAN RESPUBLİKASI
              TƏHSİL NAZİRLİYİ";
}
if (Control.MouseButtons ==
          MouseButtons.Right)
{
label1.Text = "AZƏRBAYCAN TEXNİKİ
               UNİVERSİTETİ";
label1.ForeColor = Color.Blue;
}}}}
```

Yoxlama sualları

1. Düymənin üzərindəki mətn hansı xassə ilə müəyyən edilir?
2. Düymənin üzərində təsvir olunan yazının vəziyyətini hansı xassə müəyyən edir?
3. `TextAlign` xassəsi hansı qiymətləri alır?
4. Düymənin görünüşünü müəyyən edən xassə hansıdır?
5. Düymənin koordinatlarını müəyyən edən xassələr hansılardır?
6. Düymənin ölçülərini müəyyən edən xassələr hansılardır?
7. Düymənin aktivliyini müəyyən edən xassə hansıdır?
8. Düymə hansı xassə ilə gizlədilir?

Laboratoriya işlərinə aid tapşırıqların variantları

1. Forma üzərinə üç `button`, iki `textBox` komponentləri yerləşdirin. `button1` düyməsinin sağ klavişini basdıqda `textBox` komponentləri yox olsun, sol klavişini basdıqda isə iki `button` düyməsi yox olsun, mətn sahələri isə görünsün.
2. Forma üzərindəki düymənin üzərində mausun sol düyməsini basdıqda forma şəffaf olsun və bu barədə ismarıc göndərilsin, sağ düyməsini basdıqda forma adi vəziyyətinə qayıtsin və müvafiq ismarıc göndərilsin.
3. Forma üzərindəki `button1` düyməsini basdıqda digər düymə öz koordinatlarını dəyişsin.
4. Forma üzərindəki `button1` düyməsini basdıqda digər düymə öz koordinatlarını dəyişməklə üfqi istiqamətdə titrəməyə başlasın.
5. Forma üzərindəki `button1` düyməsini basdıqda digər düymə öz koordinatlarını dəyişməklə şaquli istiqamətdə titrəməyə başlasın.

6. Forma üzərindəki `button1` düyməsini basdıqda digər düymə öz koordinatlarını dəyişməklə düzbucaqlının konturları üzrə titrəməyə başlasın. Titrəməni müşahidə edə bilməyiniz üçün `System.Threading.Thread.Sleep(10);` kodundan istifadə edin.

7. Forma üzərindəki `button1` düyməsini basdıqda Forma tam ekran boyu açılsın və digər düymə öz koordinatlarını (`0,0`) mövqeyindən (`0,Formanın sonu`) mövqeyinə dəyişdirsin. Prosesi müşahidə edə bilməyiniz üçün `System.Threading.Thread.Sleep(10);` kodundan istifadə edin.

8. Forma üzərindəki `button1` düyməsini basdıqda Forma tam ekran boyu açılsın və digər düymə öz koordinatlarını (`0,0`) mövqeyindən (`Formanın sonu,0`) mövqeyinə dəyişdirsin. Prosesi müşahidə edə bilməyiniz üçün `System.Threading.Thread.Sleep(10);` kodundan istifadə edin.

9. Forma üzərindəki `button1` düyməsini basdıqda Forma tam ekran boyu açılsın və digər düymə öz koordinatlarını (`0,0`) mövqeyindən (`Formanın sonu, Formanın sonu`) mövqeyinə diaqonal boyunca dəyişdirsin.

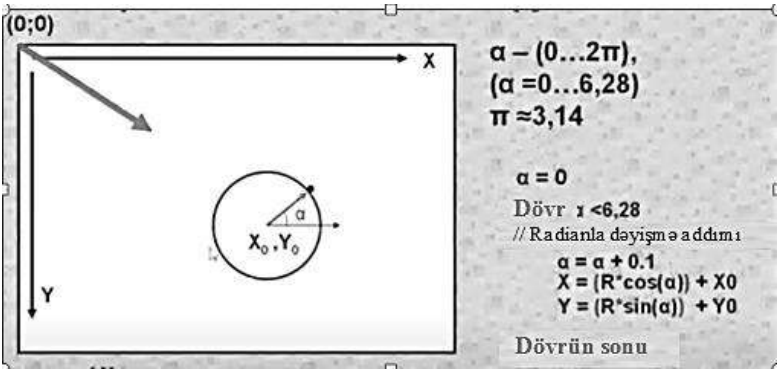
10. Forma üzərindəki `button1` düyməsini basdıqda Forma tam ekran boyu açılsın və digər düymə çevrə üzrə hərəkət etsin. Məsələni proqramlaşdırdıqda şəkil 6.1-ə əsaslanın və x_0=500, y_0=400 və R=200 qəbul edin. Prosesi müşahidə etmək üçün `System.Threading.Thread.Sleep(50);` kodundan istifadə edin.

11. İlkin verilənlərin başlanğıc qiymətlərini özünüz seçməklə 10 №-li variantdakı məsələni həll edin.

12. İlkin verilənlərin başlanğıc qiymətlərini özünüz seçməklə 10 №-li variantdakı məsələni həll edin.

13. 10 №-li variantdakı məsələni saat əqrəbinin əksinə fırlatmaqla həll edin.

14. 10 №-li variantdakı məsələni yarım çevrə boyunca fırlatmaqla həll edin.



Şəkil 6.1. 10 №-li tapşırığın həndəsi izahı

15. 10 №-li variantdakı məsələni yarım çevrə boyunca saat əqrəbinin əksinə fırlatmaqla həll edin.

16. 10 №-li variantdakı məsələni 90 dərəcəli qövs boyunca fırlatmaqla həll edin.

17. 10 №-li variantdakı məsələni 90 dərəcəli qövs boyunca saat əqrəbinin əksinə fırlatmaqla həll edin.

18. 10№-li variantdakı məsələni 270 dərəcəli qövs boyunca fırlatmaqla həll edin.

19. 10 №-li variantdakı məsələni 270 dərəcəli qövs boyunca saat əqrəbinin əksinə fırlatmaqla həll edin.

20. 10 №-li variantdakı məsələni 45 dərəcəli qövs boyunca fırlatmaqla həll edin.

21. 10 №-li variantdakı məsələni 45 dərəcəli qövs boyunca saat əqrəbinin əksinə fırlatmaqla həll edin.

22. Forma üzərindəki `button1` düyməsini basdıqda `Form2` forması ekranın sol yuxarı küncündə görünsün, `button2` düyməsini basdıqda isə `Form2` yox olsun.

23. Forma üzərindəki `button1` düyməsini basdıqda digər forma öz koordinatlarını dəyişməklə üfqi istiqamətdə titrəməyə başlasın.

24. Forma üzərindəki `button1` düyməsini basdıqda digər forma öz koordinatlarını dəyişməklə şaquli istiqamətdə titrəməyə başlasın.

25. Forma üzərindəki `button1` düyməsini basdıqda digər forma öz koordinatlarını dəyişməklə düzbucaqlının konturları üzrə titrəməyə başlasın. Prosesi müşahidə etmək üçün `System.Threading.Thread.Sleep(10);` kodundan istifadə edin.

26. Forma üzərindəki `button1` düyməsini basdıqda digər forma öz şəffaflığını müntəzəm dəyişərək ilkin şəffaflığına qayıtsın. Prosesi müşahidə etmək üçün `System.Threading.Thread.Sleep(1000);` kodundan istifadə edin. `Form2`-nin fonuna qırmızı rəng müəyyənləşdirin.

27. Forma üzərindəki `button1` düyməsinin sağ düyməsini basdıqda birinci siyahıda həmin düymənin, digər siyahıda isə Formanın koordinatları və ölçüləri təsvir edilsin.

28. Forma üzərindəki `button1` düyməsini basdıqda digər düymə koordinat başlanğıcından 45 dərəcəli bucaq altında düz xətt boyunca hərəkət etsin.

29. Forma üzərindəki `button1` düyməsinin sağ klavişini basdıqda `listBox1` komponenti yox olsun, sol klavişini basdıqda isə `listbox1` komponenti görünsün, `combox1` isə yox olsun. Hər iki halda ismarıc göndərilsin.

30. Forma üzərindəki `button1` düyməsini basdıqda `button2` düyməsinin səthinin rəngi dəyişsin, onun üzərində "`Mən dəyişdim`" mətni yazılsın və Formanin eni dəyişsin.

LABORATORİYA İŞİ № 7.
DƏYİŞDİRİCİLƏRLƏ İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən `radioButton` və `checkBox` dəyişdiricilərinin xassələrini və əlavələrin layihələndirilməsində onların tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat
`radioButton` idarəetmə elementi

Bu komponent `checkBox` komponenti kimi iki vəziyyətdən – qoşulmuş və qoşulmamış vəziyyətlərindən birində ola bilər. Lakin `radioButton` dəyişdiriciləri, adətən, qruplarda birləşirlər və bu qruplarda istənilən zaman yalnız bir dəyişdirici qoşula bilər. Qrupda bir dəyişdiricini qoşduqda o biri dəyışdiricilərin qoşulması ləğv edilir.

`radioButton` komponentinin əsas xassələri bunlardır:

❖ `Checked` – komponentin qoşulmuş və ya qoşulmamış vəziyyətdə olmasını müəyyən edir;

Komponent qoşulduqda bu xassənin qiyməti `true`, qoşulmadıqda isə `false` olur (susmaya görə) ;

❖ `Text` – komponentin sağ tərəfində təsvir edilən mətn;

❖ `TextAlign` – təsvir edilmə sahəsində mətnin vəziyyətini müəyyən edir. Bu xassənin qiymətləri `button` düyməsinin eyniadlı xassəsinin qiymətləri ilə eynidir;

❖ `CheckAlign` – komponentin sahəsində düymənin (kiçik dairənin) vəziyyətini müəyyən edir. Onun aldığı qiymətlər `TextAlign` xassəsinin qiymətləri ilə eynidir.

❖ `Appearance` – susmaya görə bu xassəyə `Normal` qiyməti mənimsədilmişdir, ona görə də o ənənəvi formada təsvir edilir. Əgər bu xassəyə `Button` qiyməti versək, onda o

düymə şəklində təsvir ediləcəkdir. Bu zaman bayraq qoyulduqda düymə basılmış vəziyyətdə olur;

❖ `Enabled` – komponentin aktivliyini müəyyən edir. Ona `false` qiyməti verdikdə komponent aktiv olmur.

❖ `Visible` – bu xassəyə `true` qiyməti verdikdə (susmaya görə bu belədir) dəyişdirici görünür, `false` qiyməti verdikdə isə dəyişdirici gizlədilir;

❖ `FlatStyle` – komponentin görünüşünü müəyyən edir:

- `Standard` – standart görünüşlü;
- `Flat` – müstəvi formasında;
- `Popup` – "peyda" olan;
- `System` – sistem dəyişdiricisi;

❖ `Image` – komponentin sahəsində şəkli müəyyən edir;

❖ `ImageAlign` – komponentin sahəsində şəklin vəziyyətini müəyyən edir. Bu xassənin qiymətləri `TextAlign` xassəsinin qiymətləri ilə eynidir.

`radioButton` komponentinin əsas hadisəsi `CheckedChanged` hadisəsidir ki, o dəyişdiricini qoşduqda və ya qoşulmanı ləğv etdikdə yaranır.

Misal. `radiobutton` dəyişdiricilərinin idarəsi ilə qaçan sətirlərin yaradılması.

Biz qaçan sətirlərin necə yaradılmasını artıq bilirik. Bu misalda da həmin alqoritm təkrar ediləcəkdir. Lakin, bu dəfə biz bir `label` komponentində olan mətni `radiobutton` dəyişdiriciləri vasitəsi ilə müxtəlif istiqamətlərdə hərəkət etdirəcəyik. Həm də `radiobutton` dəyişdiricisinin bir sıra xassələrini öyrənəcəyik.

Beləliklə, forma üzərinə bir `label`, `timer` və iki `radiobutton` komponentləri yerləşdirək. `radiobutton1` komponentini seçib onun üzərinə hərəkət istiqamətini göstərən şəkil yerləşdirək. Əgər kompüterinizdə belə şəkil yoxdursa, onu Paint şəkil redaktoru ilə yaratmaq olar. `radiobutton1`

komponentini seçib `Image` xassəsi qarşısında mausun düyməsini basaraq onun sahəsinə şəkil yükləyin (şəkil 7.1). Sonra həmin komponentin `AutoSize` xassəsinə `false` qiyməti verərək şəkli onun sahəsində yerləşdirin, `Text` xassəsinin qiymətini silin. Şəkli kiçik dairədən sağ tərəfdə yerləşdirmək üçün `CheckAlign` və `ImageAlign` xassəsinin hər ikisinə `MiddleLeft` qiyməti təyin edin. Analoji əməliyyatları `radiobutton2` komponenti üçün təkrar edin, lakin `CheckAlign` və `ImageAlign` xassəsinin hər ikisinə `MiddleRight` qiyməti təyin edin. Proqramın nəticəsi şəkil 7.1-də, kodları isə aşağıda göstərilmişdir.

Şəkil 7.1. İstiqamətini dəyişən qaçan sətir

```
namespace QACHAN_SETIR_2
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
label1.Text = "  Azərbaycan Texniki
                    Universiteti";
timer1.Enabled = true;
}

private void timer1_Tick(object sender,
                      EventArgs e)
{
```

```
string txt = label1.Text;
if (radioButton1.Checked == true)
{
string txt1 = txt.Substring(1,
              txt.Length - 1);
string txt2 = txt.Substring(0, 1);
label1.Text = txt1 + txt2;
}

if (radioButton2.Checked == true)
{
string txt1 = txt.Substring(
              txt.Length - 1, 1);
string txt2 = txt.Substring(0,
               txt.Length - 1);
label1.Text = txt1 + txt2;
}}}}
```

Bu proqramda hansı dəyişdiricinin qoşulması (`if(radioButton1.Checked==true)`) kodu ilə müəyyən edilmişdir. Hazır layihədə növbə ilə dəyişdiriciləri qoşduqda göstərilmiş istiqamətlərdə mətn qaçacaqdır.

`checkBox` idarəetmə elementi

Bu komponent də `radioButton` komponenti kimi, iki vəziyyətdən birini seçməyə imkan verən dəyişdiricidir və ona bayraq da deyirlər. `checkBox` komponenti o zaman istifadə edilir ki, istifadəçi iki variantdan birini, məsələn, hə və ya yox seçməlidir. Xarici görünüşünə görə içərisi boş və ya bayraq qoyulmuş kiçik kvadratdan ibarətdir. İstifadəçi mausun düyməsini basdıqda onun daxilində bayraq əmələ gəlir, mausun düyməsini təkrarən basdıqda isə bayraq götürülür.

checkBox komponentinin ən çox istifadə edilən xassələri bunlardır:

❖ Text – komponentin sağ tərəfində təsvir edilən mətn;

❖ TextAlign – təsvir edilmə sahəsində mətnin vəziyyətini müəyyən edir. Bu xassənin qiymətləri button düyməsinin eyniadlı xassəsinin qiymətləri ilə eynidir;

❖ CheckAlign – komponentin sahəsində kvadratın vəziyyətini müəyyən edir. Onun aldığı qiymətlər TextAlign xassəsinin qiymətləri ilə eynidir;

❖ Appearance – susmaya görə bu xassəyə Normal qiyməti mənimsədilmişdir, ona görə də o ənənəvi formada təsvir edilir. Əgər bu xassəyə Button qiyməti versək, onda o düymə şəklində təsvir ediləcəkdir. Bu zaman bayraq qoyulduqda düymə basılmış vəziyyətdə olur;

❖ Checked – bayrağın qoyulmuş və ya götürülmüş vəziyyətdə olmasını müəyyən edir. Bayraq qoyulduqda bu xassənin qiyməti true, götürüldükdə isə false olur (susmaya görə);

❖ Enabled – komponentin aktivliyini müəyyən edir. Ona false qiyməti verdikdə komponent aktiv olmur;

❖ Visible – bu xassəyə true qiyməti verdikdə (susmaya görə bu belədir) dəyişdirici görünür, false qiyməti verdikdə isə dəyişdirici gizlədilir;

❖ FlatStyle – komponentin görünüşünü müəyyən edir:

- Standard – standart görünüşlü;
- Flat – müstəvi formasında;
- Popup – “peyda” olan;
- System – sistem dəyişdiricisi;

❖ Image – komponentin sahəsində şəkli müəyyən edir;

❖ ImageAlign – komponentin sahəsində şəklin vəziyyətini müəyyən edir. Bu xassənin qiymətləri TextAlign xassəsinin qiymətləri ilə eynidir.

checBox komponentinin əsas hadisələri bunlardır:

❖ CheckedChanged — bu hadisə hər dəfə Checked xassəsinin qiyməti dəyişdikdə yaranır;

❖ CheckedStateChanged — bu hadisə Checked və ya CheckState xassələrinin qiyməti dəyişdikdə yaranır;

❖ Click — komponent üzərində mausun düyməsini basdıqda yaranır.

Misal. checkBox komponenti ilə şriftin seçilməsi.

label komponentinin mətninin şriftini dəyişdirən proqram tərtib edək. Şrifti checkBox komponenti ilə dəyişdirəcəyik. Bu məqsədlə forma üzərinə label və checkBox komponentləri yerləşdirək. Əvvəlcə Form1 və label1 komponentlərinin başlanğıc vəziyyətdə şriftini müəyyən edək. Bu işləri Form1 komponenti üçün Load hadisəsini yaratmaqla yerinə yetirək. Bu hadisə emaledicisinin kodları belədir:

```
private void Form1_Load(object sender,
                        EventArgs e)
{
  base.Text = "checBox komponentinin
                        öyrənilməsi";
  label1.Text = "Şriftin parametrlərini
                                seç";
  label1.TextAlign =
         ContentAlignment.MiddleCenter;
  label1.Font = new Font("Courier New",
                                14.0F);
  checkBox1.Focus();
}
```

Göründüyü kimi, formanın başlığı üçün mətn (base istinadı vasitəsi ilə) və label yazısının mətni müəyyən edilir. label komponentində mətn mərkəzdə yerləşdirilir və onun

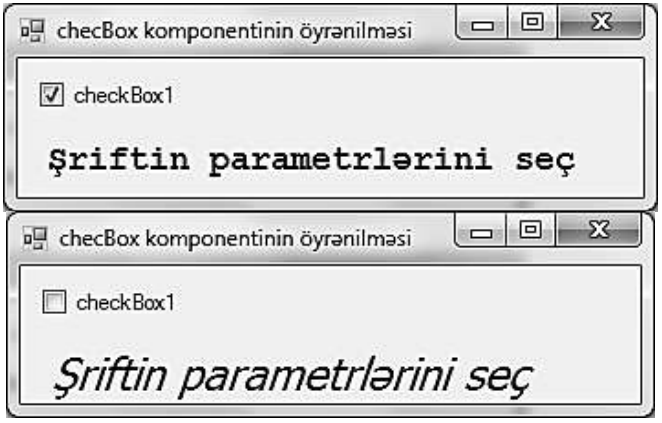
üçün 14 punkt ölçülü Courier New şrifti təyin olunur. İstifadəçi həmişə checkBox1 komponenti ilə işləyəcəyinə görə, bu komponent həmişə fokus altında olmalıdır. Ona görə də proqramda checkBox1.Focus(); yazılmışdır. Bu halda, hətta boşluq klavişini basmaqla da, bayrağı qoymaq və götürmək olar.

İndi isə checkBox1 komponenti ilə əlaqədar əməliyyatları proqramlaşdıraq.

Bayrağın vəziyyətinin dəyişdirilməsi CheckedChanged hadisəsinə uyğun gəlir. Bu hadisə emaledicisinin kodları isə belədir:

```
private void checkBox1_CheckedChanged(
            object sender, EventArgs e)
{
if(checkBox1.Checked==true)
   label1.Font = new Font("Courier New",
             14.0F,  FontStyle.Bold);
if (checkBox1.Checked == false)
   label1.Font = new Font("Tahoma",
          18.0F, FontStyle.Italic);
}
```

Kodlardan aydın olur ki, əgər bayraq qoyulmuşdursa (if(checkBox1.Checked==true)), onda label komponentinin mətni üçün Courier New, 14 punkt ölçülü, qalın şrift seçilir. Bayraq götürüldükdə isə (if(checkBox1.Checked == false)) label komponentinin mətni üçün Tahoma, 18 punkt ölçülü, kursiv şrift seçilir. Bu hallara uyğun formalar şəkil 7.2-də göstərilmişdir.



Şəkil 7.2. `checkBox` komponenti ilə şriftin seçilməsi

Misal. Aşağıdakı funksiyanı hesablayaq:

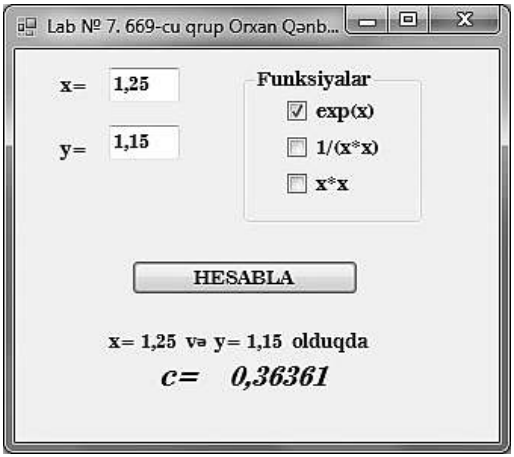
$$c=\begin{cases}\sin(f(x))+\cos(f(y)), & x-y=0\\ \operatorname{tg}(f(x+y)), & x-y\rangle 0\\ \sin^2(f(x))+\cos^2(f(y)), & x-y\langle 0.\end{cases}$$

Burada, *f(x)* funksiyası bir neçə ifadəyə malik ola bilər. Qəbul edək ki, bunlar $e^x, \frac{1}{x^2}$ və x^2 funksiyalarıdır.

Göründüyü kimi, *c* dəyişənini hesablamaq üçün ilk növbədə *x* və *y* dəyişənlərini müqayisə etmək lazımdır. Bir müqayisədə isə hər üç funksiyaya uyğun qiymətlər hesablanmalıdır. Bu məsələni həll etdikdə funksiyaların seçilməsini `checkBox` komponentləri ilə yerinə yetirək.

Məsələni həll etmək üçün iki metod yaradaq. Metodlardan biri *c* dəyişənini, digəri isə seçilmiş funksiyanı hesablamaq üçün istifadə ediləcəkdir. *c* dəyişənini hesablayacaq metod funksiyanı seçən metodu çağiracaqdır. Düyməbasma hadisə emaledicisində isə *x* və *y* dəyişənləri daxil ediləcək, birinci metod çağırılacaq və nəticələr yazı komponentlərində təsvir ediləcəkdir. Məsələni həll etmək üçün şəkil 7.3-də göstərilmiş əlavəni layihələndirək. *x* və *y* dəyişənlərinin qiymətləri

textBox komponentlərindən daxil ediləcək funksiyaları seçmək üçün isə groupBox konteynerinin üzərində üç checvkBox komponentləri yerləşdirək. Bu komponentlərin Text xassələrinə şəkil 7.3-də göstərilmiş mətnlər yazaq.



Şəkil 7.3. 7 №-li laboratoriya işinin nəticəsi

Birinci metodu ESAS_FUN adlandıraq. Bu metoda formal arqumentlər kimi *x* və *y* dəyişənləri ötürüləcək və metod funksiyanı seçən FUN metodunu çağıracaqdır. Bu metodun proqramı adi budaqlanan hesablama prosesinin kodundan ibarətdir. FUN metodunda isə checkBox komponentlərinin qoşulmasından asılı olaraq funksiya seçiləcək və onun formal arqumenti *x* dəyişəni olacaq. Proqramın hər yerindən görünməsi üçün bütün dəyişənləri qlobal dəyişən kimi elan edək.

Bu metodları kodlaşdırdıqdan sonra button düyməsi üçün Click hadisəsi yaradırıq.

Yaratdığımız əlavənin kodu aşağıda, onun nəticəsi isə şəkildə göstərilmişdir.

```
using System;
using System.Windows.Forms;

namespace Lab7
{
public partial class Form1 : Form
{
 double F;
 double x, y,c;
 public void ESAS_FUN(double x,double y)
{
  FUN(x);
  if (x-y==0) c=Math.Sin(F)+Math.Cos(F);
  else
   if (x - y > 0)
   {
     x = x + y;
     c =  Math.Tan(F);
   }
   else c = Math.Sqrt(Math.Sin(F)) +
            Math.Sqrt(Math.Cos(F));
}

public void FUN(double x)
{
  if (checkBox1.Checked == true)
F = Math.Exp(x);
  if (checkBox2.Checked == true)
F = 1 / (x * x);
  if (checkBox3.Checked == true)
F = x * x;
}
public Form1()
{
```

```
  InitializeComponent();
}

private void button1_Click(object
              sender,EventArgs e)
{
  x = Convert.ToDouble(textBox1.Text);
  y = Convert.ToDouble(textBox2.Text);
  ESAS_FUN(x, y);
  c = Math.Round(c, 5);
  label3.Text="x= "+ x.ToString()+
   "  və  "+ "y= " + y.ToString() +
                  "  olduqda";
  label5.Text = c.ToString();
} } }
```

Yoxlama sualları

1. `radioButton` komponenti nə üçündür?
2. `Checked` xassəsi nə üçündür?
3. `Text` xassəsi nə üçündür?
4. `CheckAlign` xassəsi nə üçündür?
5. `Appearance` xassəsi nə üçündür?
6. `FlatStyle` xassəsi nə üçündür?
7. `ImageAlign` xassəsi nə üçündür?
8. `radioButton` komponentinin əsas hadisəsi nədir?
9. `checkBox` komponenti nə üçündür?
10. `CheckedChanged` hadisəsi nə zaman yaranır?
11. `CheckedStateChanged` hadisəsi nə zaman yaranır?
12. `Click` hadisəsi nə zaman yaranır?

Laboratoriya işlərinə aid tapşırıqların variantları

1-14-cu variantlarda *f(x)* funksiyası kimi $e^x, \frac{1}{x^2}$ *və* x^2 funksiyalarını götürün. Bu funksiyaları `checkBox` komponenti ilə seçin.

1. $a=\begin{cases}(f(x)+y)^2-\sqrt{f(x)y}, & xy\rangle 0\\ (f(x)+y)^2+\sqrt{|f(x)\,y|}, & xy\langle 0\\ (f(x)+y)^2+1, & xy=0.\end{cases}$

2. $b=\begin{cases}\ln(f(x))+(f(x)^2+y)^3, & x/y\rangle 0\\ \ln|f(x)/y|+(f(x)+y)^3, & x/y\langle 0\\ (f(x)^2+y)^3, & x=0\\ 0, & y=0.\end{cases}$

3. $c=\begin{cases}f(x)^2+y^2+\sin(y), & x-y=0\\ (f(x)-y)^2+\cos(y), & x-y\rangle 0\\ (y-f(x))^2+\operatorname{tg}(y), & x-y\langle 0.\end{cases}$

4. $d=\begin{cases}(f(x)-y)^3+\operatorname{arctg}(f(x)), & x\rangle y\\ (y-f(x))^3+\operatorname{arctg}(f(x)), & y\rangle x\\ (y+f(x))^3+0.5, & y=x.\end{cases}$

5. $e=\begin{cases}i\sqrt{f(x)}, & i-\ təkdir,\ \ x\rangle 0\\ i/2\sqrt{|f(x)|}, & i\text{-cütdür},\ \ x\langle 0\\ \sqrt{|if(x)|}, & əks\ halda.\end{cases}$

6. $g=\begin{cases}e^{f(x)-|b|}, & 0.5\langle xb\langle 10\\ \sqrt{|f(x)+b|}, & 0.1\langle xb\langle 0.5\\ 2f(x)^2, & əks\ halda.\end{cases}$

7. $s=\begin{cases}e^{f(x)}, & 1\langle xb\langle 10\\ \sqrt{|f(x)+4*b|}, & 12\langle xb\langle 40\\ bf(x)^2, & əks\ halda.\end{cases}$

8. $j=\begin{cases}\sin(5f(x)+3m|f(x)|), & -1\langle m\langle x\\ \cos(3f(x)+5m|f(x)|), & x\rangle m\\ (f(x)+m)^2, & x=m.\end{cases}$

9. $l=\begin{cases}2f(x)^3+3p^2, & x\rangle |p|\\ |f(x)-p|, & 3\langle x\langle |p|\\ (f(x)-p)^2, & x=|p|.\end{cases}$

10. $k=\begin{cases}\ln(|f(x)|+|q|), & |xq|\rangle 10\\ e^{f(x)+q}, & |xq|\langle 10\\ f(x)+q, & |xq|=10\end{cases}$

11. $m=\dfrac{\max(f(x),y,z)}{\min(f(x),y)}+5.$

12. $n=\dfrac{\min(f(x)+y,y-z)}{\max(f(x),y)}.$

13. $p=\dfrac{|\min(f(x),y)-\max(y,z)|}{2}.$

14. $q=\dfrac{\max(f(x)+y+z,xyz)}{\min(f(x)+y+z,xyz)}.$

15. `groupBox` konteynerinin üzərində iki `radioButton` dəyişdiriciləri yerləşdirin. Konteynerin

başlığını “Piramida”, dəyişdiriciləri isə uyğun olaraq “Tam səthi” və “Həcmi” adlandırın. “Tam səthi” adlı dəyişdiricini qoşduqda piramidanın tam səthini, “Həcmi” adlı dəyişdiricini qoşduqda isə həcmini hesablayın. Piramidanın oturacağı tərəfi *b* olan kvadratdır, hündürlüyü isə *h*-a bərabərdir.

16. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Həndəsə məsələləri”, dəyişdiriciləri isə uyğun olaraq “Piramidanın həcmi” və “Üçbucağın sahəsi” adlandırın. “Piramidanın həcmi” adlı dəyişdiricini qoşduqda piramidanın həcmini hesablayın. Piramidanın oturacağı tərəfi *b* olan bərabər tərəfli üçbucaqdır, hündürlüyü isə *h*-a bərabərdir. “Üçbucağın sahəsi” adlı dəyişdiricini qoşduqda isə tərəfləri *a* və *b*, onların arasındakı bucağı isə β olan üçbucağın sahəsini hesablayın.

17. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Silindr”, dəyişdiriciləri isə uyğun olaraq “Tam səthi” və “Həcmi” adlandırın. “Tam səthi” adlı dəyişdiricini qoşduqda silindrin tam səthini, “Həcmi” adlı dəyişdiricini qoşduqda isə həcmini hesablayın. Silindrin oturacağının radiusu *r*, hündürlüyü isə *h*-dır.

18. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Düz piramida”, dəyişdiriciləri isə uyğun olaraq “Tam səthi” və “Həcmi” adlandırın. “Tam səthi” adlı dəyişdiricini qoşduqda düz piramidanın tam səthini, “Həcmi” adlı dəyişdiricini qoşduqda isə həcmini hesablayın. Piramidanın oturacağı katetləri *a* və *b* olan düzbucaqlı üçbucaqdır, hündürlüyü isə *h*-a bərabərdir.

19. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Düz prizmanın tam səthi”, dəyişdiriciləri isə uyğun olaraq “Bərabərtərəfli üçbucaq” və “Kvadrat” adlandırın. “Kvadrat” adlı dəyişdiricini qoşduqda hündürlüyü *h*, oturacağı isə kvadrat olan düz prizmanın tam səthini hesablayın. Kvadratın tərəfi *a*-ya bərabərdir. “Bərabərtərəfli üçbucaq” adlı dəyişdiricini qoşduqda isə hündürlüyü *h*, oturacağı isə bərabərtərəfli üçbucaq olan düz prizmanın tam səthini hesablayın. Üçbucağın tərəfi *b*-yə bərabərdir.

20. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Düz prizmanın həcmi”, dəyişdiriciləri isə uyğun olaraq “Bərabərtərəfli üçbucaq” və “Kvadrat” adlandırın. “Kvadrat” adlı dəyişdiricini qoşduqda hündürlüyü *h*, oturacağı isə kvadrat olan düz prizmanın həcmini hesablayın. Kvadratın tərəfi *a*-ya bərabərdir. “Bərabərtərəfli üçbucaq” adlı dəyişdiricini qoşduqda isə hündürlüyü *h*, oturacağı isə bərabərtərəfli üçbucaq olan düz prizmanın həcmini hesablayın. Üçbucağın tərəfi *b*-yə bərabərdir.

21. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Kubun tam səthi və həcmi”, dəyişdiriciləri isə uyğun olaraq “Tam səthi” və “Həcmi” adlandırın. “Tam səthi” adlı dəyişdiricini qoşduqda tili *a* olan kubun tam səthini, “Həcmi” adlı dəyişdiricini qoşduqda isə kubun həcmini hesablayın.

22. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Sahələr”, dəyişdiriciləri isə uyğun olaraq

“Halqanın sahəsi” və “Dairənin sahəsi” adlandırın. “Halqanın sahəsi” adlı dəyişdiricini qoşduqda daxili diametri *d*, xarici diametri isə *D* olan halqanın sahəsini, “Dairənin sahəsi” adlı dəyişdiricini qoşduqda isə radiusu *r* olan dairənin sahəsini hesablayın.

23. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Kürə və şar”, dəyişdiriciləri isə uyğun olaraq “Kürənin səthi” və “Şarın həcmi” adlandırın. “Kürənin səthi” adlı dəyişdiricini qoşduqda radiusu *r* olan kürənin səthini, “Şarın həcmi” adlı dəyişdiricini qoşduqda isə şarın həcmini hesablayın.

24. groupBox konteynerinin üzərində iki radioButton dəyişdiriciləri yerləşdirin. Konteynerin başlığını “Düz prizmanın səthi və kvadratın sahəsi”, dəyişdiriciləri isə uyğun olaraq “Tam səthi” və “Sahəsi” adlandırın. “Tam səthi” adlı dəyişdiricini qoşduqda düz prizmanın tam səthini hesablayın. Prizmanın oturacağı katetləri *a* və *b* olan düzbucaqlı üçbucaqdır, hündürlüyü isə *h*-dır. “Sahəsi” adlı dəyişdiricini qoşduqda isə tərəfi *d* olan kvadratın sahəsini hesablayın.

25. Forma üzərinə üç radioButton, textBox, label və button düymələri yerləşdirin. Əlavə üç müxtəlif ölçülü foto şəklin qiymətini hesablayacaqdır. radioButton komponentlərinin Text xassəsinə müvafiq olaraq 9x12, 12x15 və 18x24 yazın. 9x12 ölçülü şəklin bir ədədinin qiyməti 1 manat, 12x15 – 1 manat 50 qəpik və 18x24 – 2 manat 50 qəpikdir. textBox komponentindən nüsxələrin sayı daxil ediləcəkdir. Button düyməsini basdıqda şəklin müvafiq ölçüsünə uyğun olaraq ödəniləcək məbləğ label komponentində təsvir ediləcəkdir.

26. Banka qoyulmuş əmanətin gəlirini hesablayın. Hesabat üçün ilkin verilənlər belədir: qoyulmuş məbləğ və müddəti (1, 3, 6 və 12 ay). Faiz əmanətin müddəti ilə müəyyən edilir.

27. Yanacaq doldurma stansiyalarında benzinin qiymətini hesablayan proqram tərtib edin. Hesabat üçün ilkin verilənlər belədir: litrin miqdarı və benzinin markası (92, 95 və 98).

28. Avtomobilin icarəsı üçün icarə haqqını hesablayan proqram tərtib edin. Hesabat üçün ilkin verilənlər belədir: icarə müddəti (saatla, tam ədəddir) və avtomobilin növü (taksi, mikroavtobus və avtobus).

29. `groupBox` komponenti üzərinə 6 ədəd `radiobutton` komponentləri yerləşdirib onların `Text` xassəsinə müvafiq olaraq `Toplama`, `Çıxma`, `Vurma`, `Bölmə`, `Qüvvətə yüksəltmə` və `Bölmədə qalıq` yazın. Dəyişdiriciləri seçdikdə `textBox1` və `textBox2` komponentlərindən daxil edilmiş ədədlər üzərində müvafiq əməliyyatlar icra edilərək `textbox3` komponentində təsvir edilsin.

30. Bir `groupBox` komponentində avtomobil markaları, digərində isə rənglər müəyyən edin. “`Avtomobili seç`” düyməsini basdıqda seçilmiş marka və rəngli avtomobil `textBox` komponentində göstərilsin.

LABORATORİYA İŞİ № 8.
KONTEYNERLƏRLƏ İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən konteynerlərin xassələrini və əlavələrin layihələndirilməsində onların tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat

Konteynerlər proqramlaşdırma nöqteyi-nəzərindən məna yükü daşımırlar, sadəcə olaraq onların üzərində digər komponentlər qruplar şəklində yerləşdirilir. Adətən, qruplarda `radioButton` və `checkBox` komponentləri yerləşdirilir. Konteynerlərin yerini dəyişdirdikdə onların üzərində yerləşdirilmiş bütün komponentlər konteynerlə birlikdə yerini dəyişir. Bundan başqa, konteynerin `Visible` xassəsinə `false` qiyməti verdikdə konteynerlə birlikdə onun üzərində yerləşdirilmiş bütün komponentlər də gizlədilir.

`groupBox` idarəetmə elementi

`groupBox` komponentinin praktiki tətbiqini görmüşük. İndi isə onun bəzi xassələri ilə tanış olaq.

- `Text` – `groupBox` konteynerinin sahəsində yerləşən komponentlərin təyinatını izah edən mətn;
- `Enabled` – konteynerin aktivliyini müəyyən edir; Konteynerin bu xassəsinə `false` qiyməti verdikdə onun üzərində yerləşən bütün komponentlər də qeyri-aktiv olur.
- `Visible` –bu xassəyə `false` qiyməti verdikdə konteynerlə birlikdə onun üzərində yerləşən bütün komponentlər də gizlədilir;

❖ `Controls` – konteynerin üzərində yerləşən elementlər yığımını müəyyən edir.

`panel` idarəetmə elementi

Bu konteynerin `groupBox` konteynerindən fərqi ondadır ki, `groupBox` komponentinin başlığı var, ancaq firlatma zolaqları yoxdur, `panel` komponentinin isə başlığı yoxdur, ancaq fırlatma zolaqları vardır. Bundan başqa, `groupBox` komponenti nazik haşiyəyə malikdir, `panel` komponentinin haşiyəsi isə `BorderStyle` xassəsi ilə dəyişdirilə bilər. `panel` komponentinin əsas xassələri bunlardır:

❖ `BorderStyle` – panelin haşiyəsinin formasını müəyyən edir. Bu xassəyə aşağıdakı qiymətləri vermək olar:

- `FixedSingle` – haşiyə;
- `Fixed3D` – həcmli haşiyə;
- `None` – haşiyə yoxdur;

❖ `AutoScroll` – elementlər komponentin sahəsinə sığışmadıqda fırlatma zolaqlarının təsvir edilməsini müəyyən edir. Firlatma zolalarının görünməsi üçün bu xassəyə `true` qiyməti vermək lazımdır;

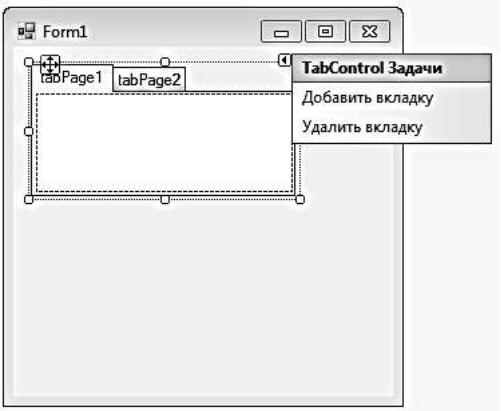
❖ `Controls` – konteynerin üzərində yerləşən elementlər yığımını müəyyən edir.

`panel` komponentinin də `Enabled` və `Visible` xassələri vardır.

`tabControl` idarəetmə elementi

Bu komponent yuxarı hissəsində səhifələri olan konteynerdir. Səhifələr formada daha çox informasiya yerləşdirməyə və verilənləri məntiqi qruplaşdırmağa imkan verir. Forma üzərinə `tabControl` elementi yerləşdirilərsə, susmaya görə komponentin `tabPage1` və `tabPage2` adlı iki səhifəsi əmələ gəlir. `tabPage` səhifələri `tabControl`

komponentinin obyektləridir və onların özlərində də digər obyektlər yerləşə bilər. Komponentdə səhifələrin sayını artırmaq olar. Bunun üçün `tabControl` komponentinin yuxarı sağ tərəfindəki kiçik kvadrat daxilində mausun düyməsini basmaq lazımdır. Bu zaman kiçik pəncərə peyda olacaqdır ki, buradakı `Add TabPage` (`Добавить вкладку`) və `Delete TabPage` (`Удалить вкладку`) əmrləri ilə yeni səhifə əlavə etmək və ya səhifəni silmək olar (şəkil 8.1). Bu əməliyyatları kontekst menyunun eyni əmrləri ilə də icra etmək olar.



Şəkil 8.1. `tabControl` elementinin görünüşü

Komponentin özü `TabControl` tipli, onun səhifələri isə `TabPage` tiplidir. `TabPage` sinfinin səhifələri öz xassələrinə görə panel komponentindən, demək olar ki, fərqlənmir. `tabControl` komponentinin isə xassələri aşağıdakılardır:

❖ `HotTrack` – kursoru səhifəyə tuşladıqda səhifə sanki işıqlanır;

❖ `MultiLine` – səhifələrin başlıqları bir sətrə sığışmadıqda bir neçə sətirdə yerləşə bilər;

❖ `SelectedIndex` – `tabPage` obyektinin seçilmiş elementinin indeksi;

❖ `SelectedTab` – seçilmiş `tabPage` obyekti;

❖ `TabCount` – səhifələrin sayını müəyyən edir;

❖ `TabPages` – bu xassədə səhifələr kolleksiyası yerləşir;

❖ `ImageList` – səhifələrə çıxarılacaq təsvirlər;

❖ `ItemSize` – səhifələrin ölçüsünü müəyyən edir.

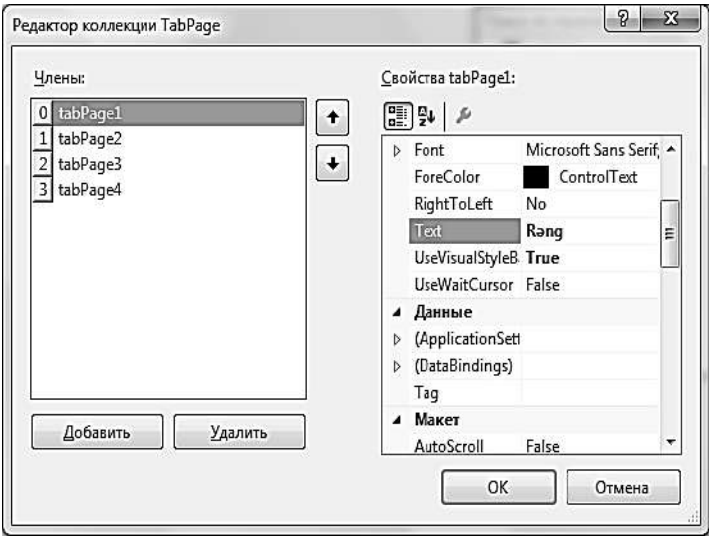
`TabControl` komponentinin ən çox istifadə edilən hadisəsi `SelectedIndexChanged` hadisəsidir. Hər dəfə hər bir səhifə üzərində mausun düyməsini basdıqda `Click` hadisəsi baş verir.

Misal. `tabControl` komponenti ilə şriftin atributlarının dəyişdirilməsi.

Forma üzərinə `tabControl` komponenti yerləşdirək. Gələcək layihənin pəncərəsi şəkil 8.2-də göstərilmişdir. İndi bu pəncərənin necə tərtib edilməsinə baxaq.

Şəkil 8.2. `tabControl` komponenti ilə şriftin atributlarının dəyişdirilməsi əlavəsinin layihəsi

Forma üzərinə, tabControl komponentinin aşağı hissəsinə, label1 komponenti yerləşdirin. Bu komponentin Text xassəsinə "Mən AzTU-da oxuyuram" yazın. Onun AutoSize xassəsinə false, AllowDrop xassəsinə true qiyməti verin ki, üzərində təsvir etdiriləcək mətn bir neçə sətirlərdə yerləşə bilsin, TextAlign xassəsinə isə MiddleCenter qiyməti mənimsədin ki, mətn komponentin mərkəzində yerləşsin. Bizə tabControl komponentdə dörd səhifə lazım olacaqdır. Ona görə də komponenti seçərək Properties pəncərəsində TabPages xassəsinin qarşısındakı üç nöqtə təsvirli düyməni basın. Ekrana TabPage Collection Editor (Редактор коллекции TabPage) pəncərəsi çixacaqdır (şəkil 8.3).



Şəkil 8.3. tabControl komponentinə səhifələrin əlavə edilməsi

Bu pəncərədə Добавить düyməsini iki dəfə basmaqla komponentə daha iki səhifə əlavə edin. Elə həmin pəncərədə

səhifələrin başlıqlarını (Text xassəsi) şəkil 8.2-yə uyğun dəyişdirin və Ok düyməsini basın. Formaya qayıdıb komponentin Rəng və Şriftin ölçüsü səhifələrini açaraq onların hər birinə üç radioButton komponenti, İsmarıc səhifəsinə iki radioButton komponenti və sonuncu Məlumat səhifəsinə isə label2 komponenti yerləşdirin. Növbə ilə birinci-üçüncü səhifələri açıb hər bir dəyişdiricini seçərək onların Text xassəsinə Qırmızı, Yaşıl, Qara, 12 punkt, 16 punkt, 20 punkt, Dərslik, Xoş xəbər mətnləri yazın.

İndi isə kodları yazaq. Biz istəyirik ki, Rəng səhifəsini açıb dəyişdiriciləri qoşduqda şriftin rəngi dəyişsin. Bunun üçün Rəng (radioButton1) dəyişdiricisini seçib xassələr pəncərəsinin Events səhifəsində CheckedChange hadisəsinin qarşısında mausun düyməsini basıb oraya aşağıdakı kodu yazın:

```
label1.ForeColor = Color.Red;
```

Analoji əməliyyatı digər rənglər üçün təkrarlayın. Şriftin ölçüsü səhifəsinə keçib 12 punkt (radioButton4) dəyişdiricisi üçün də eyni hadisə emaledicisini yaradıb, oraya aşağıdakı kodu yazın:

```
label1.Font= new Font(
        label1.Font.Name, 12);
```

Analoji əməliyyatı digər ölçülər üçün təkrarlayın. tabControl komponentinin üçüncü səhifəsi ismarıclar üçün nəzərdə tutulmuşdur. Bu səhifədə Dərslik (radioButton7) və Xoş xəbər (radioButton8) dəyişdiriciləri üçün də eyni hadisə emaledicisi yaradıb oraya müvafiq olaraq bu kodları yazın:

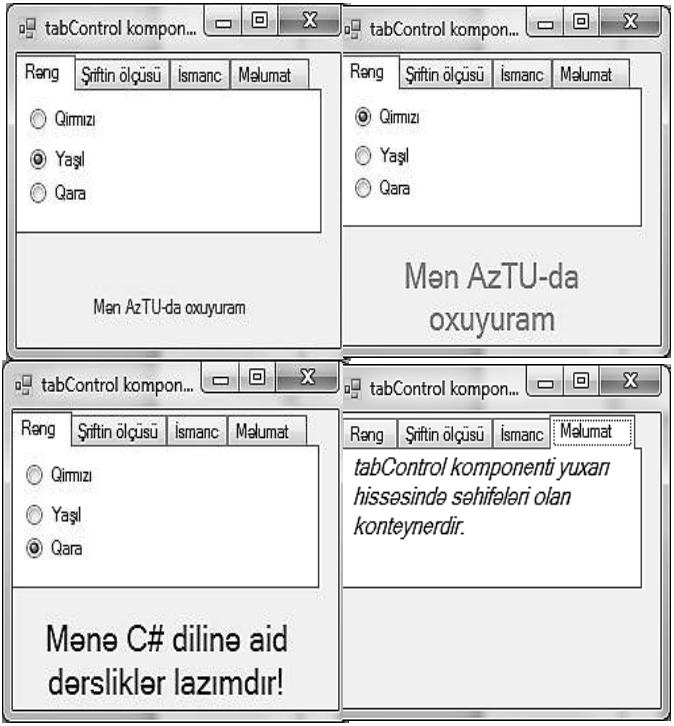
```
label1.Text = "Mənə C# dilinə aid
              dərsliklər lazımdır!";
label1.Text = "Mən C# dilinə aid
              bir neçə dərslik aldım!";
```

İndi sonuncu səhifə üçün kodlar yazaq. Bu səhifədə yalnız label2 komponenti yerləşir. Bizə lazımdır ki, Məlumat səhifəsini açan kimi, dərhal tabControl komponenti haqqında məlumat alaq. Bunun üçün Məlumat (tabPage4 obyekti) səhifəsini seçib Click hadisəsinin qarşısında mausun düyməsini basaraq aşağıdakı hadisə emaledicisini yaradın:

```
private void tabControl1_Click(object
                sender,EventArgs e)
{
if(tabControl1.SelectedIndex==3)
    label1.Visible = false;
  else label1.Visible = true;
label2.AllowDrop = true;
label2.Text = "tabControl komponenti
  yuxarı hissəsində səhifələri olan
                  konteynerdir.";
}
```

Təbii ki, hər hansı komponent haqqında məlumatın təsvir edildiyi yerdə şəxsi məlumatların olması düzgün olmaz. Ona görə də biz gərək label1 komponentini gizlədək. Bu əməliyyat birinci kodla yerinə yetirilmişdir. Bilirik ki, tabControl1 komponentinin tabPage4 obyektinin nömrəsi 3-ə bərabərdir (nömrələmə sıfırdan başlayır). Odur ki, həmin nömrəli obyektin seçilməsi şərti yoxlanılmalıdır: if(tabControl1.SelectedIndex==3). Bu şərt ödəndikdə label1 komponenti gizlədilir.

Bu proqramın bir neçə situasiyalarda nəticəsi şəkil 8.4-də göstərilmişdir.



Şəkil 8.4. `tabControl1` komponenti ilə şriftin atributlarının dəyişdirilməsi

Bu layihənin tam kodu aşağıdakı kimidir:

```
namespace TABCONTROL
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}
private void
        radioButton1_CheckedChanged(
```

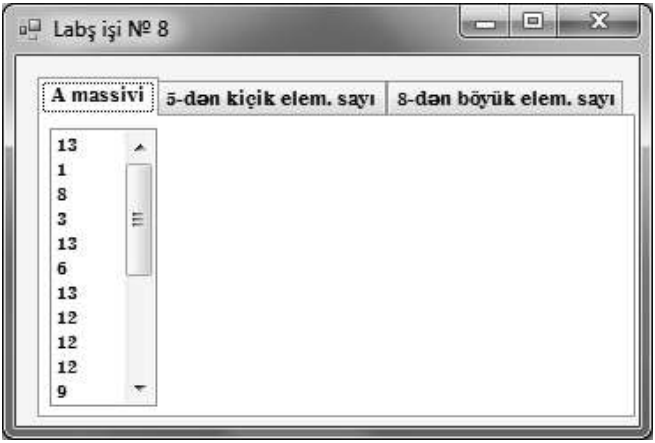
```
        object sender, EventArgs e)
{
label1.ForeColor = Color.Red;
}

private void
        radioButton2_CheckedChanged(
        object sender, EventArgs e)
{
label1.ForeColor = Color.Green;
}

private void
        radioButton3_CheckedChanged(
        object sender, EventArgs e)
{
label1.ForeColor = Color.Black;
}

private void
        radioButton4_CheckedChanged(
        object sender, EventArgs e)
{
label1.Font= new Font(
             label1.Font.Name, 12);
}

private void
        radioButton5_CheckedChanged(
        object sender, EventArgs e)
{
label1.Font = new Font(
            label1.Font.Name, 16);
}
```

```
private void
        radioButton6_CheckedChanged(
        object sender, EventArgs e)
{
label1.Font = new Font(
               label1.Font.Name, 20);
}
private void
         radioButton7_CheckedChanged(
         object sender, EventArgs e)
{
label1.Text = "Mənə C# dilinə aid
             dərsliklər lazımdır!";
}

private void
         radioButton8_CheckedChanged(
         object sender, EventArgs e)
{
label1.Text = "Mən C# dilinə aid
             bir neçə dərslik aldım!";
}

private void tabControl1_Click(
        object sender, EventArgs e)
{
if(tabControl1.SelectedIndex==3)
    label1.Visible = false;
  else label1.Visible = true;
label2.AllowDrop = true;
label2.Text = "tabControl komponenti
     yuxarı hissəsində səhifələri olan
     konteynerdir.";
```

```
}}}
```

Misal. (0, 15) intervalında təsadüfi ədədlərdən ibarət A(20) massivi yaradın. Onun 5-dən kiçik və 8-dən böyük elementlərinin sayını tapan əlavə yaradın.

Məsələni həll etmək üçün üç səhifədən ibarət `tabControl` konteyneri yerləşdirək. Onun səhifələrini şəkil 8.5-ə uyğun adlandıraq. Birinci səhifədə `listbox`, digər səhifələrdə isə `label` komponentləri yerləşdirək. Məsələni belə həll edəcəyik. `tabControl` komponentinin `A massivi` adlı səhifəsi üzərində mausun düyməsini basdıqda `listBox` komponentində A massivi təsvir ediləcək, digər düymələri basdıqda isə `label` komponentlərində səhifənin başlığına uyğun nəticələr təsvir ediləcəkdir.



Şəkil 8.5. 8 №-li laboratoriya işinin nəticəsi

Biz bu məsələni `tabControl` komponentinin `Click` hadisə emaledicisi ilə həll edəcəyik. Səhifələrin seçilməsini isə `if` operatoru ilə icra edəcəyik. Belə ki, səhifə `SelectedIndex` xassəsi ilə seçilir.

Beləliklə, məsələnin həlli üçün aşağıdakı proqramı tərtib edə bilərik:

```
using System;
using System.Windows.Forms;

namespace Lab8
{
public partial class Form1 : Form
{
int[] a = new int[20];
public Form1()
{
  InitializeComponent();
}

private void tabControl1_Click(object
                 sender, EventArgs e)
{
  Random r = new Random();
  listBox1.Items.Clear();

  if (tabControl1.SelectedIndex == 0)
  {
    for (int i = 0; i < 20; i++)
    {
      a[i] = r.Next(0, 15);
      listBox1.Items.Add(a[i]);
    }
   }

  if (tabControl1.SelectedIndex == 1)
  {
    int k = 0;
    for (int i = 0; i < 20; i++)
```

```
        if (a[i] < 5) k++;
        label1.Text = k.ToString();
    }
    if (tabControl1.SelectedIndex == 2)
    {
      int n = 0;
      for (int i = 0; i < 20; i++)
      if (a[i] > 8) n++;
        label2.Text = n.ToString();
    }
} } }
```

Qeyd edək ki, bu məsələni `switch` operatoru vasitəsi ilə də həll etmək olar.

Yoxlama sualları

1. Hansı komponentlər konteynerdir?
2. Konteynerlər nə üçündür?
3. `groupBox` komponentinin `panel` komponentindən fərqi nədədir?
4. Konteynerin üzərində yerləşən elementlər yığımını müəyyən edən xassə hansıdır?
5. `tabControl` komponentinin digər komponentlərdən fərqi nədədir?
6. `tabControl` komponentinə yeni səhifə necə əlavə edilir?
7. `tabControl` komponentinin konkret səhifəsinə necə müraciət edilir?
8. `TabControl` komponentinin ən çox istifadə edilən hadisəsi hansıdır?

Laboratoriya işlərinə aid tapşırıqların variantları

9-30-cu variantlardakı tapşırıqlarda ilkin, aralıq və son massivləri və həllin nəticələrini `listBox`, `label` və ya `textBox` komponentlərində göstərməklə şəkil 8.5-ə analoji əlavə layihələndirin.

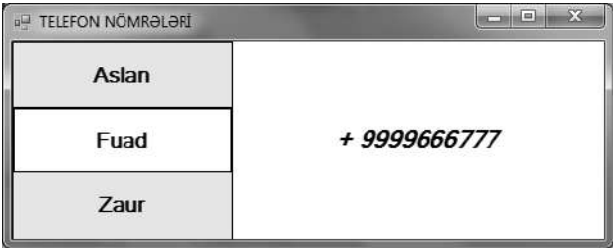
1. `groupBox` komponentinin üzərinə üç `radioButton` dəyişdiriciləri yerləşdirib onları `Qırmızı`, `Göy` və `Sarı` adlandırın. Bu dəyişdiricilərin alt hissəsində `button` düyməsi ondan aşağıda isə panel komponenti yerləşdirin. `button` düyməsini basdıqda panel komponenti dəyişdiricilərlə seçilmiş rənglə rənglənsin.

2. `groupBox` komponentinin üzərinə üç `radioButton` dəyişdiriciləri yerləşdirib onları `Göy`, `Qırmızı` və `Yaşıl` adlandırın. Bu dəyişdiricilərin alt hissəsində `button` düyməsi ondan aşağıda isə üç panel komponenti yerləşdirin. `button` düyməsini basdıqda panel komponentləri ardıcıl olaraq göy, qırmızı və yaşıl rənglə rənglənsin.

3. `groupBox` komponentinin üzərinə `label` komponentləri yerləşdirin. Onun başlığını "`panel komponentinin ölçüləri`" adlandırın. Düyməni basdıqda forma üzərində yerləşdirilmiş panel komponentinin koordinatları və ölçüləri müvafiq `label` komponentlərində təsvir edilsin.

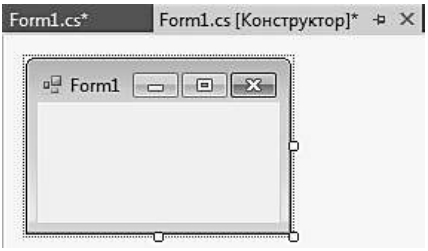
4. `panel`, düymə və `label` komponentlərindən istifadə etməklə şəkil 8.6-da göstərilən əlavəni yaradın.

5. Forma üzərinə iki `groupBox` komponentləri yerləşdirib, onlardan birini "`Xarici valyuta`" digərini isə "`Manat`" adlandırın. Birinci konteynerdə `radiobutton` düyməsi ilə ABŞ dolları və ya avro seçdikdə `textBox` komponentindən daxil edilən valyuta digər konteynerdə manatla `textBox` komponentində ifadə edilsin.



şəkil 8.6. 4 №-li tapşırığa əlavə

6. Visual Studio mühitində proqramın kompilyasiyası pəncərəsinin şəkil 8.7-də göstərilən maketini yaradın (şəkilləri yerləşdirmək üçün `pictureBox` komponentindən istifadə edin).



Şəkil 8.7. 6 №-li tapşırığa əlavə

7. Forma üzərinə `tabControl` komponenti yerləşdirin. Onun səhifələrini `A massivi` və `listbox1-dən seçilmiş elementlər` adlandırın. `tabControl` komponentinin hər səhifəsinə bir `listBox` komponenti yerləşdirin. Təsadüfi ədədlərdən ibarət A(20) massivi yaradın. `A massivi` adlı səhifəni açdıqda həmin massiv `listBox1` komponentinə köçürülsün. `listbox1-dən seçilmiş elementlər` adlı səhifəni açdıqda `listbox1`-dən seçilmiş elementlər `listbox2`-yə köçürülsün.

8. Forma üzərinə `tabControl` komponenti yerləşdirin. Onun səhifələrini `A massivi` və `listbox1-dən seçilmiş elementlər` adlandırın. `tabControl`

komponentinin birinci səhifəsinə `listBox` komponenti, ikinci səhifəsinə isə iki `radiobutton` dəyişdiriciləri, onların altında isə `listBox2` komponenti yerləşdirin. Bu dəyişdiriciləri "`10-dan kiçik elementlər`" və "`15-dən böyük elementlər`" adlandırın. Təsadüfi ədədlərdən ibarət A(20) massivi yaradın. A massivi səhifəsini açdıqda həmin massiv `listBox1` komponentinə köçürülsün. "`10-dan kiçik elementlər`" dəyişdiricisini seçdikdə 10-dan kiçik elementlər, "`15-dən böyük elementlər`" dəyişdiricisini seçdikdə isə 15-dən böyük elementlər `listBox2` komponentinə köçürülsün.

9. 20 elementdən ibarət A massivinin ən böyük elementini tapıb onun yerini birinci elementlə dəyişdirin.

10. 15 elementdən ibarət A massivinin ən kiçik elementini tapıb onun yerini sonuncudan əvvəlki elementlə dəyişdirin.

11. 20 elementdən ibarət A massivi verilmişdir. Yeni B massivinin elementlərini $B_i = 0.5 * A_i^3 - 2 * A_i + 5$ düsturu ilə hesablayıb siyahı komponentində göstərin. Digər siyahıda həmin massivin mənfi elementlərini təsvir edin.

12. 20 elementdən ibarət A massivinin mənfi elementlərini kvadrata yüksəldin, müsbət elementlərinin üzərinə 5 əlavə edin, sıfra bərabər elementlərini isə olduğu kimi saxlayın. İlkin və hesablanmış massivlərin hər ikisini göstərin.

13. 30 tam ədədlərdən ibarət A massivinin 5-ə qalıqsız bölünən elementlərinin cəmini hesablayın.

14. 30 tam ədədlərdən ibarət A massivinin mənfi və tək elementlərinin cəmini hesablayın.

15. 30 tam ədədlərdən ibarət A massivinin 5-ə qalıqsız bölünən 7-yə isə bölünməyən elementlərinin cəmini hesablayın.

16. 30 həqiqi ədədlərdən ibarət A massivi verilmişdir. Mənfi elementlərin və [2,4] parçasına düşən elementlərin sayını tapın.

17. 30 tam ədədlərdən ibarət A massivi verilmişdir. R=S+P hesablayın. Burada, S – cüt elementlərin cəmi, P isə 1-dən böyük tək elementlərin hasilidir.

18. 20 natural ədədlərdən ibarət A massivi verilmişdir. 7-yə böldükdə qalığı 1.2 və ya 5-ə bərabər olan elementləri tapın.

19. 15 elementdən ibarət A massivi verilmişdir. Birinci mənfi elementdən sonrakı elementlərin hasilini hesablayın.

20. 15 elementdən ibarət A massivi verilmişdir. Birinci mənfi elementdən əvvəlki elementlərin hasilini hesablayın.

21. 30 tam ədədlərdən ibarət A massivinin cüt elementlərinin cəmini hesablayın.

22. 30 ədəddən ibarət A massivinin bütün müsbət elementlərini kvadrata yüksəltmək, mənfi elementlərini isə 2-yə vurmaq lazımdır.

23. 30 ədəddən ibarət A massivinin bütün mənfi elementlərini 3 ilə əvəz edin.

24. [-20, 50] aralığında təsadüfi ədədlər massivi yaradıb, onun ən kiçik tək elementini tapın.

25. 15 elementdən ibarət A massivi verilmişdir. Birinci mənfi elementindən əvvəlki elementlərin ədədi ortasını tapın.

26. C və D massivlərinin uyğun elementlərinin cəmindən A massivi düzəldin.

27. C massivinin mənfi elementlərindən D massivi düzəldin.

28. A massivinin 9-dan kiçik elementlərinin cəmini və sayını tapın.

29. A massivinin müsbət elementlərinin ən kiçiyini tapın.

30. A massivin tək indeksli elementlərinin ədədi ortasını tapın.

LABORATORİYA İŞİ № 9.
DİALOQLARLA İŞ

İşin məqsədi Windows əlavələrində geniş tətbiq edilən dialoq idarəetmə elementlərinin xassələrini və əlavələrin layihələndirilməsində onların tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat
`openFileDialog` idarəetmə elementi

`openFileDialog` idarəetmə elementi *Open (Открыть)* standart dialoq pəncərəsindən ibarətdir və faylları açmaq üçün nəzərdə tutulmuşdur. Bu komponent qeyri-vizual komponentdir. Onun əsas xassələri aşağıdakılardır:

❖ `Title` – açılacaq standart pəncərənin başlığını müəyyən edir. Əgər bu xassəyə qiymət verilməzsə, onda pəncərənin başlığı `Open` (`Открыть`) mətnindən ibarət olur;

❖ `DefaultExt` – genişlənmiş hissə susmaya görə müəyyən edilir (genişlənmiş hissənin qarşısında nöqtə qoyulmur);

❖ `Filter` – dialoq pəncərəsində təsvir ediləcək fayllar üçün filtri (maskanı) müəyyən edir. Hər filtr üçün əvvəlcə onun izahı, sonra isə şaquli xətdən sonra, faylların maskasından ibarət filtrin özü yazılır. Əgər bir neçə maska olarsa, onda onların arasında nöqtəli-vergül simvolu yazılmalıdır. Şaquli xətt simvolu filtrləri bir-birindən ayırmaq üçün istifadə edilir, məsələn:

```
Mətn faylları|*.txt; *.doc;*.docx|
   Şəkil faylları |*.bmp;*.jpg;*.jpeg|
   Veb-səhifələr|*.htm;*.html;*.mht).
```

Bu təsvir *Тип файла* sahəsində əks olunur. Pəncərədə yalnız o faylların siyahısı təsvir olunur ki, onların adları maskaya

uyğun gəlir. Məsələn, əgər Filter xassəsinə |*.doc qiyməti verilərsə, onda siyahıda genişlənmiş hissəsi yalnız doc olan fayllar təsvir ediləcəkdir;

❖ FilterIndex – əgər filtr bir neçə elementdən ibarət olarsa (məsələn, Mətn|*.txt|Bütün fayllar|*.*), onda xassənin qiyməti ekranda dialoq peyda olan anda istifadə edilən filtri müəyyən edir;

❖ FileName – istifadəçi tərəfindən daxil edilmiş və ya fayllar siyahısından seçilmiş faylın adını müəyyən edir;

❖ MultiSelect – əgər bu xassəyə true qiyməti verilərsə, onda istifadəçi bir neçə faylı açа bilər;

❖ ShowReadOnly – əgər bu xassəyə true qiyməti verilərsə, onda dialoq pəncərəsində yalnız oxumaq üçün açıla biləcək faylların siyahısı təsvir ediləcəkdir.

Forma üzərinə əlavə edilmiş openFileDialog komponenti forma üzərində deyil, Windows Forms konstruktorunun aşağısındakı sahədə yerləşir. Bu komponentin ShowDialog metodu proqramın icrası zamanı dialoq pəncərəsini göstərmək üçün istifadə edilir. Yadda saxlamaq lazımdır ki, openFileDialog komponenti yalnız faylı seçməyə imkan verir, faylı açmaq üçün isə faylın açılması alqoritmi işlənməlidir.

Misal. Seçilmiş faylın məzmununun textBox komponentinə yüklənməsi.

Forma üzərinə openFileDialog, button və textBox komponentləri yerləşdirək. Bilirik ki, textBox1 komponentinə adi halda yalnız bir sətir yerləşdirmək olar. Biz isə bu komponentə hər hansı faylın məzmununu (64 kilobaytdan çox olmamaq şərti ilə) yükləyəcəyik. Ona görə də bu komponentin MultiLine xassəsinə və faylın məzmununun sətirdən-sətrə keçməsi üçün AllowDrop xassəsinə true qiyməti vermək lazımdır. Bu zaman, təbii ki, bütün sətirlər görünməyəcək. Ona görə də textBox1

komponentinin ScrollBars xassəsinə Vertical qiyməti verməliyik ki, şaquli fırlatma zolağı görünsün.

openFileDialog1 komponentinin isə xassələrini belə dəyişdirək. Açılacaq dialoq pəncərəsinin başlığını (Title) Faylların açılması adlandıraq, faylın adı sətrinin ilkin anda boş olması üçün (FileName) onun adını pozaq (susmaya görə ona openFileDialog1 adı verilmişdir) və mətn tipli faylların açılması üçün (Filter) *.txt maskası yazaq. Bütün bu xassələri Properties (Свойства) pəncərəsində müəyyən etmək olar, biz isə bunu kodlar vasitəsi ilə edəcəyik.

İndi isə faylın açılması prosesinə baxaq. Hər şeydən əvvəl qeyd edək ki, verilənlərin mətn tipli fayllardan oxunması üçün StreamReader mətn axını istifadə edilir. Form1.cs faylının məzmununda faylların daxil və xaric edilməsi üçün adlar fəzası qoşulmamışdır. Ona görə də həmin faylın using bölməsinə

Using System.IO; sətrini əlavə etmək lazımdır. StreamReader axınının konstruksiyasında verilənlərin oxunacağı faylın adı göstərilməlidir. Mətni textBox1 komponentinə yüklədikdə StreamReader sinfinin ReadToEnd() metodundan istifadə edəcəyik. Bu metod faylın bütün məzmununu oxuyur və onu bir sətir şəklində qaytarır.

İndi isə yazdığımız kodu göstərək:

```
using System.IO;
using System;
using System.Windows.Forms;

namespace OpenFileDIALOG
{
public partial class Form1 : Form
{
```

```
public Form1()
{
InitializeComponent();
openFileDialog1.Title = "Faylların
                        açılması";
openFileDialog1.FileName = "";
openFileDialog1.Filter = "Mətn
               faylları | *.txt";
openFileDialog1.DefaultExt = "txt";
textBox1.Multiline = true;
textBox1.AllowDrop = true;
textBox1.ScrollBars =
         ScrollBars.Vertical;
}
private void button1_Click(object
              sender, EventArgs e)
{
textBox1.Clear();
if (openFileDialog1.ShowDialog()==
                DialogResult.OK)
{
string faylin_adi=
        openFileDialog1.FileName;
StreamReader sr=
          new StreamReader(faylin_adi);
textBox1.Text = sr.ReadToEnd();
sr.Close();
}}}}
```

Burada, `openFileDialog1` komponenti ilə əlaqədar dialoq pəncərəsini göstərmək üçün bu komponentin `ShowDialog` metodu çağrılmışdır. Fayl seçildikdən sonra, dialoq pəncərəsi *Open* (*Открыть*) düyməsi ilə bağlandıqda, bu metod `DialogResult.OK` qiymətini qaytarır və seçilmiş faylın adı `FileName` xassəsində yadda saxlanır. Əgər dialoq

pəncərəsi *Cancel* (*Отмена*) düyməsi ilə bağlanarsa, onda bu metod `DialogResult.Cancel` qiymətini qaytarır.

Proqram belə icra olunur. `button1` (`Open`) düyməsini basdıqda dialoq pəncərəsi açılacaqdır. Diqqət yetirin ki, bu pəncərənin başlığı, təsvir ediləcək faylların tipi bizim yazdığımız kodlar əsasında reallaşmışdır. Bu pəncərədən mətn tipli fayl seçib *Open* (*Открыть*) düyməsini basdıqda seçdiyimiz faylın məzmunu `textbox1` komponentinə yüklənəcəkdir (şəkil 9.1).

Şəkil 9.1. Dialoq pəncərəsindən seçilmiş faylın məzmununun `textbox1` komponentinə yüklənməsi

`saveFileDialog` idarəetmə elementi

`saveFileDialog` idarəetmə elementi `openFileDialog` idarəetmə elementinə çox oxşayir. O, *Save (Сохранить)* standart dialoq pəncərəsindən ibarətdir və faylları yadda saxlamaq üçün nəzərdə tutulmuşdur. Bu komponent də qeyri-vizual komponentdir. Onun əsas xassələri aşağıdakılardır:

❖ `Title` – açılacaq standart pəncərənin başlığını müəyyən edir;

Əgər bu xassəyə qiymət verilməzsə, onda pəncərənin başlığı `Save As` (`Сохранить как`) mətnindən ibarət olur.

❖ `FileName` – istifadəçinin fayla verdiyi tam addır. Ümumi halda bu ad məzmunu dialoq pəncərəsində təsvir olunan qovluğun adından, istifadəçinin *Name file* (*Имя файла*) sahəsindən daxil etdiyi faylın adından və `DefaultExt` xassəsində göstərilən genişlənmiş hissədən əmələ gəlir.

❖ `InitialDirectory` – ekranda dialoq peyda olanda məzmunu təsvir edilən qovluğu müəyyən edir;

❖ `CheckPathExists` – faylın saxlanacağı qovluğun mövcudluğunun yoxlanılması əlamətini müəyyən edir. Əgər göstərilən qovluq mövcud olmazsa, onda bu barədə məlumat verilir;

❖ `CheckFileExists` – göstərilən adda faylın mövcudluğunun yoxlanılması əlamətini müəyyən edir. Əgər xassənin qiyməti `true` olarsa və göstərilən adda fayl mövcuddursa, onda sorğu pəncərəsi peyda olur, istifadəçi burada mövcud faylın dəyişdirilməsini təsdiqləyir.

Yadda saxlamaq lazımdır ki, `saveFileDialog` komponentini istifadə etdikdə faylın saxlanması üçün xüsusi alqoritm işlənməlidir. `OpenFile` metodunun köməyi ilə faylı oxumaq və yazmaq rejimlərində açmaq olar.

Misal. `textBox` komponentinə yüklənmiş faylın yadda saxlanılması.

`openFileDialog` komponentinin izahında həll etdiyimiz layihəni bir az təkmilləşdirək, yəni `textBox1` komponentindəki mətni yadda saxlayaq. Beləliklə, həmin layihəni açıb, formaya `saveFileDialog` komponenti yerləşdirək. Bu komponent üçün də `Title`, `FileName`,

`Filter` və `DefaultExt` xassələrinə `openFileDialog` komponentinə uyğun qiymətlər verək.

İndi isə faylın açılması prosesinə baxaq. `StreamWriter` sinfinin köməyi ilə faylın `textBox1` komponentinə yazılması üçün ayrıca `FaylaYaz` metodu yaradaq. `StreamReader` axınının konstruksiyasında verilənlərin yazılacağı faylın adı, yazılma rejimi (`false` –verilənlərin fayla yenidən yazılması və ya `true` – faylın məzmununa yeni verilənlərin əlavə edilməsi) və bu fayla verilənlərin yazılması formatı (`Encoding.Default`) göstərilməlidir. Sonra isə verilənlər axını `WriteLine` metodu ilə `textBox1` komponentinə yazılır və axın bağlanır.

İndi isə forma üzərinə `button2` (`Save As..`) və `button3` (`Save`) komponentləri yerləşdirək. `button2` düyməsini basdıqda fayl başqa adla, `button3` düyməsini basdıqda isə fayl verilmiş adla yadda saxlanacaqdır. Əgər fayl ilk dəfə yadda saxlanarsa, onda `button3` (`Save`) düyməsini basdıqda fayla adın verilməməsi haqqında ismarıc göndəriləcəkdir. `button2` və `button3` düymələri üçün `Click` hadisə emaledicilərində `FaylaYaz` metodu çağrılacaqdır. Layihənin tam mətni belə olacaqdır:

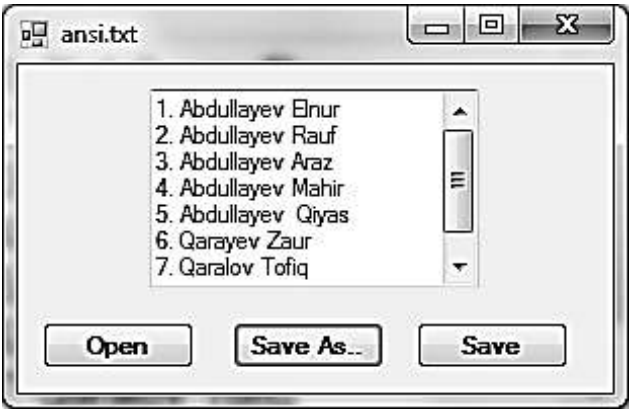
```
using System.IO;
using System;
using System.Windows.Forms;

namespace SaveFileDIALOG
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
```

```
openFileDialog1.Title = "Faylların
                      açılması";
openFileDialog1.FileName = "";
openFileDialog1.Filter = "Mətn
               faylları | *.txt";
openFileDialog1.DefaultExt = "txt";
saveFileDialog1.Title = "Faylların
                     saxlanması";
saveFileDialog1.FileName = "";
saveFileDialog1.Filter = "Mətn
               faylları | *.txt";
saveFileDialog1.DefaultExt = "txt";
textBox1.Multiline = true;
textBox1.AllowDrop = true;
textBox1.ScrollBars =
          ScrollBars.Vertical;
}

private void FaylaYaz(string faylin_adi)
{
StreamWriter sw =
         new StreamWriter(faylin_adi);
sw.WriteLine(textBox1.Text);
sw.Close();
}

private void button1_Click(object
              sender, EventArgs e)
{
textBox1.Clear();
if (openFileDialog1.ShowDialog()==
                 DialogResult.OK)
{
```

```
string faylin_adi =
        openFileDialog1.FileName;
StreamReader sr =
        new StreamReader(faylin_adi);
textBox1.Text = sr.ReadToEnd();
sr.Close();
}
}

private void button2_Click(object
              sender, EventArgs e)
{
if (saveFileDialog1.ShowDialog()==
              DialogResult.OK)
{
string faylin_adi =
            saveFileDialog1.FileName;
FaylaYaz(faylin_adi);
}
}

private void button3_Click(object
              sender, EventArgs e)
{
string faylin_adi =
         saveFileDialog1.FileName;

if (faylin_adi == "")
{
MessageBox.Show("Fayla ad
             verilməmişdir");
return;
}
else
```

```
FaylaYaz(faylin_adi);
}
}
}
```

Proqramın nəticəsi şəkil 9.2-də göstərilmişdir.



Şəkil 9.2. `textBox` komponentinə yüklənmiş faylın yadda saxlanılması

Hazır layihədə `Open` düyməsini basdıqda hər hansı bir faylın məzmunu `textBox1` komponentinə oxunur. `Save As..` və `Save` düymələrini istifadə etdikdə isə hər hansı faylın `textBox1` komponentinə oxunması vacib deyil, Siz özünüz bu komponentə klaviaturadan verilənlər yaza bilərsiniz. `Save As..` düyməsini basdıqda `Faylların saxlanması` dialoq pəncərəsi açılacaqdır. Bu pəncərədə fayla ad verib, *Сохранить* düyməsini basdıqda `textBox1` komponentinin məzmunu bu faylda yadda saxlanacaq və həmin ad proqramın (formanın) başlığında təsvir olunacaqdır. `Save` düyməsini basdıqda isə, əgər faylın adı varsa, onda redaktorda edilmiş düzəlişlər faylda yadda saxlanacaq, əgər fayla ad

verilməmişdirsə, ekranda bu barədə məlumat təsvir olunacaqdır.

`fontDialog` idarəetmə elementi

`fontDialog` idarəetmə elementi şriftin parametrlərini idarə etməyə imkan verən standart dialoq pəncərəsini ekranda təsvir etdirmək üçün nəzərdə tutulmuş qeyri-vizual komponentdir. Bu komponentin əsas xassələri aşağıdakılardır:

- `Name` – komponentin adını müəyyən edir (susmaya görə – `fontDialog`);
- `Font.Name` – şriftin adını müəyyən edir;
- `Font.Size` – şriftin ölçüsünü müəyyən edir;
- `Font.Bold`, `Font.Italic`, `Font.Strikeout`, `Font.Underline` – şriftin üslubunu (uyğun olaraq, yarımqalın, kursiv, üzərindən və altdan xətt çəkilmiş) müəyyən edir. Müvafiq üslubu seçmək üçün ona `true` qiyməti vermək lazımdır;
- `Color` – dialoq pəncərəsindən seçilmiş rəngi müəyyən edir;
- `MaxSize` və `MinSize` – şriftin maksimal və minimal ölçüsünü müəyyən edir;
- `ShowColor` – şriftin rəngini seçməyə imkan verir.

Bu komponentin `ShowDialog` metodu proqramın icrası zamanı dialoq pəncərəsini göstərmək üçün istifadə edilir.

Misal. `textBox` komponenti üçün şriftin seçilməsi.

Yuxarıdakı layihəyə şriftin seçilməsini əlavə edək. Bunun üçün forma üzərinə `fontDialoq` komponenti əlavə edib `InitializeComponent();` metodundan sonra

```
fontDialog1.ShowColor = true;
fontDialog1.MinSize = 8;
fontDialog1.MaxSize = 20;
```

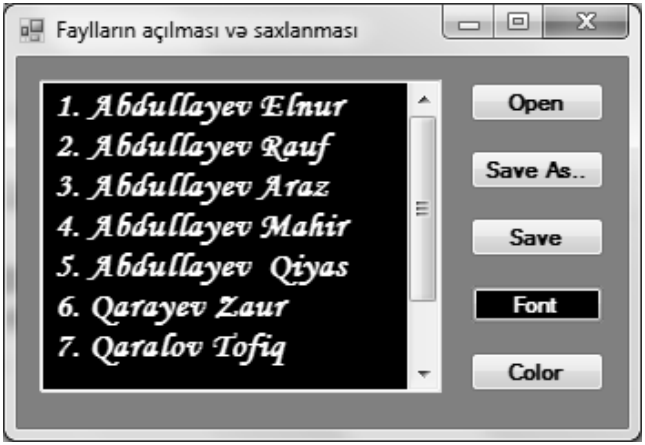
kodlarını yazaq. Birinci kod bizə dialoq pəncərəsində şriftin rəngini seçməyə imkan verəcək. İkinci və üçüncü kodlarla isə

həmin pəncərədə şriftin ölçüsünün dəyişmə intervalı 8 ilə 20 arasında müəyyənləşəcək.

Sonra, button4 (Font) komponenti əlavə edib, aşağıdakı hadisə emaledicisini yaradaq:

```
private void button4_Click(object
                sender, EventArgs e)
{
if (fontDialog1.ShowDialog()==
              DialogResult.OK)
{
textBox1.Font = fontDialog1.Font;
textBox1.ForeColor= fontDialog1.Color
button4.ForeColor = fontDialog1.Color;
}}
```

fontDialog və button4 (Font) komponentləri əlavə edildikdən sonra alınmış layihənin nəticəsi şəkil 9.3-də göstərilmişdir.



Şəkil 9.3. textBox komponenti üçün şriftin və rənginin seçilməsi

Hazır layihədə `Open` düyməsini basaraq `textBox1` komponentinə hər hansı bir mətn yükləyin. `Font` düyməsini basdıqda standart şrift dialoq pəncərəsi açılacaqdır ki, Siz buradan şriftin adını, üslubunu, ölçüsünü, rənginı və s. seçib *Ok* düyməsini basdıqda `textBox1` komponentində müvafiq dəyişikliklər baş verəcək, `Font` düyməsinin özünün mətnində isə yalnız şriftin rəngi dəyişəcəkdir.

`colorDialog` idarəetmə elementi

`colorDialog` idarəetmə elementi rəng seçməyə imkan verən rənglər standart dialoq pəncərəsini ekranda təsvir etdirmək üçün nəzərdə tutulmuş qeyri-vizual komponentdir. Bu komponentin əsas xassələri aşağıdakılardır:

- `Color` – rəng standart dialoq pəncərəsindən seçilmiş rəngi müəyyən edir;
- `FullOpen` – rəng standart dialoq pəncərəsinin görünüşünü müəyyən edir. Əgər bu xassəyə `true` qiyməti verilərsə, onda rənglərin seçilməsi üçün geniş imkanlı pəncərə təsvir edilir;
- `SolidColorOnly` – bu xassəyə `true` qiyməti verdikdə yalnız bir çalarlı rənglər içərisindən rənglər seçməyə imkan verir;

Bu komponentin `ShowDialog` metodu proqramın icrası zamanı rənglər dialoq pəncərəsini göstərmək üçün istifadə edilir.

Misal. Rənglər standart dialoq pəncərəsindən rəngin seçilməsi.

Yuxarıda yaratdığımız layihəyə sonuncu əlavəmizi edək. Formaya `colorDialog` komponenti yerləşdirib `InitializeComponent();` metodundan sonra

```
colorDialog1.FullOpen = true;
colorDialog1.Color = this.BackColor;
```

kodlarını yazaq.

Birinci kod bizə geniş imkanlı dialoq pəncərəsini ekranda təsvir etməyə imkan verəcək. İkinci kodla isə `colorDialog1` komponenti üçün başlanğıc rəng müəyyən edilir.

Sonra, `button5` (`Color`) düyməsi əlavə edib, aşağıdakı hadisə emaledicisini yaradaq:

```
private void button5_Click(object
                sender, EventArgs e)
{

if (colorDialog1.ShowDialog()==
             DialogResult.Cancel)
return;

else
{
this.BackColor = colorDialog1.Color;
textBox1.BackColor= Color.Black;
button4.BackColor = Color.Black;
} }
```

Bu kodlarla `colorDialog1` rəng dialoq pəncərəsi ekranda təsvir ediləcək. Bu pəncərədən seçilmiş rənglə formanın (`Form1`), `textBox1` və `button4` (`Font`) düyməsinin səthi rənglənəcəkdir.

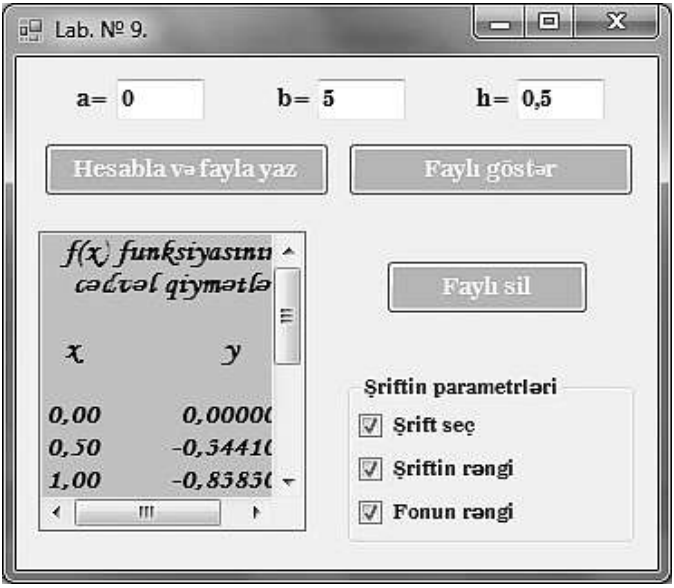
Proqramın nəticəsi şəkil 9.3-də göstərilmişdir.

Misal. $y = x^{2/3}\cos(x + e^x)$ funksiyasını $[a, b]$ parçasında h addımı ilə hesablayıb nəticəni mətn tipli fayla yazın. Sonra bu faylın məzmununu siyahı komponentində göstərin və dəyişdiricilərdən istifadə etməklə şrifti, onun fonunu və rəngini dəyişdirin. Bütün bu əməliyyatların hamısını dialoq

komponentlərindən istifadə etməklə yerinə yetirin. Sonda faylı diskdən silin.

Məsələnin həlli üçün şəkil 9.4-də göstərilən layihəni yaradaq. `Hesabla və fayla yaz` düyməsini basdıqda göstərilən parçada verilmiş addımla funksiyanın qiymətləri hesablanır və diskdə fayl yaradılır. `Faylı göstər` adlı düyməni basdıqda dialoq pəncərəsi açılacaq və buradan həmin faylı seçdikdə `listBox1` komponentində təsvir olunacaqdır. Sonra müvafiq `checkBox` komponentlərini qoşmaqla şriftin ölçüsünü, rəngini və fonun rəngini dəyişdirmək mümkün olur. Faylın məzmununu `listBox1` komponentinə yüklədikdə `StreamReader` sinfinin `ReadLine()` metodundan istifadə edəcəyik. Bu metod faylı sətirbəsətir oxumağa imkan verir.

Məsələnin həlli şəkil 9.4-də, kodu isə aşağıda göstərilmişdir.



Şəkil 9.4. 9 №-li laboratoriya işinin nəticəsi

```
using System;
using System.IO;
using System.Windows.Forms;

namespace Lab_9
{
public partial class Form1 : Form
          {
public Form1()
{
  InitializeComponent();
  openFileDialog1.Title = "Faylların
                          açılması";
  openFileDialog1.FileName = "";
  openFileDialog1.Filter = "Mətn
              faylları | *.txt";
  openFileDialog1.DefaultExt = "txt";
}

private void button1_Click(object
             sender, EventArgs e)
{

 double a, b, h, x, y;
  a = Convert.ToDouble(textBox1.Text);
  b = Convert.ToDouble(textBox2.Text);
  h = Convert.ToDouble(textBox3.Text);
  StreamWriter mf =
         File.CreateText("F:fun.txt");
  mf.WriteLine("  f(x) funksiyasının");
  mf.WriteLine("   cədvəl qiymətləri");
  mf.WriteLine();
  mf.WriteLine("  x                y");
  mf.WriteLine();
```

```
  x = a;
  while(x<=b)
  {
   y = Math.Pow(x, 2.0 / 3.0) *
       Math.Cos(x + Math.Exp(x));
   y = Math.Round(y, 4);
   mf.WriteLine("{0,3:F} {1,15:F5}",
                              x, y);
   x += h;
  }
  mf.Close();
}

private void button2_Click(object
              sender, EventArgs e)
{
  listBox1.Items.Clear();
  if (openFileDialog1.ShowDialog() ==
                 DialogResult.OK)
  {
   string faylın_adı =
          openFileDialog1.FileName;
   StreamReader sr =
          new StreamReader(faylın_adı);
   string sətirlər;
   while ((sətirlər = sr.ReadLine()) !=
                                 null)
   {
    listBox1.Items.Add(sətirlər);
   }
  }
}
private void
checkBox1_CheckedChanged(object sender,
```

```
                                EventArgs e)
{
  if (fontDialog1.ShowDialog() ==
                    DialogResult.OK)
  {
    listBox1.Font = fontDialog1.Font;
    listBox1.ForeColor =
                    fontDialog1.Color;
  }
}

private void checkBox2_CheckedChanged(
          object sender, EventArgs e)
{
  if (colorDialog1.ShowDialog() ==
                   DialogResult.OK)
  listBox1.ForeColor =
            colorDialog1.Color;
}

private void checkBox3_CheckedChanged(
           object sender, EventArgs e)
{
  if (colorDialog1.ShowDialog() ==
                    DialogResult.OK)
  listBox1.BackColor =
            colorDialog1.Color;
}

private void button3_Click(object
              sender, EventArgs e)
{
  File.Delete("F:fun.txt");
} } }
```

Yoxlama sualları

1. Dialoq elementləri hansılardır?
2. `openFileDialog` idarəetmə elementi nə üçündür?
3. `saveFileDialog` idarəetmə elementi nə üçündür?
4. `fontDialog` idarəetmə elementi nə üçündür?
5. `colorDialog` idarəetmə elementi nə üçündür?
6. Açılacaq standart pəncərənin başlığını müəyyən edən xassə hansıdır?
7. `Filter` xassəsi nə üçündür?
8. `MultiSelect` xassəsi nə üçündür?
9. `ShowReadOnly` xassəsi nə üçündür?
10. `DialogResult.OK` və `DialogResult.Cancel` qiymətləri nəyi bildirir?
12. `InitialDirectory` xassəsi nə üçündür?
13. `CheckPathExists` xassəsi nə üçündür?
14. `CheckFileExists` xassəsi nə üçündür?
15. Şriftin üslubu necə müəyyən edilir?

Laboratoriya işlərinə aid tapşırıqların variantları

Tapşırıq variantlarında verilmiş funksiyanı verilmiş $x \in [a, b]$ parçasında h addımı ilə hesablayın və şəkil 9.4-də göstərilmiş layihəyə analoji layihə hazırlayın.

Variant №	Funksiya	Arqumentin dəyişmə diapazonu
1.	$y = cosx$	$-3 \leq x \leq 3$
2.	$y = sinx$	$-3 \leq x \leq 3$
3.	$y = tgx$	$0 \leq x \leq 10$
4.	$y = ctgx$	$0 \leq x \leq 10$
5.	$y = lnx$	$0 \leq x \leq 20$

6.	$y = logx$	$5 \le x \le 15$
7.	$y = cosx/sin(x + 2\pi)$	$-5 \le x \le 5$
8.	$y = e^{sin2\pi x}$	$0 \le x \le 30$
9.	$y = e^{cos2\pi x}$	$5 \le x \le 25$
10.	$y = \sqrt[3]{\cos x}$	$25 \le x \le 50$
11.	$y = \sqrt{\lvert\cos x - \sin x\rvert}$	$-18 \le x \le 0$
12.	$y = \sqrt{\lvert\ln(\cos x - \sin x)\rvert}$	$0 \le x \le 40$
13.	$y = cos(x^3)$	$10 \le x \le 20$
14.	$y = cos^4(\mathrm{x})$	$25 \le x \le 50$
15.	$y = (cosx \cdot sinx)^{2/3}$	$0 \le x \le 60$
16.	$y = 1/cos^4 x$	$50 \le x \le 80$
17.	$y = 1/cos^4 x^2$	$6 \le x \le 24$
18.	$y = 1/e^{sin2x}$	$16 \le x \le 36$
19.	$y = 1/e^{tg5x}$	$1 \le x \le 8$
20.	$y = e^{tg5x}/\sqrt[3]{\cos x^2}$	$20 \le x \le 60$
21.	$y = e^{tg2\pi x}/\sqrt[3]{\cos \pi x^2}$	$9 \le x \le 18$
22.	$y = logx/e^{sin3x} + 2\pi x$	$0 \le x \le 1$
23.	$y = e^{\cos x - sin3\pi x}$	$-7 \le x \le 7$
24.	$y = 1/e^{\cos x - sin3\pi x} \cdot \cos x$	$10 \le x \le 40$
25.	$y = 1/e^{tg2x} \cdot \cos(x+8\pi)$	$4 \le x \le 12$
26.	$y = cos(1/e^{tg3x})$	$5 \le x \le 10$
27.	$y = cos(e^{sin2x})$	$-2 \le x \le 2$
28.	$y = sin(e^{sin2\pi x - 2cosx})$	$-5 \le x \le 5$
29.	$y = cos^4 x^2$	$12 \le x \le 18$
30.	$y = \sqrt[6]{e^{\ln(\lvert sinx\rvert)}}$	$6 \le x \le 14$

LABORATORİYA İŞİ № 10.
MENYULARLA İŞ

İşin məqsədi Windows sistemində və bütün Windows əlavələrində geniş tətbiq edilən menyu idarəetmə elementlərinin xassələrini və əlavələrin layihələndirilməsində onların tətbiq edilmə xüsusiyyətlərini öyrənməkdən ibarətdir.

Nəzəri məlumat

Menyuların yaradılması üçün nəzərdə tutulmuş komponentlər `Toolbox` (`Панель элементов`) elementlər panelinin `Menus & Toolbars` (`Меню и панели инструментов`) bölməsində yerləşir. Bu paneldə menyu yaratmaq üçün əsas komponentlər aşağıdakılardır:

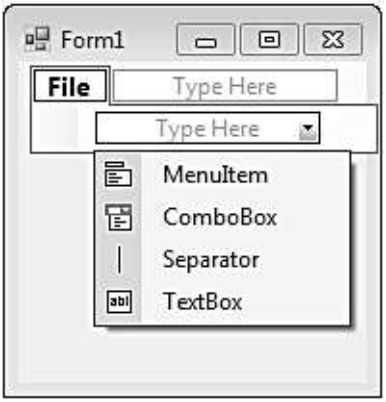
- `menuStrip` – əsas menyunun yaradılması;
- `contexMenuStrip` – kontekst menyunun yaradılması;
- `toolStrip` – alətlər panelinin yaradılması;
- `statusStrip` – vəziyyətlər sətrinin yaradılması.

`menuStrip` idarəetmə elementi

Menyu yaratmaq üçün forma üzərinə `menuStrip` komponenti yerləşdirək. Bu zaman formadan aşağıda yerləşən komponentlər panelində `menuStrip1` komponenti peyda olacaqdır. Formanın yuxarı hissəsində isə daxilində `Type Here` (`Вводить здесь`) yazısı olan sahə əmələ gələcəkdir. Bu sahəyə `Fayl` yazıb *Enter* düyməsini basaq. Bu ilk menyu bəndi olacaqdır. İndı həmin menyudan aşağıda və sağda yenidən `Type Here` (`Вводить здесь`) yazısı olan sahə əmələ gələcəkdir (şəkil 10.1). Aşağıdakı və ya sağdakı sahələrdə mausun düyməsini basıb növbəti menyu bəndlərini

yaratmaq olar. Sahələrdə aşağıya istiqamətlənmiş ox üzərində mausun düyməsini basdıqda elementlərdən ibarət açılan menyu peyda olacaq. Biz burada yerləşən aşağıdakı komponentlərdən istifadə edə bilərik:

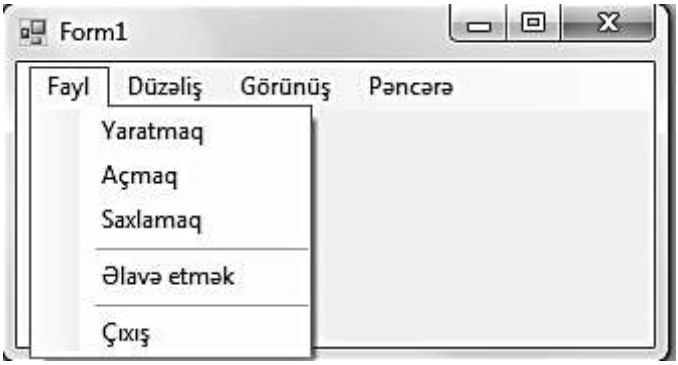
- `MenuItem` – susmaya görə yaradılan menyu elementi;
- `ComboBox` – açılan siyahı. Biz açılan siyahını menyu bəndi kimi yarada bilərik;
- `Separator` – alt menyuları qruplara bölmək üçün düz xətdən ibarət ayrıcı;
- `TextBox` – daxiletmə sahəsi. Biz daxiletmə sahəsini menyu bəndi kimi yarada bilərik.



Şəkil 10.1. `menuStrip1` komponenti

`Fayl` menyusunun alt menyularını yaradaq. Bunun üçün aşağıda yerləşən `Type Here` (`Вводить здесь`) sahəsinə yeni bəndlərin adını yazaq (`Yaratmaq...`, `Açmaq...`, `Saxlamaq...`). Bu bəndlərdən sonra ayrıcı yerləşdirək. Bunun üçün boş sahədə aşağıya istiqamətlənmiş ox üzərində mausun düyməsini basıb `Separator` seçmək lazımdır. Sonra

`Əlavə etmək` menyusunu ayıraraq `Çıxış` menyusunu yaradaq (şəkil 10.2).

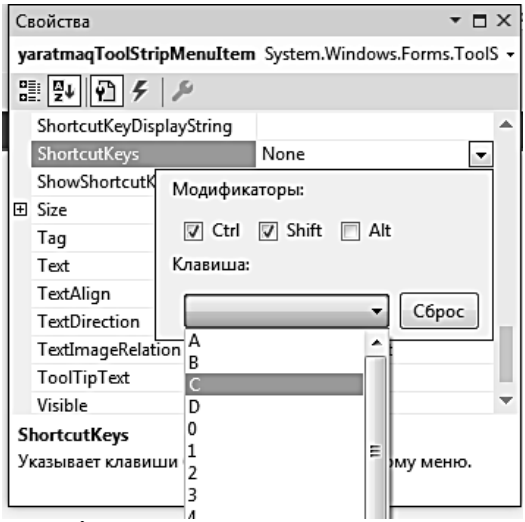


Şəkil 10.2. Menyuların yaradılması

Menyulara şəkillər də əlavə etmək olar. Bunun üçün menyu bəndini seçib `Image` xassəsinə şəkil yükləmək lazımdır.

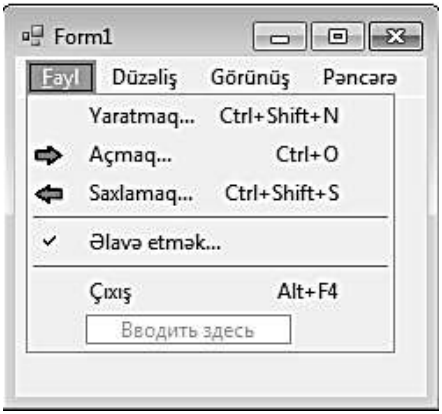
Bütün Windows əlavələrində və eləcə də əməliyyat sisteminin özündə menyularla yanaşı isti (qaynar) klavişlər də istifadə edilir. İndi menyu bəndlərinə isti klavişlər kombinasiyasını əlavə edək. Bunun üçün lazım olan simvolun qarşısına `&` simvolunu əlavə etmək lazımdır. Məsələn, `Fayl` menyusunda `F` simvolunu isti simvol etmək üçün menyu bəndinə `&Fayl` yazmaq lazımdır. Bu menyu yaradıldıqdan sonra `F` simvolunun altından xətt çəkiləcəkdir, yəni, menyu bəndi `Fayl` kimi təsvir ediləcəkdir. Hazır layihədə isə *Alt+F* klavişlərini basdıqda `Fayl` menyusu açılacaqdır. Digər klavişlər kombinasiyasını yaratmaq üçün menyu bəndini seçib `ShortcutKeys` xassəsindən istifadə etmək lazımdır. Bu xassənin qarşısındakı aşağıya istiqamətlənmiş ox təsvirli düyməni basmaqla açılan siyahıdan klavişlər kombinasiyasını seçə bilərik (şəkil 10.3). Klavişlər kombinasiyasını gizlətmək

üçün hər bir menyu bəndinin `ShowShortcutKeys` xassəsinə `false` qiyməti vermək lazımdır.



Şəkil 10.3. İsti klavişlər kombinasiyasının seçilməsi

Beləliklə, şəkil 10.4-də göstərilmiş formada menyu yarada bilərik.



Şəkil 10.4. Menyulardan ibarət forma

Menyu bəndlərini pozmaq üçün onları seçib *Delete* klavişini basmaq lazımdır.

Menyuların müəyyən əməliyyatları icra etməsi üçün onlar hadisə emalediciləri ilə əlaqələndirilməlidir. Menyular üçün bir qayda olaraq `Click` hadisəsi yaradılır. Ona görə də menyu bəndləri üzərində mausun düyməsini iki dəfə basdıqda avtomatik olaraq `Click` hadisə emaledicisi yaradılır. Biz indiyədək `button` düyməsi üçün yazdığımız proqramların hamısını menyu bəndləri ilə əlaqələndirə bilərik. Burada isə biz yalnız `Çıxış` menyusu üçün metod yaradaq, yəni `Çıxış` menyusu üzərində mausun düyməsini iki dəfə basaraq aşağıdakı metodu yaradaq:

```
private void
  çıxışToolStripMenuItem_Click(
    object sender, EventArgs e)
{
  this.Close();
}
```

Menyuların əsas xassələri aşağıdakılardır:

❖ `Checked` – menyu bəndinin seçilməsini bildirir. Əgər bu xassəyə `true` qiyməti verilərsə, onda menyu bəndinin qarşısında bayraq işarəsi qoyulmaqla seçilir;

❖ `CheckOnClick` –`Checked` xassəsi ilə seçilmiş menyu bəndi üzərində mausun düyməsini basdıqda seçmə ləğv edilir və mausun düyməsini təkrarən basdıqda menyu seçilir;

❖ `CheckState` – `checkBox` komponentinin eyniadlı xassəsinə tamamilə analojidir;

❖ `DisplayStyle` – menyu bəndinin təsvir üslubunu müəyyən edir. Bu xassəyə aşağıdakı qiymətlər seçilə bilər:

- `ImageAndText` – şəkil və mətn;
- `Image` – şəkil;
- `Text` – mətn;

- None – heç nə.

❖ Image – menyu üçün şəkil;

❖ ImageAlign – şəklin hansı tərəfə düzləndirilməsini müəyyən edir;

❖ BackColor – menyu bəndinin fonunun rəngini müəyyən edir;

❖ RenderMode – menyu bəndlərinin fonunun rəngini müəyyən edir. BackColor xassəsinə susmaya görə Control qiyməti verilmişdir, yəni menyunun fonu ağ rənglidir. Fonun rəngini formanın rənginə uyğunlaşdırmaq üçün RenderMode xassəsinə System qiyməti vermək lazımdır;

❖ ImageTransparentColor – şəffaflıq rəngini müəyyən edir;

❖ Enabled – menyunun aktivliyini müəyyən edir. Bu xassəyə false qiyməti verdikdə menyu icra oluna bilmir;

❖ RightToLeft – bu xassəyə Yes qiyməti verdikdə klavişlər kombinasiyası menyu bəndinin sol tərəfində yerləşir.

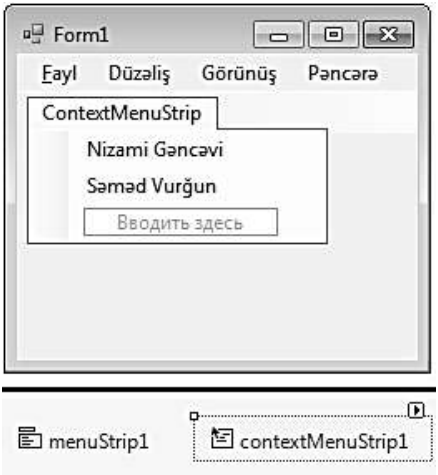
contextMenuStrip idarəetmə elementi

Bu komponent kontekst menyu yaratmaq üçün tətbiq edilir. Kontekst menyu mausun sağ düyməsini basdıqda açılan menyudur.

Kontekst menyu yaratmaq üçün forma üzərinə contextMenuStrip komponenti yerləşdirmək lazımdır. Bu zaman formanın yuxarısında vizual menyu redaktoru əmələ gələcəkdir. Bu redaktor vasitəsi ilə kontekst menyunun yaradılması əsas menyunun yaradılmasından heç nə ilə fərqlənmir. Ona görə də burada yuxarıda şərh etdiyimiz izahları təkrarlamağa ehtiyac yoxdur. Eləcə də hər iki komponentin xassələri, demək olar ki, eynidir. Nizami

`Gəncəvi` və `Səməd Vurğun` adında iki kontekst menyu bəndi yaradaq (şəkil 10.5).

Formanı və ya digər komponenti seçdikdə nəinki, `contextMenuStrip` komponenti, həm də yaratdığımız menyulardan ibarət vizual redaktor yox olur. Onu təkrarən göstərmək üçün `contextMenuStrip` komponentini seçmək lazımdır.



Şəkil 10.5. Kontekst menyu bəndlərinin yaradılması

Bəs yaratdığımız kontekst menyulardan necə istifadə edə bilərik? Sadəcə olaraq kontekst menyunu tətbiq edəcəyimiz formanı və ya komponenti seçib onun `ContextMenuStrip` xassəsinə `contextMenuStrip1` qiymətini seçmək lazımdır.

Misal. Kontekst menyu bəndlərinin tətbiqi.

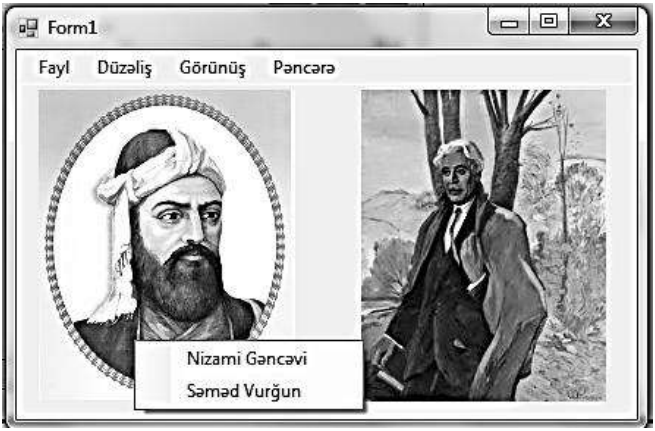
İndi isə yuxarıda yaratdığımız kontekst menyu bəndlərindən istifadə edək. Elə əlavə yaradaq ki, `Nizami Gəncəvi` adlı menyu üzərində mausun düyməsini basdıqda Nizami Gəncəvinin, `Səməd Vurğun` adlı menyu üzərində mausun düyməsini basdıqda isə Səməd Vurğunun şəkilləri ekrana

çıxsın. Bunun üçün forma üzərinə iki `pictureBox` komponentləri yerləşdirək. Hər iki komponentin `ContextMenuStrip` xassəsinə `contextMenuStrip1` qiyməti seçək. Sonra kontekst menyunun `Nizami Gəncəvi` adlı bəndi üzərində mausun düyməsini iki dəfə basıb `Click` hadisə emaledicisini yaratmaq və bu emaledicidə şəklin yüklənməsi kodunu yazmaq lazımdır. Analoji əməliyyatları `Səməd Vurğun` menyu bəndi üçün təkrar etmək lazımdır. Bu proqramın tam mətni belədir:

```
namespace Menyu
{
public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
  pictureBox1.SizeMode =
    PictureBoxSizeMode.Zoom;
  pictureBox2.SizeMode =
    PictureBoxSizeMode.Zoom;
}

private void
  çıxışToolStripMenuItem_Click(
    object sender, EventArgs e)
{
  this.Close();
}

private void
  nizamiGəncəviToolStripMenuItem_Click
      (object sender, EventArgs e)
{
  pictureBox1.Image =
```

```
    Image.FromFile(@"C:\Nizami.jpg");
  }

  private void
  səmədVurğunToolStripMenuItem_Click
    (object sender, EventArgs e)
  {
    pictureBox2.Image =
    Image.FromFile(@"C:\Səməd.jpg");
  }}}
```

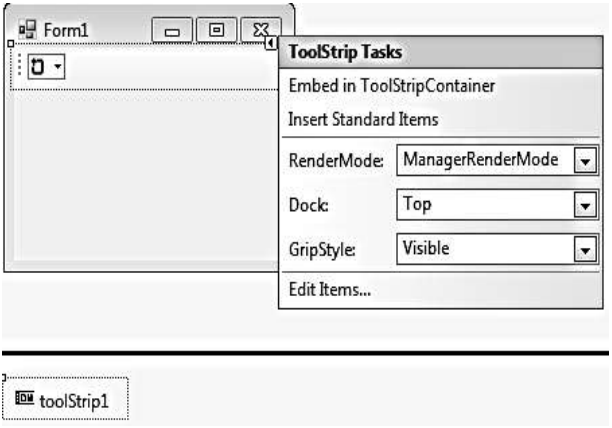
Bu proqramda şəkillərin təhrif olunmaması üçün `pictureBox` komponentlərinin `SizeMode` xassəsinə `Zoom` qiyməti verilmiş və şəkillər `FromFile()` metodu ilə yüklənmişdir. Proqramı icra etdikdən sonra forma üzərində mausun sağ düyməsini basdıqda kontekst menyu peyda olacaq və müvafiq bəndləri seçdikdə Nizami Gəncəvi və Səməd Vurğunun şəkilləri təsvir olunacaqdır (şəkil 10.6).



Şəkil 10.6. Kontekst menyu əmrlərinin icrasının nəticələri

`toolStrip` idarəetmə elementi

`toolStrip` komponenti düymələrdən, mətn sahələrindən, siyahılardan və tez-tez istifadə edilən digər interfeys komponentlərindən ibarət olan alətlər paneli yaratmaq üçün nəzərdə tutulmuşdur. Bir qayda olaraq, belə alətlər, menyu bəndlərinin yerinə yetirdiyi əmrləri əyani olaraq daha tez icra etmək üçün yaradılır. Alətlər paneli yaratmaq üçün forma üzərinə `toolStrip` komponenti yerləşdirmək lazımdır. Bu komponent də forma üzərində deyil, formadan aşağıdakı sahədə yerləşəcək, lakin formanın yuxarı hissəsində alətlər sətri (zolaq) əmələ gələcəkdir. Bu sətrin başlanğıcında açılan siyahıdan ibarət virtual düymə yerləşir. Zolağın sağ tərəfində kiçik kvadrat daxilində düymə yerləşir ki, onu basdıqda `ToolStrip Tasks (ToolStrip Задачи)` açılan menyusu əmələ gəlir. Həmin menyunun əmrləri ilə alətləri əlavə etmək və ya nizamlamaq olar (şəkil 10.7).



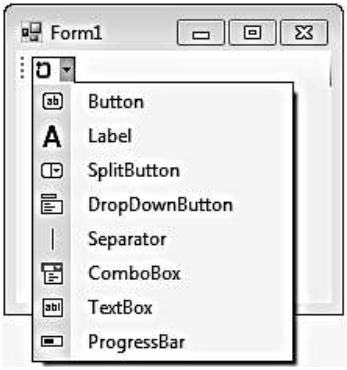
Şəkil 10.7. `ToolStrip Tasks` açılan menyusu

Alətlər paneli yaratmaq üçün alətlər zolağında yerləşən açılan siyahıdan ibarət virtual düymə üzərində mausun

düyməsini basmaq lazımdır. Bu zaman alətlər panelini yaradacaq komponentlərdən ibarət siyahı açılacaqdır (şəkil 10.8).

Bu komponentlər aşağıdakılardır:

- `Button` – düymə;
- `Label` – yazı (nişan);
- `SplitButton` – menyu çağırmaq imkanına malik olan düymə;
- `DropDownButton` – seçim siyahısını çağırmaq imkanı olan düymə;
- `Separator` – paneldə komponentləri ayıran alət (şaquli xətt);
- `ComboBox` – açılan siyahı;
- `TextBox` – mətn sahəsi;
- `ProgressBar` – prosesin indikatoru.

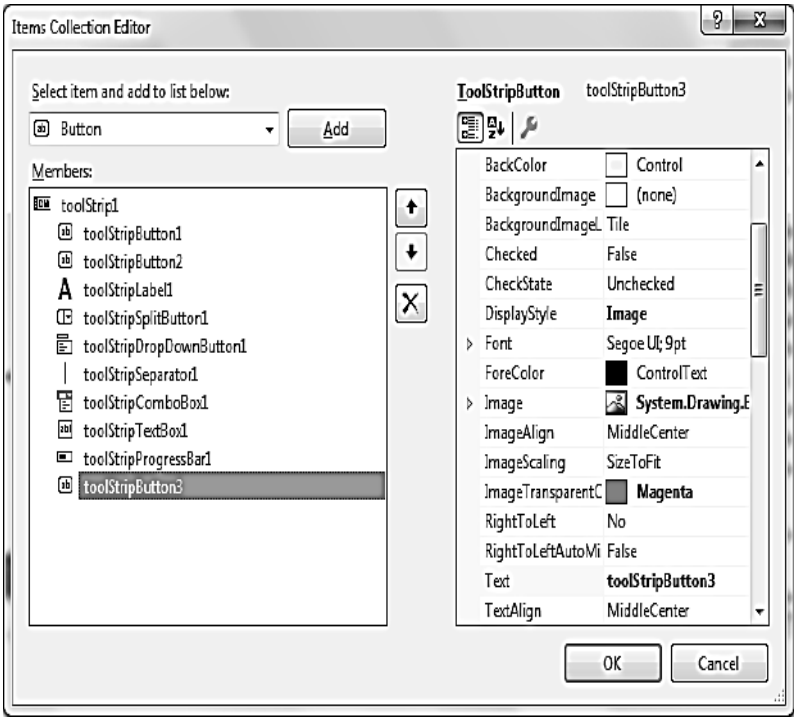


Şəkil 10.8. Alətlər paneli yaratmaq üçün açılan siyahı

Yaradılan `ToolStripButton` düyməsi və digər komponentlər (`textBox`, `comboBox` və s.) öz xassələrinə görə müvafiq olaraq adi `button` düyməsinə əvvəllər baxdığımız digər komponentlərə bənzəyir.

Alətlər panelinin strukturunu `toolStrip` komponentinin `Items` xassəsi vasitəsi ilə tərtib etmək olar. `Properties` (`Свойства`) pəncərəsində `Items` xassəsi qarşısındakı üç nöqtə təsvirli düyməni basdıqda `Items Collection Editor (Редактор коллекции строк)` pəncərəsi açılacaqdır (şəkil 10.9).

Bu pəncərənin `Selected item and to list below:` (`Выбрать элемент и добавить в список ниже:`) siyahısından komponenti seçib `Add` (`Добавить`) düyməsini basdıqda komponent alətlər panelində yerləşir. `Remove` (`Удалить`) düyməsini (X- işarəli düymə) basdıqda isə komponent alətlər panelindən silinir.



Şəkil 10.9. Alətlər panelinin tərtib edilməsi

ToolStrip Tasks açılan menyusunun digər bəndləri ilə tanış olaq:

- Embed in ToolStripContainer (Внедрить в ToolStripContainer) – alətlər panelini xüsusi konteynerdə yerləşdirir;
- RenderMode – alətlər panelinin təsvir üsulunu müəyyən edir:
  - System – sistem;
  - Professional – peşəkar;
  - Manager RenderMode – idarə olunan;
- Dock – alətlər panelinin formanın bu və ya digər tərəfinə bərkidilməsini müəyyən edir;
- GripStyle – alətlər paneli zolağının üslubunu müəyyən edir;
- Edit Items... (Правка элементов...) – bu bənd Items Collection Editor (Редактор коллекции строк) pəncərəsini açır.

Alətlər panelini sazlamaq üçün toolStrip komponentinin və menyu bəndlərinin (ToolStripItem tipli obyektlərin) xassələrinə qiymətlər vermək lazımdır. Onun xassələri aşağıdakılardır:

- ❖ Name – elementin adı;
- ❖ BackgroundImage – elementin fonunun təsvirini müəyyən edir;
- ❖ Dock – alətlər panelinin formanın bu və ya digər tərəfinə bərkidilməsini müəyyən edir;
- ❖ Enabled – elementin aktivliyini müəyyən edir;
- ❖ Items – alətlər panelinin elementləri;
- ❖ LayoutStyle – alətlər panelinin yerləşmə üslubu;
- ❖ Size – alətlər panelinin ölçüsü;
- ❖ Visible – alətlər panelinin görünməsini müəyyən edir.

ToolStripItem obyektlərinin xassələri aşağıdakılardır:

❖ `Name` – elementin adı; bu ad elementin adından və `ToolStripItem` obyektinin adından əmələ gəlir, məsələn, `toolStripButton1`;

❖ `Text` – alətlər panelinin elementinin adı;

❖ `Image` – alətlər panelinin elementində təsvir olunan şəkil;

❖ `Enabled` – alətlər panelinin elementinin aktivliyini müəyyən edir;

❖ `Checked` – alətlər panelinin elementinin seçilməsi əlaməti. Əgər element seçilərsə, onun qiyməti `true`, rəngi isə narıncı olur;

❖ `CheckState` – üç görünüşlü vəziyyəti müəyyən edir:

- `Checked` – element seçilmişdir;
- `Unchecked` – element seçilməmişdir;
- `Indeterminate` – qeyri-müəyyən vəziyyət;

❖ `DisplayStyle` – elementin xarici görünüşünü müəyyən edir:

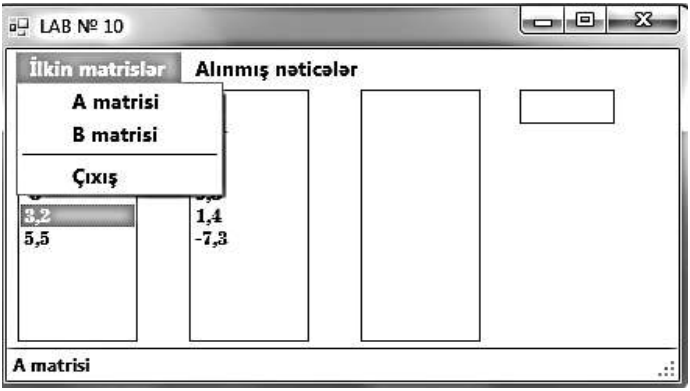
- `None` – elementdə heç nə yoxdur;
- `Text` – elementdə mətn təsvir edilir;
- `Image` – elementdə şəkil təsvir edilir;
- `ImageAndText` – elementdə şəkil və mətn təsvir edilir;

❖ `TextDirection` – mətnin yönünü (`Horizontal`, `Inherit`, `Vertical190`, `Vertical270`) müəyyən edir.

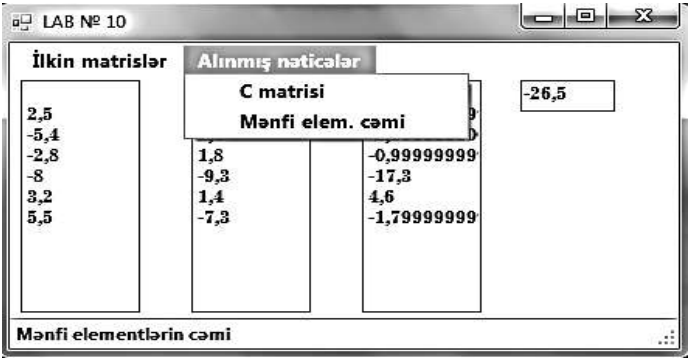
Misal. A və B matrislərinin uyğun elementlərinin cəmindən C matrisi düzəltməli. Alınmış yeni matrisin mənfi elementlərinin cəmini tapmalı.

Biz bu tip məsələləri artıq asanlıqla həll edə bilirik. İndi isə bu məsələni menyu vasitəsi ilə həll edək. Belə ki, əlavə iki menyu bəndindən ibarət olacaqdır. Onları şərti olaraq `İlkin matrislər` və `Alınmış nəticələr` adlandıraq (şəkil

10.10 və 10.11). Birinci menyu bəndi üç əmrdən ibarət olacaq: `A matrisi`, `B matrisi` və `Çıxış`. Bu əmrləri icra etdikdə müvafiq matrislər `listBox` siyahı komponentlərində təsvir ediləcəkdir. İkinci menyu bəndinin əmrləri ilə C matrisi hesablanıb siyahıda göstəriləcək və mənfi elementlərin cəmi `textBox` komponentində təsvir ediləcəkdir. Bundan başqa, müvafiq komponentlər üzərində mausun düyməsini basdıqda vəziyyətlər sətrində həmin komponentə aid məlumat təsvir ediləcəkdir.



Şəkil 10.10. 10 №-li laboratoriya işinin izahı



Şəkil 10.11. 10 №-li laboratoriya işinin izahı

Məsələni həll etmək üçün forma üzərinə `menuStrip` komponenti yerləşdirib şəkillərdə göstərilmiş menyu bəndlərini tərtib edirik. Sonra, formanın üzərinə üç `listBox`, `textBox` və vəziyyətlər sətrini yaratmaq üçün `statusStrip` komponenti yerləşdirək.

Bu konstruksiyalaşdırma işlərindən sonra menyu əmrləri üzərində mausun düyməsini iki dəfə basmaqla müvafiq kodlar yazırıq.

Vəziyyətlər sətrində matris və nəticələr haqqında məlumatın təsvir edilməsi üçün siyahı komponentləri üçün `SelectedIndexChanged`, `textBox` üçün isə `Click` emaledicilərini yaradırıq.

Bu məsələnin proqramı belədir:

```
using System;
using System.Windows.Forms;

namespace Lab_10
{
public partial class Form1 : Form
{
    public Form1()
  {
    InitializeComponent();
  }
double[,] a=new double[3,2] {
{2.5, 5.4 },{2.8, -8.0},{3.2, 5.5} ;
double[,] b= ew double[3, 2] {
{6.1, 2.6}, {1.8, 9.3}, {1.4, 7.3}};
double[,] c = new double[3, 2];
private void
  çıxışToolStripMenuItem_Click(
    object sender, EventArgs e)
{
```

```
  this.Close();
}

private void
  aMatrisiToolStripMenuItem_Click(
      object sender, EventArgs e)
{

  listBox1.Items.Clear();
  listBox1.Items.Add("");
  foreach (double element in a)
  listBox1.Items.Add(element);
}

private void
 bMatrisiToolStripMenuItem_Click(
      object sender, EventArgs e)
{
  listBox2.Items.Clear();
  listBox2.Items.Add("");
  foreach (double element in b)
  listBox2.Items.Add(element);
}

private void
  cMatrisiToolStripMenuItem_Click(
       object sender, EventArgs e)
{
  listBox3.Items.Clear();
  listBox3.Items.Add("");
  for (int i = 0; i < 3; i++)
   for (int j = 0; j < 2; j++)
    {
      c[i, j] = a[i, j] + b[i, j];
```

```
        listBox3.Items.Add(c[i,j]);
      }
}

private void
  mənfiElemCəmiToolStripMenuItem_Click(
           object sender, EventArgs e)
{
  double s=0.0;
  textBox1.Clear();
  for (int i = 0; i < 3; i++)
   for (int j = 0; j < 2; j++)
   {
    if (c[i, j] < 0)
    {
     s += c[i, j];
     textBox1.Text = s.ToString();
    }
   }
  if(s==0)
    MessageBox.Show("C matrisində mənfi
                     element yoxdur");
}

private void
  listBox1_SelectedIndexChanged(
      object sender, EventArgs e)
{
  toolStripStatusLabel1.Text =
                  "A matrisi";
}

private void
  listBox2_SelectedIndexChanged(
```

```
        object sender, EventArgs e)
  {
    toolStripStatusLabel1.Text =
                    "B matrisi";
  }

  private void
  listBox3_SelectedIndexChanged(
        object sender, EventArgs e)
  {
    toolStripStatusLabel1.Text =
                    "C matrisi";
  }

  private void textBox1_Click(object
                  sender, EventArgs e)
  {
    toolStripStatusLabel1.Text =
        "Mənfi elementlərin cəmi";
  } } }
```

Yoxlama sualları

1. Əsas menyu hansı komponentlə yaradılır?
2. Kontekst menyu nədir və hansı komponentlə yaradılır?
3. Alətlər paneli hansı komponentlə yaradılır?
4. Vəziyyətlər sətri hansı komponentlə yaradılır?
5. Menyulara şəkillər də əlavə etmək üçün hansı xassədən istifadə edilir?
6. Klavişlər kombinasiyasını yaratmaq üçün hansı xassədən istifadə edilir?
7. Menyu bəndinin seçilməsini hansı xassə bildirir?
8. `DisplayStyle` xassəsi nə üçündür?

9. `RenderMode` xassəsi nə üçündür?

10. Klavişlər kombinasiyası menyu bəndinin sol tərəfində necə yerləşdirilir?

11. Kontekst menyu digər komponentlərlə necə əlaqələndirilir?

12. Alətlər paneli komponenti hansı komponentlərdən ibarətdir?

13. `toolStrip` komponentinin `Items` xassəsi nə üçündür?

14. Kontekst menyu bəndlərinə necə müraciət edilir?

Laboratoriya işlərinə aid tapşırıqların variantları

Bütün tapşırıqlarda ilkin və aralıq massivləri `listBox` komponentinə yazmaq, son nəticələrin müxtəlif komponentlərdə göstərməklə menyu bəndlərindən istifadə edərək şəkil 10.11-də göstərilən əlavəyə analoji əlavələr yaradın.

1. A(4,5) matrisinin hər sətrindəki ən kiçik elementi tapın.

2. A(4,3) matrisinin ikinci sətrinin elementlərinin cəmini və 3-cü sütununun elementlərinin hasilini hesablayın.

3. A(5,5) matrisinin baş diaqonal elementlərinin cəmini (s) hesablayın. Əgər $s>10$ olarsa, onda ilkin matrisin elementlərini $b_{ij} = b_{ij} + 12$ düsturu ilə, $s \leq 10$ olduqda isə $b_{ij} = b_{ij}^2 - 10$ düsturu ilə hesablanmış elementlərlə əvəz edin.

4. A(5,5) matrisində 1 ədədlərinin sıra nömrələrini və ədədi ortasını hesablayın. Əgər matrisdə belə elementlər yoxdursa, onda ekrana ismarıc göndərin.

5. A(6,5) matrisinin hər sətrindəki ən böyük elementi tapın.

6. A(5,5) matrisinin baş diaqonal elementləri içərisində ən böyük elementi tapın və onun yerləşdiyi sətir elementlərini göstərin.

7. A(5,5) matrisinin hər sətrinin ən böyük elementini tapıb, onları baş diaqonala yerləşdirin.

8. A(5,5) matrisinin ən böyük elementini tapıb, onun yerləşdiyi sətir və sütunu sıfırlarla doldurun.

9. A(5,5) matrisinin hər sətrinin ən kiçik elementini tapıb, onu sətrin birinci elementinin yerinə yazın.

10. A(5,5) matrisinin sonuncu sütununun müsbət elementlərinin sayını (*k*) tapın. Əgər $k<3$ olarsa, onda matrisin bütün elementlərini kvadrata yüksəldin. Əgər $k\geq 3$ olarsa, onda baş diaqonal elementlərinin cəmini hesablayın.

11. A(5,5) matrisinin baş diaqonal elementindən yuxarıdakı elementlərin cəmini hesablayın.

12. A(5,5) matrisinin baş diaqonal elementindən aşağıdakı elementlərin cəmini hesablayın.

13. Matrisin ən az müsbət elementlərinin olduğu sütunun nömrəsini tapın.

14. Matrisin ən az müsbət elementlərinin olduğu sətrin nömrəsini tapın.

15. A(6,5) matrisinin hər sütunundakı ən böyük elementi tapın.

16. A(5,5) matrisinin baş diaqonal elementləri içərisində ən böyük elementi tapın və onun yerləşdiyi sütun elementlərini göstərin.

17. A(5,5) matrisinin ən azı bir sıfır elementi olan sətirlərin sayını tapın.

18. A(5,5) matrisinin mütləq qiymətcə ən kiçik elementinin kəsişməsində yerləşən sətir və sütunun nömrəsini tapın.

19. A(5,5) matrisinin elementlərinin cəmi ən böyük olan sətri tapın.

20. A(5,5) matrisinin elementlərinin cəmi ən böyük olan sütunu tapın.

21. A(5,5) matrisinin sətirlərinin ən böyük elementlərinin cəmini hesablayın.

22. A(5,5) matrisinin sütunlarının ən böyük elementlərinin cəmini hesablayın.

23. A(5,5) matrisinin hər bir sətrindəki maksimum elementdən B massivi düzəltməli.

24. A(5,5) matrisinin mənfi elementlərinin ən böyüyünü tapın.

25. A(5,5) matrisinin mənfi elementlərini kvadrata yüksəldin, müsbət elementlərini 2-yə vurun və sıfır elementlərini 5 ilə əvəz edin.

26. A(5,5) matrisinin $[2, 4]$ parçasına düşən elementlərinin ədədi ortasını hesablayın.

27. A(5,5) matrisinin hər bir sətrindəki minimum elementlərdən B massivi düzəldin və alınmış yeni massivin ən böyük elementini tapın.

28. A(5,5) matrisinin cüt nömrəli sütun elementlərinin hasilini tapın.

29. A(5,5) matrisinin tək nömrəli sətir elementlərinin hasilini tapın.

30. A(5,5) matrisinin tək nömrəli sətir və cüt nömrəli sütunların kəsişməsində yerləşən elementləri seçin.

ƏDƏBİYYAT

1. Zakir Məhərrəmov, Zəfər Cəfərov. Visual C#. LAP LAMBERT Academic Publishing, 2019. – 472 pagine.
2. Məhərrəmov Z.T., Abidov Ç.C., Həşimov R.H., Məmmədzadə N.F. C# dilində laboratoriya işləri (Konsol əlavələri). Bakı: ADPU-nəşriyyat, 2020. –144 s.
3. Дёминь А.Ю., Дорофеев В.А. Лабораторный практикум по инфопматике. Томск: Изд-во Томского политехнического университета, 2014. – 132 с.
4. А.А.Калинин. Разработка алгоритмов с использованием принципов ООП на языке C#. Учебно-методическое пособие.–Моск. гос. ун-т печати имени Ивана Федорова. – М.:МГУП имени Ивана Федорова, 2014. – 106 с.
5. Соловей Л.В., Мирошниченко Н.Н., Пономарёв Н.Г. Программирование на языке C#. Х. : НТУ «ХПИ», 2016. – 356 с.