МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет економіки та підприємництва
Кафедра комп'ютерних наук та економічної кібернетики

Волонтир Л.О., Денисюк В.О., Юрчук Н.П.

ПРОГРАМА
ТА МЕТОДИЧНІ ВКАЗІВКИ

до проходження навчальної практики
«**ВСТУП ДО СПЕЦІАЛЬНОСТІ: ОСНОВ ПРОГРАМУВАННЯ**»
здобувачами вищої освіти

Рівень вищої освіти Перший (бакалаврський)
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки »
Освітньо-професійна програма «Комп'ютерні науки»

ВНАУ 2021 рік

Волонтир Л.О., Денисюк В.О., Юрчук Н.П.
Програма та методичні вказівки до проходження навчальної практики до «Вступ до спеціальності: основ програмування» здобувачами вищої освіти першого (бакалаврського) рівня галузі 12 «Інформаційні технології» спеціальності 122 «Комп'ютерні науки» за освітньо-професійною програмою «Комп'ютерні науки». Вінниця: ВНАУ, 2021. 61 с.

Розробники: Волонтир Л. О., к.т.н., доцент кафедри комп'ютерних наук та економічної кібернетики
Денисюк В.О., к.т.н., доцент кафедри комп'ютерних наук та економічної кібернетики
Юрчук Н.П., к.е.н., доцент кафедри комп'ютерних наук та економічної кібернетики

Затверджено до видання науково-методичною комісією ВНАУ (протокол № 3 від 12 жовтня 2021 р.) за поданням навчально-методичної комісії факультету економіки та підприємництва (протокол №4 від 6 жовтня 2021 р.)

Програма та методичні рекомендації є навчально-методичним документом, який містить рекомендації для закріплення набутих теоретичних знань та практичних навичок студентами, що навчаються за освітньо-професійною програмою «Комп'ютерні науки» з метою формування у здобувачів компетентностей щодо основ програмування.

Програма та методичні рекомендації є навчально-методичним документом, який містить рекомендації для закріплення набутих теоретичних знань та практичних навичок студентами, що навчаються за освітньо-професійною програмою «Комп'ютерні науки» регламентує форми, організацію, здійснення та порядок захисту навчальної практики здобувачами вищої освіти першого (бакалаврського) освітнього рівня у Вінницькому національному аграрному університеті.

ЗМІСТ

ВСТУП

Проходження навчальної практики для здобувачів вищої освіти, що навчаються за ОП «Комп'ютерні науки» є обов'язковим компонентом освітньо-професійної програми для здобуття першого (бакалаврського) освітнього ступеня у Вінницькому національному аграрному університеті.

Навчальна практика призначена для студентів першого курсу та проводиться у другому семестрі після завершення вивчення основних професійних дисциплін.

Об'єктом практики є навчальний процес підготовки фахівців за освітньою програмою «Комп'ютерні науки».

Предметом практики виступають дисципліни навчального плану за перших три роки навчання.

Організатором проходження практики є кафедра комп'ютерних наук та економічної кібернетики, яка є випусковою для студентів, що навчаються за освітньо-професійною програмою Комп'ютерні науки у Вінницькому національному аграрному університеті.

Базою для проведення навчальної практики є Університет та його структурні підрозділи. Відповідно до навчального плану та графіку навчального процесу практика планується у другому семестрі навчання терміном 2 тижні для студентів, які успішно закінчили навчання, виконали індивідуальний план студента за звітний період.

1. МЕТА І ЗАВДАННЯ НАВЧАЛЬНОЇ ПРАКТИКИ

Метою навчальної практики здобувачів першого (бакалаврського) освітнього рівня є поглиблення практичних вмінь та набуття навичок і досвіду використання отриманих знань з фахових навчальних дисциплін, підвищення ефективності подальшого опанування студентами предметної області за напрямом підготовки, організації навчального процесу, а також отримання практичних навичок самостійного опрацювання навчального матеріалу.

У результаті проходження навчальної практики студенти повинні оволодіти основами навчально-методичної роботи: здатністю застосувати математичні основи, алгоритмічні принципи в моделюванні, проектуванні, розробці та супроводі інформаційних систем і технологій; здійснювати розробку, впровадження і супровід інтелектуальних систем аналізу та обробки даних в організаційних, технічних, природничих та соціально-економічних системах.

Тривалість практики – два тижні. По закінченню навчальної практики з дисципліни зі студентами проводиться співбесіда з питань виконання програми практики.

Структура навчальної практики повністю відповідає змісту навчальної програми і забезпечує цілісну підготовку здобувача першого (бакалаврського) освітнього ступеня до подальшої професійної діяльності.

Програма та методичні вказівки з проходження навчальної практики є основним навчально-методичним документом для студентів та керівників практики.

Навчальна практика студентів факультету економіки та підприємництва є важливою складовою частиною навчального процесу університету. Під час її проходження студенти не тільки закріплюють і поглиблюють теоретичні знання, одержані в процесі вивчення дисципліни, а й набувають умінь та навичок практичної діяльності з інших профільних

дисциплін.

Практична підготовка є цілісним процесом, який передбачає неперервність та послідовність отримання необхідного обсягу практичних знань та вмінь відповідно до першого (бакалаврського) освітнього ступеня.

Цілями та завданнями навчальної практики є: поглиблення і закріплення одержаних знань і навиків з дисциплін фундаментального напряму та за фаховим спрямуванням; формування професійних вмінь і навичок; розвиток професійних якостей майбутнього фахівця; систематизація, закріплення та поглиблення теоретичних знань; накопичення досвіду практичної діяльності із спеціальності.

Навчальна практика є проміжним етапом підготовки студента в галузі економічної кібернетики. У рамках навчальної практики синтезуються раніше одержані знання і навички, перевіряється на здатність студента самостійно будувати економіко-математичну модель досліджуваного об’єкта з використанням методів системного аналізу і моделювання, програмно реалізовувати окремі задачі. Знання програмного матеріалу дисциплін та набуття практичних навичок стануть у нагоді при подальшому навчанні, проведенні досліджень, вивченні інших дисциплін програми.

Результатом проходження навчальної практики є формування у майбутніх бакалаврів загальнокультурних, особистих і професійних компетенцій, спрямованих на закріплення та поглиблення теоретичної підготовки, оволодіння вміннями та навичками самостійно ставити завдання, аналізувати отримані результати і робити висновки, набуття навичок в дослідній роботі.

2. КОМПЕТЕНТНОСТІ ТА ПРОГРАМНІ РЕЗУЛЬТАТИ

Навчальна практика належить до циклу професійної підготовки освітньо-професійної програми «Комп'ютерні науки».

В результаті проходження навчальної практики здобувач освіти повинен сформувати такі програмні компетентності:

інтегральна компетентність – Здатність розв'язувати складні спеціалізовані задачі та практичні проблеми у галузі комп'ютерних наук або у процесі навчання, що передбачає застосування теорій та методів інформаційних технологій і характеризується комплексністю та невизначеністю умов;

загальні компетентності;

ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.

ЗК3. Знання та розуміння предметної області та розуміння професійної діяльності.

ЗК6. Здатність вчитися й оволодівати сучасними знаннями.

ЗК7. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.

ЗК8. Здатність генерувати нові ідеї (креативність).

ЗК9. Здатність працювати в команді.

ЗК12. Здатність оцінювати та забезпечувати якість виконуваних робіт.

спеціальні (фахові компетентності) :

СК1. Здатність до математичного формулювання та досліджування неперервних та дискретних математичних моделей, обґрунтовування вибору методів і підходів для розв'язування теоретичних і прикладних задач у галузі комп'ютерних наук, аналізу та інтерпретування

СК2. Здатність до виявлення статистичних закономірностей недетермінованих явищ, застосування методів обчислювального інтелекту, зокрема статистичної, нейромережевої та нечіткої обробки даних, методів машинного навчання та генетичного програмування тощо.

СК3. Здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв'язності та нерозв'язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем.

СК4. Здатність використовувати сучасні методи математичного моделювання об'єктів, процесів і явищ, розробляти моделі й алгоритми чисельного розв'язування, задач математичного моделювання, враховувати похибки наближеного чисельного розв'язування професійних задач.

СК5. Здатність здійснювати формалізований опис задач дослідження операцій в організаційно-технічних і соціально-економічних системах різного призначення, визначати їх оптимальні розв'язки, будувати моделі оптимального управління з урахуванням змін економічної ситуації, оптимізувати процеси управління в системах різного призначення та рівня ієрархії.

СК6. Здатність до системного мислення, застосування методології системного аналізу для дослідження складних проблем різної природи, методів формалізації та розв'язування системних задач, що мають суперечливі цілі, невизначеності та ризики.

СК7. Здатність застосовувати теоретичні та практичні основи методології та технології моделювання для дослідження характеристик і поведінки складних об'єктів і систем, проводити обчислювальні експерименти з обробкою й аналізом результатів.

СК8. Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.

СК9. Здатність реалізувати багаторівневу обчислювальну модель на

основі архітектури клієнт-сервер, включаючи бази даних, знань і сховища даних, виконувати розподілену обробку великих наборів даних на кластерах стандартних серверів для забезпечення обчислювальних потреб користувачів, у тому числі на хмарних сервісах.

ПРОГРАМНІ РЕЗУЛЬТАТИ НАВЧАННЯ

РН1. Застосовувати знання основних форм і законів абстрактно-логічного мислення, основ методології наукового пізнання, форм і методів вилучення, аналізу, обробки та синтезу інформації в предметній області комп'ютерних наук.

РН2. Використовувати сучасний математичний апарат неперервного та дискретного аналізу, лінійної алгебри, аналітичної геометрії, в професійній діяльності для розв'язання задач теоретичного та прикладного характеру в процесі проектування та реалізації об'єктів інформатизації.

РН3. Здатність продемонструвати поглиблені знання методів, способів та технологій збору інформації з різних джерел, контент-аналізу документів, аналізу та обробки даних; Використовувати знання закономірностей випадкових явищ, їх властивостей та операцій над ними, моделей випадкових процесів та сучасних програмних середовищ для розв'язування задач статистичної обробки даних і побудови прогнозних моделей.

РН4. Використовувати методи обчислювального інтелекту, машинного навчання, нейромережевої та нечіткої обробки даних, генетичного та еволюційного програмування для розв'язання задач розпізнавання, прогнозування, класифікації, ідентифікації об'єктів керування тощо.

РН5. Проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій.

РН6. Використовувати методи чисельного диференціювання та інтегрування функцій, розв'язання звичайних диференціальних та

інтегральних рівнянь, особливостей чисельних методів та можливостей їх адаптації до інженерних задач, мати навички програмної реалізації чисельних методів.

РН7. Розуміти принципи моделювання організаційно-технічних систем і операцій; використовувати методи дослідження операцій, розв’язання одно- та багатокритеріальних оптимізаційних задач лінійного, цілочисельного, нелінійного, стохастичного програмування.

РН8. Використовувати методологію системного аналізу об’єктів, процесів і систем для задач аналізу, прогнозування, управління та проектування динамічних процесів в макроекономічних, технічних, технологічних і фінансових об’єктах.

РН9. Розробляти програмні моделі предметних середовищ, вибирати парадигму програмування з позицій зручності та якості застосування для реалізації методів та алгоритмів розв’язання задач в галузі комп’ютерних наук.

3. ОРГАНІЗАЦІЯ І КЕРІВНИЦТВО НАВЧАЛЬНОЮ ПРАКТИКОЮ

Керівництво практикою та навчально-методичне забезпечення здійснюється викладачами кафедри комп'ютерних наук та економічної кібернетики.

Відповідальність за організацію, проведення і контроль практики покладається на завідувача кафедри комп'ютерних наук та економічної кібернетики або уповноважену ним особу.

Навчально-методичне керівництво практикою покладається на керівників практики, які здійснюють свою діяльність згідно програми навчальної практики за освітньою програмою 122 «Комп'ютерні науки». До керівництва навчальною практикою залучаються найбільш досвідчені та професійно підготовлені викладачі зі складу кафедри комп'ютерних наук та економічної кібернетики.

Навчальна практика проходить в аудиторіях, обладнаних комп'ютерною технікою та доступом до мережі Internet під керівництвом викладача кафедри комп'ютерних наук та економічної кібернетики.

Керівник практики:

- проводить первинний інструктаж з техніки безпеки роботи в комп'ютерних класах;

- контролює підготовленість бази практики (справність техніки, наявність необхідного програмного забезпечення);

- проводить ознайомлення з програмою практики та календарним планом;

- повідомляє студентів про систему звітування щодо результатів виконання навчальної практики;

- здійснює відповідні організаційні заходи під час проведення практики з метою її успішного проходження;

- розробляє заходи щодо удосконалення організації проведення навчальної практики;

- у складі комісії приймає заліки з практики. Зобов'язання студента

практиканта:

- своєчасно прибути на практику;
- ознайомитись та дотримуватись правил охорони праці та техніки безпеки;
- отримати індивідуальне завдання від керівника практики;
- отримати від керівника практики консультації щодо одержання необхідних для виконання навчальної практики документів;
- повністю виконати всі завдання, що передбачені програмою та методичними вказівками для проведення навчальної практики та продемонструвати результати керівнику;
- оформити звіт про виконання навчальної практики відповідно до вимог програми та методичних вказівок про проведення навчальної практики.

4. ОФОРМЛЕННЯ ЗВІТУ З НАВЧАЛЬНОЇ ПРАКТИКИ

Звіт з навчальної практики виконується студентом- практикантом відповідно програми практики на папері стандартного розміру А4 (210х297 мм) у рукописному або комп'ютерному варіанті, чітко, розбірливо (в текстовому, іншому графічному вигляді).

Рекомендований загальний обсяг звіту без додатків – в межах 20 сторінок. При потребі студент за погодженням керівника практики може коригувати кількість сторінок окремих структурних частин звіту.

На момент здачі звіту на перевірку до нього обов'язково додаються додатки, що дають змогу перевірити достовірність інформації про об'єкти дослідження. Форми необхідних документів студенти можуть знайти у відкритому Інтернет- доступі. Таблиці та графічний матеріал, які студент розробляє самостійно відповідно до індивідуального завдання доцільно також виносити в додатки.

Додатки слід позначати послідовно великими літерами української абетки, за винятком літер Ґ, Є, З, І, Ї, Й, О, Ч, Ь.

Приклад оформлення титульної сторінки звіту з навчальної практики наведено у додатку А.

5. МЕТОДИЧНІ ВКАЗІВКИ ЩОДО ВИКОНАННЯ ІНДИВІДУАЛЬНИХ ЗАВДАНЬ НАВЧАЛЬНОЇ ПРАКТИКИ

НАВЧАЛЬНИЙ ПЛАН ПРАКТИКИ

Відповідно до навчальних планів першого (бакалаврського) освітнього ступеня освітньо-професійної програми «Комп'ютерні науки» навчальна практика виконується студентами на третьому курсі протягом двох тижнів.

Навчальна практика з дисципліни проводиться за тематичним планом (табл. 1) і оцінюється окремо від дисципліни за рейтинговою оцінкою 100 балів.

Навчальна практика виконується студентами після того, як вони вже прослухали наступні курси з профільних дисциплін: «Інтелектуальний аналіз даних», «Математичні методи дослідження операцій», «Web-технології та web-дизайн», «Системний аналіз», «Теорія прийняття рішень», «Програмування Java», «Організація баз даних та знань», «Комп'ютерні мережі», «Технологія створення програмних продуктів» та ін.

Таблиця 1

Календарно-тематичний план та шкала оцінювання результатів виконання навчальної практики

Дні практики	Перелік робіт	Кількість годин/ кредитів
1	Створення Windows додатків в середовищіMS Visual C++	6/0,2
2	Розробка додатку для Windows в середовищіMS Visual C++ за допомогою конструктора Windows Forms	6/0,2
3	Створення Windows додатків в середовищі MS Visual C++ із використанням додатковихелементів вводу/виводу даних	6/0,2
4	Виведення повідомлень із використаннямMessageBox	6/0,2
5	Елементи управління роботою Windows додатків	6/0,2
6	Створення багатовіконних додатків із можливістю вводу-виводу даних у файл	6/0,2
7	Самостійна робота студентів	6/0,2
8	Створення Windows додатків в середовищіMS Visual C++	6/0,2
9	Розробка додатку для Windows в середовищіMS Visual C++ за допомогою конструктора Windows Forms	6/0,2
10	Оформлення та захист звіту	6/0,2
Всього		60/2

Таблиця 2

Шкала знань для оцінювання результатів проведеної навчальної практики

№ з/п	Види проведеної роботи	Оцінювання (кількість балів)
1.	Проходження практики (виконання програми практики)	0-50
2.	Оформлення результатів практики (індивідуальних завдань, звітів)	0-20
3.	Засвоєння програми практики (диференційований залік, захист звітів)	0-30
Разом		100

Оцінювання здійснюється за національною, чотирьох бальною шкалою та шкалою ECTS (табл. 3).

Таблиця 3

Шкала підсумкової оцінки навчальної практики

Сума балів за всі види навчальної діяльності	Оцінка ECTS	Оцінка за національною шкалою	
		для диференційованого заліку	для заліку
90-100	**A**	відмінно	зараховано
82-89	**B**	добре	
75-81	**C**		
66-74	**D**	задовільно	
60-65	**E**		
35-59	**FX**	незадовільно з можливістю повторного складання	не зараховано з можливістю повторного складання
0-34	**F**	незадовільно з обов’язковим повторним вивченням дисципліни	не зараховано з обов’язковим повторним вивченням дисципліни

Критеріями оцінювання результатів проходження та захисту практики є:

Оцінка «відмінно» (А):

– студент показав глибокі вичерпні знання всього матеріалу,

представленого у звіті, розуміння сутності і взаємозв'язку явищ, процесів і подій, уміння використовувати системні наукові відомості, які отримані під час вивчення циклів дисциплін гуманітарної, природничо-наукової, професійної (професійно-орієнтованої) та практичної підготовки при обґрунтуванні змісту відповідей на головні і додаткові запитання; глибоке розуміння нормативних документів, уміння вирішувати практичні завдання, одержувати оптимальні результати у відповідності до теоретичних положень;

– пояснювальна записка написана грамотною мовою, прийнятою у науковій літературі відповідного напрямку підготовки (спеціальності), оформлена згідно до діючих вимог;

– усі графічні елементи виконані якісно у відповідності до вимог ДСТУ;

– розрахунки та оформлення виконано за допомогою ПЕОМ;

– студент виконав звіт на основі самостійно отриманих даних;

– студент дав правильні змістовні відповіді на всі питанні, задані на захисті.

Оцінка «дуже добре» (B):

– студент показав тверді і достатньо повні знання всього матеріалу, який представлений у звіті, правильне розуміння сутності і взаємозв'язку явищ (процесів), що розглядаються; уміння самостійно з використанням сучасних методик аналізувати і застосовувати основні положення теорії для вирішення практичних завдань;

– пояснювальна записка містить незначні помилки граматичного і синтаксичного характеру;

– дав послідовні, правильні конкретні відповіді на поставлені запитання при вільному усуненні зауважень про недостатньо повне і незначне висвітлення окремих положень при постановці додаткових запитань, при цьому допустив не більше двох неповних відповідей.

Оцінка «добре» (C):

– студент показав загалом добрі знання всього матеріалу, який

представлений у звіті, правильне розуміння сутності і взаємозв’язку явищ (процесів), що розглядаються; уміння самостійно з використанням сучасних методик аналізувати і застосовувати основні положення теорії до вирішення практичних завдань, але припустив низку помітних помилок;

– пояснювальна записка містить помилки граматичного і синтаксичного характеру;

– студент дав правильні змістовні відповіді на всі питанні, задані на захисті, при цьому допустив не більше чотирьох неповних відповідей.

Оцінка «задовільно» (D):

– студент показав тверді знання і розуміння матеріалу, який представлений у звіті; правильні і конкретні, без грубих помилок відповіді на поставлені запитання для усунення неточностей та несуттєвих помилок у висвітленні окремих положень при навідних запитаннях; наявні помилки у графічній частині роботи; низькі вміння застосовувати теоретичні знання до вирішення основних практичних завдань; аналітичний апарат при доведеннях використовувався обмежено;

– пояснювальна записка містить неточності та деякі помилки;

– студент дав правильні змістовні відповіді на всі питанні, задані на захисті, при цьому допустив не більше п'яти неповних відповідей.

Оцінка «достатньо» (E):

– студент показав знання і розуміння основного матеріалу, який представлений у звіті;

– допускав помилки у відповідях на поставлені запитання;

– уміння застосовувати теоретичні знання до вирішення основних практичних завдань недостатні;

– аналітичний апарат у звіті використовувався обмежено;

– пояснювальна записка містить неточності та деякі помилки; наявні помилки у графічній частині;

– дані для виконання роботи отримані не самостійно;

– студент відповів не менш як на 10 запитань за темою роботи та по

проблемній галузі, при цьому допускав не більше п'яти невірних відповідей.

Оцінка «незадовільно» (FX):

– студент допускав грубі помилки у відповідях, показав недостатнє розуміння сутності питань, що висвітлюються, невміння застосовувати знання при вирішенні практичних завдань;

– пояснювальна записка викладена з великою кількістю помилок; – графічна частина виконана з порушенням вимог ДСТУ.

Оцінка «неприйнятно» (F):

– студент не пройшов практику у встановлений термін та/або не подав на кафедру звіт з практики у встановлений термін. Такий звіт повинен бути допрацьований і повторно винесений на захист.

6. ПРАКТИЧНІ ЗАВДАННЯ

Практичне завдання №1

Тема: Створення Windows додатків в середовищі MS Visual C++
Мета: формування навичок використання різних елементів управління у додатках **Windows Forms Application**

Для виконання роботи необхідно знати:

- порядок створення додатку Windows Forms Application;
- призначення та можливості конструктора форм;
- використання компонентів Windows Forms;
- властивості компонентів та їх налаштування;
- основні елементи управління Windows Forms;
- процедури обробки подій;
- особливості вводу та обробки коду обробки подій.

Теоретичні відомості

Додатки, що розробляються у візуальному середовищі, фактично є графічними додатками для Windows, і це принципово відрізняє їх від консольних додатків. Характерною особливістю графічних додатків на мові C++ є використання можливостей об'єктно-орієнтованого програмування, і всі елементи програми є об'єктами. При створенні нових об'єктів на візуальній формі вони будуються із уже існуючих об'єктів. Завдяки цьому нові об'єкти наслідують властивості та можливості вихідних об'єктів, і які необхідно тільки доповнити необхідними новими функціями.

Додатки із графічним інтерфейсом, що використовують форми, створюються та управляються в середовищі CLR (Common Language Runtime – загальномовне виконуюче середовище). Середовище виконує код та пропонує служби, які полегшують сам процес розробки.

Додаток Windows Forms Application дозволяє помістити у виконуваний код стандартні графічні елементи управління Windows. В процесі створення додатку у візуальному середовищі програмування MS Visual C++ користувачу надається можливість створення графічного інтерфейсу за допомогою Windows Forms. При цьому в ході створення графічного інтерфейсу в додатку на мові C/C++ необхідно забезпечити взаємодію користувача із елементами управління інтерфейсу. Для цього необхідно створити додаток, який являє собою повнофункціональну програму Forms.

Для цього необхідно знати:

- можливості використання конструктора форм **Form Design** для побудовиграфічного інтерфейсу користувача в додатку;
- можливості використання в формі елементів управління;
- можливості обробки подій, пов'язаних із елементами управління.

Розробка додатку для Windows в середовищі MS Visual C++за допомогою конструктора Windows Forms

Створення нового додатку починається із створення порожньої графічної форми. Для цього після звпуску середовища MS Visual C++ необхідно виконати команду File/New/Project, вибрати мову програмування (Visual C++), тип проекту – CLR , а тип додатку - Windows Forms Application. Необхідно також ввести ім'я додатку та задати повний шлях до файлу проекту.

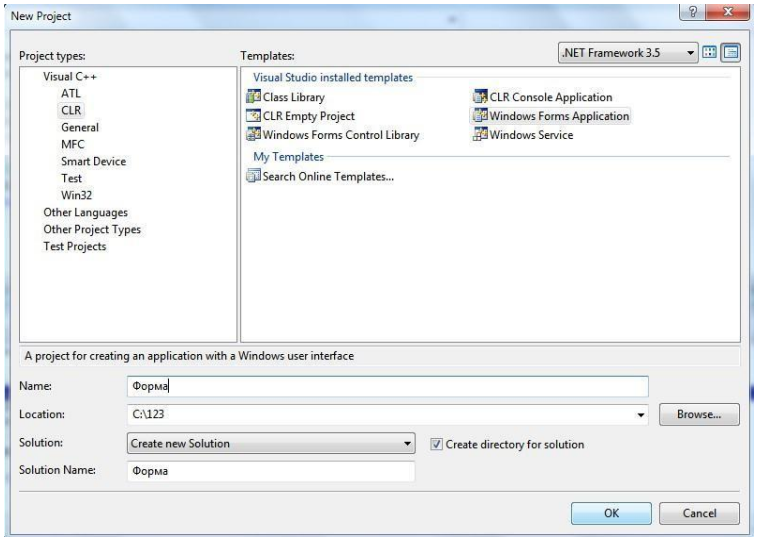


Рис.1. Вигляд вікна середовища ***Microsoft Visual Studio*** *при створеннінового проекту типу Windows Forms Application.*

Після створення нового проекту відкриється конструктор Windows Forms,у якому буде відображено порожню форму нового проекту.

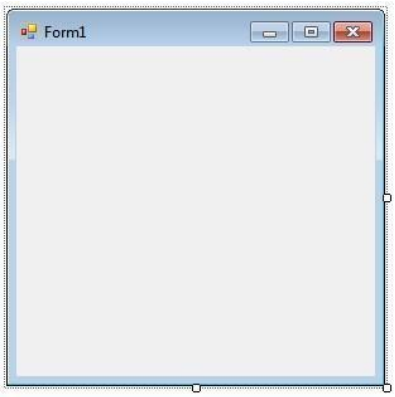


Рис.2. Вікно нової форми Form1.

Структура створеного проекту приведене на рисунку нижче

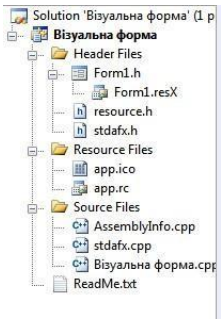


Рис.3. Структура новоствореного проекту.

Як і будь-який інший об'єкт у візуальному середовищі, форма характеризується набором властивостей, значеннями яких можна управляти за допомогою вікна властивостей об'єктів – **Properties.** Для виводу вікна властивостей форми необхідно відкрити контекстне меню форми, для чого помістити курсор мишки на форму, клікнути правою клавішею миші та виконати команду **Properties**. Властивості можна переглядати як в алфавітномупорядку так і за категоріями.

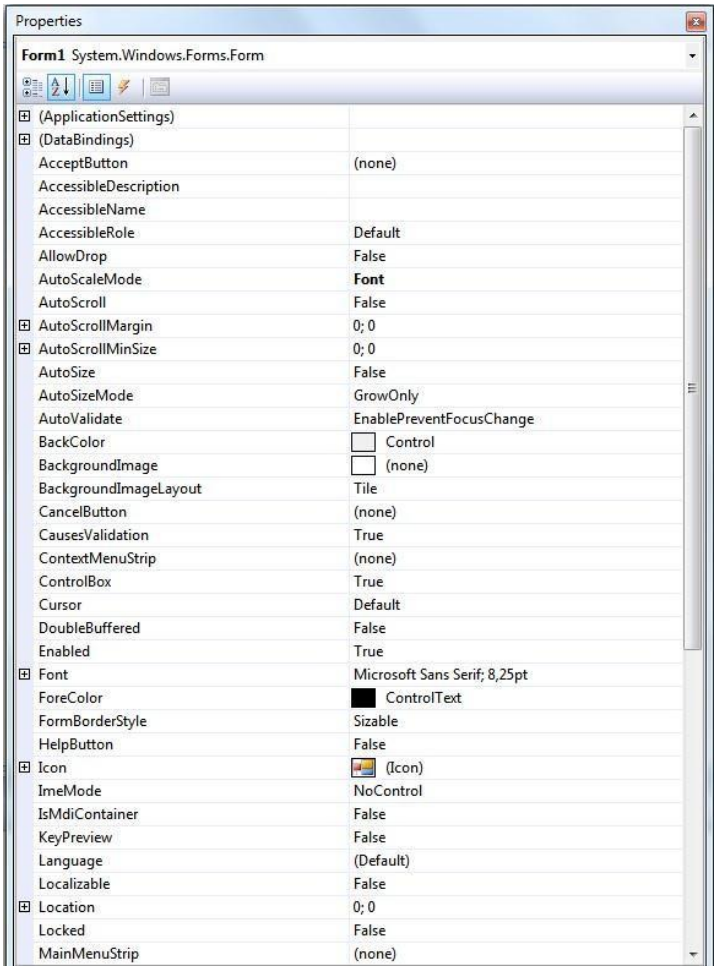


Рис.4. Частина вікна властивостей нової форми Form1.

Вікно **Properties** дозволяє задати в першу чергу загальний дизайн форми та її елементів управління. В таблиці приведено опис деяких найбільш часто використовуваних властивостей. При виборі значення властивостей,

відмінних від прийнятого по замовчування, воно виділяється жирним шрифтом, щополегшує подальше визначення введених змін.

Таблиця 1.

Деякі властивості форми

Властивість	Опис	Значення по замовчуванню
AcceptButton	Встановлюється значення кнопки, яка буде спрацьовувати при натисканні клавіші ***Enter***. Для того, щоби ця властивість була активною, необхідна наявність принаймі однієї кнопки, розміщеної на формі	None
BackColor	Колір форми.	Control
BackgroundImage	Використання зображення в якості фону	None
CancelButton	Задає кнопку, яка буде спрацьовувати при натисненні ***Esc***. Для того, щоби цявластивість була активною необхідна наявність хочаб онієї кнопки на формі	None
ControlBox	Задає наявність чи відсутність трьох стандартних кнопок в верхньому правому куті форми: "Згорнути", "Розгорнути" та "Закрити"	
Cursor	Визначає вигляд курсору при його розміщенні на формі	Default
DrawGrid	Задає наявність чи відсутність сітки, яка допомогає формувати елементи управління. Сітка відображається тільки на етапі створення додатку	True
Font	Формат шрифту, що використовується для відображення тексту на формі в елементах управління	Microsoft Sans Serif; 8,25pt
FormBorderStyle	Визначає вигляд границь форми. Можливі значення: - ***None*** — форма без границь і рядка заголовку; - ***FixedSingle*** — тонкі границі без можливості зміни розміру користувачем; - ***Fixed3D*** — границі без можливості зміни розміру із тремірним ефектом; - ***FixedDialog*** — границі без можливості зміни, без іконки додатку; - ***Sizable*** — звичайні границі: користувач може міняти розмір границь; -***FixedToolWindow*** — фіксовані границі, є тільки кнопка закриття форми. Такий вигляд мають панелі інструментів в додатках; - ***SizableToolWindow*** — границі із можливістю зміни розмірів, є тільки кнопка закриття	Sizable

Icon	Зображення іконки, що розміщується в заголовку форми. Підтримуються формати ***.ico***	
MaximizeBox	Визначає активність стандартної кнопки "Розгорнути" у верхньому правому куті форми	True
MaximumSize	Максимальний розмір ширини і висоти формив пікселях. Форма приймає заданий розмір при натисненні на стандартну кнопку "Розгорнути"	0;0 (Во весь экран)
MinimizeBox	Визначає активність стандартної кнопки "Згорнути" в верхньому правому куті форми	True
MinimumSize	Мінімальний розмір ширини и висоти формив пікселях. Форма приймає заданий розмір при зміні її границь користувачем	0;0
Size	Ширина і висота форми	300; 300
StartPosition	Визначає розміщення форми при запуску додатку. Можливими є наступні значення: - ***Manual*** — форма відкривається у верхньому лівому куті екрану; - ***CenterScreen*** — в центрі екрану; - ***WindowsDefaultLocation*** - розміщення форми по замовчуванню. Якщо користувач поміняв розмір форми, то при наступних її запусках вона буде мати той же самий виглядта розміщення; - ***WindowsDefaultBounds*** — границі форми мають фіксований розмір; - ***CenterParent*** — в центрі батьківської форми	WindowsDefaultLocation
Text	Заголовок форми. У відмінності від властивості ***Name***, це власне назва форми, яка не використовується в коді	Form1, Form 2 и т.д.
WindowState	Визначає положення форми при її запуску. Можливими є наступні значення: - ***Normal*** — форма запускається із розмірами, заданими у властивості ***Size***; - ***Minimized*** — форма запускається із мінімальними розмірами, заданими у властивості ***MinimumSize***; - ***Maximized*** — форма розгортається на весь екран	Normal

Використання компонентів Windows Forms

Створення інтерфейсу додатку у середовищі MS Visual C++ зводиться до розміщення на формі необхідних компонент, ініціалізації подій та написання програм обробки цих подій у відповідності до розв'язуваної задачі. Компоненти Windows Forms розбиті на функціональні групи та знаходяться в палітрі компонентів **Toolbox.**

Рис.5. Функціональні групи палітри Toolbox.

Для відображення палітри **Toolbox** необхідно вибрати пункт головного меню **View** і списку, що появиться вибрати пункт ToolBox. Перелік всіх доступних компонент палітри можна побачити при активізації функціональної групи **All Windows Forms**.

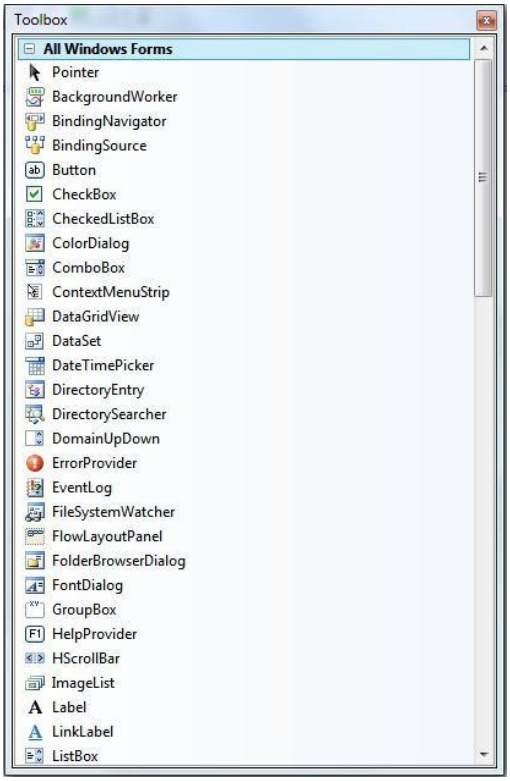


Рис.6. Частина повного переліку компоненти палітри ***Toolbox****.*

Для розміщення будь-якої компоненти на форму необхідно його знайти у палітрі компонент, клікнути на ньому мишкою, перевести курсор миші в

необхідне місце на формі та знову клікнути мишкою. Будь-який об’єкт на робочому столі форми характеризується певним набором властивостей, таких як імя, видимість, координати розміщення на формі, розмір і т.д. Властивості компонентів можна переглянути чи змінити за допомогою вікна властивостей Properties, в якому відображаються всі властивості активного об’єкта. Вікно відкривається при виконанні контекстного меню Properties.

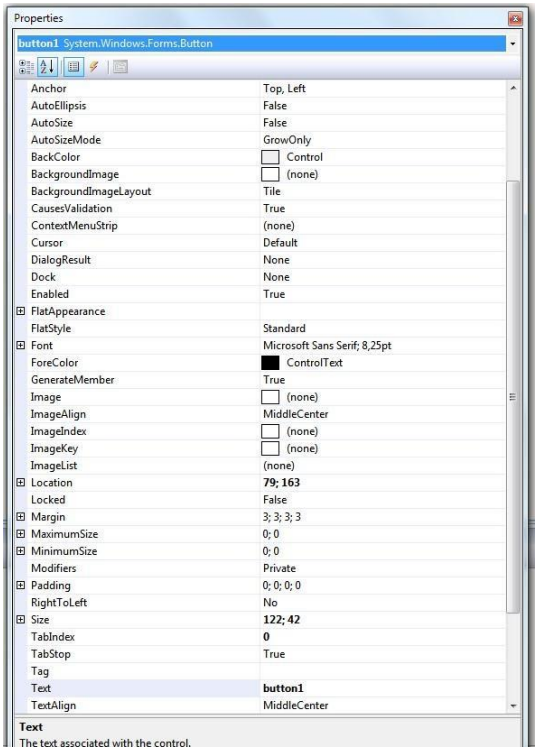


Рис.7. Частина меню Properties компоненти палітри ***Toolbox*** *кнопка* ***(Button).***

Windows Forms являє собою цілий набір засобів створення Windows - додатків, що виконуються в середовищі CLR. Форма являє собою головний контейнер, в який поміщаються віртуальні компоненти (кнопки, мітки, таблиці, поля прокрутки і т. д.), за допомогою яких і реалізується алгоритм розв’язку конкретної задачі.

До пакету Visual C++ входить стандартний набір понад 60-ти елементів управління, які можна використовувати при створенні форми для взаємодії із користувачем. Кожен із елементів управління має певне призначення та виконує певні функції.

Найбільш часто використовуваними елементами управління і компонентами Windows Forms оглянуті нижче

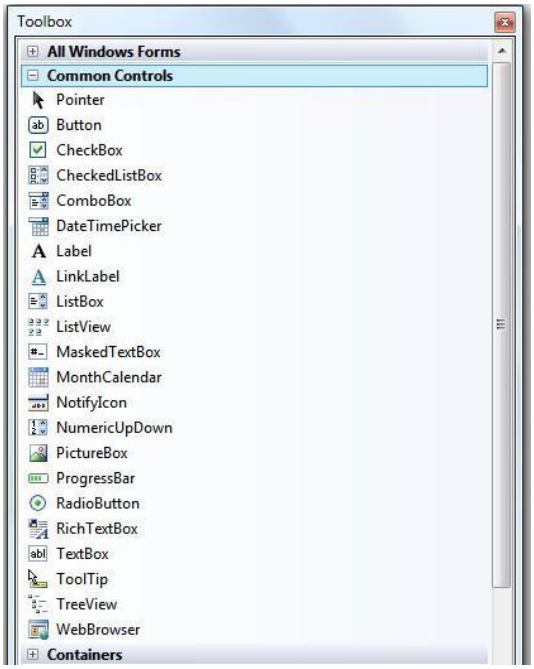


Рис.8. Елементи групи ***Common Controls****.*

Елементи групи **Common Controls** дозволяють реалізовувати ввід/вивід, відображення та редагування елементів управління. Деякі з них описанонижче:

- ***Button*** – дозволяє створювати елемент типу «кнопка» для запуску, зупинки чи переривання деякого обчислювального процесу;
- ***Label*** – відображає текст, який недоступний до редагування користувачем в ході виконання програми;
- ***TextBox*** –відображає текст, введений в процесі створення форми, якийдоступний до редагування в процесі виконання програми;
- ***ListBox*** –відображає список текстових елементів;
- ***ComboBox*** –відображає список, що розкривається;
- ***CheckedListBox*** **–**відображає список із полосою прокрутки, який складається із елементів із прапорцями;
- ***CheckBox*** –відображає прапорець із полем для тексту;
- ***RadioButton*** - відображає кнопку, що може бути включена або виключена;
- ***RichTextBox*** – дозволяє задавати текст у звичайному текстовому форматі;
- ***MaskedTextBox*** –обмежує формат даних, що вводяться користувачем;
- ***WebBrowser*** –дозволяє переміщатися по веб-сторінці в межах форми;
- ***LinkLabel*** – дозволяє добавляти веб-посилання на інше вікно чи на вебвузол;
- ***DateTimePicker*** - відображає графічний календар, який дозволяє користувачу вибирати дату/час;
- ***ProgressBar*** **–**дозволяє спостерігати за ходом виконання в часі деякогопроцесу;

- ***PictureBox*** – дозволяє відобразити графічне зображення потрібного компонента.

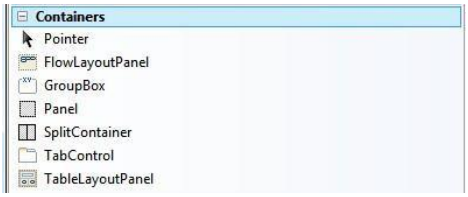


Рис.9. Елементи групи ***Containers***

Елементи управління групи **Containers** палітри компонентів**:**

- ***Panel*** – групує набір елементів управління для забезпечення загальноїповедінки;
- ***GroupBox*** – груповий контейнер, що використовується для розділеннякомпонентів на підгрупи;

Рис.10. Елементи групи ***Menus & Toolbars***

Елементи управління группи **Menus & Toolbars** палитри компонентів пов’язані із організацією меню:

- **MenuStrip** – створює головне меню додатку, за допомогою якого здійснюється управління роботою як всього додатку, так і його окремих блоків;
- **ContextMenuStrip** – створює контексні меню;
- **ToolStrip** – дозволяє створювати панелі інструментів із використанням різних стилів та принципів.

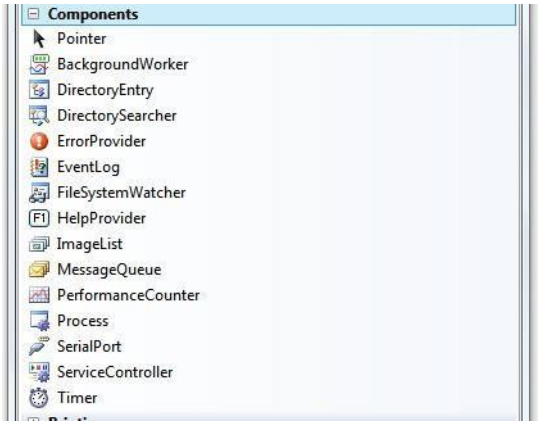


Рис.11. Елементи групи ***Components***

Елементи управління групи **Components** палітри компонентів пов’язанііз часом**:**

- ***Timer –*** задає лічильник часу;

Обробка подій

Після того, як потрібні компоненти інтерфейсу вибрані та поміщені на форму,необхідно створити процедуру обробки подій за допомогою вікна ***Properties***.

Для цього вибираємо необхідний елемент управління і у вікні ***Properties*** натискаємо кнопку **Events** і зі списку доступних вибираємо необхідну подію

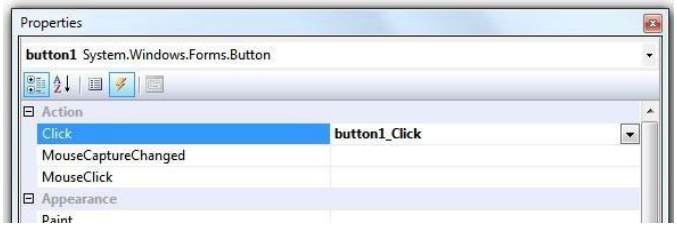


Рис.12. Налаштування процедури обробки події для кнопки ***(Button)***.

Наступним кроком є написання відповідного коду для обробки події – наприклад натискання кнопки. Подальший процес програмування полягає в тому, що в тіло шаблона необхідно записати оператори, які визначають реакцію компонента на подію з урахуванням переданих функції фактичних значень її параметрів. Шаблони всіх графічних елементів форми генеруються автоматично по мірі їх розміщення на робочому столі. Для переходу до шаблону відповідного графічного елементу достатньо двічі клікнути мишкою на ньому.

```
private: System::Void button1_Click(System::Object^
sender,System::EventArgs^  e)
    {


    }
```

Рис.12. Приклад пустого обробника події для кнопки button1 (після подвійного клікання на ній)

Особливості редагування коду

Одночасно із створенням нової форми створюється спеціальний програмний модуль із ім’ям форми та розширенням h - Form1.h. Це h–файл (заголовочний файл), в якому знаходиться опис форми та обробки подій компонентів, які використовуються в проекті. Перейти до редактора коду із вікна форми можна за допомогою комбінації клавіш **<Ctrl>+<Alt>+<0>**, або виконавши команду **Code** головного меню **View**.

Із режиму написання коду в режим дизайнера форми можна переключитися комбінацією клавіш <Shift>+<F7>.

Робота із рядковими змінними

Із рядковими змінними типу ***String*** працювати набагато зручніше на відміну від традиційних рядкових змінних типу ***char***. Для них визначено значна множина зручних методів (***Insert, Remove*** і.т.д), текстові поля вводу/виводу працюють тільки із ними, та і всі текстові поля властивостей компонент форми також є типу ***String^***. Приклад опису рядкової змінної із її одночасною ініціалізацією :

```
String^ st = "Текст рядкової змінної";
```

Приклад виводу рядкової змінної ***st*** типу ***String^*** у текстове поле ***textBox1***

```
textBox1->Text = Convert::ToString(st);
```

Приклад №1

Створення форми для вводу трьох чисел та їх впорядкування за зростанням при натисканні кнопки.

Використовуючи панель ***ToolBox*** розташовуємо на формі потрібні елементи керування та надаємо їй бажаного дизайну. В нашому прикладі:

- три текстові поля ***textBox1, textBox2, textBox3***;
- три підписи ***label*** для позначення цих полів;
- кнопка ***button1*** для запуску програми на виконання (обробки події – натискання кнопки).

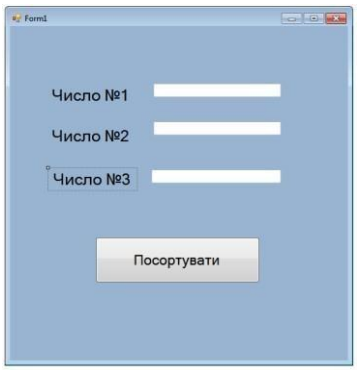


Рис.13. Вигляд форми після розміщення на ній елементів управління

Після формування форми переходимо власне до написання коду програми. Обробка події (виконання операції сортування) повинна викликатися кнопкою «Посортувати», тому двічі клацаємо на ній і переходимо до вікна редагування коду із заготовкою функції для обробки події клік кнопки:

```
private: System::Void button1_Click(System::Object^
sender, System::EventArgs^  e) {
}
```

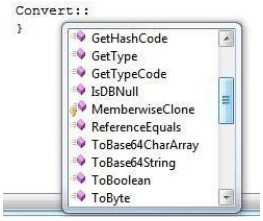
В тіло цієї функції необхідно вписати код, що власне реалізує алгоритм сортування даних. Для перетворення даних, введених у елемент керування

textBox в ціле число необхідно звернутися до класу ***Convert*** згідно наступного синтаксису:

```
int a; a=Convert::ToInt32(textBox1->Text);
```

Тут дані, введені в текстове поле ***textBox1*** будуть перетворені до цілогоформату і результат буде присвоєний змінній `a`. Можливі інші варіанти

перетворення можна побачити після введення команди ***Convert::***



Наприклад, для перетворення введених до текстового поля даних до дійсного формату необхідно ввести наступний фрагмент коду:

```
double a; a=Convert::ToDouble(textBox1->Text);
```

Для виводу результату в текстове поле необхідно скористатися зворотнімперетворенням даних до текстового формату:

```
Convert::ToString(a);
```

Вивід тексту у текстове поле **textBox1** матиме наступний вигляд:

```
textBox1->Text = Convert::ToString(a);
```

Тут **a** –імя змінної, значення якої буде перетворене до текстового формату.

Процедура читання даних із текстових полів **textBox1, textBox2, textBox3**

та їх перетворення до дійсного формату матиме наступний вигляд:

```
double a,b,c; a=Convert::ToDouble(textBox1->Text);
b=Convert::ToDouble(textBox2->Text);
c=Convert::ToDouble(textBox3-
>Text);
```

Остаточний код функції для обробки натискання кнопки **button1**:

```
double a,b,c;
a=Convert::ToDouble(textBox1->Text);
b=Convert::ToDouble(textBox2->Text);
c=Convert::ToDouble(textBox3-
>Text);

if (a<b && a<c){
//вивід значення змінної a у текстове поле textBox1
//попередньо змінна a перетворюється до типу String
textBox1->Text = Convert::ToString(a);
```

```
    if (b<c){textBox2->Text = Convert::ToString(b);
textBox3->Text = Convert::ToString(c);} else
    {textBox2->Text = Convert::ToString(c); textBox3-
>Text = Convert::ToString(b);};};

    if (b<a && b<c){ Convert::ToString(b); textBox1-
    >Text =
    if (a<c) {textBox2->Text = Convert::ToString(a);
textBox3->Text = Convert::ToString(c);} else
    {textBox2->Text = Convert::ToString(c); textBox3-
    >Text = Convert::ToString(a);};};

    if (c<a && c<b){ textBox1->Text =
Convert::ToString(c);
    if (a<b) {textBox2->Text = Convert::ToString(a);
textBox3->Text = Convert::ToString(b);} else {textBox2-
>Text =Convert::ToString(b); textBox3->Text =
Convert::ToString(a);};};
```

Примітка: при введенні дійсних чисел в текстові поля при виконанні графічної форми розділовий знак між цілою та дробовою частиною – кома!!!

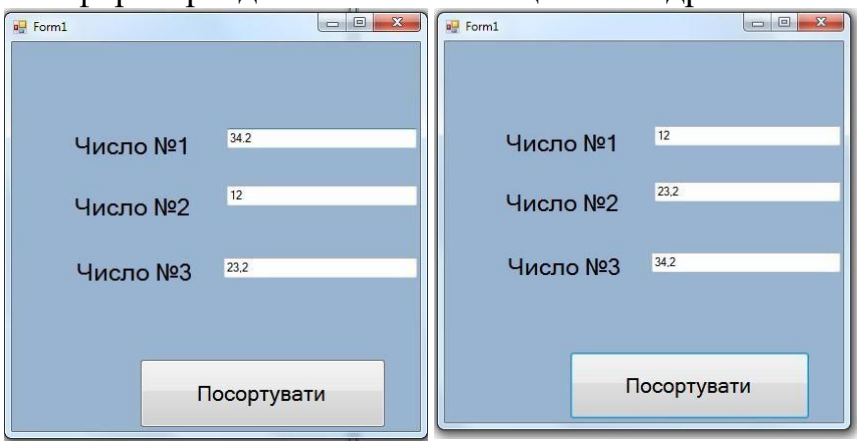


Рис.14. Результат роботи – до і після виконання операції сортування

Завдання для індивідуального виконання

Створити та запрограмувати візуальний проект ***Windows Form*** для розв'язання задачі згідно варіанту.

1. Дано три цілих числа. Знайти кількість додатних чисел в цьому наборі.
2. Дано два числа. Вивести спочатку більше, а потім менше з них.
3. Дано дві змінні дійсного типу: A, B. Перерозподілити значення даних змінних так, щоб A виявилось меншим із заданих значень, а B - більше. Вивести нові значення змінних A і B.
4. Дано дві змінні цілого типу: A і B. Якщо їхні значення не рівні, то

присвоїти кожній змінної суму цих значень, а якщо рівні, то присвоїти змінним нульові значення. Вивести нові значення змінних.

5. Дано три числа. Знайти найменше з них.

6. Дано три числа. Знайти середнє з них (тобто число, розташоване між найменшим і найбільшим).

7. Дано три числа. Вивести спочатку найменше, а потім найбільше з даних чисел.

8. Дано три числа. Знайти суму двох більших з них.

9. На числовій осі розташовані три точки: A, B, C. Визначити, яка з точок B і C розташована ближче до A, і вивести цю точку і її відстань від точкиA.

10. Дано координати точки, що не лежить на координатних осях OX та OY.Визначити номер координатної чверті, в якій знаходиться дана точка.

11. Дано цілочисельні координати трьох вершин прямокутника, сторони якого паралельні координатним осям. Знайти координати його четвертої вершини.

12. Дано номер року (додатне ціле число). Визначити кількість днів в цьому році, враховуючи, що звичайний рік нараховує 365 днів, а високосний.

13. Написати програму, яка вводить вік користувача і, якщо йому більше 18 років, повідомляє, що він має право голосу. В іншому разі вона обчислює, через скільки років користувач буде мати право голосу.

14. Дано координати двох точок. Визначити, чи розміщені вони на одному колі із центром в початку координат.

15. Із клавіатури вводиться номер місяця в році. Виводиться повідомлення про назву місяця та пору року. Якщо місяця із таким номером не існує – виводиться відповідне повідомлення.

16. Дано радіус кола та довжина сторони квадрата. Перевірити, чи пройде квадрат крізь коло.

17. Ввести із клавіатури три числа та знайти найменше та найбільше серед цих чисел.

18. Дано радіус кола та довжина сторони квадрата. Визначити, чи пройде круг через квадрат.

19. На площині задано коло радіусу R із центром в початку координат. Ввести координати точки та визначити, чи знаходиться вона на колі, чи всередині кола, чи за його межами.

20. Прямокутник задано координатами його вершин. Визначити, чи належить введена точка області прямокутника.

21. Два кола задано їхніми радіусами та координатами центрів. Визначити кількість спільних точок у цих кіл (одна, дві, чи ні однієї).

22. Із клавіатури вводиться порядковий номер місяця. Програма виводить назву пори року. Передбачити перевірку коректності вводу номера місяці.

23. Дано координати точки $M(x, y)$. Визначити, якому квадранту системи координат належить дана точка.

Дано функцію виду координати двох точок. Визначити,яка із точок

знаходиться ближче до параболи.

24. Із клавіатури вводиться п’ять цілих чисел. Визначити кількість парник чисел та кількість чисел, кратних 3.

25. Коло задано радіусом та координатами центру. Визначити, як лежить початок координат відносно кола – в середині, на колі чи за його межами.

26. Із клавіатури вводиться оцінка в п’ятибальній системі. На екран виводиться відповідна оцінка в університетській системі: відмінно, добре, задовільно або незадовільно.

27. Дано три числа (в межах від 1 до 9 як константи). Із клавіатури вводяться три довільні числа, а на екран виводиться повідомлення про кількість вгаданих чисел.

28. Дано пряма $ax + by + c = 0$ та парабола $y = a_1x^2 + b_1x + c_1$. Визначити кількість точок перетину прямої з параболою: дві, одна, жодної.

7. Практичне завдання №2
Тема: Створення Windows додатків в середовищі MS Visual C++ із використанням додаткових елементів вводу/виводу даних

Мета: формування навичок використання різних елементів управління упрограмах **Windows Forms**

Для виконання роботи необхідно знати:

- використання та налаштування *MessageBox;*
- вставка зображення на графічну форму;
- використання та налаштування ***DataGridView***.

Теоретичні відомості
Виведення повідомлень із використанням MessageBox

Для виведення повідомлень у окремому від графічного додатку вікні використовується стандартний клас **MessageBox**. Метод **Show** цього класу виводить заданий текст повідомлення користувачу:

Приклад:

```
MessageBox::Show("Сортування успішно виконано");
```

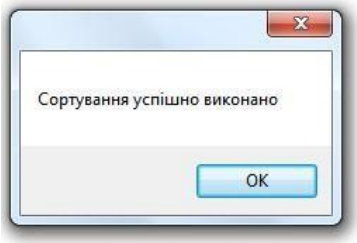


Рис.15. Результат роботи ***MessageBox***

Крім тексту повідомлення, можна задати інші параметри діалогового вікна ***MessageBox***. Так, можна вказати, які кнопки необхідно розташувати у

вікні діалогового вікна ***MessageBox***. Цей параметр задається константами з переліку ***MessageBoxButtons***.

Таблиця 2

Можливі значення параметра *MessageBoxButtons*

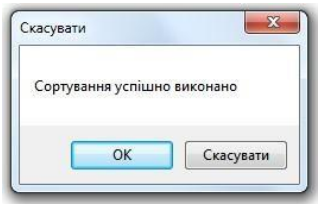
Константа	Кнопки, які відображаються у вікніMessageBox
OK	OK
OKCancel	OK, Cancel
YesNo	Yes, No
YesNoCancel	Yes, No, Cancel
RetryCancel	Retry, Cancel
AbortRetryIgnore	Abort, Retry, Ignore

Наприклад

```
MessageBox::Show("Сортування успішно виконано","Скасувати",MessageBoxButtons::OKCancel);
```

Виведе наступне вікно ***MessageBox***

Метод ***MessageBox.Show*** дозволяє також вибирати один із кількох можливих значків для розташування в лівій частині діалогового вікна. Він



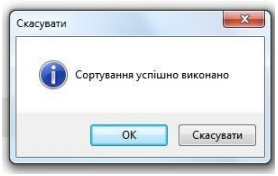
задається у вигляді константи з переліку MessageBoxIcon

Константа	Значок
Asterisk, Information	
Error, Stop	
Exclamation, Darning	
Question	
None	Значок не виводиться

Наприклад

```
MessageBox::Show("Сортування успішно виконано","Скасувати",MessageBoxButtons::OKCancel,Message
BoxIcon::Asterisk);
```

виведе наступне вікно ***MessageBox***



Якщо у вікні ***MessageBox*** є декілька кнопок, то для того, щоб визначити, яку кнопку клацнув користувач, програма повинна проаналізувати значення, яке повертає метод ***MessageBox.Show***. Він може повернути одне з декількох значень з переліку ***DialogResult***

Можливі значення DialogResult

Константа	Кнопка, при виборі якої повертається ця константа
Abort	Abort
Cancel	Cancel
Ignore	Ignore
No	No
None	Модальне вікно діалогу продовжує працювати
OK	OK
Retry	Retry
Yes	Yes

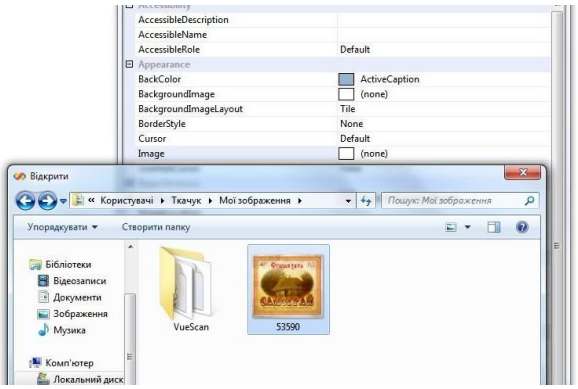
Наприклад

```
DialogResult=(MessageBox::Show("Сортування успішно виконано","Скасувати",MessageBoxButtons::OKCancel,
                MessageBoxIcon::Asterisk)) ;
if(Convert::ToString(DialogResult)=="Cancel")
                textBox3->Text = "Скасовано";
```

Тут якщо буде натиснута кнопка ***Скасувати*** у третьому полі буде виведено повідомлення «***Скасовано***».

Вставка зображень на графічну форму

Для вставлення у форму зображень служить ***PictureBox***. Після



розміщення за допомогою конструктора на формі ***PictureBox*** необхідно за допомогою вікна ***Properties***, вкладка ***Image*** задати малюнок із файлової системи комп'ютера та виконати його налаштування:

Рис.16. Вставка зображення за допомогою ***PictureBox***

За допомогою властивості ***SizeMode*** можна вписати зображення в межі всього створеного блоку ***PictureBox1***.вибравши пункт ***StretchImage***.

Створення форми з використанням *DataGridView*.

DataGridView – це елемент управління, який дозволяє відображати будьякий вид даних в комірках прямокутної сітки. Елементи управління ***DataGridView*** дозволяють відображати та редагувати прямокутний масив даних. Його модна також використовувати для відображення практично будьяких даних, згенерованих безпосередньо при виконанні програми. Дані елементу управління відображаються в прямокутному масиві комірок, які можна набір рядків та стовпців. Кожен стовпець комірок має у верхній частині комірку заголовка, яка, як правило, містить пояснювальну текстову інформацію, а на початку кожного рядка знаходиться комірка його заголовка.

Кількість стовпців у ***DataGridView*** можна задати наступним чином (буде створено три стовпці):

```
dataGridView1->ColumnCount = 3;
```

Кількість рядків ***DataGridView*** можна задати наступним чином (будестворено пять рядків):

```
dataGridView1->Rows->Add(5);
```

Для кожного стовпця, при бажанні, можна задати заголовок наступнимчином:

```
dataGridView->Columns[0]->Name="Name";
dataGrldView-
>Columns[1]->Name="Phone Number"; dataGridView-
>Columns[2]->Name="Address";
```

Приклад створення Windows – додатку, який дозволяє створити матрицюзаданої розмірності, ввести елементи матриці, або згенерувати їх за допомогою генератора випадкових чисел та підрахувати суму діагональних елементів матриці та кількість парних елементів.

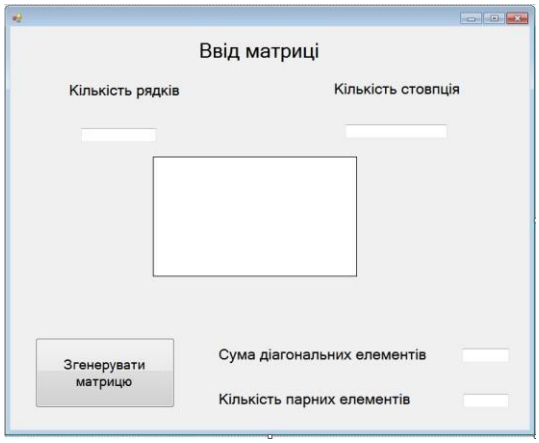


Рис.17. Можливий вигляд форми після розміщення на ній елементів управління

Код функції для обробки події - натискання кнопки ***button1*** матиме наступний вигляд:

```
    int m, n,kil,sum;
    /*Очищення стовпців компоненту DataGridView, якщо
вони непорожні */
    dataGridView1->Columns->Clear();
    //Перевірка, чи введено кількість рядків та
стовпцівif ((textBox1->Text!="")&&(textBox2->Text!=""))
    {
    //Читання кількості рядків m =
Convert::ToInt32(textBox1->Text);
    //Читання кількості стовпців n =
Convert::ToInt32(textBox2->Text);
    //Заповнення DataGridView стовпцямиdataGridView1-
>ColumnCount = n;
    //Заповнення DataGridView рядками dataGridView1-
>RowCount = m;} else
    //Вивід повідомлення у вікно MessageBox
    {MessageBox::Show( "Заповніть форму даними",
"Помилкавволу даних ",
    MessageBoxButtons::OK, MessageBoxIcon::Exclamation
    );} //Опис масиву та аналіз його елементів int
A[50][50]; sum=0; kil=0;
    for (int i = 0; i < m; i++)for (int j = 0; j < n;
j++)
    {
    A[i][j] = rand()%100-50;
    //Вивід згенерованого масиву у компоненту
DataGridView dataGridView1->Rows[i]->Cells[j]->Value=
Convert::ToString(A[i][j] ); if (i==j)
    sum=sum+A[i][j]; if (A[i][j]%2==0) {kil++;}
    }
    //Вивід результатів у відповідні вікна textBox
```

```
textBox3-
    >Text=Convert::ToString (sum); textBox4-
    >Text=Convert::ToString (kil);
```

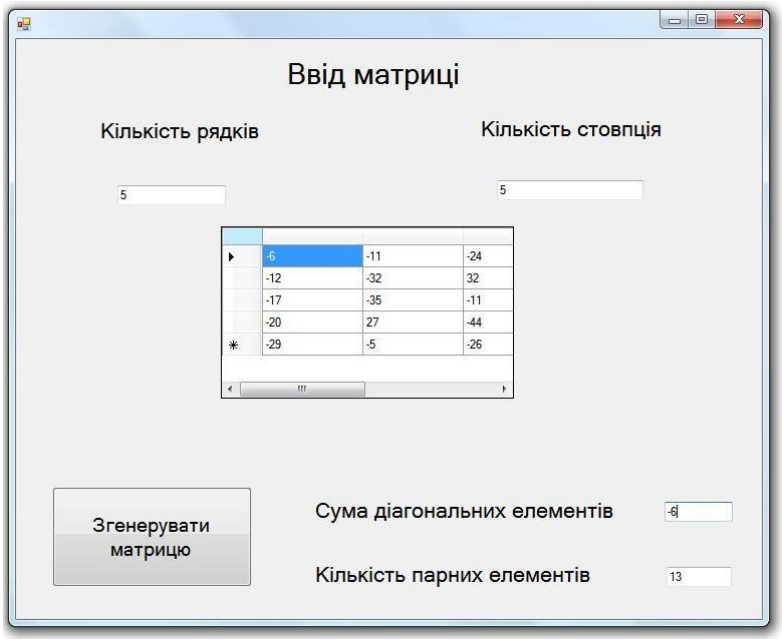


Рис.18. Результат роботи форми після виконання операції генерації матриці

Завдання для індивідуального виконання

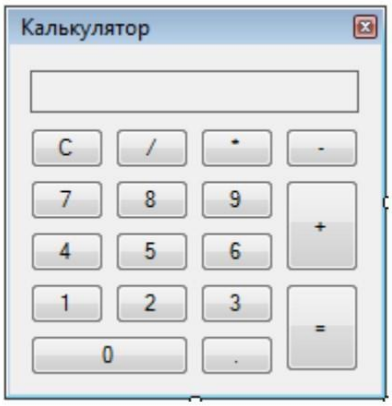
Завдання №1 Створити проект ***Windows Form*** для розв'язання задачі згідноваріанту.

№ варіанту	Завдання
1.	Визначити, чи є задана квадратна матриця A (5,5) симетричної щодоголовної діагоналі.
2.	Дано цілочисловий масив A, який складається з 12 елементів. Створити та вивести масив C, який складається з остач діленняелементів масиву A на ціле число k
3.	Задано матриця У (4,4). Визначити, чи відсортовані всі елементипершого стовпчика за зростанням.
4.	Задано матриця K (5,5). Поміняти місцями максимальний елементкожного рядка з першим елементом відповідного рядка.
5.	Переписати перші елементи кожного рядка матриці D (3,3), якібільше 10, в масив B.
6.	Задано матриця Q (5,5). Замінити останній нуль в кожному рядку на5.
7.	Задано матриця D (4,4). Визначити максимальний серед позитивних, мінімальний серед негативних і поміняти їх місцями.
8.	Задано матриця A (4,4). Замінити перший нуль в кожному стовпці накількість нулів у цьому стовпці.
9.	Задано матриця F (9,3). визначити, чи рівні всі елементи першогостовпчика відповідних елементів головної діагоналі. Якщо ні, то поміняти їх місцями.
10.	Задано матриця C (5,5). Отримати вектор B, кожен елемент якогодорівнює кількості нулів, що стоять у стовпці матриці.

11.	Задано матриця У (4,4). Якщо в рядку є хоча б одна одиниця, тозамінити цей рядок нулями.
12.	Задано матриця Q (3,3). Якщо на головній діагоналі стоїть нуль, товідповідний рядок замінити одиницями.
13.	Задано матриця D (4,4). Якщо максимальний елемент матриці стоїтьна головній діагоналі, то всі елементи головної діагоналі зробити рівними максимальному.
14.	Задано матриця З (5,5). Якщо мінімальний елемент коштує в першому рядку, то всі елементи, які стоять в рядку за ним, замінитинулями.
15.	Задано матриця А (4,4). Якщо максимальний елемент матрицідорівнює сумі елементів першого рядка, то поміняти місцями перший рядок з тим рядком, де знаходиться максимальний елемент.
16.	Задано матриця А (4,4). Якщо максимальний елемент матрицідорівнює сумі елементів першого рядка, то поміняти місцями перший рядок з тим рядком, де знаходиться максимальний елемент.
17.	Знайти мінімальний по модулю елемент та номер рядка і стовпця, уякому він знаходиться
18.	Дано цілочисловий масив А, який складається з 12 елементів. Створити та вивести масив С, який складається з непарних чиселмасиву А, полічити кількість елементів масиву С.
19.	Дано вектор, який містить К елементів. Вилучити з нього елементи,які знаходяться між максимальним і мінімальним елементами. Вивести значення максимального і мінімального елементів.
20.	Дано масив А, який складається з 19 елементів. Вивести три перші від’ємні елементи цього масиву у порядку зростання та їх порядковіномерами.
21.	В заданих двох векторах А і В однакової розмірності N знайтиокремо найбільше і найменше значення сум їх елементів.
22.	Дано одновимірний масив С, який складається з 15 елементів. Обчислити і вивести добуток непарних елементів і їх кількість.
23.	Дано одновимірний масив С, який складається з 16 елементів. Обчислити середнє арифметичне значення парних елементів масиву,які діляться на 3 з остачею 1.
24.	Дано цілочисловий одновимірний масив А, який складається з 14елементів. Обчислити і надрукувати суму парних елементів, які знаходяться на непарних позиціях, і їх кількість.
25.	Дано матрицю Е розміром 4х6. Сформувати та вивести матрицю Q, значення елементів кожного стовпця якої обчислюється як різниця відповідних елементів двох суміжних стовпців матриці Е.
26.	Дано матрицю В розміром 5х6. Поділити елементи кожного рядка на елемент, який знаходиться в третьому стовпці цього рядка.
27.	Дано квадратну матрицю А 6-го порядку. Знайти суму елементів матриці, які розміщені в рядках з від’ємним елементом на головнійдіагоналі. Обчислити кількість таких рядків.
28.	Дано матрицю Т розміром 6х7. Знайти максимальний і мінімальнийелементи для кожного стовпця матриці Т.
29.	Дана квадратна матриця С. Деякі елементи цієї матриці, які розміщені на головній діагоналі рівні нулю. Вилучити з матриці С ті стовпці, в яких елементи головної діагоналі рівні нулю. Вивести новуматрицю.

30.	Дано два одновимірні масиви, які складаються не більш як з 30 елементів кожний. Знайти півсуму максимальних елементів заданихмасивів.

Завдання №2 Написати програму «калькулятор» з графічним інтерфейсом, щовиконує чотири основні арифметичні дії.



Інтерфейс калькулятора

8. Практичне завдання №3

Тема: Елементи управління роботою Windows додатків

Мета: формування навичок використання різних елементів управління у програмах **Windows Forms**

Для виконання роботи необхідно знати:

- використання та налаштування радіо кнопки (***RadioButton***)*;*
- призначення та можливості використання елементу ***CheckBox***;
- призначення та використання елементу ***GroupBox***;
- використанням ***ListBox*** для відображення списку елементів;
- використання **`panel`** для побудови графіка функції;
- Використання горизонтальних та вертикальних полів прокрутки.

Теоретичні відомості

Створення форми з різними видами кнопок.

Для ілюстрації використання різного виду кнопок розглянемо наступний приклад:

додаток повинен виконувати наступні функції: радіокнопки (***RadioButton***) задають текст повідомлення, що буде виводитися по натисканню на звичайну кнопку. Прапорець (***CheckBox***) повинен визначати чи виводити повідомлення, чи ні. Для цього створюємо новий проект, додаємо на форму один елемент керування ***GroupBox***, три елементи ***RadioButton***, один елемент ***CheckBox*** й один елемент ***Button***. Важливо, що всі три радіокнопки поміщені в один ***GroupBox*** для їх зв’язаної роботи. Інакше вони не будуть зв'язані між собою і працюватимуть незалежно одна від одної.

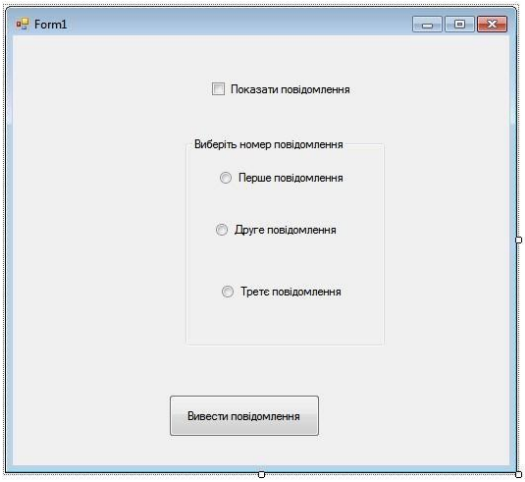
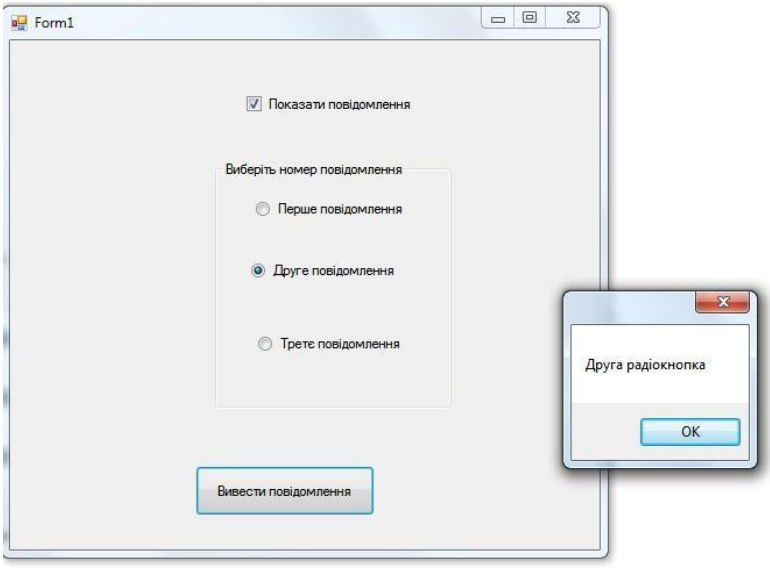


Рис.17. Проект форми для ілюстрації використання кнопок.

В тіло функції для кнопки ***button1*** введено наступний код:

```
    // перевіряємо першу радіокнопку if(radioButton1-
>Checked == true)

    //перевіряємо, чи встановлений прапорець if
(checkBox1->Checked ==true) MessageBox::Show("Перша
                        радіокнопка");;
                // перевіряємо другу радіокнопку

    if (radioButton2->Checked == true)
    //перевіряємо, чи встановлений прапорець

    if (checkBox1->Checked ==true)
MessageBox::Show("Друга радіокнопка");;

    // перевіряємо третю радіокнопку if(radioButton3-
>Checked == true)

    //перевіряємо, чи встановлений прапорецьif
(checkBox1->Checked ==true)
    MessageBox::Show("Третя радіокнопка");;
```

Відкомпілювавши та запустивши програму отримаємо:



На екрані нічого не буде виведено, якщо немає прапорця в полі «Показати повідомлення».

Створення форми з використанням бігунка, індикатора прогресу й регулятора чисельних значень.

Для ілюстрації роботи перерахованих компонент розглянемо приклад додатку у якому бігунок й елемент керування ***NumericUpDown*** управляють індикатором прогресу. При цьому бігунок і ***NumericUpDown*** повинні працювати синхронно: при зміні значення одного елемента значення іншого повинне змінюватися автоматично на ту ж величину.

Створимо новий Windows-додаток та додамо на форму наступні елементи керування:

- ***TrackBar*** ;
- ***ProgressBar*** ;
- ***NumericUpDown***.

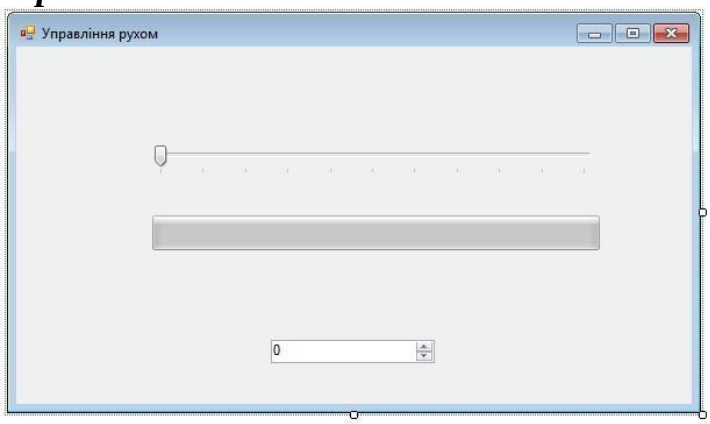


Рис. 18. Можливий варіант дизайну форми.

Для елементу ***TrackBar1***задамо наступні властивості:

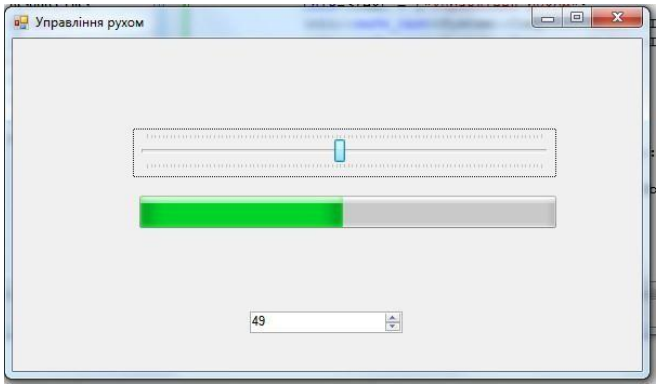
Maximum —100

TickStyle — Both

NumericUpDown1 і ***trackBar1*** є керуючими елементами, а ***progressBarl*** - керованим. Компонент ***TrackBar*** (індикатором прогресу) має подію ***Scroll***, що обробляє переміщення покажчика бігунка. У функції з ім'ям ***trackBar1_Scroll***. додамо наступний код:

```
int Value = trackBar1->Value; numericUpDown1-
>Value = Value; progressBar1->Value = Value;
```

Тепер при русі курсору бігунка ***TrackBar*** будуть змінюватися як положення індикатора прогресу ***progressBarl*** та значення елементу ***numericUpDown1***, як це показано нижче:



Для можливості синхронного керування за допомогою регулятора числовихзначень ***NumericUpDown*** додамо оброблювач події ***ValueChanged*** для елемента ***numericUpDown1***. В код програми для функції із імям

numericUpDown1_ValueChanged введемо наступний код

```
int Value = (int)numericUpDown1->Value; trackBar1->Value = Value; progressBar1->Value = Value;
```

Як наслідок, при зміні положення бігунка індикатор прогресу і числовий регулятор змінять свої значення на відповідні величини. Індикатором прогресу можна також управляти за допомогою числового індикатора.

Створення форми з використанням ListBox.

ListBox являє собою елемент управління ***Windows*** для відображення списку елементів. Проілюструємо його використання на прикладі створення форми для табуляції функції. При використанні математичних функцій необхідно обов'язково підключити математичну бібліотеку `#include <cmath>`.

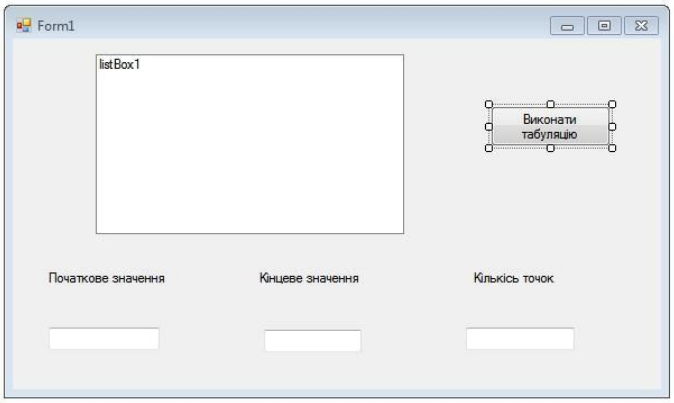


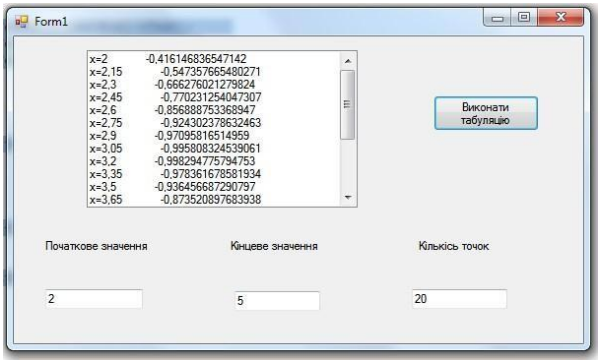
Рис.18. Проект форми, що ілюструє використання ***ListBox***

В тіло функції для кнопки ***button1*** введено наступний код:

```
double a,b,c;
// початок області табуляції
```

```
a=Convert::ToDouble(textBox1->Text);
    // кінець області табуляції
b=Convert::ToDouble(textBox2->Text);
    // кількість точок табуляції
c=Convert::ToDouble(textBox3->Text); double h;double R;
    // визначення кроку табуляції h=(b-
    a)/c;
    for(double t=a; t<=b; t=t+h)
    {
    // обчислення значення функції, яка табулюється
R=cos(t);
    //Вивід даних в listBox1 із пояснювальним текстом
listBox1->Items->Add("x="+t +" "+R); }
```

Відкомпілювавши та запустивши програму отримаємо:



Створення форми з використанням *PictureBox*.

В даному прикладі проілюстровано використання ***PictureBox*** для побудови у формі графіка функції

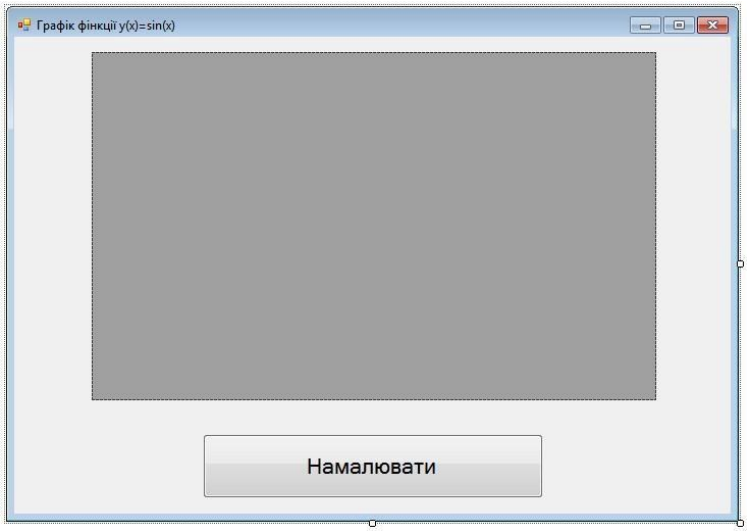
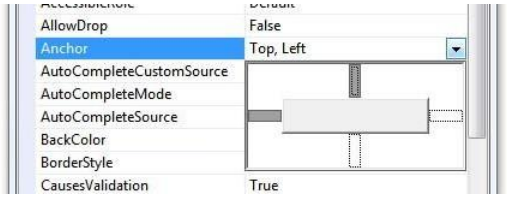
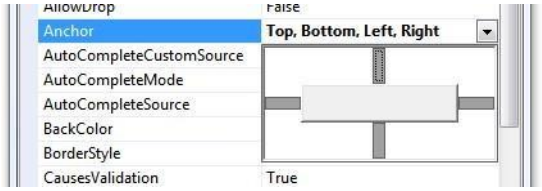


Рис.18. Проект форми для ілюстрації використання **`panel`**

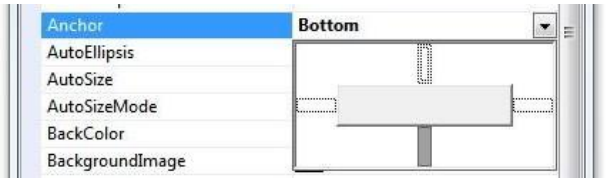
Для прив’язки розміру елементу ***PictureBox*** до форми, розміри якої можуть мінятися користувачем служить властивість ***Achor***:



Як видно, по замовчування прив’язка є до лівої верхньої частини форми. Для пропорційної зміни розміру елементу разом із вікном необхідно виділити правуі нижню частину:



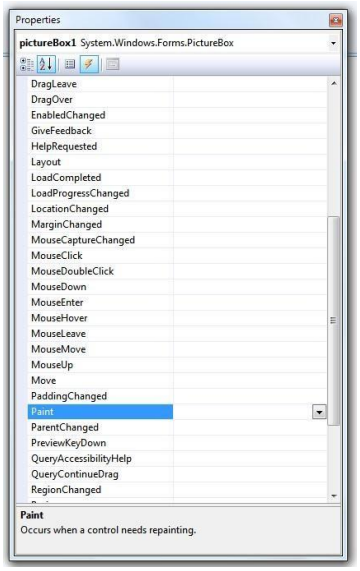
Для елементу ***button1*** задамо наступну властивість ***Achor***:



В тіло функції для кнопки ***button1*** введено наступний код (змінні ***W*** та ***H*** описати як глобальні):

```
    /*визначення параметрів вікна, у якому здійснюється
побудова графіка функції*/ W =PictureBox1->Width;
    H = PictureBox1->Height;
    //оновлення вікна PictureBox1_Paint, якщо
помінялисявхідні значення для побудови графіка функції
pictureBox1->Refresh();
```

Для добавлення події ***PictureBox1_Paint*** необхідно скористатися панеллювластивостей даного елементу згідно приведеного нижче малюнку:

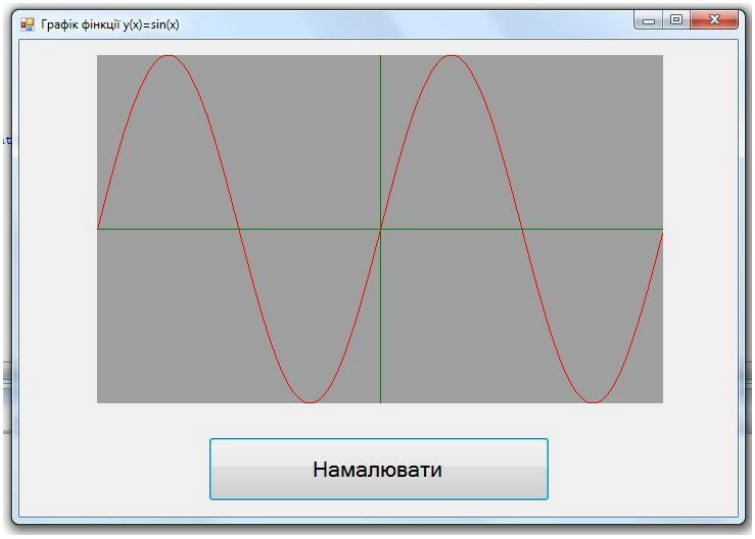


Та два рази клікнути лівою клавішею миші на вибраному пункті меню. Длядобавленої події ***PictureBox1_Paint*** введено наступний код:

```
    //заадння тексту
    String^ Text="Графік фунуції";
    //колір тексту
    Brush^ Кисть = gcnew SolidBrush(Color::Black);
    //шрифт(діє для всього додатку)
    Font = gcnew System::Drawing::Font("Times New
Roman",14,
    FontStyle::Bold);
    //вивід тексту у елемент PictureBox
    e->Graphics->DrawString(Text, Font, Кисть, 150,
50);
    // Координаты размещения текста double halfW = W /
2., halfH = H /2.;
    //побудова системи координат
    e->Graphics->DrawLine(System::Drawing::Pens::Green,
0,halfH, W, halfH);
    e->Graphics->DrawLine(System::Drawing::Pens::Green,
halfW, 0, halfW, H);
```

```
    //початкові дані для побудови  графіка функції int ixPrev
= -1, iyPrev =(int)halfH; // побудова графіка функціх;
for (int ix = 0; ix < W; ix++)
    {
    // перевід значення x в діапазон -1..1 floatx = (ix -
halfW) / halfW;
    // перевід значення x в діапазон  -2*pi..2*pi
    x =2*x*(float)Math.PI;
    // обчислення значення функції sin(x) float y =
(float)((sin(x))); // переводим yiз -1..1 в пікселі на
формі int iy = (int)(halfH - y * halfH);
    // Color визначає колір лінії графіка //графік
будуєтьсяяк відрізки, що зєднують попередне та наступне
    //обчислене значення функції
    e->Graphics->DrawLine(System::Drawing::Pens::Red,
ixPrev, iyPrev, ix, iy); ixPrev = ix; iyPrev = iy;
    }
```

Відкомпілювавши та запустивши програму отримаємо (текст на графічній формі не виводився):



Якщо приведений вище код помістити не в тіло функції для кнопки ***button1***, ав тіло функції ***panel1***, то графік функції буде будуватися зразу після відкриття форми без необхідності натиснути кнопку «***Намалювати***» .

Використання горизонтальних та вертикальних полів прокрутки.

Для ілюстрації створимо форму, у якій значення у відповідних текстових полях можна задавати за допомогою вертикальної та горизонтальної полоспрокрутки.

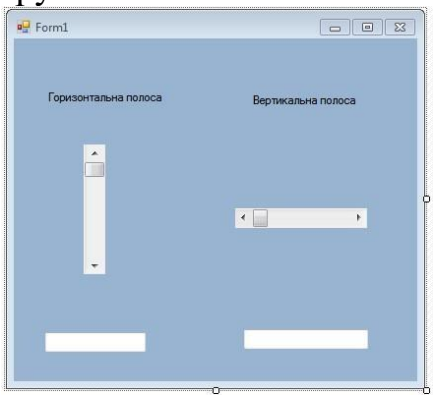


Рис.19. Проект форми для ілюстрації використання ***panel***

В тіло функції для вертикальної полоси прокрутки ***vScrollBar*** введемонаступний код:

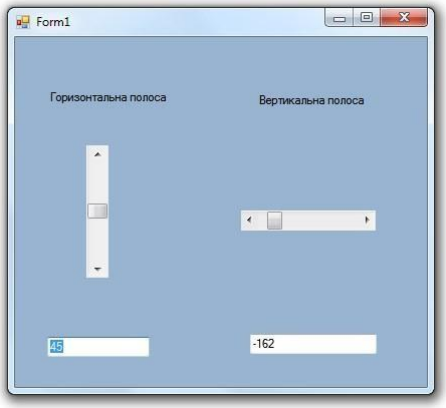
```
int rezult;
rezult=(vScrollBar1->Value); textBox1-
>Text = Convert::ToString(rezult);
```

В тіло функції для горизонтальної полоси прокрутки ***hScrollBar*** введено наступний код:

```
int rezult1;
rezult1=(hScrollBar1->Value); textBox2-
>Text = Convert::ToString(rezult1);
```

Діапазон зміни значень, що отримуються із полоси прокрутки можна налаштувати за допомогою пунктів меню вікна ***Properties Minimum*** та ***Maximum*** відповідно для ***vScrollBar*** та ***hScrollBar***

Відкомпілювавши та запустивши програму отримаємо:



Значення, що виводяться в полях ***textBox1*** та ***textBox2*** можуть

бути задані за допомогою відповідних полос прокрутки

Завдання для індивідуального виконання

Використовуючи елементи управління роботою Windows додатків, описаних у теоретичній частині до лабораторної роботи створити форму для табуляції функції у заданому діапазоні та побудувати її графік.

№ п/п	Функції	Область побудови та крок
1.	$f(x)=5\cos(2x+\pi)$	$[-1,2]$, крок $h=0.25$
2.	$f(x)=\cos^2(x)+2x$	$[-2,2]$, крок $h=0.1$
3.	$f(x)=3x^5+20x^2-20x+3$	$[1,4]$, крок $h=0.2$
4.	$f(x)=\sin(x-0.5)$	$[1,5]$, крок $h=0.4$
5.	$f(x)=\cos(\sqrt{x})$	$[2,5]$, крок $h=0.2$
6.	$f(x)=\left(x+\frac{2}{x}\right)$	$[1,3]$, крок $h=0.1$
7.	$f(x)=\cos\left(x+\frac{1}{x}\right)$	$[1.3,5.1]$, крок $h=0.3$
8.	$f(x)=\frac{x}{x^2-2.5}$	$[3,7]$, крок $h=0.15$

9. Практичне завдання №4

Тема: Створення багатовіконних додатків із можливістю вводу-виводу даних у файл.

Мета: формування навичок використання різних елементів управління у програмах **Windows Forms**

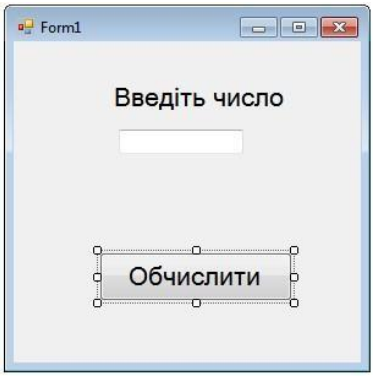
Для виконання роботи необхідно знати:

- створення двох зв’язаних форм;
- форматний вивід даних;
- створення меню за допомогою MenuStrip;
- призначення та можливості елементаму ***openFiledialog***;
- призначення та можливості елементаму ***saveFiledialog***;
- використання параметра ***Achor*** для налаштування роботи елементу ***textBox***

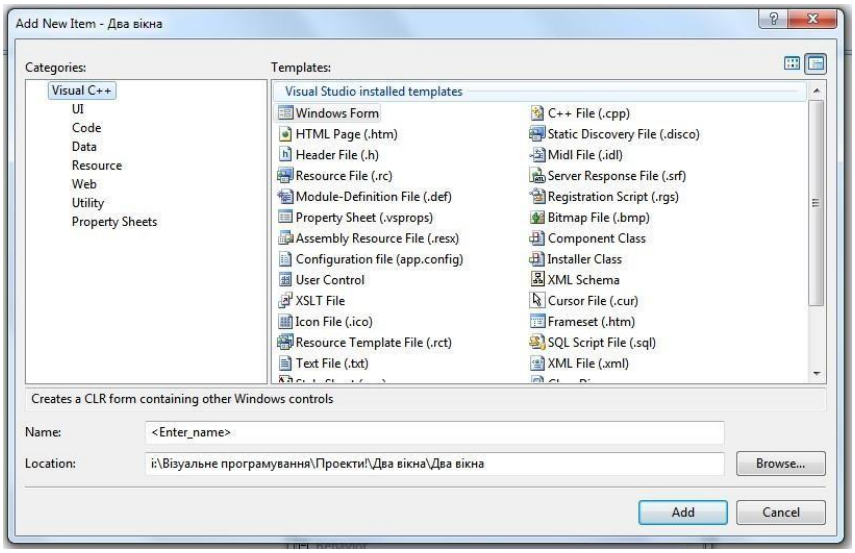
Теоретичні відомості Створення двох зв’язаних форм.

Вихідна (батьківська) форма може містити кілька додаткових (дочірніх) вікон. Проілюструємо порядок створення таких двох зв’язаних

формна наступному прикладі: перша форма містить два елементи управління - ***Toolbox*** та ***Button.*** Після вводу числа у текстовому полі та натискання кнопки результат піднесення числа до квадрату відкривається в дочірній формі. Для прикладу дизайн батьківської форми наступний:



Другу форма створюємо в цьому – проекті наступним чином: в пункті менюрозділу ***"Project"*** вибираємо ***"AddNewItem…"***.



Після цього вибираємо новий елемент ***" Windows Forms"***, задавши длянього імя – ***Form2***. Після цього на панелі появиться друга форма конструктора:

На другій формі розміщуємо елемент - ***Toolbox***:

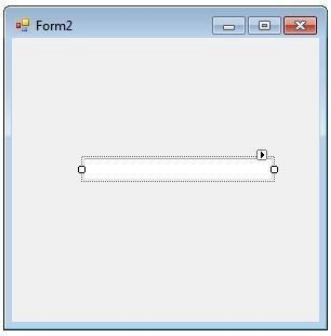
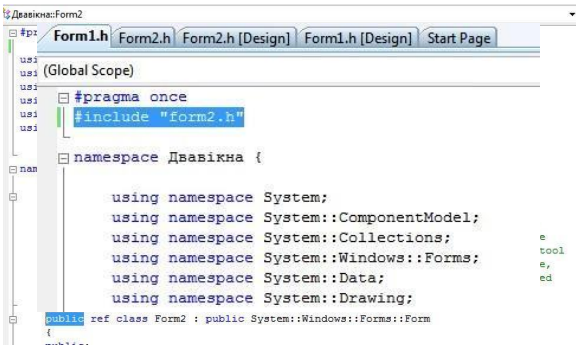


Рис.20. Приклад дизайну дочірної форми ***Form2***

Важливим є те, що елемент ***Toolbox1*** одночасно біде знаходитись на обох формах, тому його необхідно оголосити відритим членом класу (***public***) у верху коду форми ***Form2***(виділений фрагмент коду):

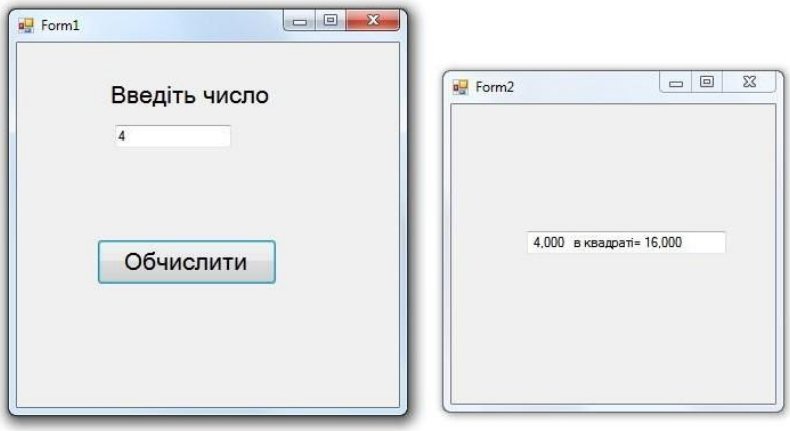
В коді форми ***Form1*** необхідно підключити бібліотеку другої форми ***#include "form2.h"***:



Для обробки події, пов'язаної із ***Button1***, що знаходиться на ***Form1*** необхідно написати наступний код:

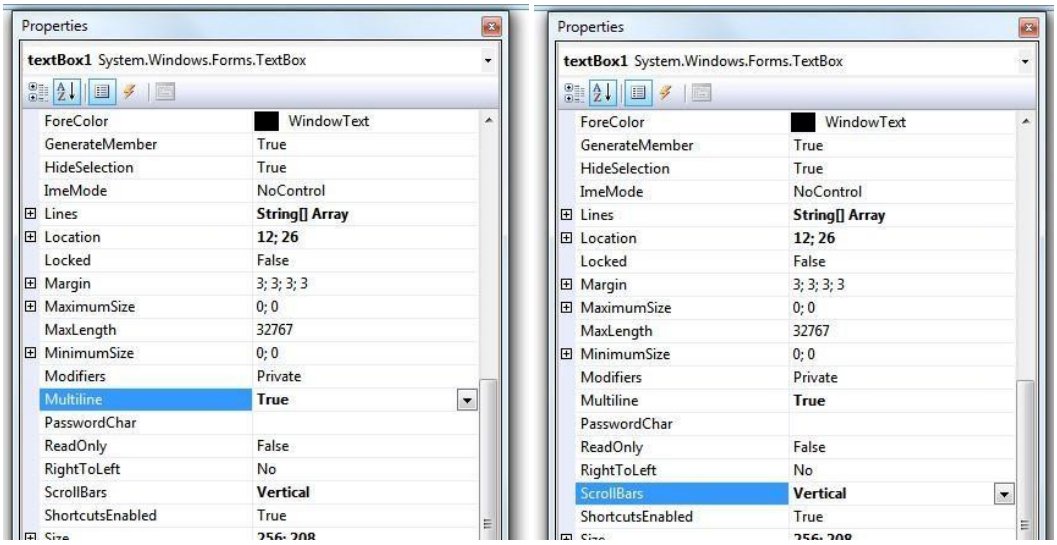
```
    double a, b = 0; a=Convert::ToDouble(textBox1-
>Text);b = a*a;
    Form2^ gform2 = gcnew Form2;
    //вивід другої форми gform2-
    >Show();
    //форматний вивід числа а та б
    gform2->textBox1->Text=String::Format("{0:F3} в
квадраті=
    {1:F3}",a, b);
```

Зверніть увагу на форматний вивід обчисленого та вихідного значення чисел `a` та `b` . Тут `{0:F3}` визначає місце в тексті, де буде виведена перша змінна та кількість знаків, що будуть виведені після коми, а `{0:F3}` – друга змінна (`b`) та формат її виводу. Результат роботи відкомпільованого проекту:

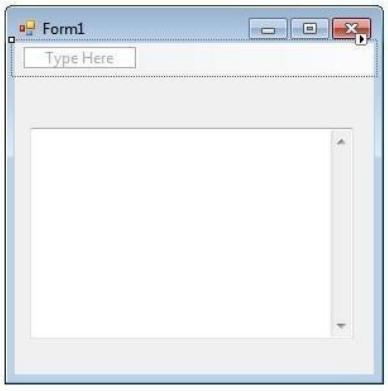


Створення меню за допомогою *MenuStrip*.

Елемент дозволяє випадаючі меню із набором можливих операцій. Для ілюстрації на формі будуть знаходитися елементи ***MenuStrip*** та ***textBox***. Для ***textBox*** задано налаштування:

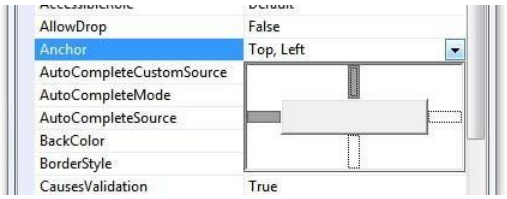


Дизайн форми наступний:

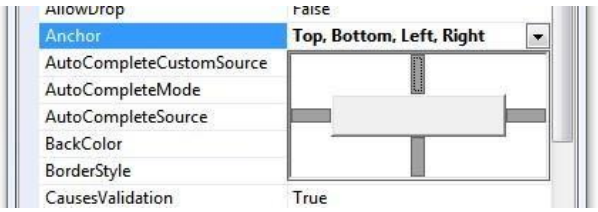


Для елементу ***MenuStrip*** формуємо наступні команди

Для прив’язки розміру елементу ***textBox*** до форми, розміри якої можуть мінятися користувачем служить властивість ***Achor***:

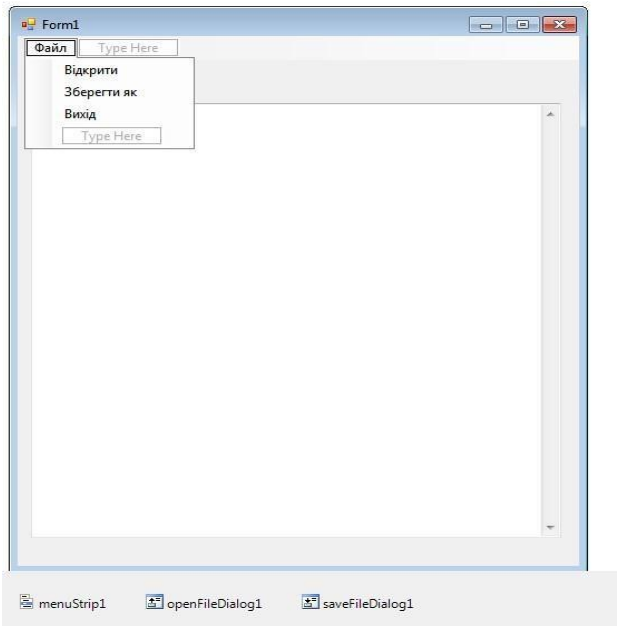


Як видно, по замовчування прив’язка є до лівої верхньої частини форми. Для пропорційної зміни розміру елементу разом із вікном необхідно виділити правуі нижню частину:



Відкриття та запис текстового файлу.

Як продовження до попереднього пункту створимо простий текстовий редактор. Доповнимо попередню форму елементами ***openFiledialog***, ***saveFiledialog*** та добавивши в ***MenuStrip*** пункт меню ***Вихід***:



Для налаштування обробки подій, пов’язаних із вибором відповідних пунктів меню введено наступні фрагменти коду:

- для функції завантаження графічного додатку ***Form1_Load***:

```
private: System::Void Form1_Load(System::Object^
sender, System::EventArgs^ e) { //задання заголовка
```

```
додатку this->Text = "Текстовый редактор";
     //задання імені файлу openFileDialog1->FileName =
"*.txt"; //фільтр для відображення файлів при відкритті
файлу openFileDialog1->Filter = "Текстовіфайли
     (*.txt)|*.txt|All files (*.*)|*.*";
     //фільтр для відображення файлів при закритті
saveFileDialog1->Filter = "Текстові файли
(*.txt)|*.txt|All files (*.*)|*.*";
     }
```

- для функції закриття графічного додатку ***Form1_Load*** для перевірки, чизміни в файлі:

```
     //перевірка, чи були зміни у файлі if(textBox1-
>Modified == false) return;
     //Вавід повідомлення, що файл булозмінено
     DialogResult=MessageBox::Show("Текст було змінено.
     \nЗберегти зміни?","Простий редактор",
MessageBoxButtons::OKCancel,
MessageBoxIcon::Exclamation);
     //Перевірка, яка кнопка була натичнута у MessageBox
if(Convert::ToString(DialogResult)=="Cancel") return;
     //textBox3->Text = "Скасовано"; else{  //Якщо
натиснуто "ОК", то викликається функція // Save()для
зберкження текстового файлу
     {
     Save(); return;
     }
```

- для пункту меню ***Відкрити***

```
     if(openFileDialog1->ShowDialog() ==
System::Windows::Forms::DialogResult::OK) {
     System::IO::StreamReader ^ sr = gcnew
     //відкриття елемента openFiledialog
     System::IO::StreamReader(openFileDialog1-
     >FileName);
     //читання даних із файлу у textBox1
     textBox1->Text = Convert::ToString(sr-
>ReadToEnd());
     //закриття елемента openFiledialogsr->Close();
     }
```

- для пункту меню ***Зберегти як:***

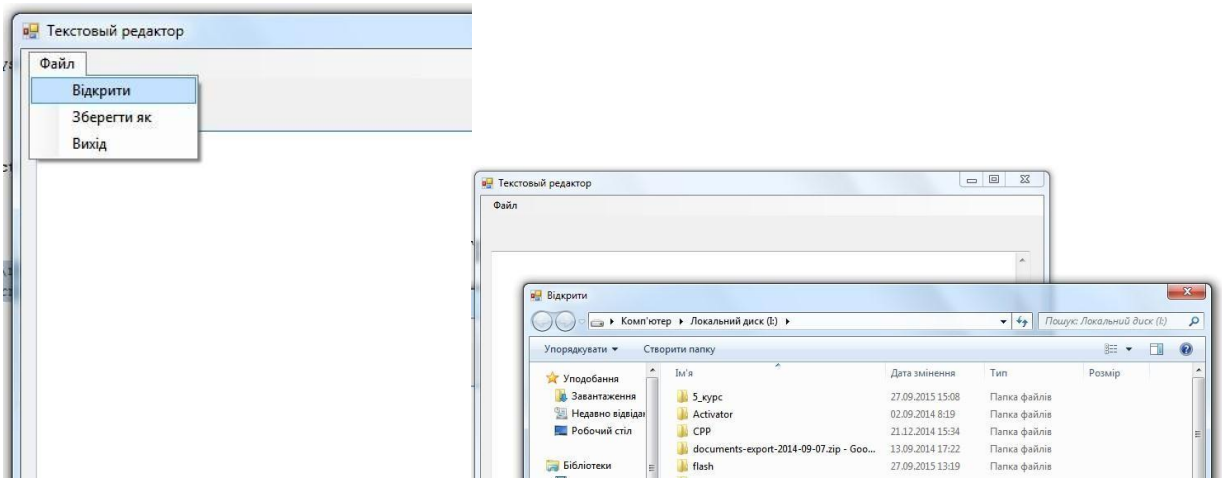
```
     //відкриття елемента saveFileDialog
saveFileDialog1-
     >FileName = openFileDialog1->FileName;
     if (saveFileDialog1->ShowDialog() ==
```

```
Windows::Forms::DialogResult::OK) Save();
     }
     //функція для зберкження текстового файлуvoid
Save()
     {
     try
     {
     SaveFileDialog ^saveFileDialog1 = gcnew
SaveFileDialog();
     //фільтр для відображення файлів при закритті файлу
saveFileDialog1->Filter = "Текстові файли|*.txt" ;
saveFileDialog1->FilterIndex = 2 ; saveFileDialog1-
     >RestoreDirectory = true ; if(saveFileDialog1-
     >ShowDialog() ==
System::Windows::Forms::DialogResult::OK)
     {
     //вивід даних у файл
IO::File::WriteAllText(saveFileDialog1-
     >FileName,textBox1->Text);
     }
     }
     catch (Exception^ Ситуация)
     {
     }
```

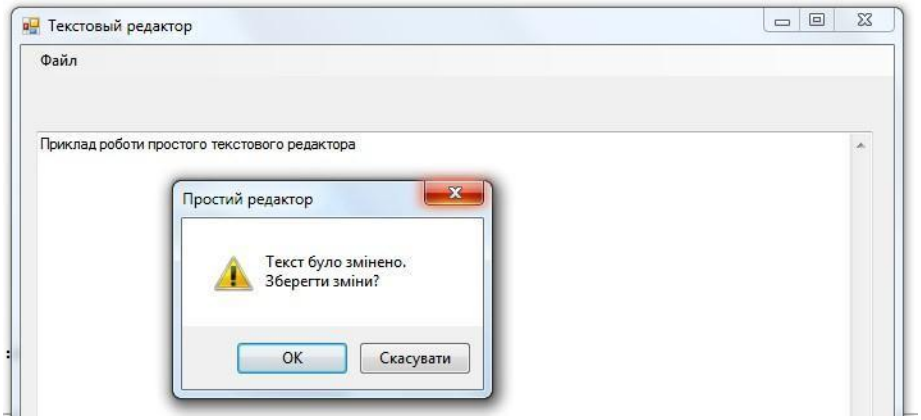
- для пункту меню ***Вихід:***

```
Close();
```

Результати виконання програми:відкриття файлу:



Спроба закриття документу чи виходу із віконного додатку, коли текстовийдокумент було редаговано:



Завдання для самостійної роботи

Використовуючи елементи управління роботою Windows додатків, описаних у теоретичній частині до лабораторної роботи створити форму для роботи із текстовими документами. Дизайн та призначення форми розробити самостійно.

7. СПИСОК РЕКОМЕНДОВАНИХ ДЖЕРЕЛ

1. С++. Алгоритмізація та програмування : підручник / Трофименко О. Г. [та ін.]. 2-е вид., перероб. і допов. Одеса : Фенікс, 2019. 476 с.

2. Mochurad Lesya I. Technologies of distributed systems and parallel computation [Text] : [monograph] / L. I. Mochurad, N. I. Boyko ; Lviv polytechnic nat. univ. Lviv : Bona, 2020. 261 p.

3. Nesterenko B. B. Formal means of the simulation of parallel processes and systems [Text] / B. B. Nesterenko, M. A. Novotarskyi ; Nat. acad. of sciences of Ukraine, Inst. of mathematics. Kyiv : Akademperiodyka, 2016. 192c.

4. Волонтир Л. О., Зелінська О. В., Потапова Н. А., Чіков І. А. Чисельні методи. Навчальний посібник. Вінниця: ВНАУ, 2021. 322 с.

5. Галісеєв Г.В.. Системне програмування : навч. посіб. / Галісеєв Г.В. Київ : Ун-т «Україна», 2019. 112 с.

6. Герасимов В.В. Розробка програмного забезпечення на платформі Java. Багатопоточне програмування і паралельні обчислення : навч. посіб. / В.В. Герасимов, Н. О. Матвєєва ; Дніпров. нац. ун-т ім. Олеся Гончара. Дніпро : Ліра, 2020. 173 с.

7. Глибовець М. М. Моделі обчислень у програмній інженерії : [навч. посіб.] / Глибовець М. М., Кирієнко О. В., Проценко В. С. ; Нац. ун-т «Києво-Могилян. акад.». Київ : Києво-Могилянська академія, 2019. 209 с.

8. Зелінська О.В., Потапова Н.А., Волонтир Л.О. Інформаційні системи та технології в галузі. Навчальний посібник. Вінниця: ВНАУ. 2020. 263 с.

9. Крячок О.С. Основи програмування : навч. посіб. : у 2 ч. / О.С. Крячок, Л. Г. Полягушко ; Нац. техн. ун-т України «Київ. політехн. ін-т ім. Ігоря Сікорського». Київ : КПІ ім. Ігоря Сікорського : Політехніка, 2019 . Ч. 1 : Комп'ютерний практикум. 2019. 51 с.

10. Кузніченко С.Д. Основи алгоритмізації та програмування : навч. посіб. / С. Д. Кузніченко, Л. Б. Коваленко ; Одес. держ. екол. ун-т. Одеса : ТЕС, 2019. 338 с.

11. Куліков С.І. Основи програмування та обчислювальної математики : навч. посіб. / Куліков С. І., Волкова С. А., Чернишов А. А. ; ДВНЗ «Укр. держ. хім.-технол. ун-т». Дніпро : ДВНЗ УДХТУ, 2018. 159 с.

12. Лахно В.А. Комп'ютерна логіка : [навч. посіб.] / Лахно В. А., Гусєв Б. С., Касаткін Д. Ю. ; Нац. ун-т біоресурсів і природокористування України. Київ : Компринт, 2018. 417 с.

13. Лінійне програмування : навч.-метод. посіб. : / [О. О. Ємець та ін.] ; Харків. нац. авт.-дорож. ун-т. Харків : ХНАДУ, 2020. 105 с.

14. Методи математичного моделювання та ідентифікації складних процесів і систем на основі високопродуктивних обчислень (нейрота нанопористі кіберфізичні системи із зворотніми зв'язками моделі з даними розрідженої структури, паралельні обчислення) : [монографія] / Хіміч О. М. [та ін.] ; НАН України, Ін-т кібернетики ім. В. М. Глушкова. Київ : Вид-во НАН України, 2019. 175 с.

15. Мироненко О. А., Шрамко І. І. Застосування сучасних програмних засобів в управлінні підприємствами аграрного сектору. Socio-economic aspects of economics and management. Taunton : Aspekt Publishing, 2015. Т. 1. С. 78-82.

16. Нужна С. А. Математичні аспекти моделювання та планування діяльності агропромислових підприємств в умовах невизначеності. Вісник Дніпропетровського державного аграрно-економічного університету. 2016. № 3(41). С. 128-133.

17. Нужна С.А., Смарець Н.М. Оптимізація використання виробничих ресурсів підприємствами аграрного сектору. Економічний аналіз. 2018. Том 28. №4. С.225-234.

18. Основи програмної інженерії : навч. посіб. / В. М. Юрчишин [та ін.] ; Івано-Франків. нац. техн. ун-т нафти і газу. Івано-Франківськ : ІФНТУНГ, 2021. 180 с.

19. Офіційний сайт Державної служби статистики України [Електронний ресурс]. Режим доступу: http://www.ukrstat.gov.ua/. 5

20. Павлик В. П. Використання моделювання в управлінні сільськогосподарськими підприємствами. Економіка АПК. 2018. № 4. С. 70-78.

21. Рольщиков В.Б. Застосування засобів інтерфейсу передачі повідомлень при програмуванні розподілених систем мовою Java : навч. посіб. / В. Б. Рольщиков ; Одес. держ. екол. ун-т. Одеса : ТЕС, 2018. 208 с.

22. Семеренко В.П. Технології паралельних обчислень : навч. посіб. / В. П. Семеренко ; Вінниц. нац. техн. ун-т. Вінниця : ВНТУ, 2018. 103 с.

23. Сорокатий Р.В. Основи об'єктивно-орієнтованого програмування : навч. посіб. / Р. В. Сорокатий, О. А. Пасічник, Т. К. Скрипник. Хмельницький : ХНУ, 2019. 175 с.

24. Тверитникова О.Є. Базові алгоритми та основи програмування. Теорія і практика : навч. посіб. / О. Є. Тверитникова, В. А. Крилова, О.Г. Васильченков ; Нац. техн. ун-т «Харків. політехн. ін-т». Харків : НТУ «ХПІ», 2020. 263 с.

Додаток А

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ВІННИЦЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет економіки та підприємництва

Кафедра комп'ютерних наук та економічної кібернетики

ЗВІТ

про виконання завдань навчальної практики

«**ВСТУП ДО СПЕЦІАЛЬНОСТІ: ОСНОВИ ПРОГРАМУВАННЯ**»

Виконав: студент (ка) групи
П.І.Б. студента

Перевірив: П.І.Б. керівника практики

Вінниця 2021

НАВЧАЛЬНО - МЕТОДИЧНЕ ВИДАННЯ

Волонтир Людмила Олексіївна
Денисюк Валерій Олександрович
Юрчук Наталія Петрівна

ПРОГРАМА ТА МЕТОДИЧНІ ВКАЗІВКИ ДЛЯ ПРОХОДЖЕННЯ НАВЧАЛЬНОЇ ПРАКТИКИ

Набір і редагування авторські

Технічний редактор Юрчук Наталія Петрівна
Верстка Юрчук Наталія Петрівна

Підписано до друку Формат 60х84/16. Папір офсетний. Друк різографічний. Тираж 50 прим.

Віддруковано у редакційно-видавничому відділі Вінницького державного аграрного університету 21008, м. Вінниця, вул. Сонячна, 3