**Волинський національний університет імені Лесі Українки**
**Факультет інформаційних технологій і математики**
**Кафедра загальної математики та методики навчання інформатики**

Вікторія ПАСТЕРНАК, Світлана ЯЦЮК

# МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ З ПРОГРАМУВАННЯ

для студентів спеціальності 014 Середня освіта (Інформатика)
першого (бакалаврського) рівня

Луцьк 2022

УДК 004.42(072)
П 19

Рекомендовано до друку науково-методичною радою Волинського національного університету імені Лесі Українки (протокол № 4 від 19 грудня 2022 року)

**Рецензенти:**
Н.М. Ліщина, кандидат технічних наук, доцент, завідувач кафедри інженерії програмного забезпечення Луцького національного технічного університету;

Я.М. Пастернак, доктор фізико-математичних наук, професор кафедри комп'ютерних наук та кібербезпеки Волинського національного університету імені Лесі Українки.

**Укладачі:** В.В. Пастернак, С.М. Яцюк
П 19 Методичні вказівки до виконання курсової роботи з програмування: методичні вказівки / Вікторія Валентинівна Пастернак, Світлана Миколаївна Яцюк. Луцьк: ВНУ ім. Лесі Українки, 2022. 71 с.

Анотація: Методичні вказівки містять загальні положення щодо мети, змісту та організації виконання курсової роботи з програмування, детальний опис всіх структурних елементів роботи та вимоги до оформлення. Подається порядок і процедура допуску до захисту та самого захисту й критерії оцінювання. Представлені тематики та приклад виконання курсової роботи з програмування. А також, у додатках наведено зразки документів, які використовуються при підготовці (написанні) курсової роботи з програмування для здобувачів першого (бакалаврського) рівня вищої освіти за спеціальністю 014 Середня освіта (Інформатика).

УДК 004.42(072)
© Пастернак В.В., © Яцюк С.М., 2022
© Волинський національний
університет імені Лесі Українки, 2022

**ЗМІСТ**

## ВСТУП

Згідно з Положенням про організацію навчального процесу у вищих навчальних закладах України, курсова робота виконується з метою закріплення, поглиблення і узагальнення знань, одержаних студентами за час навчання та їх застосування до комплексного вирішення конкретного фахового завдання. Написання та захист курсової роботи є важливим підготовчим етапом для реалізації наступного, складнішого завдання – виконання бакалаврської роботи. Курсова робота з програмування є обов'язковим компонентом освітньо-професійної програми «Середня освіта (Інформатика)» для здобуття освітнього рівня бакалавр за спеціальністю 014 Середня освіта (Інформатика). Підготовка закладами вищої освіти фахівців спеціальності 014 Середня освіта (Інформатика), професійна кваліфікація: вчитель інформатики закладу загальної середньої освіти для базової школи передбачає обов'язкову практичну підготовку, що частково реалізується під час написання курсових робіт. Порядок організації, написання та оформлення курсових робіт регламентується Положенням про організацію навчального процесу на першому (бакалаврському) та другому (магістерському) рівнях у Волинському національному університеті імені Лесі Українки [2].

Курсова робота повинна представляти закінчену розробку прикладної фахової проблеми, а саме:

- бути актуальною, мати наукову новизну, виконуватися на рівні сучасних досягнень науки і техніки;
- мати конкретне спрямування на вирішення практичних завдань майбутньої професійної діяльності;
- стимулювати у студентів творчий пошук нових пріоритетних проблемних рішень;
- вимагати опрацювання спеціальної наукової і методичної літератури;
- передбачати вибір оптимальних рішень на основі застосування математичних методів та сучасних методів моделювання з використанням різноманітних засобів обчислювальної техніки.

Метою виконання курсової роботи є вивчення основних принципів програмування та формування навичок застосування цих знань під час розв'язання конкретних практичних задач в предметній області спеціальності середня освіта (інформатика). Дисципліна повинна зосередити студента на найбільш важливих рисах програмування, які не залежать від таких обставин, як тип процесора або операційна система.

Крім того, під час захисту курсової роботи студент повинен проявити такі властивості як впевненість у власних знаннях, вміння відстоювати власну думку, вміння виступати перед аудиторією.

Завдання курсової роботи з програмування:

1) поглиблене вивчення принципів структурного програмування, сучасних

процедурно-орієнтованих та об’єктно-орієнтованих мов, основних структур даних і здатність їх застосовувати під час програмної реалізації алгоритмів професійних завдань;

2) отримання практичних навиків розробки програм із застосуванням різних парадигм програмування: узагальненого, об’єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами та алгоритмами обчислень, структурами даних і механізмами управління;

3) отримання практичних навиків щодо вирішення різноманітних задач з програмування процесів та об’єктів;

4) отримання практичних навиків щодо обґрунтування вибору середовища розробки;

5) реалізація у вигляді програми одного чи кількох взаємопов’язаних алгоритмів, що вирішують поставлену прикладну задачу;

6) застосування основних нормативних документів, необхідних для проєктування, розробки та оформлення програмних продуктів;

7) тестування розробленого програмного забезпечення.

Працюючи над курсовою роботою з програмування, студент повинен вивчити окремі фази розробки програмного забезпечення і навчитися поєднувати їх в одне ціле – у свій проект. І хоч неможливо очікувати, що за той обмежений час, який є у студента, він створить справжній програмний продукт, викладач повинен максимально наблизити його роботу до реальних умов та достовірних результатів.

## 1. Підготовчий етап виконання курсової роботи та вимоги до курсових робіт

У відповідності до основної мети університетської освіти – підготовки педагогічних і наукових кадрів для самостійної творчої роботи – навчальним планом передбачено виконання кількох курсових робіт (проектів). Курсова робота – форма творчої самостійної роботи студентів. Головна мета курсової роботи полягає в тому, щоб розвинути у студентів навики і здібності до творчого наукового пошуку та сприяти отриманню ними досвіду публічного оголошення результатів власної роботи. Курсову роботу студент виконує, як правило, на четвертому курсі.

Мета курсової роботи полягає в систематизації, закріпленні та розширенні теоретичних знань і практичних навичок, ознайомленні та відпрацюванні методів і методик вже здобутих у певній галузі знань. Завдання до курсових робіт повинні мати елементи пошуковості, що проявляються у виборі методів, методик і способів розрахунків і вимірювань, прийнятті самостійних проектних рішень. Наявність у роботі творчого елементу (самостійний підбір і огляд студентом джерел літератури, визначення кардинальних моментів роботи, вибір оригінальних шляхів розв'язування задач, нестандартна трактовка отриманих результатів тощо) повинна оцінюватися найвищим балом.

Курсова робота є своєрідним визначенням професійного потенціалу студента.

Курсова робота може бути:

1) реферативною (огляд літературних джерел за визначеною темою);
2) теоретичною (розрахунковою);
3) експериментальною.

До курсової роботи висувають наступні основні вимоги:

- тема роботи повинна бути тісно пов'язана з програмою даної дисципліни;
- завдання до роботи має складатися з кількох невеликих за обсягом задач, які студент може розв'язати самостійно на базі знань, отриманих при вивченні даної дисципліни; повинна мати елементи пошуковості;
- робота повинна передбачати систематизацію, закріплення й розширення теоретичних і практичних знань із даної дисципліни й використання знань, набутих при вивченні попередніх дисциплін;
- оформлення роботи має відповідати вимогам державних стандартів.

### 1.1. Організація взаємодії керівника та студента

Керівником курсової роботи з програмування призначається науково-педагогічний працівник кафедри, за яким закріплено викладання навчальної дисципліни «Програмування», або який викладає дисципліни, пов'язані з програмуванням. Студенту надається можливість самостійно обирати керівника із переліку науково-педагогічних працівників, запропонованих кафедрою. При

здійсненні вибору студенту слід ознайомитись із науковими інтересами керівника, а також переліком його наукових публікацій. Робота над виконанням курсової роботи з програмування спрямована на оволодіння навиками самостійно ставити та розв'язувати завдання, самостійно працювати з літературою, обирати та вивчати потрібний інструментарій для реалізації поставленої задачі. Курсову роботу студент повинен виконувати самостійно, консультуючись з керівником, головне завдання якого – допомогти у пошуку шляхів та методів вирішення проблеми.

До основних обов'язків керівника курсової роботи відносять:

1) допомога студенту у виборі та формулюванні актуальної теми курсової роботи з врахуванням його наукових та практичних інтересів;

2) допомога студенту при формуванні плану виконання роботи та визначення термінів виконання;

3) надання консультації з приводу формування структури роботи та її змістового наповнення;

4) надання консультацій щодо збору та обробки інформаційних джерел за темою дослідження;

5) надання консультацій щодо функціоналу розроблюваного програмного продукту та підбору оптимальних алгоритмів для розв'язку поставленої задачі;

6) контроль за виконанням кожного з етапів роботи, згідно плану виконання;

7) контроль за дотримання студентом академічної доброчесності;

8) контроль за підготовкою студента до захисту курсової роботи;

9) оцінювання якості та можливості допуску курсової роботи до захисту.

До основних обов'язків студента при написанні курсової роботи з програмування відносяться наступні основні обов'язки:

1) вчасно звернутися до керівника курсової роботи для надання консультації з приводу вибору та формулювання теми курсової роботи;

2) складання плану (виконання етапів) написання курсової роботи згідно зазначених термінів;

3) вчасно виконувати всі поставлені завдання (етапи) згідно плану написання курсової роботи з програмування;

4) обов'язково дотримуватись академічної доброчесності при написанні курсової роботи;

5) на кожному етапі написання курсової роботи подавати результати роботи на розгляд керівнику і відповідно до його зауважень уточнювати, доповнювати та вдосконалювати і в разі потреби їх доопрацьовувати.

Слід відмітити, що курсова робота повинна виконуватись відповідно до затвердженого календарного плану. Важливо зазначити і те, що на кафедрі складається графік консультацій наукових керівників, в якому вказується час і місце їх проведення. А також, консультації з керівником повинні проводитися не менше, як 1 раз на тиждень.

## 1.2. Вибір теми курсової роботи та її актуальність

Курсова робота повинна містити **теоретичну** та **практичну** частини. Теоретична частина повинна містити огляд сучасних технологій, які використовуються при виготовленні аналогічних програмних продуктів. Практична частина повинна відтворювати готовий програмний продукт. Іншими словами студент повинен виконати всі етапи створення програмного продукту (під ОС Windows) з можливістю його безпосереднього використання. Виходячи з цього, студенту пропонуються (на вибір) наступні тематики курсової роботи з програмування, наприклад:

1) Розробка мультимедійних ресурсів навчального призначення з використанням HTML, CSS, JavaScript;
2) Розробка веб-системи лекцій з курсу «Програмування»;
3) Розробка програмного засобу доповненої реальності навчального призначення;
4) Розробка веб-системи тестів з курсу «Програмування»;
5) Створення електронних освітніх ресурсів навчального призначення засобами об'єктно-орієнтованого програмування;
6) Створення програми науковий калькулятор мовою C++ з використанням середовища Visual Studio.
7) Дослідження технологій створення 3D об'єктів для віртуальних лабораторних робіт з інформатики;
8) Підготовка відеоматеріалів навчального призначення;
9) Створення віртуального 3D туру навчальних лабораторій кафедри та його інтеграція на сайті;
10) Технології побудови Інтернет-магазинів;
11) Веб-дизайн шкільного сайту;
12) Проектування сайту комп'ютерної тематики;
13) Розробка електронного підручника засобами C#;
14) Створення довідника з інформатики засобами об'єктно-орієнтованого програмування;
15) Учнівський органайзер засобами ООП (об'єктно-орієнтованого програмування);
16) Журнал учителя засобами C#;
17) Розробка ППЗ (прикладного програмного забезпечення) для мобільних платформ;
18) ППЗ (прикладне програмне забезпечення) «Учительська»;
19) Розробка програм-тренажерів з інформатики;
20) Розробка ігрової програми для закріплення навичок програмування;
21) Розробка ППЗ (прикладного програмного забезпечення) для вивчення будови комп'ютера;
22) Розробка веб-додатків мовою програмування JavaScript;
23) Розробка серверних веб-додатків мовою PHP.

Слід зазначити, що тематика курсової роботи повинна відповідати професійним завданням, зафіксованим в освітньо-кваліфікаційній характеристиці відповідно до напряму підготовки (освітньої програми). Тематика курсової роботи з програмування повинна бути актуальною, відповідати завданням і сучасним тенденціям та перспективам розвитку сьогодення. Назва курсової роботи повинна бути короткою та відповідати меті дослідження. Курсова робота із програмування повинна бути спрямована на розв'язання однієї або кількох споріднених задач (прикладного або наукового характеру) і обов'язково включати в себе програмну реалізацію розв'язання поставленої задачі. Задача повинна бути повністю розв'язаною та завершеною. Основним результатом курсової роботи з програмування повинен бути програмний продукт.

Перелік тематик курсової роботи з програмування формується випусковою кафедрою та оновлюється кожного навчального року. Студенти мають право запропонувати свою тему з обґрунтуванням доцільності її написання або самостійно вибрати із переліку запропонованих. Студенту, що не обрав тему або керівника курсової роботи у встановлені терміни, на засіданні кафедри призначається керівник та тема роботи. Тематика курсових робіт пропонується студентам на початку семестру та затверджується рішенням кафедри і оприлюднюється не пізніше, як за 3 місяці до планового терміну її захисту. Уточнення у формулюванні затвердженої теми може бути внесене лише за згодою наукового керівника і затверджене на засіданні кафедри, але не пізніше, як за місяць до планового захисту. Довільна зміна студентом теми своєї роботи не допускається. Не допускається виконання курсових робіт на однакову тему різними студентами. Допускається робота кількох студентів над одним проєктом, де кожен учасник виконує свою частину проєкту.

Основними критеріями вибору теми курсової роботи з програмування є:

- актуальність;
- елементи новизни;
- перспективність (затребуваність) обраної теми.

## 1.3. План роботи та терміни виконання курсової роботи

Після визначення теми курсової роботи, здобувач вищої освіти повинен отримати перше завдання на консультації у свого керівника курсової роботи. Під час консультації визначаються загальні вимоги до роботи, порядок її виконання, план написання та терміни виконання основних етапів роботи, літературні джерела, які підлягають вивченню, зміст та методика проведення конкретного дослідження. У таблиці 1 подано орієнтовний календарний план написання курсової роботи.

**Табл. 1**

**Календарний план курсової роботи**

| № з/п | Назва основних етапів написання курсової роботи | Терміни виконання етапів роботи |
|---|---|---|
| 1. | Вибір теми курсової роботи. Подання заяви про обрання теми. Затвердження теми курсової роботи. | 3-4 тиждень семестру |
| 2. | Затвердження графіку виконання роботи. | 4 тиждень семестру |
| 3. | Аналіз літературних джерел. Формування основної літератури до обраної теми. | 4-5 тиждень семестру |
| 4. | Формування попереднього плану курсової роботи. | 5 тиждень семестру |
| 5. | Опис теоретичних аспектів дослідження. Написання першого розділу курсової роботи. | 5-7 тиждень семестру |
| 6. | Проєктування і розробка програмного засобу. | 8-12 тиждень семестру |
| 7. | Тестування і налагодження розробленого програмного продукту. | 13 тиждень семестру |
| 8. | Загальний опис технології розробки продукту. Написання другого розділу курсової роботи. | 3-14 тиждень семестру |
| 9. | Оформлення курсової роботи згідно вимог. | 3-14 тиждень семестру |
| 10. | Подача курсової роботи керівнику. Підготовка презентації. Допуск до захисту. | 15 тиждень семестру |
| 10 | Захист перед комісією. | 16-17 тиждень семестру |

## 2. Об’єм, структура та правила оформлення курсової роботи

Рекомендований обсяг курсової роботи – 25 – 30 сторінок. Обсяг може відхилятися в межах не більше 10%. Додатки до курсової роботи до загальної кількості сторінок *не додаються*.

Рекомендована наступна *структура* курсової роботи:

- ТИТУЛЬНИЙ АРКУШ;
- ЗМІСТ;
- СПИСОК УМОВНИХ ПОЗНАЧЕНЬ (за потреби);
- ВСТУП;
- РОЗДІЛ 1. Теоретична частина;
- РОЗДІЛ 2. Практична частина:

  1) блок-схема алгоритму програмного продукту. Специфікація

класів;

2) макет інтерфейсу програмного продукту. Реалізація класів;
3) створення програмного продукту;
4) тестування програмного продукту;

- ВИСНОВКИ;
- СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ;
- ДОДАТКИ.

На ***титульному аркуші*** повинно бути відображено основні дані, що визначають місце виконання роботи, її назву і тип роботи (курсова робота), керівника і виконавця роботи (див. додаток 1). Слід відмітити, що титульний аркуш оформляється згідно зразку, який представлений у додатку 1.

***Перелік умовних позначень.*** Перелік умовних позначень є не обов'язковою структурною частиною роботи. Даний перелік складається, якщо в роботі зустрічаються маловідомі скорочення, специфічні терміни або абревіатури. Перелік друкується двома колонками. В першій колонці подається термін, в другій його детальне пояснення. Якщо в роботі зустрічається один і той же термін не більше 3 разів, то його тлумачення не подається в переліку умовних позначень, а розшифровується в самому тексті роботи при першому його згадуванні.

У ***вступі*** коротко описують актуальність теми, мету роботи, завдання, об'єкт дослідження, предмет дослідження, апробація результатів роботи (у випадку наявності), публікації (у випадку наявності). Слід відмітити, що об'єкт дослідження необхідно викладати у стислій формі (а в разі необхідності його обґрунтовувати). Тобто висвітлити основні етапи роботи, необхідність виконання тих чи інших розрахунків, вимірювань тощо. Обсяг вступу – 2-3 сторінки.

***Актуальність теми.*** Шляхом критичного аналізу та порівняння з відомими розв'язаннями проблеми (наукової задачі) обґрунтовують актуальність теми та її доцільність для розвитку спеціальності 014 Середня освіта (Інформатика). Висвітлення актуальності не повинно бути багатослівним. Досить кількома реченнями висловити головне – сутність проблеми або наукового завдання. Якщо робота виконується у межах досліджень кафедри (відділу, організації тощо), коротко визначають зв'язок вибраного напряму з планами організації, галузевими, державними планами та програмами. Обов'язково потрібно зазначити особистий вклад автора у виконанні цих науково-дослідних завдань.

***Мета і завдання дослідження.*** Мета роботи – це кінцевий результат, якого прагне досягнути студент (автор) роботи у процесі власного дослідження. Формулювання мети курсової роботи повинне бути співзвучне з темою курсової роботи. Завдання – це більш конкретніші шляхи, засоби для досягнення поставленої мети. У свою чергу, перелік завдань повинен бути співзвучний зі змістом роботи.

***Об'єкт дослідження*** – це процес або явище, що породжує проблемну ситуацію.

***Предмет дослідження*** знаходиться в межах об'єкта і становить частину від цілого (тобто об'єкта). Об'єкт і предмет співвідносяться як загальне і часткове. В об'єкті виділяють ту його частину, яка й стане предметом дослідження. Саме на предмет спрямована увага студента (автора), оскільки він має визначати назву роботи.

***Апробація результатів роботи.*** Якщо студент брав участь у наукових конференціях, семінарах, засіданнях наукового гуртка з оголошенням результатів своєї роботи та має цьому підтвердження (опубліковані тези, статті або програму конференції за тематикою роботи), то слід зазначити їх назву, рік та місце проведення.

***Публікації.*** Якщо студент (автор) має публікації, то потрібно подати їх у загальний перелік літературних джерел.

***Основна частина.*** Текстова частина курсової роботи складається з двох розділів (теоретична частина та практична частина). Розділи повинні мати бібліографічні посилання на джерела, що дозволяє аналізувати власні дані та порівнювати їх з даними інших джерел.

У першому розділі **(теоретична частина)**, як правило, описують теоретичні дослідження з теми курсової роботи, написані з використанням першоджерел. Подається огляд існуючих підходів до розв'язання поставленої задачі, аналіз існуючих алгоритмів розв'язання, їх аналіз або порівняльна характеристика. Велике значення при написанні першого розділу має правильне трактування понять теми, їх точність і науковість. Використані терміни мають бути загальновживаними чи подаватися з посиланням на їх автора. Останнім підрозділом першого розділу повинен бути «Огляд та аналіз аналогічних програмних розробок». В даному пункті потрібно проаналізувати існуючі програмні продукти аналогічного призначення (орієнтовна кількість 2-3), вказавши їх переваги та недоліки та можливість використання аналогів для вирішення поставленої задачі.

При проведенні порівняльного аналізу аналогічних програмних розробок, зокрема калькулятор, студент для кожного аналогу програмного забезпечення повинен визначати наступні основні характеристики:

- назва;
- розробник (дистриб'ютор);
- архітектура (desktop application, client-server, web application, mobile application);
- мова реалізації;
- перелік основних функцій, основні характеристики;
- аналіз переваг та недоліків даного ПЗ;
- джерело інформації або веб-сайт.

Обов'язковим є використання скріншотів (рисунків) проаналізованих програмних продуктів (1-2 для кожного програмного засобу).

На основі результатів аналізу, поданого в даному підрозділі, будуть

сформовані вимоги до програмного продукту, який реалізує мету та завдання курсової роботи.

Назва першого розділу формується згідно текстового наповнення першого розділу і повинна відповідати темі курсової роботи.

Другий розділ **(практична частина)** містить опис процесу розробки програмного продукту. Назва другого розділу повинна бути співзвучна з назвою курсової роботи. Другий розділ обов'язково повинен мати наступні ОСНОВНІ підрозділи:

1) блок-схема алгоритму програмного продукту. Специфікація класів;
2) макет інтерфейсу програмного продукту. Реалізація класів;
3) створення програмного продукту;
4) тестування програмного продукту;

Дані підрозділи можуть бути переформульовані згідно поставленої теми курсової роботи, розширені і доповнені додатковими підпунктами, але при цьому структурний зміст порядку оформлення не повинен бути порушеним.

Слід зазначити, що склад ***теоретичної та практичної частини*** випливає безпосередньо із завдання до курсової роботи.

У ***висновках*** необхідно висвітлити основні результати роботи і рекомендації щодо їх застосування. Важливо, щоб сформульовані висновки відповідали поставленим завданням, які фігурують у ***Вступі***. Необхідно зазначити не тільки позитивні результати, яких вдалося досягнути в результаті виконання дослідження, але й недоліки та проблеми, а також конкретні шляхи їх усунення.

***Список літератури*** оформляють за алфавітним порядком. Кількість використаних літературних джерел повинна бути не меншою, ніж 25 пунктів. Бібліографічний опис джерел та літератури складають відповідно до чинних стандартів з бібліотечної або видавничої справи (Інформація та документація. Бібліографічне посилання: Загальні положення та правила складання згідно ДСТУ 8302:2015):

https://kubg.edu.ua/images/stories/podii/2017/06_21_posylannia/dstu_8302.pdf

Зразок оформлення літератури представлений у додатку 4.

У ***додатках*** розміщують тексти програм, схеми, викладки допоміжних матеріалів, додаткові таблиці, регламенти технологічних процесів тощо. На кожен додаток повинно бути розміщене посилання в тексті.

**Оформлення:**

При оформленні курсової роботи необхідно використовувати набір через півтора інтервали (1,5 інтервали) шрифтом Times New Roman, розмір шрифту – 14. Абзацний відступ – 1,25 мм. Інтервал перед і після абзацу – 0 пунктів. Виділення тексту напівжирним, курсивом, підкресленим або іншим шрифтом, окрім Times New Roman, розмір 14, не допускається, окрім випадків, зазначених нижче (заголовки, окремі слова вступу).

Роботу оформлюють на стандартних аркушах білого паперу формату А4 (210х297), допуск на формат – 10 мм. Розміри полів: зверху – 20 мм, знизу – 20 мм, зліва – 30 мм, справа – 10 мм. Вимоги до оформлення курсової роботи не

можуть мати узагальнюючого характеру й визначаються профілем дисципліни, з якої вона виконується. Для нумерації сторінок використовують колонтитули. Нумерація сторінок проставляється у верхньому правому куті сторінки арабськими цифрами. Крапка після цифри не ставиться. Нумерація починається з титульного аркуша, але номер сторінки на ньому не проставляється. Розмір цифр такий самий, як і розмір шрифту тексту роботи.

Переплітають роботу (курсова робота) по лівому полю за допомогою м'якої палітурки.

При оформленні не плутати «дефіс» клавіша «Minus» і «тире» [Ctrl+NumMinus].

Використовувати нерозривний пробіл [Ctrl+Shift+пробіл] та нерозривний дефіс [Ctrl+Shift+дефіс].

**Формування змісту.** Зміст створюється виключно засобами автоматичного генерування змісту текстового процесора і повинен містити посилання на структурні елементи курсової роботи, включаючи номери сторінок.

У змісті заголовки розділів подаються великими буквами, а підрозділів, пунктів – маленькими з першої великої; у змісті не виділяти заголовки напівжирним.

Заголовки структурних частин «ЗМІСТ», «ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ», «ВСТУП», «РОЗДІЛ», «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТКИ», (заголовки першого рівня) у змісті відображати без відступу від лівого краю сторінки, заголовки другого рівня з відступом 0,75 см від лівого краю сторінки, заголовки третього рівня – з відступом 1,5 см від лівого краю сторінки. Зразок оформлення змісту представлений у додатку 2.

**Оформлення розділів (підрозділів).** Текст основної частини поділяють на розділи, підрозділи, пункти і підпункти. Заголовки структурних частин: «ЗМІСТ», «ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ», «ВСТУП», «РОЗДІЛ», «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТКИ» друкують великими літерами симетрично до тексту. Кожну структурну частину роботи починають з нової сторінки, крім назв підрозділів і пунктів у межах розділу. Після заголовків першого рівня: «ЗМІСТ», «ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ», «ВСТУП», «РОЗДІЛ», «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», «ДОДАТКИ» пропускається один рядок (один абзац, шрифт Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0).

Назви розділів друкують великими літерами симетрично до тексту. Розділи нумерують арабськими цифрами. Після номеру розділу крапку не ставлять. З нового рядка (натискаючи не [Enter], а [Shift+Enter]) друкують назву розділу великими літерами симетрично до тексту без крапки в кінці.

Відстань між заголовком і подальшим чи попереднім текстом має бути один рядок: шрифт Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5.

Відстань між основами рядків заголовку, а також між двома заголовками

приймають такою, як у тексті.

Заголовки підрозділів (заголовки другого рівня) друкують маленькими літерами (крім першої великої) з абзацу, абзацний відступ – 1,25 см (текст заголовку повинен відступати від номера заголовку на 0,5 см) шрифтом Times New Roman, розмір шрифту – 14, напівжирний, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0. Крапку в кінці заголовка не ставлять.

Заголовки пунктів (заголовки третього рівня) друкують маленькими літерами (крім першої великої) з абзацу, абзацний відступ – 1,75 см, шрифтом Times New Roman, розмір шрифту – 14, напівжирний, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0. Крапку в кінці заголовка не ставлять.

Перенесення слів у заголовках не допускається. Якщо заголовок складається з кількох рядків, другий і наступні рядки заголовку другого рівня друкують з відступом у 1,25 см від лівого краю сторінки; другий і наступні рядки заголовку третього рівня друкують з відступом у 1,75 см від лівого краю сторінки. Якщо заголовок складається з кількох речень, їх розділяють крапкою.

Назви підрозділів і пунктів друкуються в межах розділу, а не починаються з нової сторінки. Не допускається розташування назв підрозділів, пунктів у нижній частині сторінки, якщо після них міститься лише один рядок тексту. В такому випадку такі окремі заголовки примусово переносяться на наступну сторінку.

При нумерації заголовків другого та наступного рівнів відступ від номера заголовку до назви заголовку повинен становити 0,5 см.

У вступі лише слова «Актуальність теми», «Мета роботи», «Завдання», «Об’єкт дослідження», «Предмет дослідження», «Апробація результатів роботи», «Публікації» виділяють напівжирним шрифтом.

Слово «ВИСНОВКИ», а не «ВИСНОВОК» друкують великими літерами симетрично до тексту шрифтом Times New Roman, розмір шрифту – 14, напівжирний. Після слова «ВИСНОВКИ» пропускається один рядок (один абзац, шрифт Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0).

Слова «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ», друкують великими літерами симетрично до тексту шрифтом Times New Roman, розмір шрифту – 14, напівжирний. Після слів «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ» пропускається один рядок (один абзац, шрифт Times New Roman, розмір шрифта – 14, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0).

Додатки слід позначати послідовно цифрами, наприклад: 1, 2, 3, 4, 5, 6, 7, 8 і т.д. Тобто: Додаток 1, Додаток 2, Додаток 3 т.д.

Додаток повинен мати заголовок, надрукований вгорі малими літерами з першої великої симетрично відносно тексту сторінки. З правого боку рядка над заголовком малими літерами з першої великої повинно бути надруковано слово «Додаток 1» (наприклад).

Після назви додатку пропускається один рядок (один абзац, шрифт Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5, інтервал перед і після абзацу – 0).

**Вимоги до оформлення рисунків, таблиць, формул.** У роботі обов'язкова наявність ілюстративного матеріалу: рисунки, графіки, схеми, діаграми.

**Рисунки** слід розташовувати безпосередньо після тексту, у якому вони згадуються вперше, або на наступній сторінці. На всі рисунки повинні бути посилання у тексті. Якщо рисунки створені не автором роботи, необхідно зробити певні посилання, дотримуючись вимог чинного законодавства щодо авторських прав.

Креслення, рисунки, графіки, схеми, діаграми, які містяться у тексті, повинні відповідати вимогам державних стандартів.

Слід також відмітити, що рисунки повинні мати назву, яку розміщують під зображенням. За необхідності під рисунками розміщують пояснювальні дані. Рисунки позначають словом «Рис.», яке разом із назвою представленої ілюстрації розташовують після пояснювальних даних і вирівнюють по центру сторінки. Наприклад: «Рис. 1.2 – Приклад вікна введення та редагування коду Unit 1… .

Приклад рисунка:

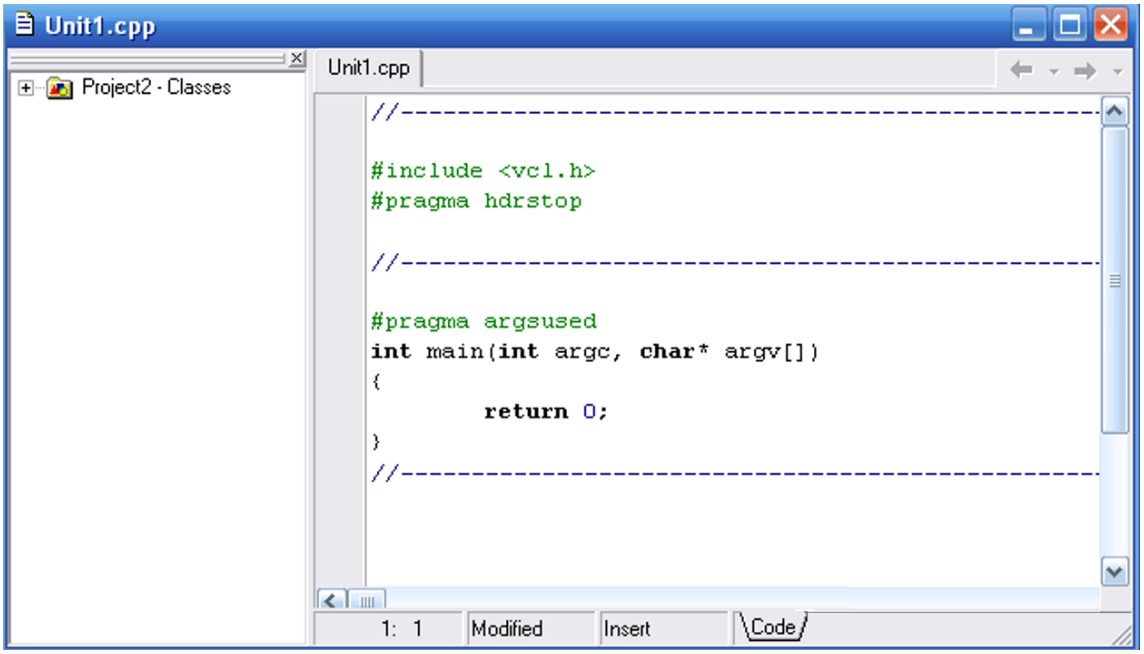



Рисунок 1.2 – Приклад вікна введення та редагування коду Unit 1

Слід відмітити, що рисунки слід нумерувати арабськими цифрами порядковою нумерацією в межах розділу, за винятком ілюстрацій, наведених у додатках. Номер рисунка складається з номера розділу і порядкового номера рисунка, відокремлених крапкою. Наприклад: «Рисунок 3.2» – це другий рисунок третього розділу.

Відстань між текстом і рисунком, рисунком та наступним текстом має бути один рядок: шрифт Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5. Відступ між рисунком і назвою рисунка не роблять.

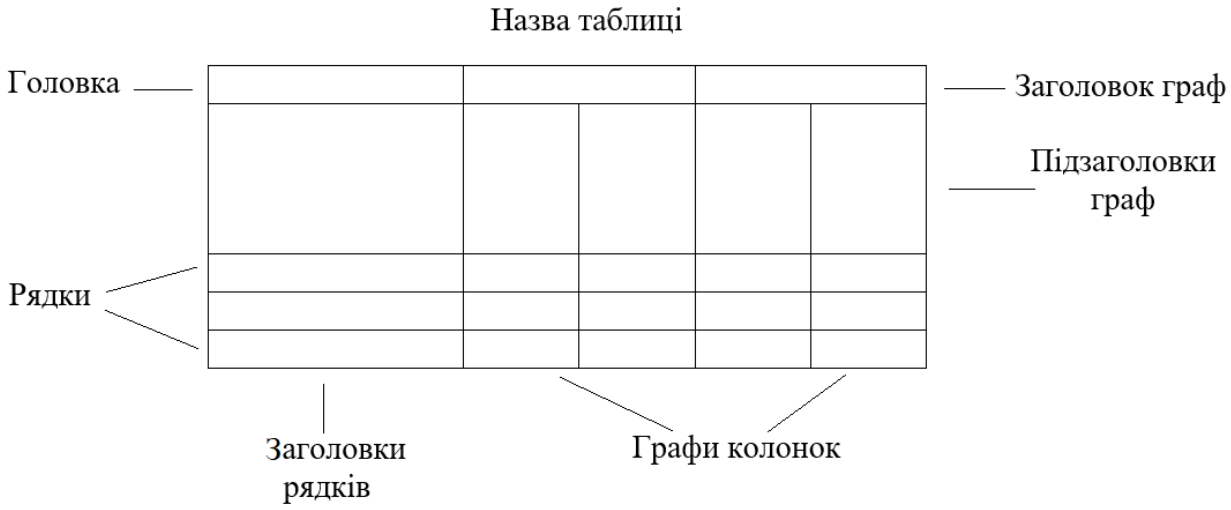
Якщо рисунок не поміщається на одній сторінці, можна переносити його на іншу сторінку, вміщуючи назву рисунка на першій сторінці, пояснювальні дані – на кожній сторінці, і під ними позначають: «Рисунок_, аркуш_».

**Таблиці.** Цифровий матеріал, як правило, оформляють у вигляді таблиць. Таблицю слід розташовувати після тексту, у якому вона згадується вперше, або на наступній сторінці. На всі таблиці повинні бути зроблені посилання [] у тексті.

Таблиці потрібно нумерувати арабськими цифрами порядковою нумерацією в межах розділу (за винятком додатків). Номер таблиці складається з номера розділу й порядкового номера таблиці, які відокремлюються крапкою. Крапку після номера таблиці не ставлять. Наприклад: перша таблиця другого розділу – «Таблиця 1.2».

Наприклад:

Таблиця 1.2




Варто зазначити, що слово «Таблиця» вписують один раз і розташовують праворуч над першою частиною таблиці. Якщо ж таблиця переходить на наступну сторінку, то над іншими частинами таблиці необхідно написати «Продовження таблиці 1.2» із зазначеним відповідним номером. Відстань між основним текстом і назвою таблиці має бути один рядок. Шрифт – Times New Roman, розмір шрифту – 14, міжрядковий інтервал – 1,5. Відступ між назвою таблиці і таблицею не роблять. Заголовки таблиці починають з великої літери і вирівнюють по центрі. У кінці заголовків таблиць крапки не ставлять.

**Формули.** При використанні формул необхідно дотримуватися певних орфографічних правил. Довгі та громіздкі формули, які мають у складі знаки суми, добутку, диференціювання, інтегрування, розміщують на окремих рядках. Це стосується також і всіх нумерованих формул.

Для коротких однотипних формул, які відокремлені від тексту, можна подати в одному рядку разом з основним текстом. Невеликі і нескладні формули, що не мають самостійного значення, вписують всередині рядків тексту.

Пояснення значень символів і числових коефіцієнтів треба подавати безпосередньо під формулою в тій послідовності, в якій вони дані у формулі. Перший рядок пояснення починають зі слова «де» без двокрапки і без абзацного відступу. Рівняння і формули треба виділяти до тексту і після тексту вільними рядками (тобто: вище і нижче кожної формули потрібно залишити один вільний рядок).

Якщо рівняння не поміщається в один рядок, його слід перенести після знаку рівності (=) або після знаків плюс (+), мінус (-), множення (х) і ділення (:). Нумерувати слід лише ті формули, на які є посилання в основному тексті.

Порядкові номери позначають арабськими цифрами в круглих дужках біля правого краю сторінки без крапок від формули до її номера. Номер формули, який не поміщається у рядку разом із формулою, переносять у наступний (нижче) рядок формули. Якщо формула знаходиться у рамці, то номер такої формули записують зовні рамки з правого боку навпроти основного рядка формули. Номер формули складається з номера розділу і порядкового номера формули, відокремлених крапкою, наприклад: (1.4) – де 1 – це номер розділу, а 4 – це четверта формула першого розділу. Крапку після номера формули не ставлять.

Наприклад:

$$y = \begin{cases} \prod_{i=0}^{n-1} \sum_{j=1}^{n} (x - i + j)^3, x < 0 \\ \sum_{i=1}^{n-1} \frac{x}{i}, x \geq 0 \end{cases} \qquad 1.4$$

**Додатки.** Додатки потрібно оформляти як продовження рукопису на подальших сторінках, розташовуючи їх відповідно у порядку посилань на них в основному тексті. Додатки повинні мати спільну з рукописом наскрізну нумерацію сторінок.

У додатках представляють матеріал, який не може бути послідовно розташований в основній частині тексту через великий обсяг, суто технічний характер або внаслідок неможливості способу відтворення (викладений на папері іншого формату). У додатках можуть бути представлені окремі рисунки (окремі фрагменти до них), таблиці, схеми тощо. А також, у додатках текстової частини курсової роботи з програмування, обов'язково, повинно бути: «Технічне завдання» і «Вказівки користувачу».

**Правила оформлення списку використаних джерел** подано у додатку 4. Список використаних джерел оформляється згідно вимог Національного стандарту України ДСТУ 8302:2015.

Загальні вимоги до оформлення списку використаних джерел:

1) список використаних джерел складається за алфавітним порядком;

2) кожне джерело включається у загальний список лише один раз, навіть якщо на нього присутні декілька посилань у тексті курсової роботи;

3) джерела описуються мовою оригіналу (не перекладаються);

4) список використаних джерел оформляється у вигляді нумерованого списку за наступними правилами: список з використанням арабських цифр, після номеру ставиться крапка, порядковий номер джерела друкують без відступу від лівого краю сторінки, відступ від номера до тексту 0,5 см, другий та наступний рядки друкують з відступом 0,75 см від лівого поля сторінки.

Джерела можна розміщувати одним із таких способів: у порядку появи посилань у тексті (найбільш зручний для користування і рекомендований при написанні наукових робіт), а також в алфавітному порядку прізвищ перших авторів або заголовків.

Рекомендована кількість використаних джерел – не менше 25.

## 2.1. Вимоги до оформлення презентаційних матеріалів курсової роботи

Презентаційні матеріали (презентація) є невід'ємною частиною процедури захисту курсової роботи з програмування, яка враховується при оцінюванні, та повинна бути правильно оформлена.

Основні цілі презентаційних матеріалів:

- показати здатність студента самостійно осмислити матеріали наукової роботи, систематизувати їх та доповісти (донести) у доступній для сприйняття формі слухачам;
- проявити ораторські та комунікаційні здібності студента під час викладення основних положень роботи, а також відповідей на запитання та участі у загальній дискусії;
- сформувати вміння студента працювати з електронними засобами створення, редагування та демонстрації презентацій, тобто комп'ютерною технікою та відповідним програмним забезпеченням, враховуючи специфіку дисципліни, в межах якої захищається робота.

Слід відмітити, що при оформленні презентаційних матеріалів необхідно дотримуватися певних вимог до змісту презентації, зокрема:

- відповідність змісту презентації поставленим цілям і завданням курсової роботи;
- дотримання правил орфографії, пунктуації, скорочень і правил оформлення тексту;
- відсутність орфографічних помилок, достовірність представленої інформації;
- лаконічність тексту на слайді (не більше 4 речень);
- завершеність (зміст кожної частини текстової інформації повинен бути логічно завершеним);
- розташування інформації на слайді (на слайди виноситься найбільш

важлива інформація; якщо на слайді є зображення (Рис.), то напис повинен розташовуватися під ним (рисунком);

- використання єдиного стилю оформлення;
- використання для фону слайдів комфортних тонів (рекомендується – білий);
- дозволяється виділяти, відтіняти, підкреслювати найбільш важливу інформацію;
- доцільність використання анімаційних ефектів;
- читабельність тексту на слайдах презентаційних матеріалів (висока якість зображення, таблиць, формул і т.д.);

Нижче наведено типову структуру для розробки презентаційних матеріалів, яка представляється на захисті курсової роботи із програмування.

Наприклад:

Слайд 1. Тема, виконавець, науковий керівник;

Слайд 2. Обґрунтування актуальності обраної теми (1-2 речення);

Слайд 3. Основна мета курсової роботи;

Слайд 4. Завдання курсової роботи;

Слайд 5. Об’єкт дослідження та предмет дослідження;

Слайд 6. Основні поняття (найбільш вживані), які використовуються у курсовій роботі;

Слайд 7. Основні алгоритми, які були проаналізовані, або основні методи розв’язування проблеми, які були розглянуті в першому розділі;

Слайд 8. Аналогічні програмні продукти або прототипи програмних продуктів, які були проаналізовані;

Слайд 9. Завдання, призначення та вимоги до програмних продуктів (які були представлені на попередньому слайді);

Слайд 10. Загальна структура курсової роботи;

Слайд 11. Інструментальні засоби розробки (логотипи);

Слайд 12. Особливості програмної реалізації. Переваги та недоліки;

Слайд 13-18. Основні режими роботи;

Слайд 17. Загальні висновки;

Слайд 18. Дякую за увагу!

Слід зазначити, що дана структура є шаблоном для формування презентаційних матеріалів і носить рекомендаційний характер. Під час підготовки презентаційних матеріалів (презентації) кількість слайдів може змінюватися, але рекомендовано дотримуватись наведеного вище порядку викладання матеріалів.

## 2.2. Вимоги до відеоролика курсової роботи

Одним із важливих елементів курсової роботи є створення відеоролика на курсову роботу, який демонструватиме роботу готового програмного продукту. А також, є підтвердженням готовності та результату курсової роботи студента.

Відеоролик курсової роботи повинен презентувати основні можливості програмної розробки та відповідати наступним основним вимогам:

- тривалість від 3 до 5 хвилин;
- відео високої якості, наприклад: 720p, не нижче 30fps (контейнер: mkv, кодек: H264), співвідношенням кадру 16:9;
- відео може бути знято у декілька етапів, допускається монтаж;
- основний звуковий супровід - лише оригінальний (або титри);
- титри, при наявності, повинні бути вмонтовані (не зовнішнім файлом).

Основні вимоги до структури відеоролика:

- відомості про автора, назву розробленого програмного продукту;
- демонстрація основних етапів роботи із розробленим програмним продуктом;
- коментарі до курсової роботи можна подавати у вигляді текстових повідомлень або супроводжуватися голосовими коментарями самого студента.

## 2.3. Підготовка до захисту курсової роботи

До захисту курсової роботи, здобувач вищої освіти (студент) повинен підготувати доповідь та презентацію, яку йому належить представити на засіданні комісії.

Слід відмітити, що регламент доповіді складає 10-15 хвилин, яка у свою чергу має супроводжуватися обов'язково презентацією наукової роботи. Презентаційні матеріали мають бути оформлені згідно вимог. У доповіді здобувача вищої освіти повинно бути чітко сформульовано та обґрунтовано актуальність вибраної теми, її затребуваність, предмет дослідження, об'єкт дослідження, мета роботи та її основне завдання. У наступній частині доповіді потрібно послідовно охарактеризувати кожен розділ курсової (наукової) роботи, звернувши особливу увагу на її підсумкові результати дослідження, зроблені висновки та внесені пропозиції.

## 2.4. Апробація отриманих результатів при написанні курсової роботи

Апробація являє собою різновид наукової діяльності у формі проведення певних перевірок результатів дослідження. Мета апробації полягає у встановленні придатності результатів для реалізації конкретних завдань. Слід також зазначити, що результати тієї чи іншої наукової діяльності дослідника тільки тоді мають практичну цінність і значущість, коли вони оприлюднені суспільству. Однією із найпоширеніших форм такої діяльності є наукові публікації, які виконують наступні функції:

- оприлюднюють результати наукової роботи автора (студента) і сприяють встановленню його пріоритету;
- свідчать про відповідний особистий внесок дослідника у розробку

наукової проблеми;

- підтверджують достовірність основних результатів і висновків, новизни і наукового рівня досліджень;
- підтверджують факт апробації та впровадження результатів та висновків наукової роботи;
- відбивають основний зміст наукової (курсової) роботи;
- фіксують завершення певного етапу дослідження або роботи в цілому;
- забезпечують первинною науковою інформацією суспільство, сповіщають наукове співтовариство про появу нового наукового знання, передають індивідуальний результат у загальне надбання тощо.

Види апробації:

- участь у бесідах, круглих столах та семінарах з обговоренням результатів роботи;
- представлення власних доповідей до Дня науки факультету інформаційних технологій і математики;
- опублікування результатів роботи в рецензованих фахових журналах, а також на міжнародному рівні;
- представлення основних результатів роботи в рамках наукових конференцій чи вебінарах;
- публікація тез, наукових статей, підручників і т.д;
- використання результатів роботи дослідження та пропозицій студента при підготовці викладачем окремих лекцій, практичних занять, наочного та інформаційного забезпечення навчальних дисциплін у закладах вищої та середньої освіти.

## 2.5. Академічна доброчесність

Дотримання студентами (здобувачами вищої освіти) академічної доброчесності при написанні курсової роботи з програмування регламентується ст. 42 Закону України «Про освіту» [6], Кодексу академічної доброчесності Волинського національного університету імені Лесі Українки [7], Положенням про систему запобігання та виявлення академічного плагіату у науково-дослідній діяльності здобувачів вищої освіти і науково-педагогічних працівників ВНУ імені Лесі Українки [8].

У разі виявлення науковим керівником у науковій роботі студента одного або кількох видів порушень академічної доброчесності, а саме: академічного плагіату, самоплагіату, фабрикації, фальсифікації, обману до нього можуть бути застосовані види відповідальності, передбачені Кодексом академічної доброчесності Волинського національного університету імені Лесі Українки, зокрема: повторне проходження оцінювання; повторне проходження відповідного освітнього компонента освітньої програми; відрахування з університету; позбавлення академічної стипендії; позбавлення наданих університетом пільг з оплати навчання.

З метою запобігання та виявлення плагіату у науковій (курсовій) роботі, сприяння дотриманню вимог наукової етики та поваги до інтелектуальних надбань, активізації самостійності й оригінальності тексту в університеті проводиться перевірка на плагіат. Слід відмітити, що дана перевірка проходить за допомогою сервісу Unicheck. Слід також зазначити, що Unicheck – це онлайн-інструмент для швидкої перевірки наукових текстів на плагіат. Сервіс виконує швидку перевірку по інтернет, а також по базах наукових робіт університетів чи репозитаріїв. Працює із різними форматами тексту, з необмеженою кількістю користувачів одночасно. Здійснюючи пошук на плагіат протягом декількох секунд, програма забезпечує найбільш точні результати в режимі реального часу. А також, дана перевірка авторського тексту курсової роботи за допомогою сервісу Unicheck може здійснюватися лише один раз.

У випадку, якщо порушення виявлені не менш, як за три-чотири тижні до захисту курсової роботи, студенту надається можливість виправити порушення. Якщо порушення виявлені менше, як за два-три тижні до захисту, курсова (наукова) робота не допускається до захисту, студент отримує оцінку «незадовільно» із можливістю повторного захисту.

### 2.6. Основні документи курсової роботи, які подаються до захисту

Перелік основних документів, які повинні бути представлені на кафедру перед захистом курсової роботи:

1) електронний варіант текстової частини курсової роботи у форматі *doc.* або *doxc*.;
2) електронний варіант програмної розробки курсової роботи з виконуваним файлом;
3) переплетений друкований примірник текстової частини курсової роботи, який повинен містити на титульному аркуші резолюцію «До захисту», дату та підпис наукового керівника;
4) розроблені презентаційні матеріали програмної розробки або відеоролик-презентація програмної розробки.

## 3. Порядок захисту та критерії оцінювання наукової (курсової) роботи

До захисту допускаються здобувачі вищої освіти (студенти), які у повному обсязі виконали завдання курсової (наукової) роботи і мають рецензію керівника, що є основою для оцінювання роботи студента.

Курсова робота (пояснювальна записка і розробка програмного продукту) здається керівнику для перевірки та рецензування за три тижні до запланованого захисту курсових робіт. Не пізніше ніж за 10 днів до захисту рецензована курсова робота подається на кафедру.

Комісія із захисту курсової роботи у визначений термін заслуховує повідомлення студентів про основні положення наукової роботи (до 15 хвилин), після чого студент дає відповіді на запитання членів комісії (загалом захист курсової роботи не може перевищувати 20 хвилин).

Обговорення результатів та виставлення оцінки проводиться комісією у присутності студентів. Студент, який не захистив курсову роботу, допускається, до її доопрацювання, до повторного захисту згідно з графіком ліквідації академічної заборгованості студентів.

Під час оцінювання курсових робіт враховуються відповідність змісту курсової роботи темі, меті і завданням, що визначались у вступі, новизна теми, якість оформлення курсової роботи, вчасність подання роботи у навчальну частину та якість захисту курсової роботи.

### 3.1. Послідовність процедури захисту

Допуск здобувача вищої освіти до захисту курсової роботи здійснює науковий керівник. Основними критеріями допуску є:

1) наявність електронного варіанту текстової частини курсової роботи у форматі *.doc* або *.docx* оформленого згідно вимог;
2) наявність електронного варіанту працюючої програмної розробки (згідно основних задач та мети, поставлених у роботі);
3) наявний переплетений друкований примірник текстової частини курсової роботи, оформлений згідно вимог, завізований керівником;
4) відповідність змісту текстової частини і теми курсової роботи;
5) наявність у додатках текстової частини курсової роботи технічного завдання та інструкції користувачу для використання програмної розробки;
6) наявність презентаційних матеріалів або відеоролика програмної розробки курсової роботи;
7) дотримання академічної доброчесності під час написання курсової роботи, відповідно до нормативних документів.

Захист курсової роботи проводиться перед комісією, яка складається не менше як з двох викладачів кафедри та за участю керівника курсової роботи. Дата захисту передбачається графіком підсумкового семестрового контролю на факультеті.

Захист курсової роботи включає в себе короткий виступ студента з презентацією та його відповіді на запитання, які задають члени комісії. У виступі студента відображаються актуальність теми, завдання курсової роботи, її основні результати та демонстрація роботи програмного продукту. Здобувач вищої освіти (студент) повинен продемонструвати вміння вільно та обґрунтовано відповідати на питання з предметної області курсової роботи, а також вести наукову дискусію.

Після закінчення процедури захисту комісія ухвалює рішення щодо підсумкової сумарної оцінки кожного здобувача за написання курсової роботи з урахуванням орієнтовних критеріїв. Слід відмітити, що результати захисту в той

же день оголошуються здобувачам вищої освіти. Диференційована оцінка за курсову роботу вноситься у заліково-екзаменаційну відомість, індивідуальний навчальний план (залікову книжку) студента за підписами членів комісії і враховується під час рейтингування на отримання стипендії разом з іншими підсумковими оцінками. У разі отримання підсумкової сумарної оцінки менше 60 балів за 100-бальною шкалою або у випадку, якщо курсова робота не була допущена до захисту, у заліково-екзаменаційній відомості робиться відповідний запис про академічну заборгованість з курсової роботи.

### 3.2. Ліквідація академічної доброчесності

Слід зазначити, що студент (здобувач вищої освіти) не допускається до захисту курсової роботи у деяких випадках, наприклад:

- недотримання критеріїв допуску;
- порушення термінів подачі курсової роботи керівнику (на кафедру) без поважних причин;
- порушення академічної доброчесності.

Таким чином, ліквідація академічної заборгованості здійснюється шляхом повторного виконання та захисту курсової роботи за новою темою (у випадку грубих порушень академічної доброчесності), або після виправлення недоліків у поданій курсовій роботі та її повторного захисту. Студент може бути допущений до повторного захисту курсової роботи у встановлений термін ліквідації академічної заборгованості. Інші випадки (хвороба, відрядження тощо) регламентуються Положенням про організацію навчального процесу на першому (бакалаврському) та другому (магістерському) рівнях у Волинському національному університеті імені Лесі Українки [2].

### 3.3. Основні критерії оцінювання та шкала оцінювання

Оцінку **«відмінно»** отримує студент, робота якого оформлена відповідно до вимог; зміст курсової роботи в повному обсязі відповідає темі та визначеній меті; у роботі містяться елементи наукового пошуку в теоретичному аспекті на основі опрацювання достатньої кількості наукової літератури; практичний результат обраної теми свідчить про вміння студента систематизувати зібраний матеріал, робити узагальнюючі висновки, які відповідають завданням дослідження; студент вільно володіє спеціальними термінами, не робить граматичних помилок.

Оцінку **«добре»** отримує студент робота якого оформлена відповідно до вимог; зміст курсової роботи в повному обсязі відповідає темі та визначеній меті, але є незначні недоліки методичного або стилістичного характеру. Під час захисту студент дає правильні відповіді, але недостатньо обґрунтовані та аргументовані.

Оцінку **«задовільно»** отримує студент, курсова робота якого розкриває теоретичні питання недостатньо повно, містить необґрунтований емпіричний матеріал, неадекватно застосовані методики дослідження, відсутність інтерпретації результатів методик, аналіз зроблено поверхнево; висновки фрагментарні, не відповідають завданням дослідження, робота оформлена неохайно.

Оцінку **«незадовільно»** виставляють здобувачу, якщо курсова робота виконана не в повному обсязі та з відхиленнями від завдання; оформлена без врахування визначених вимог; мають місце суттєві помилки; студент слабо володіє мовою викладу матеріалу і т.д.

Слід також відмітити, що оцінки за якість виконання курсової роботи та результати її захисту відображаються в сумарній підсумковій оцінці і виставляються за спільної згоди членами комісії.

**Шкала оцінювання (національна та ECTS)**

| **Сума балів за всі види навчальної діяльності** | **Оцінка ECTS** | **Оцінка за національною шкалою** |
|---|---|---|
| 90 – 100 | A | Відмінно |
| 82 – 89 | B | Добре |
| 75 - 81 | C | |
| 67 -74 | D | Задовільно |
| 60 - 66 | E | |
| 1 – 59 | FX | Незадовільно |

## 4. Приклад виконання курсової роботи з програмування на тему: «Створення програми науковий калькулятор мовою С++ з використанням середовища Visual Studio»

### 4.1. Вступ

Мова С++ багато в чому є надмножиною мови програмування Сі. Нові можливості Сі++ включають оголошення у вигляді виразів, перетворення типів у вигляді функцій, оператори new і delete, тип bool, посилання, розширене поняття константності та змінності, функції, що підставляються, аргументи за замовчанням, перевизначення, простори імен, класи (включаючи всі пов'язані з класами можливості, такі як успадкування, функції-члени (методи), віртуальні функції, абстрактні класи і конструктори), перевизначення операторів, шаблони, оператори, обробку винятків, динамічну ідентифікацію і багато іншого.

C++ є мовою програмування строгого типування і накладає більше функцій (операцій) щодо дотримання типів, порівняно з Сі. Базова мова, С – це підмножина C++, спроектована так, що існує дуже близька відповідність між його типами, операціями й операторами і комп'ютерними об'єктами, з якими безпосередньо приходиться мати справу: числами, символами й адресами Microsoft Visual C++ містить безліч інтегрованих засобів візуального програмування. Компілятор Visual C++ містить багато нових інструментальних засобів і поліпшених можливостей, надає величезні можливості в плані оптимізації додатків, внаслідок чого можна отримати якісний результат відносно розміру програми, так і відносно швидкості її виконання.

Система Microsoft Visual C++ дозволяє створювати як маленькі програми і утиліти для персонального використання, так і корпоративні системи, що працюють з базами даних на різних платформах.

Варто також зазначити, що C++ мова програмування високого рівня з підтримкою декількох парадигм програмування: об'єктно-орієнтованої, узагальненої та процедурної.

***Примітка. Обсяг вступу 2-3 сторінки.**

## 4.2. Актуальність теми, мета дослідження та основні завдання дослідження

Актуальність теми полягає в тому, інформатизація суспільства сьогодення впливає на усі сфери суспільного життя, у тому числі й освіти, завданням якої є виховання компетентнісного учня. Учня, який не тільки отримує знання, а й доцільно уміє використовувати їх у різних сферах своєї діяльності. А це, безперечно, передбачає опрацювання додаткової інформації, обмін нею, опрацювання і зберігання.

Головним технічним засобом обміну інформацією і її опрацювання у наш час є комп’ютер, який, на думку вчених-дослідників, найбільше відповідає структурі навчального процесу. Вважається, що він найбільш повно задовольняє дидактичні вимоги і дозволяє керувати навчальним процесом, а також адаптувати його до індивідуальних особливостей учня.

Важливо розуміти, що комп’ютеризація освіти, використання комп’ютерних технологій дозволить зробити навчальний процес більш ефективним, якщо його застосовувати як засіб пізнання, а не передавання знань.

У світі існують тисячі програмних середовищ, які використовують люди у своїй діяльності. При цьому будь-яке програмне середовище, яке на пряму пов’язане із калькулятором потребує постійного вдосконалення та розвитку на якісному професійному рівні.

На нашу думку, тема курсової роботи є досить актуальною у даний час, оскільки велика кількість людей використовують калькулятори для обчислення будь-яких арифметичних операцій. А також, сьогодення тісно пов’язане із цифровими технологіями, де важливе місце посідає обчислення числових даних

за допомогою калькуляторів у будь-яких сферах діяльності і спеціальностей, зокрема спеціальності 014 Середня освіта (Інформатика).

**Мета дослідження** – створення програми науковий калькулятор за допомогою мови програмування С++ з використанням середовища Visual Studio.

**Завдання дослідження:**

- охарактеризувати основні поняття мови програмування С++;
- проаналізувати основні прототипи (аналоги) калькуляторів;
- визначити основні можливості мови програмування С++ з використанням середовища Visual Studio;
- розробити основні етапи (алгоритм) створення програми науковий калькулятор;
- протестувати розроблену програму на якісному рівні.

### 4.3. Об’єкт дослідження та предмет дослідження

**Об’єкт дослідження** – математичні розрахунки.

**Предмет дослідження** – інженерний калькулятор.

### 4.4. Розділ 1. Опис мови і середовище програмування

У першому розділі **(теоретична частина)**, як правило, описують теоретичні дослідження з теми курсової роботи, написані з використанням першоджерел. Подається огляд існуючих підходів до розв’язання поставленої задачі, аналіз існуючих алгоритмів розв’язання, їх аналіз або порівняльна характеристика. Велике значення при написанні першого розділу має правильне трактування понять теми, їх точність і науковість. Використані терміни мають бути загальновживаними чи подаватися з посиланням на їх автора. Останнім підрозділом першого розділу повинен бути «Огляд та аналіз аналогічних програмних розробок». В даному пункті потрібно проаналізувати існуючі програмні продукти аналогічного призначення (орієнтовна кількість 2-3), вказавши їх переваги та недоліки та можливість використання аналогів для вирішення поставленої задачі.

При проведенні порівняльного аналізу аналогічних програмних розробок, зокрема калькулятор, студент для кожного аналогу програмного забезпечення повинен визначити наступні основні характеристики:

- назва;
- розробник (дистриб’ютор);
- архітектура (desktop application, client-server, web application, mobile application);
- мова реалізації;
- перелік основних функцій, основні характеристики;

- аналіз переваг та недоліків даного ПЗ;
- джерело інформації (веб-сайт).

Обов’язковим є використання рисунків (скріншотів) проаналізованих програмних продуктів (1-2 для кожного програмного засобу).

На основі результатів аналізу, поданого в даному розділі (підрозділі), будуть сформовані вимоги до програмного продукту, який реалізує мету та завдання даної курсової роботи.

## 4.5. Розділ 2. Розробка програмного середовища за допомогою мови програмування С++

### 4.5.1. Шаблонні спискові класи. Специфікація шаблонного класу Stack

Для того, щоб розширити можливості наукового калькулятора написаного на мові програмування С++ необхідно на першому етапі розширити клас Stack із шаблонами, який буде використовуватися у наступних наших розділах, наприклад: 2.2. для обчислення інфіксного виразу. Шаблонна версія цього класу включена у програмний додаток 6. Слід відмітити, що перевизначення класу Stack включає просту шаблонну техніку. Починаємо з оголошення, розміщуючи список параметрів шаблону перед оголошенням класу та замінюючи DataType на T.

**Специфікація шаблонного класу Stack**

**ОГОЛОШЕННЯ**

```
#include <iostreara.h>
#include <stdlib.h>

const int MaxStackSize = 50;

template <class T>
class Stack
{

   private:
      // закриті дані-члени
      T stacklist [MaxStackSize];
      int top;

      public:
      // конструктор
      Stack (void);
```

```
    // стікові методи доступу
    void Push (const T& item);
    T Pop (void);
    T Peek (void);

    // методи тестуванні і очищення
    int StackEmpty (void) const;
    int StackFull (void) const;
    void ClearStack (void);
};
```

### 4.5.2. Реалізація класу Stack, заснованого на шаблоні

Кожен метод класу окреслюється зовнішньою шаблонною функцією. Це вимагає розміщення списку параметрів шаблону перед кожною функцією та заміни класу типу Stack на Stack <T>. У фактичному визначенні функції ми маємо замінити параметризований тип DataType на шаблонний тип T. Тому наступний лістинг буде задавати нові методи розв'язків Push і Pop[1]. Отримаємо:

```
// constructor
template <class T>
Stack <T> :: Stack (void)
{}

// uses the LinkedList method ClearList to clear the stack
template <class T>
void Stack <T> :: ClearStack(void)
{
    stackList. ClearList ();
}

// use the LinkedList method InsertFront to push item
template <class T>
void Stack <T> :: Push (const T& item)
{
stackList. InsertFront (item);
}

// use the LinkedList method DeleteFront to pop stack
template <class T>
T Stack <T> :: Pop (void)
{
```

```
        // check for an empty linked list
        if (stackList. ListEmptyO)
        {
        cerr «"Popping an empty stack" «endl;
        exit (l);
        }
        // delete first node and return its data value
        return stackList. DeleteFront ();
}

// returns the data value of the first first item on the stack
template <class T>
T Stack <T> :: Peek(void)
{
        // check for an empty linked list
        if (stackList. ListEmpty ())
        {
        cerr «"Calling Peek for an empty stack" «endl;
        exit (l);
        }
        // reset to the front of linked list and return value
        stackList. Reset ();
        return stackList. Data ();
}
// use the LinkedList method ListEmpty to test for empty stack
template <class T>
int Stack <T> :: StackEmpty (void) const
{
return stackList. ListEmpty ();
}
```

### 4.5.3. Ранг та алгоритм інфіксного виразу

У даному розділі ми показуємо використання стеків для обчислення виразів постфіксного інверсного запису (RPN). Тема обчислення інфіксних виразів була навмисне опущена раніше, оскільки їх реалізація вимагає використання двох стеків: одного для операндів, та іншого для операторів. Оскільки у двох стеках є дані різних типів, при інфіксному обчисленні ефективно використовуються шаблони. Тому необхідно розробити алгоритм обчислення інфіксного виразу, який реалізується із шаблонним стіковим класом, які поєднуються із арифметичними операціями. Наприклад, такі операції можна об'єднати в унарний оператор – , бінарні оператори +, -, *, /, дужки та операнди з плаваючою точкою:

8.5 + 2 * 3 – 7 * (4/3 – 6.25) + 9

Слід відмітити, що ці вирази використовують переважно інфіксний запис (форму) із бінарними операторами, розташованими між операндами. Пара дужок створює вираз, що обчислюється окремо. В умовах високого рівня є порядок виконання операцій (order of precedence) та асоціативність (associativity) між операторами. Оператор із високим пріоритетом виконується першим. Якщо більше одного бінарного оператора мають той самий пріоритет, першим виконується крайній ліворуч оператор у разі лівої асоціативності (+, -, *, /) і крайній правий оператор – у разі правої асоціативності (унарний плюс, унарний мінус).

Табл. 4.1

Порядок виконання операцій та їх оператор

| Порядок виконання операцій (від низького до високого) | Оператор |
|---|---|
| 1 | +, – |
| 2 | *, / |
| 3 | унарний плюс, унарний минус |

Наприклад:
1. 8.5 + 2*3 = 14.5 // * виконується перед +;
2. (8.5 + 2) * 3 = 31.5 // дужки створюють окремі підвирази;
3. 9 – – 6 = 15 // унарний мінус має найвищий пріоритет.

**Ранг виразу.** Алгоритм для обчислення інфіксного виразу використовує поняття рангу (rank), який надає значення –1, 0 або 1 кожному наступному виразу:

Ранг операнда з плаваючою точкою дорівнює 1;

Ранг унарних операторів +, – дорівнює 0;

Ранг бінарних операторів +, –, *, / дорівнює –1.

Ранг лівої дужки дорівнює 0.

Коли розглядати більш детально представлені вирази вище, то можна зробити висновок про асоціацію сумарного рангу (cumulative rank), який є сумою рангу окремих термів від першого символу до даного терму. Сумарний ранг використовується для контролю за тим, щоб кожен бінарний оператор мав два навколишні операнди і щоб ніякі операнди у виразі не існували без інфіксного оператора. Наприклад, у простому виразі 2 + 3 послідовні значення рангу наступні:

Сканування 2: сумарний ранг = 1;

Сканування +: сумарний ранг = 1 + –1 = 0;
Сканування 3: сумарний ранг = 1.

Важливо відмітити, що для кожного терму у виразі сумарний ранг виразу повинен дорівнювати 0 або 1. А ранг повного виразу повинен дорівнювати 1.

Наприклад: Наступні вирази є невірними, що визначається функцією rank:

| Вираз | Невірний ранг | Причина |
|---|---|---|
| 1. 2.5A + 3 | Ранг 4 = 2 | Занадто багато послідовних операндів |
| 2. 2.5 + ∗, – 3 | Ранг * = –1 | Занадто багато послідовних операндів |
| 3. 2.5 + 3 – | Кінцевий ранг = 0 | Немає одного операнда |

**Алгоритм інфіксного виразу**. Алгоритм інфіксного виразу використовує стек операндів (operand stack) – стек значень з точкою для зберігання операндів і результатів проміжних обчислень. Другий стек, званий стеком операторів (operator stack) містить оператори та ліві дужки і дозволяє реалізувати порядок пріоритетів. При скануванні виразу терми поміщаються у відповідні стеки. Операнд поміщається у стек операндів, якщо він зустрічається у процесі сканування і витягується (popped) із стека, що є необхідним для операції. Варто зазначити, що оператор поміщається у стек лише тоді, коли вже були оцінені всі оператори із вищим або рівним пріоритетом, і звільняється зі стека, коли настає час його виконання. Це відбувається при введенні наступного оператора з нижчим або рівним пріоритетом або наприкінці виразу. Розглянемо наступний вираз:

2 + 4 – 3*6

Введення 2: Помістити 2 у стек операндів.
Введення +: Помістити + у стек операндів.
Введення 4: Помістити 4 у стек операндів.

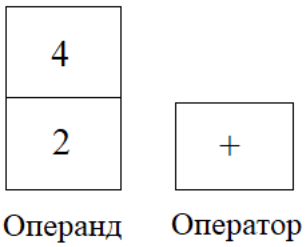



Рис. 4.1 – Операнди та оператори

Введення – :

Оператор – має той же порядок пріоритетів, що і оператор + в стеку. Спочатку необхідно витягнути 4 із стека операторів, витягнути два операнда і виконати операцію додавання. Результат (2 + 4 = 6) помістити назад у стек операндів.

Помістити – у стек операторів. Отримаємо:

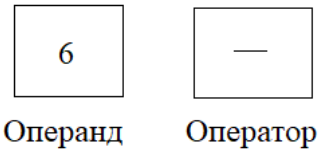



Рис. 4.2 – Стек операторів

Введення 3: Помістити 3 у стек операндів.

Введення * та : , отримаємо: оператор * має більший пріоритет, ніж оператор – у стеку. Необхідно помістити * у стек операторів.

Введення 6: Помістити 6 у стек операндів.

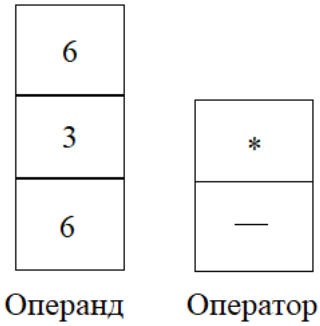



Рис. 4.3 – Стек операторів

Виконати: Трансполювати * та виконати операцію з операндами 6 та 3 із стека операндів. Помістити результат 18 у стек операндів.

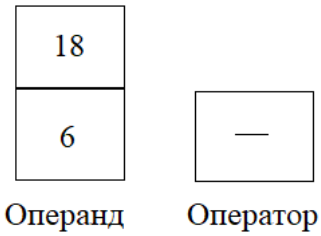



Рис. 4.4 – Стек операторів

Витягти зі стека і виконати наступну операцію – відняти разом з операндами 18 і 3 зі загальних стеків операндів. Тобто отримаємо: 6 – 18 = –12. Це результат.

Пріоритет оператора визначається двічі: спочатку при введенні оператора, потім, коли він знаходиться в стеку. Початкове значення, є вхідним пріоритетом (input precedence), використовується порівняння відносної важливості оператора з операторами в стеку. Як тільки оператор поміщається в стек, йому задається новий пріоритет, що називається стековим пріоритетом (stack precedence). Відмінність між вхідним та стіковим пріоритетом використовується для дужок та правих асоціативних операторів. У таких випадках стічний пріоритет менше, ніж вхідний пріоритет. Коли виявляється оператор з тим же пріоритетом та

асоціативністю, вхідний пріоритет перевищує стічний пріоритет оператора у вершині стека. Перший оператор не виймається, а новий оператор поміщається у стек. Порядок обчислення визначається з наступною послідовністю: зправа на наліво. У таблиці 4.2 наводиться вхідний та стіковий пріоритет та ранг, що використовуються для обчислення інфіксного виразу. Ці оператори включають дужки та бінарні оператори +, –, *, та /. Бінарні оператори є лівими асоціативними та мають рівний вхідний та стіковий пріоритет. Алгоритм обчислення виразів стає трохи складнішим за наявності дужок. Коли виявляється ліва дужка, вона представляє початок виразу і, отже, має бути негайно поміщена у стек. Це виконується привласненням лівої дужки вхідного пріоритету, який більший за пріоритет будь-якого з операторів. Якщо ліва дужка поміщена в стек, вона може бути видалена тільки тоді, коли знаходиться відповідна їй права дужка, і вираз обчислюється повністю. Слід відмітити, що пріоритет лівої дужки у стеку повинен бути меншим, ніж пріоритет будь-якого оператора, щоб вона не витягувалась зі стека при обчисленні всіх термів у виразі.

**Алгоритм.** Алгоритм реалізується із двома стеками. Стек операнда містить числа з плаваючою точкою, в той же час як елементами стека операторів є об'єкти класу типу MathOperator.

Табл. 4.2

Вхідний та стіковий пріоритет з рангом

| Символ | Пріоритет | | |
|---|---|---|---|
| | Вхідний пріоритет | Стіковий пріоритет | Ранг |
| + - (бінарний) | 1 | 1 | –1 |
| * / | 2 | 2 | –1 |
| ( | 3 | –1 | 0 |
| ) | 0 | 0 | 0 |

```
// клас, керуючий операторами в стеку-операторів
class MathOperator
{
   private:
      // оператор та два його значення пріоритету
      char op;
      int inputprecedence;
      int stackprecedence;

   public:
      // конструктор; включає конструктор замовчування та
      // конструктор, ініціалізуючий об'єкт
      MathOperator (void);
```

```
        MathOperator (char ch);

        // функції-члени для керування оператором у стеку
        int operators (MathOperator a) const;
        void Evaluate (Stack <float> & OperandStack);
        char GetOp (void);
};
```

MathOperator зберігає оператор і значення пріоритету, пов'язані з цим оператором. Конструктор задає як вхідний, і стіковий пріоритет оператора.

```
MathOperator :: MathOperator (char ch)
{
   op = ch;
   switch(op)
   {
      // '+' и '-' мають вхідний та стіковий пріоритет 1
      case ' +' :
      case '-': inputprecedence = 1;
                stackprecedence = 1;
                break;

      // '*' и '/' мають вхідний та стіковий пріоритет 2
      case '*' :
      case '/': inputprecedence = 2;
                stackprecedence « 2;
                break;

      // ' (' має вхідний пріоритет 3 і стіковий пріоритет -1
      case ' C : inputprecedence = 3;
                 stackprecedence = -1;
                 break;

      // ')' мають вхідний та стіковий пріоритет 0
      case ')' : inputprecedence « 0;
                 stackprecedence «■ 0;
                 break;
}
>
```

Клас MathOperator перевантажує C++ оператор ">", який використовується для порівняння значень пріоритетів.

```
        // Перевантажити оператор >= порівнянням стікового;
        // Пріоритету поточного об'єкта та вхідного пріоритету a;
```

```
// Використовується під час читання оператора визначення;
// Того, чи слід обчислювати оператори зі стека перед тим;
// Як помістити у нього новий оператор

int MathOperator :: operator>= (MathOperator a) const
<
return stackprecedence >* a. inputprecedence;
}
```

Цей клас містить функцію-член Evaluate, яка відповідає за виконання операцій та витягує два операнди. Після виконання даної функції операції та її результат поміщається у стек операндів. Отримаємо:

```
void MathOperator :: Evalute (Stack <double> & OperandStack)
{
        double operand1 = OperandStack. Pop ();
        if (op=='$') OperandStack. Push (-operand1);
        else if (!func)
        {
                double operand 2 = OperandStack. Pop ();

                switch(op)
                {
                case '+':
                        OperandStack. Push (operand 2 + operand1);
                        break;
                case '-':
                        OperandStack. Push (operand 2 - operand1);
                        break;
                case '*':
                        OperandStack. Push (operand 2 * operand1);
                        break;
                case '/':
                        OperandStack. Push (operand 2 / operand1);
                        break;
                case '^':
                        OperandStack. Push (pow (operand 2, operand1));
                }
        }
        else
        {
                switch (func)
                {
                case 1:
```

```
                OperandStack. Push (acos (operand1));
                break;
        case 2:
                OperandStack. Push (asin (operand1));
                break;
        case 3:
                OperandStack. Push (atan (operand1));
                break;
        case 4:
                OperandStack. Push (cos (operand1));
                break;
        case 5:
                OperandStack. Push (cosh (operand1));
                break;
        case 6:
                OperandStack. Push (exp (operand1));
                break;
        case 7:
                OperandStack. Push (fabs (operand1));
                break;
        case 8:
                OperandStack. Push (log (operand1));
                break;
        case 9:
                OperandStack. Push (log10 (operand1));
                break;
        case 10:
                OperandStack. Push (sin (operand1));
                break;
        case 11:
                OperandStack. Push (sinh (operand1));
                break;
        case 12:
                OperandStack. Push (tan (operand1));
                break;
        case 13:
                OperandStack. Push (tanh (operand1));
                break;
        case 14:
                OperandStack. Push (sqrt (operand1));
                break;

        }
}
```

}

Алгоритм обчислення інфіксного виразу зчитує кожен терм виразу, поміщає їх у відповідний стек і оновлює сумарний ранг. Введення завершується в кінці виразу або якщо ранг знаходиться поза діапазоном. Наступні перераховані нами правила застосовуються при зчитуванні термів:

*Введення операнда*: Помістити операнд у стек операндів.

*Введення оператора*: Вийняти з стека всі оператори, які мають пріоритет більший або дорівнює вхідному пріоритету поточного оператора. Виконати порівняння, використовуючи метод класу MathOperator ">=". Коли оператори будуть видалені зі стека, виконати оператор за допомогою методу Evaluate.

*Введення правої дужки ")"*: Вийняти та виконати всі оператори у стеку, що мають стіковий пріоритет більший або рівний вхідному пріоритету дужки ")", який дорівнює 0. Зауважте, що стіковий пріоритет дужки ")" дорівнює -1, так що процес зупиняється, коли зустрічається дужка "(". Виконанням є обчислення всіх операторів методу дужками. Якщо ніякої дужки "(" не виявлено, вираз є невірним ("немає лівої дужки")).

*Наприкінці виразу очистити стек операторів*: Ранг має бути 1. Якщо ранг менше 1, це означає, що не вистачає операнда. Якщо при очищенні стека виявляється "(", то вираз є неправильним ("немає правої дужки"). Коли оператори видаляються зі стека, функція Evaluate виконує кожне обчислення. Кінцевий результат виразу знаходять вибіркою зі стека операндів."

### 4.5.4. Обчислення інфіксного виразу

Даний підрозділ описує та ілюструє обчислення інфіксного виразу. Ми зчитуємо кожен терм виразу, пропускаємо всі символи прогалин, доки не знаходимо "=". Під час цього процесу виконується перевірка помилок із печаткою спеціальних повідомлень про помилки. Після завершення введення обчислюються терми виразу, що залишилися, і його значення виводиться для друку. Отримаємо:

```
int isoperator (char *ch, bool unary)
{
        if (unary && * ch=='-')
        {
                *ch='$';
                return 1;
        }
        if (*ch=='+' || *ch=='-' || *ch=='*' ||
                *ch=='/' || *ch=='(' || *ch=='^')
                return 1;
        else
```

```
		return 0;
}
int iswhitespace (char ch)
{
	if (ch==' ' || ch=='\t' || ch=='\n')
		return 1;
	else
		return 0;
}


double Calculate (char *string, int &error)
{
	MathOperator opr1, opr2;
	Stack <double> OperandStack;
	Stack <MathOperator> OperatorStack;
	Expression expr (string);

	int rank = 0;
	bool issign = TRUE;
	double number;
	char ch;

	error =  -1;

	while (ch = expr. GetChar ())
	{
		if (isdigit(ch) || ch=='.')
		{
			number = expr. GetNumber ();
			issign = FALSE;
			rank ++;
			if (rank >1)
			{
				error=0;
				return NULL;
			}
			OperandStack. Push (number);
		}
		else if (isoperator (& ch, issign))
		{
			Issign = TRUE;
			if (ch!='(' && ch!='$')
				rank--;
```

```
        if (rank<0)
        {
                error = 1;
                return NULL;
        }

        opr1 = MathOperator(ch);
        while (! (OperatorStack.StackEmpty ()) &&
                 (opr2 = OperatorStack. Peek ()) > = opr1)
        {
                opr2 = OperatorStack. Pop ();
                opr2. Evalute (OperandStack);
        }
        OperatorStack. Push (opr1);
}

else if (ch==')')
{
        opr1 = MathOperator(ch);
        while (! (OperatorStack. StackEmpty ()) &&
                 (opr2 = OperatorStack. Peek ()) > = opr1)
        {
                opr2 = OperatorStack. Pop ();
                opr2. Evalute (OperandStack);
        }
        if (OperatorStack. StackEmpty ())
        {
                error = 2;
                return NULL;
        }
        opr2 = OperatorStack. Pop ();
        if (! OperatorStack. StackEmpty ())
        {
                opr2 = OperatorStack. Peek ();
                if (opr2. IsFunction ())
                {
                        opr2 = OperatorStack. Pop ();
                        opr2. Evalute (OperandStack);
                }
        }
}

else if (! iswhitespace(ch))
{
```

```
                if (int fnumber = expr. GetFunNumber ())
                {
                        if (fnumber == 15)
                        {
                                issign = FALSE;
                                rank ++;
                                if (rank >1)
                                {
                                        error = 0;
                                        return NULL;
                                }
                                OperandStack. Push (PI);
                        }
                        else
                        {
                                opr2 = MathOperator (fnumber);
                                OperatorStack. Push (opr2);
                        }
                }
                else
                {
                        error = 4;
                        return NULL;
                }
        }
}

if (rank! = 1)
{
        error = 1;
        return NULL;
}

while (! OperatorStack. StackEmpty ())
{
        opr1 = OperatorStack. Pop ();
        if (opr1. GetOp () ='(')
        {
                error = 3;
                return NULL;
        }
        opr1. Evalute (OperandStack);
}
return OperandStack. Pop ();
```

}/*.

**Висновки:**

Під час розробки програми науковий калькулятор мовою програмування С++ з використанням середовища Visual Studio мною були засвоєні основні функції мови С++. А також, ми охарактеризували основні поняття даної мови програмування, виокремили основні їх види: машинно-залежні, машинно-орієнтовані, об’єктно-орієнтовані.

У процесі дослідження ми визначили основні можливості мови програмування С++ і переконалися у тому, що ця мова програмування є найзручнішою для практичної роботи з комп’ютером, має зрозумілий інтерфейс, тому ідеально підходить для створення наукового калькулятора.

Слід відмітити, що створивши програму науковий калькулятор, я здійснив (ла) можливість користувачу виконувати в ній основні математичні дії. У програмі закладені основні способи та прийоми роботи з вказівниками та різноманітними арифметичними функціями. Основна задача полягає у виконанні арифметичних дій (операцій), виведення таких констант як: PI, Long та експоненти, виконання дій з ними. А також, розроблений науковий калькулятор дозволяє застосовувати та здійснювати розрахунок тригонометричних функцій таких як: sin, cos, tan, ctg і т.д. Важливо зазначити і те, що розроблений науковий калькулятор мовою програмування С++ з використанням середовища Visual Studio дозволяє також виконувати математичні дії використовуючи будь-які дужки. Також доцільним було тестування на логічні та синтаксичні помилки.

Слід також зазначити, що використовувати програму науковий калькулятор можна у будь-яких організаціях, а також за різним спрямуванням та призначенням. Також неможна не згадати, що завдяки написанню даної курсової роботи з програмування на тему: «Створення програми науковий калькулятор мовою С++ з використанням середовища Visual Studio» я отримав (ла) хороший практичний досвід у використанні функцій програмування. Було засвоєно усі аспекти створення та програмування програми калькулятор, що дає змогу у подальшому розробляти більш серйозні проекти.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. Положення про організацію навчального процесу у вищих навчальних закладах. Режим доступу: https://zakon.rada.gov.ua/laws/show/z0173-93#Text.
2. Положення про організацію навчального процесу на першому (бакалаврському) та другому (магістерському) рівнях у Волинському національному університеті імені Лесі Українки. Режим доступу: https://ed.vnu.edu.ua/wp-content/uploads/2022/08/2022Polozhennya_pro_org_anizatsiyu_navch._pr_otsesu_u_VNU_%D1%80%D0%B5%D0%B4.pdf.
3. Положення про випускні кваліфікаційні роботи (прєкти). Режим доступу: https://vnu.edu.ua/uk/normativno-pravova-baza.
4. Приклади оформлення бібліографічного опису відповідно до ДСТУ 8302:2015. Режим доступу: https://kubg.edu.ua/images/stories/podii/2017/06_21_posylannia/dstu_8302.pdf.
5. Марченко А.В. Проектування інформаційних систем. Режим доступу: http://kist.ntu.edu.ua/textPhD/PIS_Marchenko.pdf.
6. Закон України «Про освіту». Режим доступу: https://zakon.rada.gov.ua/laws/show/2145-19#Text.
7. Кодекс академічної доброчесності Волинського національного університету імені Лесі Українки. Режим доступу: https://ra.vnu.edu.ua/wp-content/uploads/2020/11/Kodeks-akademichnoyi-dobrochesnosti.pdf.
8. Положення про систему запобігання та виявлення академічного плагіату у науково-дослідній діяльності здобувачів вищої освіти і науково-педагогічних працівників ВНУ імені Лесі Українки. Режим доступу: https://vnu.edu.ua/uk/normativno-pravova-baza.
9. Ришковець Ю.В., Висоцька В.А. Алгоритмізація та програмування: навчальний посібник / за редакцією Ю.В. Ришковець, В.А. Висоцької. Львів: Новий Світ-2000, 2021. 336 с. Режим доступу: https://ns2000.com.ua/alhorytmizatsiia-ta-prohramuvannia-chastyna-1-navchal-nyy-posibnyk/
10. Данілова А.В. Об’єктно-орієнтоване програмування. Практикум: навчальний посібник / за редакцією А.В. Данілової. Київ: КПІ ім. Ігоря Сікорського, 2021. 121 с. Режим доступу: https://ela.kpi.ua/bitstream/123456789/45183/1/OOP_praktykum.pdf

**ДОДАТОК 1**

*Зразок титульної сторінки курсової роботи з програмування*

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ ВОЛИНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТІМЕНІ ЛЕСІ УКРАЇНКИ**

**Кафедра загальної математики та методики навчання інформатики**

Курсова робота з програмування

# СТВОРЕННЯ ПРОГРАМИ НАУКОВИЙ КАЛЬКУЛЯТОР МОВОЮ С++ З ВИКОРИСТАННЯМ СЕРЕДОВИЩА VISUAL STUDIO

Виконав:
Бороненко Владислав Ігорович,
студент групи Інф – 45 О
факультету інформаційних
технологій і математики

Науковий керівник:
Пастернак Вікторія
Валентинівна,
к.т.н., доцент
кафедри загальної математики
та методики навчання
інформатики

Луцьк 2022

**ДОДАТОК 2**

*Шаблон формування змісту курсової роботи*

РЕФЕРАТ

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ (за потреби)

ВСТУП

РОЗДІЛ 1. НАЗВА РОЗДІЛУ …

1.1. Назва …

1.1.1. Назва …

1.2. Назва …

1.2.1. Назва …

1.2.2. Назва …

1.3. Назва …

1.4. Огляд та аналіз аналогічних програмних розробок

РОЗДІЛ 2. НАЗВА РОЗДІЛУ…

2.1. Постановка задачі, призначення та вимоги до програмного засобу «*назва програмної розробки*»

2.2. Загальний опис проєкту

2.3. Обґрунтування вибору інструментальних засобів розробки «*назва програмної розробки*»

2.4. Особливості програмної реалізації «*назва програмної розробки*»

2.5. Організація тестування та налагодження програмного засобу «*назва програмної розробки*»

2.6. Рекомендації по використанню та впровадженню програмного засобу «*назва програмної розробки*»

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТКИ

## ДОДАТОК З

*Вимоги до змісту і оформлення технічного завдання*

Unified system for program documentation.Technical specification for development.

Даний стандарт встановлює порядок побудови і оформлення технічного завдання на розробку програми чи програмного продукту для ПЕОМ, комплексів і систем незалежно від їх призначення та сфери застосування.

**Загальні положення.**

1.1. Технічне завдання оформляють за допомогою комп'ютера на одній стороні аркуша білого паперу формату А4 (210х297 мм) через півтора міжрядкових інтервали. (Текстовий редактор Word 7.0, розмір шрифту - 14, Tymes New Roman).

Текст документу необхідно друкувати, залишаючи наступні поля розмірів: лівий – 20 мм, правий – 10 мм, верхній – 20 мм, нижній – 20 мм.

Структурними елементами технічного завдання є розділи.

Заголовки розділів пишуть прописними буквами і розміщають симетрично відносно правої і лівої межі тексту (розмір шрифту – 14, жирний, Times New Roman).

Технічне завдання повинно містити такі розділи:

1) вступ;
2) підстави для розробки; призначення розробки;
3) вимоги до програми чи програмного продукту; вимоги до програмної документації;
4) техніко-економічні показники; стадії та етапи розробки; порядок контролю і приймання;
5) в технічне завдання допускається включати додатки.
6) залежно від особливостей програми, її специфіки допускається вдосконалювати зміст розділів, вводити нові (додаткові) розділи чи об'єднувати окремі підпунтки.

**ДОДАТОК 4**

*Приклади*
*оформлення бібліографічного опису*
*у списку використаних джерел з урахуванням національного стандартуУкраїни*
*ДСТУ 8302:2015*

| ***Опис книги з одним автором*** |
|---|
| Бондаренко В. Г. Історія України. Львів, 2017. 153 с. |
| Гурманова Л. І. Релігієзнавство : навч. посіб. 2-ге вид., переробл. та допов. Київ : ЦУЛ, 2017. 193 с. |
| Дичківська О. О. Інноваційний менеджмент : конспект лекцій. Київ : ДІА, 2018. 82 с. |
| Ваш О. М. Етика : навч.-метод. посіб. Запоріжжя : ЗНУ, 2018. 104 с. |
| Parker J. Principles of scientific research. 7th ed. London : Editorial, 2017. 301 p. |
| ***Опис книги з двома авторами*** |
| Боярин М. В., Нетробчук І. М. Основи гідроекології: теорія й практика : навч. посіб. Луцьк : Вежа-Друк, 2016. 365 с. |
| Мартиненко З. Е., Макар І. В. Управління підприємством: теоретико-методичні засади : монографія. Харків : Щедра садиба плюс, 2017. 296 с. |
| Палеха В. І., Карпова П. В. Менеджмент організацій : навч. посіб. Запоріжжя : ЗНУ, 2015. 120 с. |
| Білоус С. І., Корнійчук В. П. Філософія освіти : навч.-метод. посіб. Переяслав-Хмельницький, 2016. 176 с. |
| Мороз І. С., Василенко Н. Ю. Маркетинг : конспект лекцій. Київ : Молодь, 2016. 102 с. |
| Вердіна С. А., Волков А. А. Контролінг : навч. посіб. Запоріжжя : ЗНУ, 2016. 131 с. |
| Вердіна С. А., Волков А. А. Контролінг : навч. посіб. Вид. 3-тє., переробл. та допов. Херсон, 2017. 212 с. |
| ***Опис книги з трьома авторами*** |
| Тарнавська Г. Я., Марценюк Н. С., Герасимова Т. М. Фінанси : навч. посіб. Львів : Магнолія 2006, 2017. 412 с. |
| Пустовенко В. В., Максименко І. Л., Яким А.С. Безпека життєдіяльності : монографія. Харків : ХНПУ, 2017. 348 с. |
| Коваленко А. Д., Герасимчук О. П., Данилюк А. С. Міжнародне кредитування. 2-ге вид. Київ : Наука, 2018. 155 с. |
| Wilson D., Lister P., Andrews A. Modern surgery. Manchester : MAN, 2019. 240 p. |
| ***Опис книги з чотирма та більше авторами*** *(за необхідності в області заголовку перелічуються всі автори)* |

Інновації : навч. посіб. / Гуревич Д. Т., Чекан О. С., Грибан О. М., Макарова В. В. Запоріжжя : ЗНУ, 2016. 389 с.

Вища математика : конспект лекцій / Ткачук Т.С. та ін. Київ, 2015. 82 с.

Світ рослин у творчості І. П. Котляревського : науково-популярні нариси / М. В. Гриньова та ін. Полтава, 2017. 112 с.

*або*

Світ рослин у творчості І. П. Котляревського : науково-популярні нариси. Полтава, 2017. 112 с.

*або*

Гриньова М. В., Оніпко В. В., Купріян К. В., Ходунай В. В. Світ рослин у творчості І. П. Котляревського : науково-популярні нариси. Полтава, 2017. 112 с.

***Опис книги без зазначення автора (організація як автор або колективний автор)***

30 років історичному факультету: історія та сьогодення (1986-2016) : ювіл. вип. / під заг. ред. В. В. Черепані. Запоріжжя : ЗНУ, 2016. 340 с.

Етнографія : конспект лекцій / за заг. ред. В. І. Гарапка; уклад. А. І. Гарапко. Київ : ЦУЛ, 2018. 320 с.

Міжнародні відносини : монографія / за ред. М. А. Березовського. Київ : ЦУЛ, 2016. 162 с.

Міжнародні економічні відносини : навч. посіб. / за ред.: П. О. Бедрія, О. О. Петренка. Одеса : ОНУ, 2015. 306 с.

Науково-практичний коментар Цивільного кодексу України / за заг. ред. Т. А. Тарнавського. Київ : ЦУЛ, 2016. 186 с.

Підготовка фахівців у ВНЗ в умовах реформування вищої освіти : матеріали Всеукр. наук.-практ. конф., м. Мукачево, 4-5 жовт. 2018 р. Мукачево : МДУ, 2018. 226 с.

Освіта в Україні: виклики модернізації : зб. наук. пр. / редкол.: П. М. Марценюк (відп. ред.) та ін. Київ : Ін-т всесвітньої історії НАН України, 2017. 319 с.

Естетична освіта педагога : колективна монографія. Київ, 2015. 172 с.

***Опис книги on-line***

Палеха Ю. І., Леміш Н. О. Загальне документознавство. Режим доступу : https://textbook.com.ua/dokumentoznavstvo/1473445811 (дата звернення: 26.09.2017).

***Багатотомні видання***

Енциклопедія рослин / редкол.: І. М. Деркач та ін. Київ : ЦУЛ, 2016. Т. 8. 812 с.

Бюджетна система України: історія, стан та перспективи : у 3 т. / Акад. прав. наук України. Львів : Право, 2017. Т. 2 : Бюджетний менеджмент / заг. ред. Ю. П. Бубряка. 476 с.

Кучеренко Н. П. Казначейська справа : в 6 т. Київ : Право, 2016. Т. 3 : Контроль у системі Державного казначейства. 432 с.

***Переклад з іншої мови***

Гарфорд, Тім. Речі, що змінили світ. Історія економіки в 50 винаходах : пер. з англ. Київ, 2018. 352 с.

***Опис стандарту***

ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Чинний від 2016-07-01. Вид. офіц. Київ : УкрНДНЦ, 2016. 16 с.

ДСТУ 8746:2017. Автомобільні дороги. Методи вимірювання зчіпних властивостей поверхні дорожнього покриття. На заміну ДСТУ Б В.2.3-2-97 (ГОСТ 30413-96), ДСТУ Б В.2.3-8-2003, СОУ 45.2-00018112-042:2009 ; чинний від 2019-01-01. Вид. офіц. Київ : УкрНДНЦ, 2019. 20 с.

ДСТУ ISO 14024:2018. Екологічні маркування та декларації. Екологічне маркування типу I. Принципи та процедури. На заміну ДСТУ ISO 14024:2002 ; чинний від 2020-01-01. Вид. офіц. Київ : УкрНДНЦ, 2019. 18 с.

***Опис патенту***

Спосіб лікування синдрому дефіциту уваги та гіперактивності у дітей: пат. 76509 Україна. № 2004042416 ; заявл. 01.04.2014 ; опубл. 01.08.2016, Бюл. № 8 (кн. 1). 120 с.

Верстат для поздовжнього розпилювання круглих колод : пат. 123197 Україна : B27B 7/00. № u 2017 10306 ; заявл. 25.10.2017 ; опубл. 12.02.2018, Бюл. № 3. 4 с.

Рентгенодіагностичний препарат на основі конусно-променевого комп'ютерного томографа для обстеження кінцівок : пат. 140662 Україна : A61B 6/03, A61B 8/13, H05G 1/02, G03B 42/02, G01N 23/04. № u 2019 07999 ; заявл. 12.07.2019 ; опубл. 10.03.2020, Бюл. № 5 (кн. 1). 1 с.

Розбірний візок для транспортування надувного човна : пат. 121790 Україна : B60Q 5/00. № u 2017 09803 ; заявл. 24.07.2017 ; опубл. 11.12.2017, Бюл. № 23. 5 с.

***Опис дисертації***

Винниченко О. М. Контроль соціально-економічного розвитку промислових підприємств : дис. д-ра екон. наук: 08.00.04. Київ, 2018. 344 с.

Устьян О. Ю. Клієнтоорієнтований маркетинг підприємств сфери розваг і відпочинку : автореф. дис. канд. екон. наук : 08.00.04. Полтава, 2018. 20 с.

***Опис довідника***

Середняцька Г. В. Історія України. Універсальний довідник для школярів та абітурієнтів. Опорні конспекти : довідник. Київ, 2017. 272 с.

***Опис словника***

Великий тлумачний словник сучасної української мови / уклад. та голов. ред. В. Т. Бусел. Київ, 2017. 1728 с.

***Матеріали конференції***
***Тези доповіді***

Святецька А. В. Діалектизми у повісті М. Коцюбинського «Тіні забутих предків» : семантико-стилістичний аспект. *Стратегії розвитку та пріоритетні завдання філологічних наук* : матеріали Всеукр. наук.-практ. конф., м. Запоріжжя, 19-20 жовт. 2018 р. / Класич. приват. ун-т. Запоріжжя, 2018. С. 19-23.

Киридон А., Троян С. М. Грушевський і основні етапи українського державотворення. Міжнародна наукова конференція до 150-річчя М. С. Грушевського : тези доп., 17 верес. 2016 р., Острог / редкол.: Винар Л.-Р. та ін. Острог, 2016. С. 44-47.

Івченко В. О. Проблема правового регулювання імпічменту в Україні. *Актуальні проблеми сучасної науки і правоохоронної діяльності* : тези доп. учасників XXV Наук.-практ. конф. курсантів та студентів, м. Харків, 17 трав. 2018 р. / Харків. нац. ун-т внутр. справ. Харків, 2018. С. 35-36.

Цехмістров І. І., Перець І.П. Про бюджет. *Дослідження проблем в Україні очима молодих вчених* : матеріали Міжнар. наук.-практ. конф., м. Запоріжжя, 3-4 берез. 2016 р. Запоріжжя, 2016. С. 50-53.

Бібліотечне краєзнавство у культурному просторі України : зб. матеріалів Всеукр. наук.-практ. конф., м. Київ, 2 листоп. 2017 р. Київ : Національна академія керівних кадрів культури і мистецтв, 2017. 246 с.

Майстренко В. М., Осадчук О. П. Теоретичні основи впровадження системи управління якістю. *Проблеми управління та економіки підприємств в сучасних умовах* : матеріали XV міжнар. наук.-практ. конф., м. Київ, 23-24 квіт. 2019 р. Київ : НУХТ, 2019. С. 18-21.

***Опис статті***

Коваль Л., Коваль П. Переваги дистанційної роботи. *Урядовий кур'єр*. 2017. 1 листоп. (№ 205). С. 5.

Bletskan D. I., Glukhov K. E., Frolova V. V. Electronic structure of 2H-SnSe2. *Semiconductor Physics Quantum Electronics and Optoelectronics*. 2017. Vol. 18, No 2. P. 109-118.

Мурашко І. С. Біономічний підхід до сталого розвитку підприємства. *Вісник Запорізького національного університету. Серія «Економічні науки»*. 2017. № 4. С. 43-49.

Близнюк О. П., Ставерська Т. О., Іванюта О. М. Формування кредитно-грошового механізму забезпечення сталого розвитку підприємств торгівлі України. *Бізнес Інформ*. 2019. № 7. С. 240-249.

Андрущенко В. Академічна недоброчесність як виклик інтелектуальній спроможності нації. *Голос України*. 2018. 20 лип. С. 10.

***Опис статті зі збірника on-line***

Іващенко В. Л. Когнітивне термінознавство: перспективи розвитку // Термінологічний вісник 2017. Вип. 1. С. 47-54. URL : http://nbuv.gov.ua/UJRN/terv_2011_1_7 (дата звернення: 26.09.2017).

*або*

Іващенко В. Л. Когнітивне термінознавство: перспективи розвитку. Термінол. вісн. 2017. Вип. 1. С. 47-54. URL : http://nbuv.gov.ua/UJRN/terv_2011_1_7 (дата звернення 26.09.2017).

***Електронні ресурси***
***Книги***

Академічна чесність як основа сталого розвитку університету / за заг. ред. Т. В. Фінікова, А. Є. Артюхова. Київ : Таксон, 2016. 234 с. URL: http://www.univer.kharkov.ua/images/redactor/news/2016-09-07/chesnist_osnova_rozvitk_Univers.pdf (дата звернення: 02.11.2017).

Україна очима дітей : фотовиставка. URL: http://www.kmu.gov.ua/control/uk/photogallery/gallery?galleryId=15725757& (дата звернення: 15.11.2017).

Хміль А. А. Функції державної служби за законодавством України. *Юридичний науковий електронний журнал*. 2017. № 5. С. 115-118. URL: http://lsej.org.ua/5_2017/32.pdf.

Куцкір Я. С., Махно Б. А., Борислав С. Г. Трансформація науково-педагогічної системи України протягом 90-х років XX століття: період переходу до ринку. *Наука та інновації*. 2016. Т. 12, № 6. С. 6-14. DOI: https://doi.org/10.15407/scin12.06.006.

***Електронні ресурси***
***Статті з періодичних видань***

Костюченко Я. М. Механізми вирішення спорів в угоді про асоціацію між Україною та ЄС. Науковий вісник Ужгородського національного університету. Серія : Право. 2019. Вип. 56, т. 2. С. 144-147. URL: http://www.visnyk-juris.uzhnu.uz.ua/file/No.56/part_2/31.pdf (дата звернення: 23.08.2019).

Мірошниченко О. Ю., Карюк В. І. Етапи формування організаційно-економічного механізму інноваційної діяльності підприємств. Ефективна економіка. 2017. № 2. URL:

http://www.economy.nayka.com.ua/?op=1&z=932 (дата звернення: 22.01.2018)

***Законодавчі документи***

Про стандартизацію : Закон України від 11 лют. 2014 р. № 1315. URL: https://zakon.rada.gov.ua/laws/show/1315-18 (дата звернення: 02.09.2019).

Про затвердження Порядку використання документів через обмінні бібліотечні фонди : наказ Міністерства культури України від 31 жовт. 2017 р. № 1131. URL: https://zakon3.rada.gov.ua/laws/show/z1583-17 (дата звернення: 02.08.2019).

***Сторінки веб-сайтів***

Органічне землеробство та його розвиток в Україні. *Agronews* : веб-сайт. URL: https://agronews.ua/node/24264 (дата звернення: 02.09.2019).

Красива і дивовижна Полтава. *Моя планета* : веб-сайт. URL: http://myplanet.com.ua/?p=10440 (дата звернення: 10.09.2019).

Чайка А. С. Інклюзивна освіта - шлях до повноцінної соціалізації учнів з особливими освітніми потребами. *Всеосвіта* : веб-сайт. URL: https://vseosvita.ua/library/inkluzivna-osvita-slah-do-povnocinnoi-socializacii-ucniv-z-oop-1906.html (дата звернення: 12.08.2019).

Офіційний курс гривні щодо іноземних валют на дату 11.06.2020. Національний банк України. URL: https://bank.gov.ua/ua/markets/exchangerates?date=11.06.2020&period=daily (дата звернення: 11.06.2020).

Приймак Д. М., Томіленко О. В., Ковальчук З. Ю. «Підодіяльник»: як правильно сказати українською?. Kyiv Dictionary. URL: https://www.kyivdictionary.com/uk/grammar/uk/how-to-say/pidodiialnyk/ (дата звернення: 09.06.2020).

Київський національний університет імені Тараса Шевченка. URL: http://www.univ.kiev.ua/ (дата звернення: 05.11.2019).

APA Style Introduction. Purdue University. URL: https://owl.purdue.edu/owl/research_and_citation/apa_style/apa_style_introduction.html (date of access: 09.06.2020).

***Диск***

Тараненко Ю. А. Енциклопедія українського козацтва. Запоріжжя, 2016. 2 електрон.-опт. диски (DVD-R). Тартак. Сімка. Наш формат, 2016. 1 електрон.-опт. диск (CD) .

***Опис статті з журналу***

*(за необхідності в області заголовку перелічуються всі автори)*

Загірняк М., Костенко А. Про користування можливостями міжнародної бази даних Scopus. *Вищашкола.* 2017. № 5-6. С. 48-55.

*або*

Загірняк М., Костенко А. Про користування можливостями міжнародної бази даних Scopus. *Вища школа*. 2017. № 5-6. С. 48-55.

| ***Опис статті з журналу on-line*** |
|---|
| Мар'їна О. Контент-стратегія бібліотек у цифровому середовищі. *Бібл. вісн*. 2016. № 4. С. 8-12. URL : http://nbuv.gov.ua/UJRN/bv_2016_4_4 (дата звернення: 26.09.2017).<br>*або*<br>Мар'їна О. Контент-стратегія бібліотек у цифровому середовищі. *Бібліотечний вісник,* 2016. № 4.С. 8-12. URL : http://nbuv.gov.ua/UJRN/bv_2016_4_4 (дата звернення: 26.09.2017). |
| ***Опис статті з газети on-line*** |
| Кобець В. Різдвяні свята з Василем Стусом. *Літературна Україна*. 2017. 2 лют. (№ 5). URL : http://litukraina.kiev.ua/r-zdvyan-svyata-z-vasilem-stusom (дата звернення 15.09.2017). |
| ***Опис матеріалів конференції*** |
| Етико-естетична традиція у вітчизняній культурі : VII Всеукраїнська студентська науково-практична конференція з міжнародною участю, Київ, 23 листоп. 2017 р. Київ. 127 с.<br>Економіка, менеджмент, освіта в системі реформування агропромислового комплексу : матеріали Всеукр. конф. молодих учених-аграрників «Молодь України і аграрна реформа», (Харків, 11-13 жовт. 2020 р.). Харків : Харк. держ. аграр. ун-т ім. В. В. Докучаєва, 2020. 167 с. |
| ***Опис збірника тез доповідей*** |
| Етико-естетична традиція у вітчизняній культурі: тези VII Всеукраїнської студентської науково-практичної конференції з міжнародною участю, 23 листоп. 2017 р. Київ, 2017. 127 с. |
| ***Опис офіційного документа*** |
| Земельний кодекс України : офіц. видання : текст прийнятий ВР України 25 жовт. 2021 р. Київ, 2021. 171 с.<br>*або*<br>Земельний кодекс України. Київ, 2021. 171 с.<br>Цивільний кодекс України : чинне законодавство України зі змінами та допов. станом на 11 жовтня 2012 р. Київ, 2012. 272 с.<br>*або*<br>Цивільний кодекс України. Київ, 2012. 272 с. |
| ***Закон. Нормативний акт*** |
| Про забезпечення функціонування української мови як державної : Закон України від 25.04.2019 р. № 2704-VIII : станом на 19 квіт. 2020 р. URL: https://zakon.rada.gov.ua/laws/show/2704-19 (дата звернення: 11.06.2020).<br>Митний кодекс України : Закон України від 13.03.2012 р. № 4495-VI : станом на 02 квіт. 2020 р. URL: https://zakon.rada.gov.ua/laws/show/4495-17 (дата звернення: 09.06.2020).<br>Про грошове забезпечення військовослужбовців, осіб рядового і начальницького складу та деяких інших осіб : Постанова Каб. Міністрів України від 30.08.2017 р. № 704 : станом на 29 січ. 2020 р. |

URL: https://zakon.rada.gov.ua/laws/show/704-2017-%D0%BF (дата звернення: 05.05.2020).

Про відзначення державними нагородами України працівників Національного університету «Чернігівська політехніка» : Указ Президента України від 09.06.2020 р. № 215/2020. URL: https://www.president.gov.ua/documents/2152020-34077 (дата звернення: 11.06.2020).

Про іменні стипендії Верховної Ради України для молодих учених – докторів наук : Постанова Верховної Ради України від 05.02.2019 р. № 2676-VIII. Голос України. 2019. 19 лют. С. 4.

***Препринт***

Головач Ю., Пляцко Р., Сварник Г. Петер Пулюй і архів Івана Пулюя. Львів : Ін-т фізики конденс. систем НАН України, 2020. 24 с. (Препринт. Ін-т фізики конденс. систем НАН України ; ICMP–20–01).

Протидія агресивному податковому плануванню в Україні / С. С. Брехов та ін. Ірпінь : Ун-т ДФС України, 2017. 108 с. (Препринт. Наук.-дослід. ін-т фіскал. політики Ун-ту ДФС України).

Simon J. Physics of oscillations. Poznań : University of Poznań, 2019. 121 p. (Preprint. University of Poznań ; UP-02).

**ДОДАТОК 5**

*Приклад оформлення інструкції користувача*

```
#include <windows.h>
#include <math.h>

class Expression
{
private:
	char *instring;
	int isfunction(char *s,int nfunc);

public:
	Expression(char *string);
	char GetChar(void);
	double GetNumber(void);
	int GetFunNumber(void);
};

Expression::Expression(char *string)
{
	instring=string;
	strupr(instring);
}

char Expression::GetChar(void)
{
	char temp=*instring;
	instring++;
	return temp;
}

double Expression::GetNumber(void)
{
	double temp=0;
	double divisor=10;
	int exponent=0;
	bool eminus=0;

	instring--;
	while (isdigit(*instring))
	{
		temp=temp*10+(*instring-'0');
		instring++;
```

```
	}

	if (*instring=='.')
	{
		instring++;
		while (isdigit(*instring))
		{
			temp=temp+(*instring-'0')/divisor;
			divisor*=10;
			instring++;
		}
	}

	if (*instring=='E')
	{
		instring++;
		if (*instring=='+')
			instring++;
		if (*instring=='-')
		{
			instring++;
			eminus=1;
		}
		while(isdigit(*instring))
		{
			exponent=exponent*10+(*instring-'0');
			instring++;
		}
		if (eminus)
			temp=temp/pow(10,(double)exponent);
		else
			temp=temp*pow(10,(double)exponent);
	}

	return temp;
}

int Expression::isfunction(char *s, int nfunc)
{
	int len=strlen(s);
	instring--;
	if (strncmp(s, instring, len)==0)
	{
		instring+=len;
```

```
                return nfunc;
        }
        instring++;
        return 0;
}

int Expression::GetFunNumber(void)
{
        int FunNumber=0;
        if ((FunNumber=isfunction("ARCCOS",1)) ||
                (FunNumber=isfunction("ARCSIN",2))      ||
                (FunNumber=isfunction("ARCTG", 3))      ||
                (FunNumber=isfunction("COSH", 5))       ||
                (FunNumber=isfunction("COS", 4))  ||
                (FunNumber=isfunction("EXP", 6))  ||
                (FunNumber=isfunction("ABS", 7))  ||
                (FunNumber=isfunction("LN", 8))           ||
                (FunNumber=isfunction("LG", 9))           ||
                (FunNumber=isfunction("SINH", 11))        ||
                (FunNumber=isfunction("SIN", 10))  ||
                (FunNumber=isfunction("TGH", 13))||
                (FunNumber=isfunction("TG", 12))  ||
                (FunNumber=isfunction("SQRT",14))       ||
                (FunNumber=isfunction("PI",15)))

                return FunNumber;
        else
                return 0;
}



#include <math.h>

class MathOperator
{
private:
        char op;
        int func;
        int inputprecendence;
        int stackprecendence;

public:
```

```
        MathOperator(void);
        MathOperator(char ch);
        MathOperator(int nfunc);

        int IsFunction(void);
        int operator>=(MathOperator a) const;
        void Evalute(Stack<double> &OperandStack);
        char GetOp(void);
};

MathOperator::MathOperator(void)
{}

MathOperator::MathOperator(char ch)
{
        func=0;
        op=ch;
        switch(op)
        {
        case '+':
        case '-':
                inputprecendence=1;
                stackprecendence=1;
                break;
        case '*':
        case '/':
                inputprecendence=2;
                stackprecendence=2;
                break;
        case '(':
                inputprecendence=6;
                stackprecendence=-2;
                break;
        case ')':
                inputprecendence=0;
                stackprecendence=0;
                break;
        case '^':
                inputprecendence=3;
                stackprecendence=3;
                break;
        case '$':
                inputprecendence=5;
                stackprecendence=5;
```

```
	}
}

MathOperator::MathOperator(int nfunc)
{
	func=nfunc;
	inputprecendence=4;
	stackprecendence=-1;
}

int MathOperator::IsFunction(void)
{
	return func;
}


int MathOperator::operator >=(MathOperator a) const
{
	return stackprecendence>=a.inputprecendence;
}

void MathOperator::Evalute(Stack<double> &OperandStack)
{
	double operand1=OperandStack.Pop();
	if (op=='$') OperandStack.Push(-operand1);
	else if (!func)
	{
		double operand2=OperandStack.Pop();

		switch(op)
		{
		case '+':
			OperandStack.Push(operand2+operand1);
			break;
		case '-':
			OperandStack.Push(operand2-operand1);
			break;
		case '*':
			OperandStack.Push(operand2*operand1);
			break;
		case '/':
			OperandStack.Push(operand2/operand1);
			break;
		case '^':
```

```
                OperandStack.Push(pow(operand2,operand1));
        }
}
else
{
        switch(func)
        {
        case 1:
                OperandStack.Push(acos(operand1));
                break;
        case 2:
                OperandStack.Push(asin(operand1));
                break;
        case 3:
                OperandStack.Push(atan(operand1));
                break;
        case 4:
                OperandStack.Push(cos(operand1));
                break;
        case 5:
                OperandStack.Push(cosh(operand1));
                break;
        case 6:
                OperandStack.Push(exp(operand1));
                break;
        case 7:
                OperandStack.Push(fabs(operand1));
                break;
        case 8:
                OperandStack.Push(log(operand1));
                break;
        case 9:
                OperandStack.Push(log10(operand1));
                break;
        case 10:
                OperandStack.Push(sin(operand1));
                break;
        case 11:
                OperandStack.Push(sinh(operand1));
                break;
        case 12:
                OperandStack.Push(tan(operand1));
                break;
        case 13:
```

```
                OperandStack.Push(tanh(operand1));
                break;
            case 14:
                OperandStack.Push(sqrt(operand1));
                break;

            }
        }
}

char MathOperator::GetOp(void)
{
        return op;
}



//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by CalcRes.rc
//
#define VERSION_INFO            1
#define IDD_MFORM               102
#define IDI_ICON1               104
#define IDI_ICON_MAIN           104
#define IDR_MENU_MAIN           105
#define IDD_ABOUT               106
#define IDC_CALC_IN             1000
#define IDC_CALCULATE           1003
#define IDC_BUTTON1             1008
#define ID_ABOUT                40001

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        107
#define _APS_NEXT_COMMAND_VALUE         40002
#define _APS_NEXT_CONTROL_VALUE         1018
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif
```

```
#include <windows.h>

#define MAX_SIZE 50

template <class T>
class Stack
{
	private:
		T stacklist[MAX_SIZE];
		int top;

	public:
		Stack(void);

		void Push(const T& item);

		T Pop(void);
		T Peek(void);

		int StackEmpty(void);
		int StackFull(void);
		void ClearStack(void);
};

template <class T>
Stack<T>::Stack(void):top(-1)
{}

template <class T>
void Stack<T>::ClearStack(void)
{
	top=-1;
}

template <class T>
void Stack<T>::Push(const T& item)
{
	if (top==MAX_SIZE-1)
	{
		MessageBox(NULL,"Переповнення стеку!", "Помилка", MB_OK);
		exit(1);
	}
	top++;
```

```
	stacklist[top]=item;
}

template <class T>
T Stack<T>::Pop(void)
{
	T temp;
	if (top==-1)
	{
		MessageBox(NULL,"Стек порожній!", "Помилка", MB_OK);
		exit(1);
	}

	temp=stacklist[top];
	top--;

	return temp;
}

template <class T>
T Stack<T>::Peek(void)
{
	if (top==-1)
	{
		MessageBox(NULL,"Стек порожній! Тяжко знайти вершину!",
			"Помилка",MB_OK);
		exit(1);
	}

	return stacklist[top];

}

template <class T>
int Stack<T>::StackEmpty(void)
{
	return top==-1;
}

template <class T>
int Stack<T>::StackFull(void)
{
	return top==MAX_SIZE-1;
}
```

```
#include <windows.h>
#include <math.h>
#include "resource.h"
#include "stack.h"
#include "moperator.h"
#include "expression.h"
#define PI          3.14159265358979323846

int isoperator(char *ch, bool unary)
{
        if (unary && *ch=='-')
        {
                *ch='$';
                return 1;
        }
        if (*ch=='+' || *ch=='-' || *ch=='*' ||
                *ch=='/' || *ch=='(' || *ch=='^')
                return 1;
        else
                return 0;
}


int iswhitespace(char ch)
{
        if (ch==' ' || ch=='\t' || ch=='\n')
                return 1;
        else
                return 0;
}


double Calculate(char *string, int &error)
{
        MathOperator opr1, opr2;
        Stack<double> OperandStack;
        Stack<MathOperator> OperatorStack;
        Expression expr(string);

        int rank=0;
        bool issign=TRUE;
```

```
double number;
char ch;

error=-1;

while(ch=expr.GetChar())
{
	if (isdigit(ch) || ch=='.')
	{
		number=expr.GetNumber();
		issign=FALSE;
		rank++;
		if (rank>1)
		{
			error=0;
			return NULL;
		}
		OperandStack.Push(number);
	}
	else if (isoperator(&ch,issign))
	{
		issign=TRUE;
		if (ch!='(' && ch!='$')
			rank--;
		if (rank<0)
		{
			error=1;
			return NULL;
		}

		opr1=MathOperator(ch);
		while(!(OperatorStack.StackEmpty()) &&
			 (opr2=OperatorStack.Peek())>=opr1)
		{
			opr2=OperatorStack.Pop();
			opr2.Evalute(OperandStack);
		}
		OperatorStack.Push(opr1);
	}

	else if (ch==')')
	{
		opr1=MathOperator(ch);
		while(!(OperatorStack.StackEmpty()) &&
```

```
                (opr2=OperatorStack.Peek())>=opr1)
        {
                opr2=OperatorStack.Pop();
                opr2.Evalute(OperandStack);
        }
        if (OperatorStack.StackEmpty())
        {
                error=2;
                return NULL;
        }
        opr2=OperatorStack.Pop();
        if (!OperatorStack.StackEmpty())
        {
                opr2=OperatorStack.Peek();
                if (opr2.IsFunction())
                {
                        opr2=OperatorStack.Pop();
                        opr2.Evalute(OperandStack);
                }
        }
}

else if (!iswhitespace(ch))
{
        if (int fnumber=expr.GetFunNumber())
        {
                if(fnumber==15)
                {
                        issign=FALSE;
                        rank++;
                        if (rank>1)
                        {
                                error=0;
                                return NULL;
                        }
                        OperandStack.Push(PI);
                }
                else
                {
                        opr2=MathOperator(fnumber);
                        OperatorStack.Push(opr2);
                }
        }
        else
```

```
                {
                        error=4;
                        return NULL;
                }
        }
    }

    if (rank!=1)
    {
        error=1;
        return NULL;
    }

    while(!OperatorStack.StackEmpty())
    {
        opr1=OperatorStack.Pop();
        if (opr1.GetOp()=='(')
        {
            error=3;
            return NULL;
        }
        opr1.Evalute(OperandStack);
    }
    return OperandStack.Pop();
}

LRESULT CALLBACK About(HWND hDlg,
                                    UINT message,
                                    WPARAM wParam,
                                    LPARAM lParam)
{
    switch (message)
    {
    case WM_INITDIALOG:
        return TRUE;

    case WM_COMMAND:
        if (LOWORD(wParam) == IDOK)
        {
            EndDialog(hDlg, LOWORD(wParam));
            return TRUE;
        }
        break;
    }
```

```
        return FALSE;
}

LRESULT CALLBACK DlgProc(HWND hDlg,
                                        UINT message,
                                        WPARAM wParam,
                                        LPARAM lParam)
{
        static char *errormsgs[]={
                                "Відсутній оператор!",
                                "Відсутній операнд!",
                                "Нема лівої дужки!",
                                "Нема правої дужки",
                                "Невідома функція"
                                        };
        char *pstr,string[MAX_SIZE];
        int errornumber;
        double result;

        switch (message)
        {
        case WM_INITDIALOG:
                return TRUE;

        case WM_COMMAND:
                switch(LOWORD(wParam))
                {
                case IDCANCEL:
                        EndDialog(hDlg, LOWORD(wParam));
                        break;

                case IDOK:

                        GetDlgItemText(hDlg,IDC_CALC_IN,string,MAX_SIZE);
                        result=Calculate(string, errornumber);

                        if (errornumber==-1        )
                                pstr=gcvt(result,10,string);
                        else
                                pstr=errormsgs[errornumber];

                        SetDlgItemText(hDlg,IDC_CALC_IN,pstr);
                        break;
                case ID_ABOUT:
```

```
                    DialogBox(NULL, (LPCTSTR)IDD_ABOUT,
                                NULL,(DLGPROC)About);
                    break;
                }
                break;
        }
        return FALSE;
}

int APIENTRY WinMain(HINSTANCE hInstance,
                HINSTANCE hPrevInstance,
                LPSTR     lpCmdLine,
                int       nCmdShow)
{
        return DialogBox(hInstance, (LPCTSTR)IDD_MFORM, NULL,
(DLGPROC)DlgProc);

}
```

**УДК 004.42(072)**
**П 19**

*Електронне мережне навчальне видання*

Вікторія ПАСТЕРНАК, Світлана ЯЦЮК

# МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ КУРСОВОЇ РОБОТИ З ПРОГРАМУВАННЯ

для студентів спеціальності 014 Середня освіта (Інформатика)
першого (бакалаврського) рівня

Друкується в авторській редакції