Методичні вказівки
до виконання курсових робіт з дисципліни «Програмування» зі спеціальності «Інформаційні системи і технології»

Міністерство освіти і науки України
Вінницький національний технічний університет

Методичні вказівки до виконання курсових робіт з дисципліни «Програмування» зі спеціальності «Інформаційні системи і технології»

Вінниця
ВНТУ
2026

Рекомендовано до видання Радою з якості освіти Вінницького національного технічного університету Міністерства освіти і науки України (протокол № 7 від 23.01.2026 р.)

Рецензенти:
Д. Х. Штофель, кандидат технічних наук, доцент
В. В. Ковтун, доктор технічних наук, професор
М. Г. Тарновський, кандидат технічних наук, доцент

Методичні вказівки до виконання курсових робіт з дисципліни «Програмування» зі спеціальності «Інформаційні системи і технології» / уклад. : К. В. Овчинников, В. Г. Сторчак. Вінниця : ВНТУ, 2026. 69 с.

У методичних вказівках наводяться основні рекомендації для виконання курсових робіт з дисципліни «Програмування» здобувачами спеціальності «Інформаційні системи і технології» денної та заочної форм навчання. Наводяться приклади оформлення окремих частин роботи з урахуванням специфіки дисципліни. Методичні вказівки будуть корисни-ми здобувачам вищої освіти а також магістрантам, аспірантам і викладачам, що працюють в галузі інформаційних технологій.

ЗМІСТ

ВСТУП

Курсова робота з дисципліни «Програмування» є однією з ключових форм підсумкового контролю знань студентів спеціальності «Інформаційні системи і технології». Її виконання спрямоване на систематизацію та поглиблення теоретичних знань, а також на формування практичних навичок розробки програмного забезпечення. У процесі роботи студент набуває досвіду самостійного розв'язання прикладних задач із використанням сучасних підходів програмування. Особлива увага приділяється розвитку алгоритмічного мислення та інженерного підходу до створення програмних продуктів.

Методичні вказівки до виконання курсових робіт визначають загальні вимоги до змісту, структури та оформлення результатів роботи. Вони регламентують послідовність виконання основних етапів розробки програмного забезпечення, починаючи з аналізу предметної області та закінчуючи тестуванням і формулюванням висновків. Запропоновані рекомендації спрямовані на забезпечення логічної цілісності роботи та коректного викладення технічного матеріалу. Дотримання цих вимог сприяє уніфікації підходів до виконання курсових робіт.

Здобувач вищої освіти має продемонструвати вміння формулювати вимоги до програмного продукту, обґрунтовувати вибір інструментальних засобів і реалізовувати програмні рішення відповідно до розробленої архітектури. Важливим є і вміння виконувати ґрунтовний аналіз, проєктувати та оцінювати якість програмного забезпечення оскільки отримані компетентності є необхідними для подальшого професійного становлення фахівця.

Актуальність виконання курсової роботи з дисципліни «Програмування» для студентів спеціальності «Інформаційні системи і технології» також зумовлена зростанням ролі програмних засобів у сучасних інформаційних системах. Практична орієнтованість завдань дозволяє поєднати теоретичні знання з реальними задачами розробки програмного забезпечення. Виконання курсової роботи сприяє підготовці студентів до професійної діяльності в умовах швидкого розвитку інформаційних технологій. Саме тому курсова робота є важливим етапом фахової підготовки майбутніх спеціалістів.

1 МЕТА І ЗАВДАННЯ

Курсова робота з дисципліни «Програмування» є важливою складовою частиною підготовки технічних фахівців галузі знань F «Інформаційні технології». Написання курсової роботи є обов'язковим етапом у вивченні програмного матеріалу з названої дисципліни.

На виконання курсової роботи виділяється 1,5 кредити ЄКТС (45 годин самостійної роботи здобувачів вищої освіти).

Метою виконання курсової роботи є:

- поглиблення набутих теоретичних знань з дисципліни «Програмування»;
- формування практичних навичок створення програмного забезпечення, вирішення актуальних питань, пов'язаних з проєктуванням та тестуванням;
- застосування набутого студентом у процесі навчання науково-дослідницького потенціалу.

У процесі досягнення зазначеної мети вирішуються такі *завдання*:

- закріпити та поглибити знання з дисципліни «Програмування»;
- систематизувати методичний інструментарій, оволодіти конкретними комп'ютерними засобами проєктування;
- сформулювати задачу на розробку та обґрунтувати методи та підходи для реалізації;
- логічно і послідовно пройти відповідні етапи проєктування;
- проаналізувати результати роботи, зробити відповідні висновки.

Мета й *завдання* в межах виконання курсової роботи визначаються її темою, структурою, специфікою об'єкта, предмета та інформаційною базою дослідження.

Виконання студентами курсової роботи сприяє поєднанню в цілісну систему знань із галузі теоретичного та практичного проєктування програмних застосунків та процесів, що, в свою чергу, дозволяє їм –- майбутнім інженерам – сформувати чіткі уявлення про методологію проєтування та навчитись використовувати її на практиці.

Керівник курсової роботи надає допомогу в уточненні змісту, складанні завдання для виконання курсової роботи. Керівник також сприяє процесу збирання та отримання необхідного матеріалу для написання курсової роботи, рекомендує основну та додаткову літературу, проводить регулярні консультації; розробляє календарний графік виконання етапів роботи та слідкує за його дотриманням, перевіряє роботу, робить відповідні зауваження і вирішує питання про можливість допуску до захисту.

2 ПІДГОТОВКА ДО ВИКОНАННЯ

На початку семестру студент отримує індивідуальне завдання на курсову роботу, оформлене у відповідності до вимог, що висуваються до такого роду документів. Як правило, індивідуальне завдання представляє з себе аркуш паперу формату А4, на якому в стислій формі подаються вихідні дані для проведення роботи, окреслюється напрямок дослідження, визначаються числові значення необхідних параметрів та наводиться орієнтовний зміст роботи. Пакет індивідуальних завдань для студентів представляється на засіданні кафедри і візується завідувачем кафедри на початку семестру. При отриманні індивідуального завдання у відповідній графі бланку студент ставить свій підпис. Свій підпис у відповідній графі ставить і керівник курсової роботи.

Після отримання індивідуального завдання студент розробляє план роботи, який узгоджує з керівником. На базі розробленого плану формується зміст роботи, перелік розділів та додатків пояснювальної записки.

Після того, як буде сформовано зміст роботи студент приступає безпосередньо до виконання робіт окреслених розробленим планом та формування пояснювальної записки згідно представленого змісту. Формуючи пояснювальну записку необхідно дотримуватись вимог, що описані в «Положенні про курсове проектування у Вінницькому національному університеті» [1].

Написання курсової роботи передбачає вивчення літературних джерел і підбір ілюстративного матеріалу. В першу чергу доцільно звертатися до навчальних посібників, які в системному порядку викладають основний зміст курсу. Інформаційною базою для виконання курсової роботи є наукова література по вибраній темі дослідження; підручники і навчальні посібники, які в системному порядку викладають основні проблемні і актуальні питання в галузі розробки програмного забезпечення.

Особливу увагу слід приділити вивченню змісту основоположних теоретичних і практичних питань розробки та тестування програмного забезпечення. При вивченні монографій, журнальних статей, іншої спеціальної літератури з питань, що безпосередньо відносяться до теми курсової роботи, необхідно скласти конспект, викладаючи зміст своїми словами. Такий підхід дозволить забезпечити правильне розуміння вивченого матеріалу, а також дасть можливість самостійно викласти зміст курсової роботи. В якості ілюстративного матеріалу слід підбирати аналітичні таблиці, діаграми та графіки, схеми, алгоритми вирішення задач, схеми взаємозв’язку програмних модулів, тощо.

3 ЗМІСТ, СТРУКТУРА ТА ОБСЯГ

План курсової роботи студент розробляє самостійно на основі індивідуального завдання, методичних рекомендацій кафедри після огляду та обробки переліку рекомендованої літератури.

Незалежно від того, яким буде план виконання курсової роботи пояснювальна записка до курсової роботи повинна містити такі структурні елементи:

- титульний аркуш (додаток А);
- індивідуальне завдання (додаток Б);
- анотація (додаток В);
- зміст;
- вступ;
- основна частина;
- висновки;
- список використаних джерел;
- додатки.

3.1 Вимоги до анотації

В анотації наводиться коротка характеристика основного змісту курсової роботи, при цьому використовуються переважно прості синтаксичні конструкції і стандартизована термінологія в рамках наукового стилю мовлення, притаманні діловим документам.

Анотація повинна містити прізвище та ініціали автора, тему курсової роботи, стислий опис змісту та основних результатів, а також перелік ключових слів. Обсяг анотації (без теми і ключових слів) – 3-4 речення, але не більше 1/3 сторінки для анотації однією мовою (додатково анотація може подаватись і англійською мовою).

3.2 Вимоги до вступу

Вступ до курсової роботи є відповідальною частиною, в якій коротко викладають оцінку сучасного стану проблеми, відзначаючи практично розв’язані або ж нерозв’язані задачі, наукові підходи, що існують у науковому світі, провідних вчених і фахівців, світові тенденції розв’язання поставлених задач та обов’язково обґрунтування доцільності проведення наукових досліджень.

У вступі коротко розкривається актуальність теми, чітко формулюється мета дослідження і завдання, які треба розглянути, щоб досягти поставленої у курсовій роботі мети.

У вступній частині обов'язково необхідно:

– розкрити актуальність теми курсової роботи;
– розкрити ступінь розробленості теми курсової роботи у наукових працях вітчизняних і закордонних учених;
– чітко сформулювати мету та завдання дослідження;
– визначити об'єкт і предмет дослідження;
– описати основні методи дослідження;

Опис актуальності теми курсової роботи не повинен бути багатослівним, оскільки цьому передувала характеристика сучасного стану розвитку явищ, що стосуються курсової роботи.

Метою написання курсової роботи, як правило, є «закріплення отриманих знань на практиці …».

Завдання курсової роботи, сформульовані у вступі, обов'язково формуються за розділами роботи і мають відповідати задачам, поставленим науковим керівником в індивідуальному завданні.

Відповідно до мети дослідження ставляться такі завдання:

– проаналізувати …
– дослідити …
– оцінити …
– розробити …
– застосувати …

Об'єктом курсової роботи є процес створення програмного забезпечення згідно даних наведених в індивідуальному завданні.

Предметом роботи є певна частина об'єкту дослідження (алгоритм, модель взаємодії, тощо).

Методи курсової роботи можуть формулюватись як загальнонаукові (синтез, аналіз, …) так і специфічні.

3.3 Вимоги до основної частини роботи

Основну частину роботи рекомендується розбити на чотири етапи:

I аналіз існуючих варіантів;
II розробка структури програми;
III розробка програми;
IV тестування програми.

Кожен етап роботи доцільно (не обов’язково) оформити окремим розділом пояснювальної записки.

І Аналіз існуючих варіантів

Перший розділ курсової роботи має виконувати оглядово-аналітичну функцію та слугувати теоретичним підґрунтям для подальшого проєктування програмного рішення. У цьому розділі необхідно чітко сформулювати постановку задачі, визначити її функціональні та нефункціональні вимоги, а також окреслити обмеження предметної області. Аналіз має базуватися на сучасних підходах і практиках програмної інженерії. Особлива увага приділяється коректності термінології та узгодженості використаних понять. Виклад матеріалу повинен бути логічно структурованим і послідовним.

Обов’язковим елементом розділу є аналіз існуючих підходів, алгоритмів, бібліотек або програмних платформ, що можуть бути застосовані для розв’язання поставленої задачі. Для кожного розглянутого варіанту доцільно навести його архітектурні особливості, принципи роботи та умови застосування. Порівняння має здійснюватися за обґрунтованими критеріями, такими як продуктивність, масштабованість, складність реалізації та супроводжуваність коду. Аналіз не повинен зводитися до простого переліку рішень, а має відображати критичне осмислення їх переваг і недоліків.

Завершальною частиною розділу є узагальнення результатів аналізу та обґрунтування вибору підходу, який буде реалізовано в курсовій роботі. Обраний варіант має відповідати сформульованим вимогам і бути технічно доцільним з огляду на рівень підготовки студента та рамки завдання. У висновках слід чітко показати зв’язок між аналізом існуючих рішень і подальшими етапами розробки програмного забезпечення. Текст розділу повинен бути стислим, інформативним і орієнтованим на демонстрацію розуміння сучасних методів програмування.

Результати аналізу, виконаного в першому розділі, доцільно подавати в структурованому вигляді з використанням порівняльних таблиць, діаграм та графіків. Такий спосіб подання інформації забезпечує наочність, спрощує зіставлення альтернативних рішень і полегшує інтерпретацію отриманих даних. Візуальні засоби дозволяють чітко відобразити ключові відмінності між підходами за визначеними критеріями оцінювання. Їх використання сприяє підвищенню аналітичної якості розділу та обґрунтованості подальших проєктних рішень.

ІІ Розробка структури програми

Другий розділ курсової роботи має бути присвячений проєктуванню структури програмного забезпечення та опису логіки його функціонування. У цьому розділі необхідно визначити загальну архітектуру програми, обґрунтувати вибір структурної або модульної організації та окреслити основні компоненти системи. Опис має відображати взаємозв'язки між елементами програми та послідовність виконання основних операцій. Чіткість і формалізованість викладу є обов'язковими вимогами до цього розділу.

Перед переходом до етапу реалізації програмного коду доцільно розробити детальну структурну модель розв'язання задачі. Залежно від характеру роботи це може бути формалізований алгоритм, ієрархія програмних модулів або їх комбінація. Такий підхід дозволяє зменшити кількість логічних помилок на етапі кодування та спростити подальше тестування. Розробка структури програми розглядається як ключовий етап життєвого циклу програмного забезпечення.

У розділі необхідно описати призначення кожного структурного елемента програми, його вхідні та вихідні дані, а також відповідальність у межах загальної логіки роботи. Особлива увага приділяється визначенню інтерфейсів взаємодії між модулями та способам передавання даних. Такий опис сприяє забезпеченню модульності, повторного використання коду та зниженню зв'язності компонентів. Усі рішення мають бути логічно обґрунтованими та узгодженими між собою.

Важливим аспектом другого розділу є використання графічних і формалізованих засобів подання структури програми. Доцільним є застосування блок-схем алгоритмів, структурних схем, діаграм потоків даних або UML-діаграм. Такі засоби дозволяють наочно представити логіку роботи програми та взаємодію її компонентів. Візуальні моделі повинні доповнювати текстовий опис, а не замінювати його.

Завершуючи розділ, необхідно показати, що розроблена структура програми є достатньо повною та готовою до етапу реалізації. Описана архітектура має забезпечувати коректну реалізацію функціональних вимог, визначених у першому розділі. Чітко сформована структура програми слугує основою для ефективного кодування, тестування та подальшого супроводу програмного продукту. Це підкреслює принципову важливість етапу проєктування в процесі розробки програмного забезпечення.

ІІІ Розробка програми

Третій розділ курсової роботи має бути присвячений безпосередній реалізації програмного забезпечення відповідно до структури, розробленої на попередньому етапі. У цьому розділі необхідно описати процес розробки програми, дотримуючись принципів програмної інженерії та обраної архітектури. Реалізація має логічно відповідати спроєктованим алгоритмам і модульній організації. Виклад матеріалу повинен бути послідовним і технічно обґрунтованим.

Важливою вимогою є обґрунтований вибір інструментальних засобів розробки програмного забезпечення. У роботі необхідно вказати обрану мову програмування та пояснити її доцільність з огляду на специфіку задачі, продуктивність і доступні парадигми програмування. Також слід описати використане середовище розробки та його функціональні можливості, що сприяють підвищенню ефективності кодування. Усі інструменти мають бути сучасними, стабільними та відповідати поставленим вимогам.

Окрему увагу слід приділити вибору та використанню програмних бібліотек, фреймворків і сторонніх модулів. Необхідно обґрунтувати їх застосування, описати основні функції та роль у реалізації програмного рішення. Використання бібліотек повинно зменшувати складність реалізації, підвищувати надійність коду та забезпечувати повторне використання типових рішень. Недопустимим є застосування сторонніх компонентів без розуміння принципів їх роботи.

Перед початком реалізації програми необхідно сформулювати чіткі вимоги до програмного продукту, що розробляється. Ці вимоги мають охоплювати функціональні можливості, обмеження, умови експлуатації та очікувані результати роботи програми. Доцільно оформити їх у вигляді технічного завдання, яке слугуватиме формалізованою основою для розробки. Технічне завдання забезпечує узгодженість між постановкою задачі та її програмною реалізацією.

У розділі слід описати структуру програмного коду, принципи іменування, організацію файлів і модулів, а також підходи до обробки помилок і виняткових ситуацій. Особлива увага приділяється дотриманню стандартів кодування та рекомендацій щодо стилю програмування. Такий підхід підвищує читабельність, супроводжуваність і масштабованість програмного забезпечення. Наведені фрагменти коду мають бути репрезентативними та супроводжуватися поясненнями.

Завершальна частина третього розділу має демонструвати готовність програми до тестування та подальшої експлуатації. Необхідно показати, що реалізоване програмне рішення повністю відповідає вимогам технічно-

го завдання та проєктній структурі. Опис реалізації повинен підтверджувати коректність прийнятих інженерних рішень. Це дозволяє розглядати розроблену програму як завершений і цілісний програмний продукт.

IV Тестування програми

Четвертий розділ курсової роботи має бути присвячений питанням тестування програмного забезпечення як обов'язковому етапу його розробки. У цьому розділі необхідно обґрунтувати роль тестування в забезпеченні коректності, надійності та відповідності програми сформульованим вимогам. Тестування розглядається як невід'ємна складова життєвого циклу програмного продукту, а не як допоміжна або формальна процедура. Опис має бути логічно пов'язаний із технічним завданням і реалізованою програмою.

У межах курсової роботи доцільно застосовувати базові методи тестування, зокрема модульне, інтеграційне та функціональне тестування. Модульне тестування спрямоване на перевірку коректності роботи окремих функцій або програмних модулів. Інтеграційне тестування дозволяє оцінити правильність взаємодії між компонентами програми. Функціональне тестування використовується для перевірки відповідності реалізованого програмного продукту функціональним вимогам технічного завдання.

Окрему увагу слід приділити тестуванню граничних випадків, обробці помилкових або некоректних вхідних даних, а також перевірці стійкості програми до збоїв. За можливості можуть бути застосовані елементи регресійного або автоматизованого тестування з використанням стандартних засобів обраної мови програмування. Вибір методів тестування має бути обґрунтованим і відповідати складності програмного рішення. Недоцільним є формальне перерахування тестів без аналізу їх результатів.

Результати тестування необхідно оформлювати у структурованому та наочному вигляді. Доцільно використовувати таблиці з описом тестових сценаріїв, вхідних даних, очікуваних і фактичних результатів. За потреби можуть наводитися скріншоти, логи виконання або повідомлення про помилки. У висновках до розділу слід узагальнити результати тестування та зробити висновок щодо коректності й готовності програмного продукту до експлуатації.

3.4 Вимоги до висновків

Висновки є завершальною структурною частиною курсової роботи та підсумовують результати виконаного дослідження й розробки. У цьому розділі необхідно стисло узагальнити основні положення роботи без детального повторення змісту попередніх розділів. Виклад має бути лаконічним, логічно завершеним і орієнтованим на підбиття підсумків.

У висновках слід подати тезовий виклад основних результатів, отриманих у процесі аналізу, проєктування, реалізації та тестування програмного забезпечення. Необхідно відобразити ключові інженерні рішення, застосовані методи та використані інструментальні засоби. Усі твердження мають базуватися на фактичних результатах виконаної роботи.

Обов'язковим є встановлення відповідності між поставленими у вступі завданнями та досягнутими результатами. У висновках має бути чітко показано ступінь вирішення кожного із сформульованих завдань. Це дозволяє об'єктивно оцінити повноту та коректність виконання курсової роботи.

За потреби у висновках можуть бути окреслені напрями подальшого вдосконалення програмного продукту або розширення його функціональних можливостей. Такі зауваження мають носити узагальнений характер і не виходити за межі тематики роботи. Загалом висновки повинні створювати цілісне уявлення про результати та практичну цінність виконаної курсової роботи.

3.5 Вимоги до додатків

Додатки є допоміжною, але обов'язковою частиною курсової роботи, призначеною для розміщення матеріалів, які за обсягом або форматом недоцільно включати до основного тексту. Вони забезпечують повноту подання результатів роботи без перевантаження основних розділів. Наявність додатків підвищує наочність і технічну обґрунтованість виконаного дослідження.

У додатках доцільно розміщувати розширені та багатосторінкові документи, зокрема технічне завдання, оформлене як окремий структурований документ із титульною сторінкою. Такі матеріали слугують формалізованою основою для розробки програмного забезпечення. Винос технічного завдання в додатки дозволяє зберегти логічну цілісність основного тексту роботи.

Як правило, до додатків включають схеми роботи програми, алгоритмічні блок-схеми, структурні та функціональні діаграми. Також доцільним є розміщення UML-діаграм, схем взаємодії модулів і потоків даних. Ці матеріали доповнюють опис структури та логіки програмного продукту.

Крім того, в додатки можуть бути винесені фрагменти лістингу програмного коду, результати роботи програми у вигляді скріншотів, тестові дані та інші допоміжні матеріали. Оформлення додатків має відповідати загальним вимогам до науково-технічної документації. Усі додатки повинні мати нумерацію, а в основному тексті роботи мають бути посилання на кожен.

4 ВИМОГИ ДО ОФОРМЛЕННЯ

Пояснювальна записка до курсової роботи виконується згідно ДСТУ 3008:2015 [2]. Мова курсової роботи державна, стиль науковий, чіткий, без орфографічних і синтаксичних помилок; послідовність логічна.

4.1 Текст

Текст курсової роботи друкується на комп'ютері з одного боку стандартного аркуша односортного паперу формату А4 (210×297 мм). Гарнітура Times New Roman, розмір шрифту 14 пунктів, інтервал 1,5 (≈ 28-30 рядків на сторінку). При написанні дотримуються наступних розмірів полів: верхній, лівий і нижній – не менше 20 мм, правий – не менше 10 мм. Абзаци в тексті починають відступом, що дорівнює 1,27 см

Під час виконання курсової роботи необхідно дотримуватись рівномірної щільності, контрастності й чіткості зображення. Всі лінії, літери, цифри і знаки повинні бути чіткими та однаково чорними впродовж усієї роботи.

Номера сторінок слід проставляти арабськими цифрами у правому верхньому куті аркушу без крапки в кінці, дотримуючись наскрізної нумерації впродовж усього тексту роботи. Титульний аркуш включають до загальної нумерації сторінок роботи, проте номер сторінки на титульному аркуші не проставляють.

Заголовки структурних частин (розділів) курсової роботи пишуть великими літерами симетрично до тексту, крапка в кінці заголовку не ставиться. Переноси частини слів в заголовку не допускаються, на інший рядок слово переноситься повністю. Якщо заголовок складається з двох речень, то вони розділяються крапкою. Кожний наступний розділ роботи починають з нової сторінки. Розділи нумеруються арабськими цифрами в межах всієї курсової роботи, проте розділам «ЗМІСТ», «ВСТУП», «ВИСНОВКИ», «СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ» номера не присвоюють. Крапка після цифри не проставляється. Заголовки розділів відбивають знизу від основного тексту порожнім рядком або інтервалом в 1-1,5 розміру основного шрифту.

Заголовки підрозділів пишуться малими літерами окрім першої і розміщуються з абзацу. Переноси частини слів в підзаголовку не допускаються, на інший рядок слово переноситься повністю. Якщо підзаголовок складається з двох речень, то вони розділяються крапкою. Не допускається

розміщувати назву підрозділу, а також пункту й підпункту в нижній частині сторінки, якщо після неї розміщено тільки один рядок тексту. Підрозділи нумерують арабськими цифрами в межах розділу («1.1 Перший підрозділ першого розділу», «2.3 Третій підрозділ другого розділу»), крапку після останньої цифри не проставляють. Заголовки підрозділів відбивають знизу і згори від основного тексту порожнім рядком або інтервалом в 1-1,5 розміру основного шрифту.

4.2 Формули

Формули, що входять до курсової роботи, нумерують в межах розділу. Номер формули складається з номера розділу та порядкового номера формули, розділених крапкою. Номер формули розташовують з правого боку на рівні формули в круглих дужках. Посилання в тексті на номер формули дають в дужках, наприклад, «… за формулою (4.1)».

$$e^{i\pi} + 1 = 0, \tag{4.1}$$

де e – математична константа, основа натурального логарифма;
i – уявна одиниця;
π – фундаментальна математична константа.

Пояснення символів та числових коефіцієнтів наводять під формулою. Пояснення кожного символу подається з нового рядка в тій послідовності, в якій символи зустрічаються в формулі. Перший рядок пояснення починається зі слова «де» без двокрапки після нього.

Формули, що записані одна за одною та не розділені текстом, розділяються комою. Рівняння і формули необхідно виділяти з тексту в окремий рядок. Формули відбивають знизу і згори від основного тексту порожнім рядком або інтервалом в 1-1,5 розміру основного шрифту.

4.3 Таблиці і рисунки

Ілюстративні матеріали (таблиці і рисунки) розміщуються в тексті пояснювальної записки до курсової роботи або виносяться в додатки. Ілюстрація має розташовуватись одразу після посилання на неї в тексті («… дивись рис. 4.1»), або на наступній сторінці, якщо для розміщення її на поточній сторінці не вистачає місця.

Всі ілюстрації нумеруються арабськими цифрами в межах розділу і повинні мати назву. Номер ілюстрації складається з номера розділу та порядкового номера ілюстрації, розділених крапкою, а назва ілюстрації подається після номеру і відділяється від нього знаком «тире», наприклад, «Рисунок 4.1 – Діаграма класів», «Таблиця 4.1 – Результати комп'ютерного моделювання». Крапка в кінці заголовка ілюстрації не ставиться.

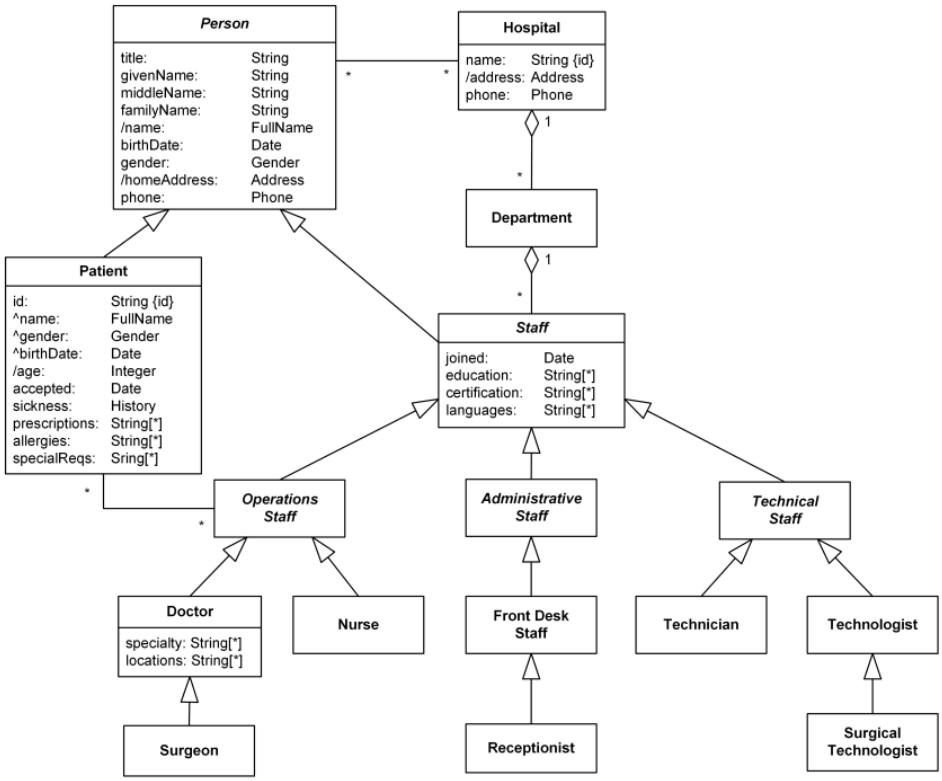

Рисунок 4.1 – Діаграма класів

Рисунки підписують знизу симетрично до тексту і відбивають від основного текту порожнім рядком або інтервалом в 1-1,5 розміру основного шрифту.

Таблиці підписують згори вирівнюючи назву по лівому краю таблиці і відбивають від основного тексту порожнім рядком або інтервалом в 1-1,5 розміру основного шрифту.

У разі перенесення частини таблиці на інший аркуш (сторінку) слово «Таблиця» та її номер вказують лише один раз – ліворуч над першою частиною таблиці; над іншими частинами пишуть «Продовження табл.» із зазначенням номера таблиці, наприклад: «Продовження табл. 4.1».

Таблиця 4.1 – Порівняльна характеристика методів пошуку

Метод	Гарантія оптимальності	Складність реалізації	Швидкодія
Повний перебір	100 %	проста	низька
Динамічний	100 %	середня	висока
Жадібний	не гарантована	проста	надвисока
Евристичний	не гарантована	висока	висока

4.4 Лістинги

Лістинги програмного коду, що наводяться в основному тексті курсової роботи, мають виконувати ілюстративну та пояснювальну функцію. Вони повинні відображати ключові фрагменти реалізації, без яких неможливо зрозуміти принципи роботи програми. Недоцільним є включення до тексту великих обсягів коду, що не мають безпосереднього аналітичного значення.

Розмір лістингу, розміщеного в основному тексті роботи, не повинен перевищувати 3/4 сторінки. Об'ємні фрагменти програмного коду слід виносити в додатки із відповідним посиланням у тексті. Такий підхід забезпечує зручність сприйняття матеріалу та збереження логічної структури документа. Можливий варіант оформлення лістингу в тексті роботи приведений нижче.

```
private String LoadData(String sourceName, int seconds) {
        try {
            TimeUnit.SECONDS.sleep(seconds);
        } catch (InterruptedException e) {
            e.printStackTrace();
        }
        return sourceName + '' завантажено за '' + seconds +
         ↪  '' сек.'';
}
```

Усі лістинги мають бути оформлені з використанням шрифту, призначеного для написання програмного коду, з фіксованою шириною символів. Код повинен чітко відрізнятися від основного тексту та бути візуально

структурованим. Для цього лістинги необхідно відокремлювати від тексту інтервалами або порожніми рядками до та після фрагмента коду.

Кожен лістинг має супроводжуватися пояснювальним текстом і, за потреби, нумерацією та заголовком. У поясненнях слід коротко описувати призначення наведеного коду та його роль у загальній логіці програми. Оформлення лістингів повинно бути уніфікованим по всій роботі та відповідати загальним вимогам до технічної документації.

4.5 Додатки

Ілюстративний матеріал може бути оформлений у вигляді додатків. Додатки представляють з себе окремі розділи пояснювальної записки до курсової роботи, що розташовуються після переліку посилань. Як і будь-який розділ додатки мають відображатись в змісті пояснювальної записки і мати наскрізну нумерацію сторінок.

На відміну від звичайних розділів заголовок додатку записують маленькими літерами окрім першої і позначають великими літерами української абетки, починаючи з А, за винятком літер Ґ, Є, З, І, Ї, Й, О, Ч, Ь, наприклад, «Додаток А». Заголовок додатку розташовують симетрично відносно тексту окремим рядком. Кожний наступний додаток починають з нової сторінки.

Текст кожного додатка, за необхідності, може бути поділений на підрозділи, пронумеровані в межах кожного додатка: перед кожним номером ставлять позначення додатка (літеру) і крапку, наприклад: «А.2» (другий підрозділ додатка А). Рисунки, таблиці та формули, розміщені в додатках, нумерують у межах кожного додатка, наприклад: «Рисунок Д.2» (другий рисунок додатка Д).

4.6 Список використаних джерел

Список використаних джерел повинен мати суцільну нумерацію. Використані джерела можна розміщувати в один з таких способів: за абеткою (за першою літерою прізвища автора або першого слова заголовка) або в порядку розташування посилань у тексті. В курсових роботах рекомендується використовувати другий спосіб.

Оформлення літературних джерел здійснюється відповідно до ДСТУ 8302:2015 [3].

5 АКАДЕМІЧНА ДОБРОЧЕСНІСТЬ

Законами України «Про вищу освіту», «Про освіту» [4], академічну доброчесність визначено як «сукупність етичних принципів та правил, якими мають керуватися учасники освітнього процесу під час навчання, викладання та провадження наукової (творчої) діяльності з метою забезпечення довіри до результатів навчання та/або наукових (творчих) досягнень».

5.1 Основні поняття

Академічна відповідальність здобувача та керівника кваліфікаційної роботи передбачається за наведені нижче порушення.

Академічний плагіат:

- плагіат фрагментів письмових робіт та повних текстів;
- плагіат ідей, даних, моделей, ілюстрацій тощо;
- відсутність належних посилань;
- помилки цитування.

Самоплагіат:

- дуплікація публікацій – публікація однієї і тієї самої наукової роботи (цілком або з несуттєвими змінами) в декількох виданнях, а також повторна публікація (цілком або з несуттєвими змінами) раніше оприлюднених статей, монографій, інших наукових робіт як нових наукових робіт;
- дуплікація наукових результатів – публікація повністю чи частково одних і тих самих наукових результатів у різних статтях, монографіях, інших наукових працях як нових результатів, які публікуються вперше;
- подання у звітах із виконання різних наукових проєктів тих самих результатів як таких, що отримані при виконанні відповідного проєкту;
- повторне подання здобувачами освіти письмових робіт, які вже подавалися як звітність із інших дисциплін, без дозволу викладача;
- агрегування чи доповнення даних – суміщення раніше опублікованих і нових даних без їх поділу з відповідними посиланнями на попередню публікацію;

– повторний аналіз раніше опублікованих даних без посилання на попередню публікацію цих даних та раніше виконаного їх аналізу.

Фабрикація:

– наведення у письмових роботах здобувачів та в наукових роботах вигаданих чи неперевірених даних, зокрема статистичних даних, результатів експериментів, розрахунків чи емпіричних досліджень, фотографій, аудіо- та відеоматеріалів тощо;
– посилання на вигадані джерела інформації або навмисне посилання не на справжнє джерело;
– приписування іншим особам текстів, думок чи ідей, яких вони не висловлювали чи не публікували.

Фальсифікація:

– необґрунтоване корегування результатів власних наукових досліджень чи виконання навчальних завдань (таке, що не базується на повторних чи додаткових дослідженнях, вимірюваннях або розрахунках, виправленні виявлених помилок тощо);
– наведення у письмових роботах здобувачів та в наукових роботах свідомо змінених літературних даних та даних, отриманих із інших джерел; зокрема, статистичних даних, результатів експериментів, розрахунків чи емпіричних досліджень, фотографій, аудіо- та відеоматеріалів тощо без належного обґрунтування причин і зазначення методики їх корегування;
– наведення неповної або викривленої інформації про апробацію результатів досліджень та розробок.

Обман:

– включення до співавторів наукових публікацій осіб, що не брали кваліфікованої участі в їх підготовці;
– невключення до співавторів наукових публікацій осіб, що брали кваліфіковану участь в їх підготовці;
– подання як результатів власної праці робіт, виконаних на замовлення іншими особами, або робіт, стосовно яких справжні автори не дали згоду на таке використання;
– здавання або представлення різними особами робіт з однаковим змістом як результату власної навчальної діяльності;
– написання чужих варіантів завдань на контрольних заходах;
– використання системи прихованих сигналів (звукових, жестових та ін.) при виконанні групових контрольних заходів з однаковими варіантами;

– несамостійне виконання завдань у випадках, коли не дозволяється отримання допомоги, або не зазначення інформації про отриману допомогу, консультації, співпрацю;
– проходження процедур контролю знань підставними особами;
– симуляція погіршення стану здоров'я, хвороби з метою уникнення контрольних заходів;
– надання відгуків або рецензій на наукові або навчальні роботи без належного проведення їх експертизи.

Необ'єктивне оцінювання:

– свідоме завищення або заниження оцінки результатів навчання здобувачів освіти;
– невчасне повідомлення здобувачів освіти про систему оцінювання результатів навчання;
– застосування системи оцінювання, що не відповідає декларованим цілям та завданням теми, дисципліни, практики, освітньої програми тощо;
– відсутність об'єктивних критеріїв оцінювання.

5.2 Основні види відповідальності

Основні види відповідальності педагогічних, науково-педагогічних та наукових працівників за порушення академічної доброчесності встановлює ч. 5 ст. 42 Закону України «Про освіту» [4] такі:

– відмова у присудженні наукового ступеня чи присвоєнні вченого звання;
– позбавлення присудженого наукового (освітньо-творчого) ступеня чи присвоєного вченого звання;
– відмова в присвоєнні або позбавлення присвоєного педагогічного звання, кваліфікаційної категорії;
– позбавлення права брати участь у роботі визначених законом органів чи займати визначені законом посади.

Ч. 6 ст. 42 того ж Закону визначає основні види відповідальності здобувачів освіти за порушення академічної доброчесності, якими є:

– повторне проходження оцінювання (контрольна робота, іспит, залік тощо);
– повторне проходження відповідного освітнього компонента освітньої програми;
– відрахування із закладу освіти (крім осіб, які здобувають загальну середню освіту);

– позбавлення академічної стипендії;
– позбавлення наданих закладом освіти пільг з оплати навчання.

Попередження плагіату в академічному середовищі університету здійснює Центр моніторингу якості освіти ВНТУ відповідно до «Положення про запобігання академічного плагіату та порядок його виявлення у наукових, кваліфікаційних, навчальних та науково-методичних роботах у Вінницькому національному технічному університеті» [5].

6 ПОРЯДОК ВИКОНАННЯ, ЗАХИСТУ І ОЦІНЮВАННЯ

Курсова робота (далі просто робота) вважається готовою до захисту перед екзаменаційною комісією (ЕК) після проходження всіх обов'язкових етапів підготовки та перевірки.

6.1 Порядок виконання

Здобувачі вищої освіти денної форми навчання обирають собі тему КР не пізніше першого тижня теоретичного семестру. Здобувачі вищої освіти заочної форми навчання обирають собі тему протягом сесії, під час якої проводяться лекційні заняття з відповідної навчальної дисципліни. Якщо здобувач вищої освіти не обрав собі тему, її призначає для нього керівник

На основі визначених тем КР керівник формує індивідуальні завдання. Тема та індивідуальне завдання на виконання КР видається керівником здобувачам вищої освіти не пізніше другого тижня теоретичного семестру. Для здобувачів освіти заочної форми навчання індивідуальне завдання видається до завершення сесії, під час якої проводяться лекційні заняття з відповідної навчальної дисципліни. Індивідуальне завдання підписується керівником та здобувачем вищої освіти із зазначенням дати його отримання. Також на індивідуальне завдання затверджує завідувач кафедри своїм підписом.

Упродовж теоретичного семестру керівник проводить періодичні консультації з питань виконання КР та контролює дотримання графіка виконання КР здобувачами вищої освіти. Здобувачі вищої освіти повинні виконувати індивідуальне завдання, дотримуючись встановленого графіка.

Пояснювальна записка до курсової роботи виконується у вигляді комп'ютерного файлу за допомогою довільного програмного текстового редактора, який здатний забезпечити виконання вимог до оформлення КР.

Якщо КР виконано у повному обсязі, у відповідно до індивідуального завдання, не містить ознак академічної недоброчесності, не містить суттєвих помилок, оформлена згідно з встановленими вимогами та надіслана здобувачем вищої освіти у вигляді одного файлу у форматі Portable Document Format (`*.pdf`), керівник приймає КР до захисту, про що повідомляє здобувача вищої освіти у відповіді на файл, надісланий через інструмент «Файл-Експрес» системи JetIQ.

Якщо остаточний файл КР виконано не в повному обсязі, або не у відповідності до індивідуального завдання, або він містить ознаки академічної недоброчесності, або суттєві помилки, або не оформлено згідно з встановленими вимогами, або файл не надійшов через інструмент «Файл-Експрес» системи JetIQ (станом на визначений у розкладі день захисту), така КР визнається керівником недопущеною до захисту із виставленням незадовільної оцінки у відомість успішності (від 0 до 59 балів). Якщо оцінка за стобальною шкалою склала від 35 до 59 балів включно, то здобувач вищої освіти має право на доопрацювання і захист КР з виставленням оцінки у другу відомість. Якщо оцінка за стобальною шкалою склала від 0 до 34 балів включно, то здобувач вищої освіти вважається таким, що має академічну заборгованість. Для її ліквідації здобувач повинен виконати КР за новою темою (або зміненим індивідуальним завданням).

Перевірку КР на наявність помилок у їх змісті та оформленні, нормоконтроль (за необхідності) здійснює керівник КР, а в разі його тимчасової відсутності – інший науково-педагогічний працівник відповідної кафедри, визначений розпорядженням завідувача кафедри.

6.2 Порядок захисту

Захист КР відбувається під час заліково-екзаменаційної сесії згідно з розкладом контрольних заходів, визначених деканатом.

До захисту допускаються роботи, які виконані з дотриманням вимог описаних в «Положенні про курсове проєктування у Вінницькому національному університеті» [1].

Якщо здобувач вищої освіти не з’явився на захист КР у зв’язку з поважною причиною (перешкода стихійного характеру, хвороба, воєнні дії на території перебування здобувача або інші обставини, які позбавили його можливості особисто і своєчасно прибути на захист), такому здобувачу вищої освіти надається можливість захистити КР в інший день за узгодженням з керівником, але не пізніше останнього дня заліково-екзаменаційної сесії. Про поважну причину неявки на захист здобувач вищої освіти повинен повідомити керівника і деканат відповідного факультету протягом доби від призначеного часу захисту.

Якщо здобувач вищої освіти не з’явився на захист КР без поважної причини, такий здобувач вищої освіти отримує незадовільну оцінку (від 0 до 59 балів за стобальною шкалою) з виставленням її у відомість успішності. Якщо оцінка за стобальною шкалою склала від 35 до 59 балів включно, то здобувач вищої освіти має право на доопрацювання і захист КР з ви-

ставленням оцінки у другу відомість. Якщо оцінка за стобальною шкалою склала від 0 до 34 балів включно, то здобувач вищої освіти вважається таким, що має академічну заборгованість. Для її ліквідації здобувач повинен виконати КР за новою темою (або зміненим індивідуальним завданням).

Захист КР повинен мати публічний характер і прийматися комісією, до складу якої входять не менше 2-х осіб з-поміж викладачів кафедри, один з яких є керівником КР. У разі тимчасової відсутності керівника в день захисту, його замінює інший науково-педагогічний працівник відповідної кафедри, визначений розпорядженням завідувача кафедри.

Процедура захисту включає коротку доповідь здобувача вищої освіти щодо основних результатів виконання КР та відповіді на запитання членів комісії або інших присутніх на захисті осіб.

За результатами захисту КР комісія на закритому засіданні визначає оцінку, яку керівник оголошує здобувачам вищої освіти в день захисту та виставляє у відомість успішності.

6.3 Оцінювання курсових робіт

Оцінювання якості виконання та захисту КР здійснюється згідно таких критеріїв:

- ступінь і якість виконання індивідуального завдання;
- ступінь і якість виконання основної частини КР;
- ступінь і якість виконання ілюстративної або графічної частини (в разі наявності);
- відповідність встановленим вимогам до змісту і оформлення КР;
- рівень і якість представлення результатів курсового проєктування здобувачем вищої освіти під час захисту;
- відповіді на запитання в процесі захисту КР.

Оцінювання КР здійснюється за стобальною шкалою та за шкалою ЄКСТ. Оцінка за стобальною шкалою та за шкалою ЄКТС виставляється керівником КР у відомість успішності в день проведення захистів КР. Також оцінка за стобальною шкалою та за шкалою ЄКТС проставляється керівником КР на титульному аркуші роботи.

Підсумкову оцінку захисту КР визначає комісія. Рішення комісії є остаточним.

Таблиця 6.1 – Критерії оцінювання

№	Орієнтовні критерії оцінювання	Кількість балів
1	Рівень виконання та відповідність КР індивідуальному завданню: – дотримання вимог та параметрів, сформульованих в завданні; – повнота обґрунтування аналізу предметної області; – рівень використання джерел інформації; – глибина обґрунтування застосування методів та засобів, які використовуються при виконанні КР; – точність та технічний рівень розрахункових матеріалів; – якість та технічний рівень проєктних рішень, представлених у графічній або ілюстративній частині; – обґрунтування прийнятих проєктних рішень, формулювання висновків та рекомендацій	до 50 балів
2	Відповідність оформлення КР встановленим вимогам	до 10 балів
3	Рівень представлення результатів КР на захисті: – якість доповіді; – якість та оформлення презентації, – якість відповідей на запитання	до 40 балів
	Максимальна оцінка	100 балів

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. Положення про курсове проєктування у Вінницькому національному технічному університеті. СУЯ ВНТУ-03-01-П.001.01:2024 / Д. Х. Штофель [та ін.]. 2024. URL: https://vntu.edu.ua/uploads/2024/StateofKurs.pdf (дата звернення: 23.12.2025).
2. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. ДСТУ 3008:2015. ДП «УкрНДНЦ» / В. Земцева [та ін.]. 2016. URL: https://science.kname.edu.ua/images/dok/derzhstandart_3008_2015.pdf (дата звернення: 23.12.2025).
3. Петрова Н., Плиса Г., Жигун Т. Бібліографічне посилання. Загальні положення та правила складання. ДСТУ 8302:2015. ДП «УкрНДНЦ». 2017. URL: https://pdf.lib.vntu.edu.ua/books/2018/%D0%94%D0%A1%D0%A2%D0%A3%208302%20%D0%BF%D0%BE%D0%B2%D0%BD%D0%B8%D0%B9.pdf (дата звернення: 23.12.2025).
4. Закон України «Про освіту» (редакція № 2145-VIII від 05.09.2017 р.). 2017. URL: https://zakon.rada.gov.ua/laws/show/2145-19#n1854 (дата звернення: 24.12.2025).
5. Положення про запобігання академічному плагіату та порядок його виявлення у наукових, кваліфікаційних, навчальних та науково-методичних роботах Вінницького національного технічного університету / С. Тужанський [та ін.]. 2024. URL: https://vntu.edu.ua/uploads/2024/Stateofplag.pdf (дата звернення: 29.12.2025).

Додатки

Додаток А
Зразок титульного аркуша

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра автоматизації та інтелектуальних інформаційних технологій

КУРСОВА РОБОТА
з дисципліни: «Програмування»
на тему: програма, що вирішує «задачу про рюкзак»

Виконав студент: II курсу 1ІСТ-24б групи спеціальності: F6 – Інформаційні системи і технології,
Анна КУДРІНСЬКА _____

Керівник: к.т.н., доцент каф. АІІТ
Костянтин ОВЧИННИКОВ

Кількість балів: _____ Оцінка: ECTS _____

Члени комісі: _____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

Вінниця ВНТУ – 2025 р.

Додаток Б
Зразок індивідуального завдання

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації

ЗАТВЕРДЖУЮ
зав. кафедри АІІТ, проф., д.т.н.
_____О. В. Бісікало
(підпис)
«____» вересня 2025 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу з дисципліни «Програмування»
студенту Кудрінська Анна Павлівна групи 1ІСТ-24б

Розробити програму, що вирішує «задачу про рюкзак».
Вимоги до розробки:

1. Передбачити роботу програми в графічному режимі;
2. Необхідний обсяг оперативної пам'яті не менше 1 ГБ.
3. _____

Примітка:

1. _____
2. _____
3. _____

Затверджено на засіданні кафедри 26 серпня 2025 р. протокол №1

Орієнтовний зміст ПЗ до курсової роботи:

Зміст
Анотація
Вступ
Огляд методів вирішення поставленої задачі
Розробка алгоритмічного забезпечення
Розробка програмного забезпечення
Висновки
Список використаних джерел
Додатки

Дата видачі «____» _____ 2025 р. керівник _____
(підпис)

Строк подання виконаної роботи «____» _____ 2025 р.

завдання отримав _____
(підпис)

Додаток В
Зразок анотації

АНОТАЦІЯ

Кудрінська А. П. програма, що вирішує «задачу про рюкзак» : курсова робота з дисципліни «Програмування».

У курсовій роботі проведена розробка програмного забезпечення для розв'язання класичної оптимізаційної задачі про «Рюкзак», яка широко застосовується в задачах планування ресурсів, логістики, економічного аналізу та систем підтримки прийняття рішень. Наведено огляд основних підходів до розв'язання задачі, проаналізовано їх переваги та недоліки, обґрунтовано вибір методу повного перебору як такого, що гарантує оптимальний результат. У роботі створено алгоритмічні засади розв'язання задачі та структуру програмного застосунку. Програму створено мовою Java, використанні бібліотеки графічного інтерфейсу для забезпечення зручності введення даних і зручності представлення результатів роботи програми.

Ключові слова: задача про рюкзак, оптимізація, повний перебір, алгоритм, програмне забезпечення, Java, графічний інтерфейс користувача.

Додаток Г
Завдання на курсову роботу

Задачі, рішення яких часто потребують науковці та спеціалісти певних галузей.

Захист інформації

1 Реалізація шифрування інформації вказаним шифром (3 способи);
2 Реалізація шифрування інформації вказаним шифром із використанням фреймворку Spring;
3 Реалізація генератора випадкових чисел та аналізатора випадковості;
4 Реалізація програми, яка перевіряє цифровий підпис;
5 Реалізація шифру Віженера та простої підстановки;
6 Реалізація геометричного шифру;

Різні задачі

7 Реалізація програми для балістичної експертизи;
8 Реалізація програми для відображення дерева версій Unix з можливістю додавання нових версій;
9 Реалізація запису структурованого тексту в базу даних з можливістю підтримки NoSQL;
10 Реалізація для Linux Ubuntu запису форматованого тексту в базу даних;
11 Реалізація будильника з використанням JBoss Seam;
12 Реалізація програми для розрахунку та відображення поточної фази місяця;
13 Реалізація мобільної програми для виведення інформації про речовину за введеним Е-кодом;
14 Реалізація програми, яка переводить цифровий код у штрих-код і визначає виробника товару;
15 Реалізація програми, яка переводить цифровий код у QR-код і визначає виробника товару;
16 Реалізація програми для малювання української вишивки, характерної для Вінниччини;
17 Реалізація програми для малювання української вишивки за способом «Бродівське письмо»;

Мобільні пристрої

18 Реалізація будильника для мобільного пристрою;
19 Реалізація програми для упорядковувала музики для мобільного пристрою;
20 Реалізація простого диспетчера задач для мобільного пристрою;
21 Реалізація програми для визначення поточного місця знаходження для мобільного пристрою;
22 Реалізація програми для визначення довжини маршруту від поточного місця знаходження для мобільного пристрою;

Соціальні мережі

23 Реалізація автоматичного відповідача у нічний час для соцмережі Facebook.
24 Реалізація програми для автоматичного написання відтермінованих повідомлень у заданий час для соцмережі Twitter.
25 Реалізація програми для автоматичного виставлення фотографій у заданий час для соцмережі Instagram.

Інтернет та локальна мережа

26 Реалізація автоматизованого перекладача з використанням Hibernate;
27 Реалізація програми для відстежування курсу валют із графічним відображенням коливань на основі Spring Web Flow (SWF);
28 Реалізація програми для автоматичної відправлення повідомлень по електронній пошті у вказаний час за певних умов;
29 Реалізація власного простого завантажувача сайту для автономного перегляду при відсутності доступу до мережі інтернет;
30 Реалізація програми для збереження картинок з веб-сайту на комп'ютер;
31 Реалізація програми для обміну повідомленнями (месенджера). Частина 1. Реалізація інтерфейсу користувача (GUI);
32 Реалізація програми для обміну повідомленнями (месенджера). Частина 2. Реалізація обміну текстовими повідомленнями через інтернет та локальну мережу;
33 Реалізація програми для обміну повідомленнями (месенджера). Частина 3. Реалізація обміну файлами через інтернет та локальну мережу;
34 Реалізація програми для обміну повідомленнями (месенджера). Частина 4. Реалізація збереження історії повідомлень у базі даних із можливістю пошуку;

35 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 1. Реалізація інтерфейсу користувача (GUI);
36 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 2. Реалізація простого torrent-клієнта;
37 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 3. Реалізація простого torrent-сервера (для роздачі);
38 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 4. Реалізація програми для оптимізації швидкості при нестабільній та різній швидкості з'єднання у користувачів;
39 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 5. Реалізація програми для оптимізації часу доставки та маршруту проходження пакетів;
40 Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 6. Реалізація програми для захищеного передавання (криптографічне шифрування пакетів та аутентифікація користувачів);

Веб-технології

41 Створення інформаційного веб-сайту на HTML5;
42 Створення інформаційного веб-сайту з можливістю перегляду фотографій на DHTML;
43 Створення веб-сайту-форуму із БД DB/2 на HTML не нижче 4.0;
44 Створення інформаційного веб-сайту, який здатен працювати з БД;
45 Створення інформаційного веб-сайту із можливістю перегляду відео на JavaFX;
46 Створення веб-сайту для музиканта з можливістю керувати доступом відтворенням композицій залежно від рейтингу користувача на сайті;
47 Створення веб-сайту для перегляду інформації про певний товар зі списку із використанням JavaServer Faces (JSF);
48 Створення веб-сайту, який доступний на відкритому сервері та зберігає інформацію, захищену від хакерських атак (будь-хто зі групи має можливість спробувати викрасти цю інформацію);
49 Створення веб-сайту для масової гри у «хрестики-нулики» онлайн на скриптах (PHP/JavaScript/JScript/ін.);
50 Створення веб-сайту для масової гри у «шашки» онлайн із БД на Google Web Toolkit;

51 Створення веб-сайту для масової гри у «морський бій» онлайн. Частина 1. Створення веб-сайту із авторизацією користувачів;
52 Створення веб-сайту для масової гри у «морський бій» онлайн. Частина 2. Створення графічного інтерфейсу гри у «морський бій»;
53 Створення веб-сайту для масової гри у «морський бій» онлайн. Частина 3. Створення штучного інтелекту для гри у «морський бій»;
54 Створення веб-сайту для масової гри у «морський бій» онлайн. Частина 4. Створення інтерфейсу та функцій для ведення статистики на сайті та записом у БД;

Ігри

55 Реалізація гри «морський бій» із автоматичним встановленням програми. Частина 1. Реалізація інтерфейсу із авторизацією користувачів та можливістю обміну повідомленнями по локальній мережі;
56 Реалізація гри «морський бій» із автоматичним встановленням програми. Частина 2. Реалізація графічного інтерфейсу гри у «морський бій»;
57 Реалізація гри «морський бій» із автоматичним встановленням програми. Частина 3. Реалізація штучного інтелекту для гри у «морський бій»;
58 Реалізація гри «морський бій» із автоматичним встановленням програми. Частина 4. Реалізація ведення статистики та запису у БД Cloudscape;
59 Реалізація гри «морський бій» із автоматичним встановленням програми. Частина 5. Реалізація програми встановлення гри із визначенням платформи та перевірки можливості встановлення гри;
60 Реалізація гри «пятнашки» для мобільного пристрою;
61 Реалізація гри «хрестики-нулики» для мобільного пристрою на Android;
62 Реалізація гри «монополія»;
63 Реалізація програми для гри у пінг-понг по локальній мережі. Частина 1. Реалізація графічного інтерфейсу та штучного інтелекту способом 1;
64 Реалізація програми для гри у пінг-понг по локальній мережі. Частина 2. Реалізація з'єднання по Internet та штучного інтелекту способом 2;
65 Реалізація гри «лучник». Частина 1. Реалізація інтерфейсу користувача (GUI);
66 Реалізація гри «лучник». Частина 2. Реалізація модуля для прорахування траєкторії стріли залежно від кута, сили запуску, напряму та сили вітру;
67 Реалізація гри «лучник». Частина 3. Реалізація модуля для створення перешкод (непробивні блоки, відбивачі, важелі, механізми);

68 Реалізація гри «лучник». Частина 4. Реалізація набору рівнів гри та етапу навчання;
69 Реалізація гри «мушкетер». Частина 1. Реалізація інтерфейсу користувача (GUI);
70 Реалізація гри «мушкетер». Частина 2. Реалізація модуля для прорахування траєкторії кулі залежно від кута, кількості пороху, напряму та сили вітру;
71 Реалізація гри «мушкетер». Частина 3. Реалізація модуля властивостей матеріалів поверхонь(міцність, крихкість, можливість рикошету та роздроблення) та розрахунок імпульсу кулі до і після влучання;
72 Реалізація гри «мушкетер». Частина 4. Реалізація модуля штучного інтелекту для ворогів (різних типів);
73 Реалізація гри «мушкетер». Частина 5. Реалізація модуля вдосконалень для зброї гравця та ворогів;
74 Реалізація програми з віртуальною реальність (VR);
75 Реалізація програми з доповненою реальність (AR);
76 Реалізація програми з доповненою реальність (AR) з допомогою Unity 3D;
77 Реалізація шпигунської програми, замасковану під гру з доповненою реальність (AR);
78 Реалізація гри «симулятор каменю»;
79 Реалізація гри «лабіринт»;
80 Реалізація гри на основі ігрового рушія MineCraft.

Детальний опис завдань

1. Реалізація шифрування інформації вказаним шифром (3 способи).
Програма повинна реалізувати 3 різні способи шифрування тексту довільної довжини (наприклад 3-DES, AES, ДСТУ 28147:2009), генерувати ключі (відкриті або закриті), які не будуть відомі нікому окрім тих, хто читає повідомлення (для відкритого ключа це не строга умова). Шифр повинен мати крім достатньої стійкості також додаткові способи захисту (наприклад-додавання «криптографічної солі (salt)»). Реалізовувати самі шифри не потрібно.
Вхідні дані: незашифрований текст.
Вихідні дані: зашифрований текст.
Складність: середня.
Примітки: **3**

2. Реалізація шифрування інформації вказаним шифром із використанням фреймворку Spring.
Програма повинна реалізувати 3 різні способи шифрування тексту довільної довжини, генерувати ключі (відкриті або закриті), які не будуть відомі нікому окрім тих, хто читає повідомлення (для відкритого ключа це не строга умова). Шифр повинен мати крім достатньої стійкості також додаткові способи захисту (наприклад-додавання «криптографічної солі (salt)»). Пропонуються на вибір для використання шифри 3-DES, AES, ДСТУ ГОСТ 28147:2009, IDEA, Twofish, BlowFish, TreeFish, SAFER, SHARK, FEAL, KASUMI, RC5, TEA, CRYPTON, Hierocrypt-3, Serpent, Camellia, CAST-256, MARS, RC6. Реалізовувати самі шифри не потрібно. Обов'язкове використання для реалізації програми каркасу Spring.
Вхідні дані: незашифрований текст.
Вихідні дані: зашифрований текст.
Складність: середня.
Примітки: **З**, **Т**.

3. Реалізація генератора випадкових чисел та аналізатора випадковості.
Програма повинна реалізувати 5 різних способів генерації випадкових чисел. Оцінка випадковості виконується або теоретично, або шляхом статистичної обробки (знаходження математичного сподівання, дисперсії, моди, медіани) та кількості повторюваних чисел у достатньо великій послідовності. Імовірність повтору-відношення кількості повторів на кількість чисел. За необхідності викладач надає спеціаліста, який пояснює статистичної обробки та випадкових чисел. Студент реалізує програму генератора випадкових чисел з оцінкою їх якості.
Вхідні дані: ступінь випадковості, статистичні параметри.
Вихідні дані: послідовність випадкових чисел, оцінка випадковості.
Складність: середня.
Примітки: **З**, **М**, **С**.

4. Реалізація програми, яка перевіряє цифровий підпис.
Реалізувати програму, яка буде перевіряти наявність цифрового підпису у деякої програми, його дійсність та відповідність реальному, і на основі цих даних робитиме висновок про ступінь довіри до цієї програми (довірена, підозріла, не довірена) та пояснюватиме причину.
Вхідні дані: програма, яка потребує перевірки цифрового підпису.
Вихідні дані: стан цифрового підпису і оцінка довіри програмі.
Складність: середня.
Примітки: **З**.

5. Реалізація шифру Віженера та простої підстановки.
Програма повинна реалізувати шифрування тексту довільної довжини шифром Віженера та простої підстановки. Шифр повинен мати крім достатньої стійкості також додаткові способи захисту (наприклад-додавання «криптографічної солі (salt)», перевірка користувача, затримку введення в часі).
Вхідні дані: незашифрований текст.
Вихідні дані: зашифрований текст.
Складність: низька.
Примітки: **З**.

6. Реалізація геометричного шифру.
Програма повинна реалізувати шифрування тексту довільної довжини геометричним шифром. Літери повідомлення виписуються у вигляді шестикутника, а записуються у порядку ейлеревого шляху. Дешифрування відбувається навпаки. Шифр повинен мати також додаткові способи захисту (наприклад-додавання «криптографічної солі (salt)», перевірка користувача, затримку введення в часі).
Вхідні дані: незашифрований текст.
Вихідні дані: зашифрований текст.
Складність: середня.
Примітки: **З**, **М**.

7. Реалізація програми для балістичної експертизи.
Балістична експертиза полягає у знаходженні місця, напряму, кута, початкової швидкості руху твердого предмета (наприклад кулі) по відомому куту та глибині проникнення, масі цього предмета у матеріалі (механічна міцність, в'язкість цього матеріалу також відомі). Програма повинна за заданими вхідними даними визначити всі початкові дані. Відтворення польоту тіла за відомими даними необхідно показати графічно.
Вхідні дані: глибина проникнення, маса предмета, матеріал.
Вихідні дані: місце, напрям, кут, початкова швидкість руху предмета.
Складність: середня.
Примітки: **М**, **С**.

8. Реалізація програми для відображення дерева версій Unix з можливістю додавання нових версій.
Програма повинна показати схематично повністю або частково гілки розвитку системи Unix із класифікацією за поколінням, системою та роком випуску. Програма за бажанням користувача повинна показати частину

всього дерева із версіями, створеними за вказаний період. Повинна бути можливість додавання нових версій користувачем.
Вхідні дані: існуючі версії Unix, нові версії Unix.
Вихідні дані: дерево версій Unix.
Складність: середня.
Примітки: **Г**.

9. Реалізація запису структурованого тексту в базу даних з можливістю підтримки NoSQL.
Програма повинна записувати із файлу текст, який має певну структуру (список) зі збереженням цієї структури у базу даних типу не SQL (або SQL-подібну), і при потребі витягати текст із цієї бази даних у новий текстовий файл зі збереженням структури. База даних та технологія запису у неї повинна уточнюватись на етапі технічного завдання. Дозволяється використати відкритий код Jasper Report з проекту Jasper Studio.
Вхідні дані: структурований текст у файлі.
Вихідні дані: запис у базі даних типу NoSQL.
Складність: висока.
Примітки: **Б**, **Т**.

10. Реалізація для Linux Ubuntu запису форматованого тексту в базу даних.
Програма повинна записувати із файлу текст зі збереженням форматування у базу даних, і при потребі витягати текст із цієї бази даних у новий текстовий файл зі збереженням форматування. Дозволяється використовувати не лише Ubuntu, а й іншу систему Linux/Unix. Якщо даної ОС немає у розробника, для установки емулятора можна використати програму VirtualBox, всередині якої буде запущена сама програма.
Вхідні дані: форматований текст у файлі.
Вихідні дані: запис у базі даних.
Складність: середня.
Примітки: **Б**, **Т**.

11. Реалізація будильника з використанням JBoss Seam.
Програма повинна реалізовувати будильник, який видаватиме звуковий сигнал (можна разом із графічним), поки користувач не скасує цей сигнал, або не пройде 5 хвилин від старту будильника. Дія вимкнення будильника має бути не дуже очевидна, щоб її не можна було миттєво зрозуміти, як вимкнути-наприклад через математичний приклад. Користувач повинен мати можливість поставити затримку на 5 хвилин, змінити звуковий сигнал на свій та періодично його змінювати. Обов’язкове використання для

реалізації програми каркасу JBoss Seam.
Вхідні дані: час запуску будильника, звуковий файл користувача.
Складність: висока.
Примітки: **Т**.

12. Реалізація програми для розрахунку та відображення поточної фази місяця.
Програма повинна реалізовувати розрахунок фази місяця залежно від заданої дати та часу і зображувати графічно. Також у програму може бути додана опція врахування географічного положення.
Вхідні дані: бажана дата та час.
Вихідні дані: рисунок фази місяця.
Складність: середня.
Примітки: **М**, **Г**.

13. Реалізація мобільної програми для виведення інформації про речовину за введеним Е-кодом.
Програма повинна реалізовувати на мобільному пристрої пошук харчової добавки за введеним Е-кодом та виводити інформацію про неї: назва речовини, ступінь шкідливості, чи є заборонена/не рекомендована в Україні, США, ЄС, галузь використання, призначення, основні властивості.
Вхідні дані: номер Е-коду.
Вихідні дані: опис речовини.
Приклад: Е621 – Глутамат натрію однозаміщений, Monosodium Glutamate; підсилювач смаку і аромату; схвалений в ЄС, Україні. Е330 - Лимонна кислота, Citric acid; додається за технологічною необхідністю, в тому числі в какао, шоколадні вироби; схвалений в ЄС, Україні. Е1510 - Етиловий спирт; розчинник; схвалений в Україні. Складність: середня.
Примітки: **П**.

14. Реалізація програми, яка переводить цифровий код у штрих-код і визначає виробника товару.
Програма повинна реалізовувати (на мобільному пристрої або ПК) за введеним числовим кодом малювання штрих-коду, який можна буде роздрукувати, та виводити інформацію про країну-виробника. Вхідні дані: номер штрих-коду.
Вихідні дані: рисунок штрих-коду, країна-виробник.
Складність: низька.
Примітки: **П**.

15. Реалізація програми, яка переводить цифровий код у QR-код і визначає виробника товару.
Програма повинна реалізовувати (на мобільному пристрої або ПК) за введеним числовим кодом(або іншим текстом) малювання QR-коду, який можна буде роздрукувати.
Вхідні дані: число/текст.
Вихідні дані: рисунок QR-коду.
Складність: середня.
Примітки: **П**.

16. Реалізація програми для малювання української вишивки, характерної для Вінниччини.
Програма повинна виконувати за введеними розмірами (у точках) малювання української вишивки, враховуючи симетрію, у вигляді прямокутного зображення або у вигляді рамки (всередині будь-яке зображення, наприклад логотип фірми). Дозволяється відхилятися від традиційного орнаменту та використовувати орнаменти будь-яких регіонів. Залежності від конкретизованого технічного завдання, складність може варіюватися від низької до високої.
Вхідні дані: розмір рисунка.
Вихідні дані: рисунок вишивки.
Складність: висока.
Примітки: **Г**.

17. Реалізація програми для малювання української вишивки за способом «Бродівське письмо».
Програма повинна виконувати за введеним текстом малювання української вишивки за способом «Бродівське письмо», враховуючи симетрію. Зображення може бути у вигляді кільцевого надпису, у вигляді простого тексту, так і у вигляді рамки (всередині будь-яке зображення, наприклад логотип фірми). Можна використовувати як прямі, так і косі літери. Дозволяється відхилятися від традиційного орнаменту. Детальний опис способу вишивки «бродівське письмо» описаний у книзі Підгірняка В. П. «Текстова вишивка. Бродівське письмо», 2008 року. Залежності від конкретизованого технічного завдання, складність може варіюватися від низької до високої.
Вхідні дані: розмір рисунка.
Вихідні дані: рисунок вишивки.
Складність: висока.
Примітки: **Г**.

18. Реалізація будильника для мобільного пристрою.
Програма повинна реалізовувати будильник для мобільного пристрою (телефону, планшета), який видаватиме звуковий сигнал (можна разом із графічним), поки користувач не скасує цей сигнал, або не пройде 5 хвилин від старту будильника. Дія вимкнення будильника має бути не дуже очевидна, щоб її не можна було миттєво зрозуміти, як вимкнути-наприклад повторення руху на екрані. Користувач повинен мати можливість поставити затримку на 5 хвилин, змінити звуковий сигнал на свій та періодично його змінювати.
Вхідні дані: час запуску будильника, звуковий файл користувача.
Складність: середня.
Примітки: **П**.

19. Реалізація програми для упорядковувала музики для мобільного пристрою.
Програма повинна реалізовувати для мобільного пристрою (телефону, планшета) створення таблиці всіх наявних на пристрої музичних файлів, де вказуватиме їх місце знаходження, розмір, дату створення. Також програма дозволяє шукати у ній, сортувати та запускати ці файли, рахує кількість запусків кожного файлу через програму, дає можливість додавати автора, жанр, групу, виконавця композиції.
Вхідні дані: музичні файли на мобільному пристрої.
Вихідні дані: впорядкована таблиця музичні файли на мобільному пристрої.
Складність: висока.
Примітки: **П**.

20. Реалізація простого диспетчера задач для мобільного пристрою.
Програма повинна реалізовувати для мобільного пристрою (телефону, планшета) створення таблиці всіх запущений на пристрої програм, відображати запущені процеси та потоки, а також вказувати їх місце знаходження, розмір використовуваної пам’яті, час запуску. У програмі повинна бути можливість запускати свій процес (запустити програму від користувача).
Вхідні дані: запущені програми на мобільному пристрої.
Вихідні дані: впорядкована таблиця запущених програми на мобільному пристрої.
Складність: висока.
Примітки: **П**.

21. Реалізація програми для визначення поточного місця знаходження для мобільного пристрою.
Програма повинна реалізовувати для мобільного пристрою (телефону, планшета) знаходження користувача у поточний момент, відобразити його положення на карті, вивести його координати та при потребі відправити їх через повідомлення/СМС/електронну пошту. Програма повинна однаково відображатись на планшеті та мобільному телефоні. Можна використати взаємодію з картами Google.
Вихідні дані: поточні координати користувача та його положення на карті.
Складність: низька.
Примітки: **П**.

22. Реалізація програми для визначення довжини маршруту від поточного місця знаходження для мобільного пристрою.
Програма повинна реалізовувати для мобільного пристрою (телефону, планшета) знаходження маршруту для користувача за відомими координатами або точкою на карті початку і кінця маршруту. Програма повинна відобразити маршрут на карті і розрахувати його довжину. Можна використати взаємодію з картами Google.
Вхідні дані: початкові і кінцеві координати маршруту користувача і їх положення на карті.
Вихідні дані: маршрут на карті та його довжина.
Складність: середня.
Примітки: **П**.

23. Реалізація автоматичного відповідача у нічний час для соціальної мережі Facebook.
Програма повинна реалізовувати для соціальної мережі Facebook автоматичний відповідач, який буде відписувати (без втручання користувача сторінки) всім користувачам певний текст (наприклад «користувач на даний момент спить, напишіть йому вранці») у нічний час (з 0:00 до 7:00). Даний текст задається в програмі. У програмі потрібно передбачити кілька варіантів тексту, які вибираються залежно від певних умов (часу написання, номеру повідомлення, людини, яка пише, або категорії дописувачів). Причому ця програма запам’ятовує користувача, який писав вночі, та у час, коли в нього буде поздоровлення, привітає також вночі. У програмі потрібно передбачити кілька варіантів тексту поздоровлення, які вибираються залежно від певних умов (дати та часу поздоровлення, людини, яку вітають, або її категорії).
Вхідні дані: час та текст відповіді.

Вихідні дані: повідомлення, написані іншими користувачами вночі.
Складність: середня.
Примітки: **C**.

24. Реалізація програми для автоматичного написання відтермінованих повідомлень у заданий час для соціальної мережі Twitter.
Програма повинна реалізовувати для соціальної мережі Twitter автоматичний відповідач, який буде писати певні тексти («твіти») (без втручання користувача сторінки) у заданий час всім підписаним користувачам, або заданим користувачам чи категоріям користувачів. Даний текст (з хештегами) задається в програмі. У програмі потрібно передбачити кілька варіантів тексту, які вибираються залежно від певних умов (часу викладення, певного повідомлення іншого користувача, номеру повідомлення).
Вхідні дані: час та текст повідомлень.
Вихідні дані: коментарі, кількість лайків та репостів кожного повідомлення.
Складність: середня.
Примітки: **C**.

25. Реалізація програми для автоматичного виставлення фотографій у заданий час для соціальної мережі Instagram.
Програма повинна реалізовувати для соціальної мережі Instagram автоматичне виставлення фотографій (без втручання користувача сторінки) у заданий час. Зображення з текстовим підписом(за бажанням з хештегами, геомітками) задається в програмі у вигляді файлу або шляху до нього. У програмі потрібно передбачити можливість виставлення фотографії залежно від певних умов (часу викладення, певного повідомлення іншого користувача, номеру фотографії, дати, поточного перебування користувача, певних коментарів інших користувачів).
Вхідні дані: час, дата і файли фотографій для виставлення, або шлях на ПК до них, текст надпису до них.
Вихідні дані: коментарі, кількість лайків та репостів кожної фотографії.
Складність: середня.
Примітки: **C**.

26. Реалізація автоматизованого перекладача з використанням Hibernate.
Програма повинна перекладати у автоматичному режимі текст, який вписаний у дану програму, або перекладати вказаний текстовий файл. Переклад можна виконувати будь-яким способом. Обов’язкове використання каркасу (фреймворка) Hibernate (будь-якої її частини) або аналогічного.

Вхідні дані: Текст для перекладу/файл для перекладу.
Вихідні дані: Перекладений текст/перекладений файл.
Складність: низька.
Примітки: **Т**.

27. Реалізація програми для відстежування курсу валют із графічним відображенням коливань на основі Spring Web Flow (SWF).
Програма повинна у автоматичному режимі відстежувати зміни курсу вказаних валют протягом певного часу і оновлювати дані. У якості джерела інформації можна використати офіційний сайт НБУ. Користувач повинен мати можливість побачити коливання однієї або кількох валют протягом певного часу у вигляді графіка. Обов'язкове використання каркасу (фреймворка) Spring Web Flow (будь-якої її частини) або аналогічного.
Вхідні дані: Значення курсу валют відносно гривні, періодичність оновлення, період коливань.
Вихідні дані: Поточний курс вказаних валют, значення коливань за вказаний період.
Складність: низька.
Примітки: **Т**.

28. Реалізація програми для автоматичної відправлення повідомлень по електронній пошті у вказаний час за певних умов.
Програма повинна у вказаний час у автоматичному режимі (без участі користувача) відправляти повідомлення з підготовленим текстом за вказаною адресою. Також можуть бути додаткові умови для часу та адреси відправки повідомлення (зміст тексту, значення ціни на нафту у даний час, перемога чи поразка деякої футбольної команди). Ці додаткові умови повинен або створюватись користувачем (для цього треба передбачити відповідний інтерфейс), або отримувати у вигляді програмного модуля, або отримувати у вигляді значення у текстовому файлі (яке може змінюватись, і залежить від цих умов).
Вхідні дані: Текст повідомлення для відправки, адресат, відправник, час відправлення.
Вихідні дані: Повідомлення про стан виконання.
Складність: середня.
Примітки: **Т**.

29. Реалізація власного простого завантажувача сайту для автономного перегляду при відсутності доступу до мережі інтернет.
Програма повинна виконувати завантаження цілого сайту у вказану папку,

при необхідності змінювати посилання всередині. Сайт повинен у автономному режимі зберігати зображення, відео, вбудовані об'єкти. Повинні працювати внутрішні посилання, при необхідності змінити посилання відносно папки завантаження як кореневої папки.
Вхідні дані: IP-адреса/ім'я хоста сайта, тип програми-клієнта (її номер порту), папка для збереження.
Вихідні дані: Час збереження, збережений сайт.
Складність: середня.
Примітки: **Т**.

30. Реалізація програми для збереження зображення з веб-сайту на комп'ютер.
Програма повинна зберігати зображення з вказаного веб-сайту на жорсткий диск комп'ютера користувача за вказаною адресою. Користувач повинен мати можливість обмежити розмір, якість (кількість пікселів) та тип зображень для збереження. Про стан завантаження (успішно/неуспішно) формується звіт у вигляді списку із вказуванням параметрів кожного збереженого зображення.
Вхідні дані: Посилання на сайт, список типів зображень для збереження, максимальний і мінімальний розмір і якість зображень, папка для збереження.
Вихідні дані: Кількість збережених зображень, список, розмір і тип збережених зображень, збережені зображення.
Складність: середня.
Примітки: **Г**.

31. Реалізація програми для обміну повідомленнями (месенджера). Частина 1. Реалізація інтерфейсу користувача (GUI).
Програма повинна реалізовувати графічний інтерфейс користувача (GUI), який надає можливість введення користувача, його авторизація (перевірка паролю користувача, завантаження його даних і статистики), а також інтерфейс (форми для введення даних) для під'єднання по локальній мережі. Також програма повинна реалізувати графічно панель меню, відображає кнопки в меню, відображає прийнятий та відправлений текст (взаємодія із модулями частини 2-4, графічний формат та використовувані технології визначає розробник цієї частини).
Вхідні дані: ім'я користувача, пароль, адреса в мережі (ім'я хоста, IP-адреса), номер порту, текст повідомлень для відправки супернику.
Складність: середня.
Примітки: **К**, **Г**.

32. Реалізація програми для обміну повідомленнями (месенджера). Частина 2. Реалізація обміну текстовими повідомленнями через інтернет та локальну мережу.
Програма повинна реалізовувати можливість доставки текстових повідомлень віддаленому користувачу через локальну (LAN) або глобальну (Internet) мережу, використовуючи введені користувачем дані з'єднання (отримані від модуля частини 1). Отриманий та відправлений текст передається для відображення модулю частини 2. Програма повинна відслідковувати час отримання повідомлення і видати повідомлення у випадку неможливості спілкування з іншим користувачем. Також повинна бути можливість відправити текст із підготовленого текстового файлу.
Вхідні дані: Текст для відправлення, час відправлення.
Вихідні дані: Отриманий текст, час отримання, стан доставки повідомлення.
Складність: середня.
Примітки: **К**.

33. Реалізація програми для обміну повідомленнями (месенджера). Частина 3. Реалізація обміну файлами через інтернет та локальну мережу.
Програма повинна реалізовувати можливість доставки файлових повідомлень віддаленому користувачу через локальну (LAN) або глобальну (Internet) мережу, використовуючи введені користувачем дані з'єднання (отримані від модуля частини 1). При успішному отриманні та відправлені файлу відображається відповідне повідомлення. Програма повинна відслідковувати час отримання файлу, показувати, яку частину даних уже отримано і видати повідомлення у випадку неможливості спілкування з іншим користувачем.
Вхідні дані: Файл для відправлення, адреса розміщення файлу час відправлення.
Вихідні дані: Отриманий файл, час отримання, стан доставки повідомлення.
Складність: висока.
Примітки: **К**.

34. Реалізація програми для обміну повідомленнями (месенджера). Частина 4. Реалізація збереження історії повідомлень у базі даних із можливістю пошуку.
Програма повинна реалізовувати можливість збереження листування (в т.ч. час доставки і отримання кожного повідомлення та файлу) користувача за даний сеанс і дописувати у базу даних до попередньої історії, фіксуючи

дату, час початку і кінця спілкування, кількість повідомлень, список отриманих файлів. Програма повинна надавати можливість пошуку у історії у базі даних за фрагментом тексту, знаходити текст у вказаний період (задається час початку і кінця).
Вхідні дані: Текст (із часом) листування під час сеансу, попередня історія, час початку і кінця для пошуку, фрагмент тексту для пошуку.
Вихідні дані: Текст історії, знайдений текст за вказаний період, знайдений текст.
Складність: середня.
Примітки: **К**, **Б**.

35. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 1. Реалізація інтерфейсу користувача (GUI).
Програма повинна реалізовувати графічний інтерфейс користувача (GUI), надавати поля введення даних IP-адреси і порт у режимі «ручного» підключення, надавати меню для всіх операцій, необхідних розробникам частин 2-6, надати відображення графіку (або діаграми) зміни швидкості з'єднання, часу передавання за отриманою послідовністю даних, відображати кількість переданих пакетів від кожного сервера та кількість переданих пакетів кожному клієнту.
Вхідні дані: IP-адреси і порт підключення програм торент-клієнтів.
Вихідні дані: IP-адреси і порт підключення програм торент-клієнтів.
Складність: середня.
Примітки: **К**, **Г**.

36. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 2. Реалізація простого torrent-клієнта.
Програма повинна використовувати технологію «торент-трекер» в режимі клієнта, тобто мати можливість отримувати файли частинами від віддаленої програми, або яка працює торент-сервером (роздає даний файл, сервер може бути не один). Технологію «торент-трекер» можна використовувати будь-якої версії, або реалізувати власну, зберігши дотримання лише загальних принципів (для програми важлива працездатність, а не відповідність технології). Також у програмі повинні показуватись кількість отриманих пакетів (або фрагментів файлів), графік швидкості закачки а після закачки середню швидкість і час завантаження.
Вхідні дані: Торент-файл (торент-посилання для під'єднання до серверів), адреса для збереженні на комп'ютері, IP-адреса і порт підключення програм торент-серверів.
Вихідні дані: Час закачування файлу, кількість фрагментів файлу, середня

швидкість завантаження файлу, графік швидкості завантаження, ступінь (%) завантаження файлу в кожний момент часу.
Складність: середня. Примітки: **К**.

37. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 3. Реалізація простого torrent-сервера (для роздачі).
Програма повинна використовувати технологію «торент-трекер» в режимі сервера, тобто мати можливість віддавали файли частинами будь-якій віддаленій програмі, або яка працює торент-клієнтом (отримує даний файл, клієнт може бути не один) і відправила запит на отримання файлу. Технологію «торент-трекер» можна використовувати будь-якої версії, або реалізувати власну, зберігши дотримання лише загальних принципів (для програми важлива працездатність, а не відповідність технології). Також у програмі повинні показуватись кількість переданих пакетів (або фрагментів файлів), графік швидкості вивантаження а після вивантаження середню швидкість і час завантаження, частка втрачених пакетів і переданих повторно.
Вхідні дані: Торент-файл, адреса файлу на комп'ютері, IP-адреси і порт підключення програм торент-клієнтів.
Вихідні дані: Час викачування файлу, кількість фрагментів файлу, середня швидкість вивантаження файлу, графік швидкості вивантаження, ступінь (%) вивантаження файлу в кожний момент часу, частка (%) втрачених пакетів і переданих повторно.
Складність: середня. Примітки: **К**.

38. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 4. Реалізація програми для оптимізації швидкості при нестабільній та різній швидкості з'єднання у користувачів.
Програма повинна виконувати оптимізацію ресурсу інтернет-каналу у програми сервера, якщо швидкість під'єднання у кожного користувача різна і може змінюватись у часі. Програма повинна показувати зі зміною у часі розподіл ресурсу між користувачами. Надавати ресурс потрібно пропорційно можливостям клієнта. Для вирішення даної задачі можна використати розділ теорії оптимізації «системи масового обслуговування».
Вхідні дані: Торент-файл, адреса файлу на комп'ютері, IP-адреси і порт підключення програм торент-клієнтів, середня та зміна в часі швидкості з'єднання кожного користувача.
Вихідні дані: Частка ресурсу інтернет-каналу, яка виділяється кожному користувачу у даний момент часу. Час викачування файлу, кількість фра-

гментів файлу, середня швидкість вивантаження файлу, графік швидкості вивантаження, ступінь (%) вивантаження файлу в кожний момент часу, частка (%) втрачених пакетів і переданих повторно.
Складність: висока. Примітки: **К**, **М**.

39. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 5. Реалізація програми для оптимізації часу доставки та маршруту проходження пакетів.
Програма повинна виконувати оптимізацію ресурсу інтернет-каналу у програми сервера, розраховуючи, що швидкість під'єднання у кожного користувача більш-менш стабільна. Оптимізація проводиться шляхом вирішення задачі маршрутизації (зазвичай цю задачу вирішує відповідне обладнання маршрутизаторів) пошуку інших, більш коротких (у часі) шляхів доставки пакетів. При знаходженні коротшого шляху доставка пакетів відбувається по цьому маршруту. Програма повинна показувати зміну часу доставки пакетів та шлях (повний зі список усіх проміжних точок, або скорочений) проходження пакетів. Для вирішення даної задачі можна використати будь-який розділ теорії оптимізації, який виконує знаходження оптимального маршруту між двома точками (пошук в ширину, пошук в глибину, жадібний, швидкий алгоритми …).
Вхідні дані: Торент-файл, адреса файлу на комп'ютері, IP-адреси і порт підключення програм торент-клієнтів.
Вихідні дані: Повний (або скорочений) шлях проходження пакетів під час доставки із вказуванням усіх проміжних пристроїв маршрутизації. Час викачування файлу, кількість фрагментів файлу, середня швидкість вивантаження файлу, графік швидкості вивантаження, ступінь (%) вивантаження файлу в кожний момент часу, частка (%) втрачених пакетів і переданих повторно.
Складність: висока. Примітки: **К**, **М**.

40. Реалізація власної простої програми для доставки файлів за технологією торент-трекер. Частина 6. Реалізація програми для захищеного передавання (криптографічне шифрування пакетів та аутентифікація користувачів).
Програма повинна виконувати аутентифікацію користувачів на клієнтській частині (перевірка імені та хешованого паролю користувача на серверній частині, та виводити повідомлення про стан аутентифікації-успішно, неуспішно) та виконувати криптографічне шифрування пакетів з відкритим ключем на серверній частині перед відправленням на розшифрування пакетів у клієнтській програмі. Обрахувати зменшення швидкості пере-

давання за рахунок шифрування та час, витрачений на шифрування та дешифрування.
Вхідні дані: Торент-файл, адреса файлу на комп'ютері, IP-адреси і порт підключення програм торент-клієнтів, ім'я і пароль користувача, відкритий ключ шифрування.
Вихідні дані: Зашифровані пакети (або фрагменти файлу) для відправлення, стан аутентифікації користувача.
Складність: середня.
Примітки: **К**, **З**.

41. Створення інформаційного веб-сайту на HTML5.
Необхідно написати інформаційний веб-сайт (портал) з динамічним контентом (відео, анімація, скрипти, аплети) на основі HTML5. На самому сайті повинні бути посилання на інші сторінки порталу, а також посилання на сторінки «карту сайту», «головна», «детальна інформація», «про авторів». Необхідно зберігати IP-адреси всіх відвідувачів (можна із класифікацією за регіоном). Обов'язкове тестування розміщення сайту на власному сервері(можна взяти Denwer, Tomcat, GlassFish або безкоштовні онлайн-хостинги).
Вхідні дані: Контент сайту.
Вихідні дані: IP-адреси відвідувачів сайту.
Складність: середня.
Примітки: **В**, **Т**.

42. Створення інформаційного веб-сайту з можливістю перегляду фотографій на DHTML.
Необхідно написати інформаційний веб-сайт (портал) з динамічним контентом (перегляд багатьох фотографій) на основі DHTML. Повинна бути можливість додавання фотографій певним категоріям привілейованих користувачів. На самому сайті повинні бути посилання на інші сторінки порталу, а також посилання на сторінки «карту сайту», «головна», «детальна інформація», «гостьова сторінка», «про авторів».
Вхідні дані: Контент сайту, фотографії, логін та пароль користувача.
Вихідні дані: Категорія користувача.
Складність: середня.
Примітки: **В**, **Т**.

43. Створення веб-сайту-форуму із БД DB/2 на HTML не нижче 4.0.
Необхідно написати веб-сайт (форум), з динамічним контентом (перегляд зображень користувача та завантаження доданих файлів) на HTML 4.0

і вище. Повинна бути можливість додавання фотографій та файлів (дозволених типів) певним категоріям користувачів(які написали певну кількість повідомлень за певний час). На форумі повинен бути список тем для обговорень, а обговорення у певній темі відображається у вигляді вітки повідомлень. Відповідь одного користувача на повідомлення іншого позначається у вигляді вітки та іншим кольором. На сайті існує ієрархія зареєстрованих користувачів. На самому сайті повинні бути посилання на інші сторінки порталу, а також посилання на сторінки «карту сайту», «головна», «список тем», «гостьова сторінка», «авторизація», «особистий кабінет», «про авторів». Весь текст обговорення та файли зберігаються у базі даних DB/2.
Вхідні дані: Контент сайту, фотографії, прикріплені файли, список користувачів, логіни та паролі користувачів.
Вихідні дані: Текст усіх тем у БД.
Складність: середня.
Примітки: **В**, **Т**, **Б**.

44. Створення інформаційного веб-сайту, який здатен працювати з БД.
Необхідно написати інформаційний веб-сайт (спеціалізований новинний портал) з динамічним контентом (зображення, відео, анімація, скрипти, аплети), який надає можливість певним категоріям користувачів додавати інформацію у вигляді стрічки новин RSS. Користувачам-гостям доступні не всі новини, авторизованим користувачам-майже всі, авторам, адміністраторам, VIP-користувачам-усі. На самому сайті повинні бути посилання на інші сторінки порталу, а також посилання на сторінки «карту сайту», «головна», «список тем», «гостьова сторінка», «авторизація», «особистий кабінет», «новини», «про авторів».
Для робот из БД можна використати технології JSP або Servlet. Вхідні дані: Контент сайту, текст усіх новин, текст новини для додання.
Вихідні дані: Текст усіх новин, користувачі у БД, категорія користувача.
Складність: низька.
Примітки: **В**, **Б**.

45. Створення інформаційного веб-сайту із можливістю перегляду відео на JavaFX.
Необхідно написати інформаційний веб-сайт (портал збірник блогів) з динамічним контентом (перегляд відео) на основі HTML та з використанням JavaFX. Відео може бути вставлене з відеосерверів (YouTube) або бути доданим користувачем. Повинна бути можливість вибрати кращу або гіршу якість відео(за наявності обох видів). Має бути можливість дода-

вання відео з описом певним категоріям користувачів. На самому сайті повинні бути посилання на інші сторінки користувачів, а також посилання на сторінки «карту сайта», «головна», «пошук відео по темі», «гостьова сторінка», «про авторів».
Вхідні дані: Контент сайту, відео, якість і опис відео, логін та пароль користувача.
Вихідні дані: Категорія користувача, доступні відео для певного користувача.
Складність: висока.
Примітки: **В**, **Т**.

46. Створення веб-сайту для музиканта з можливістю керувати доступом відтворенням композицій залежно від рейтингу користувача на сайті.
Необхідно написати інформаційний веб-сайт (музичний портал-блог) на з динамічним контентом (прослуховування музичних файлів). Звукові файли можуть бути вставлене з відповідних сервісів (SoundCloud) або бути доданим користувачем. Кожен звуковий файл має відомості: якість, формат файлу, автор, виконавець, дата запису, жанр, група-виконавець, альбом, тип (музика, голос, професійна музика, мелодія) та опис. Має бути можливість додавання звукових файлів з описом певним категоріям користувачів та прослуховування звукових файлів залежно від рейтингу користувача (кількість авторських композицій, якість-оцінка від інших користувачів). На самому сайті повинні бути посилання на інші сторінки користувачів, а також посилання на сторінки «карту сайту», «головна», «доступні композиції», «мої композиції», «пошук композиції» (по жанру, виконавцю, автору, альбому …), «гостьова сторінка», «авторизація», «про авторів».
Вхідні дані: Контент сайту, аудіо-файли, відомості про них, логін та пароль користувача.
Вихідні дані: Категорія користувача, рейтинг користувача, доступні аудіо для певного користувача.
Складність: середня.
Примітки: **В**.

47. Створення веб-сайту для перегляду інформації про певний товар зі списку із використанням JavaServer Faces (JSF).
Необхідно написати інформаційний веб-сайт, який надає інформацію про певні товари з інтернет-магазину з динамічним контентом (зображення, відео, скрипти). Зображення та відео, та інформацію про товар можуть додавати лише адміністратор. Користувач запитує товар за ключовими словами, пошук яких виконується у базі даних серверною частиною за

технологією JSF. На самому сайті повинні бути посилання на сторінки «карту сайту», «головна», «доступні товари», «пошук товару», «про нас».
Вхідні дані: Контент сайту, список товарів, опис всіх товарів, текст пошуку.
Вихідні дані: Знайдений результат.
Складність: середня.
Примітки: **В**, **Т**, **Б**.

48. Створення веб-сайту, який доступний на відкритому сервері та зберігає інформацію, захищену від хакерських атак (будь-хто зі групи має можливість спробувати викрасти цю інформацію).
Необхідно написати веб-сайт, який розміщений на одному з безкоштовних хостів (або хост створений на власному комп'ютері), зберігає деяку інформацію, доступну для деяких користувачів, мають доступ. Кількість доступів, користувач і час доступу суворо фіксуються у журналі. Інформація має бути захищену від хакерських атак (підбір пароля, переповнення стека, обхід доступу через помилки пам'яті, …). Будь-яка людина може спробувати викрасти цю інформацію (і на сайті про це написано). Якщо інформацію викрав студент(и), він отримує(ють) підвищення оцінки з предмету «програмування» на 10 балів. Якщо інформація була викрадена, розробник повинен внести вдосконалення і закрити вразливість системи. Атаки вважаються витримані, якщо на протязі тижня інформація не була викрадена.
Вхідні дані: Інформація, користувачі, які мають доступ.
Вихідні дані: Кількість доступів, користувач і час доступу.
Складність: середня.
Примітки: **В**, **З**.

49. Створення веб-сайту для масової гри у «хрестики-нулики» онлайн на скриптах (PHP/JavaScript/JScript/Node JS/Ruby/Python/JQuery/Elixir/ін.).
Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «хрестики-нулики» (розмір 100×100) по парам онлайн, який використовує скрипти (PHP/JavaScript/JScript/Node JS/Ruby/Python та ін.). Користувачі мають можливість авторизуватись або грати анонімно. Сайт реєструє та відображає кількість перемог користувача, час гри, 5 кращих гравців за тиждень.
Вхідні дані: Ім'я користувача і пароль.
Вихідні дані: Кількість перемог користувача, час гри, 5 кращих гравців.
Складність: середня.
Примітки: **В**, **Т**.

50. Створення веб-сайту для масової гри у «шашки» онлайн із БД на Google Web Toolkit.
Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «шашки» по парам онлайн. Сама повинна використовувати Google Web Toolkit. Користувачі мають можливість авторизуватись або грати анонімно. У чемпіонаті приймають участь лише зареєстровані користувачі. Сайт реєструє та відображає кількість перемог користувача, час гри, 5 кращих гравців за тиждень.
Вхідні дані: Ім’я користувача і пароль.
Вихідні дані: Кількість перемог користувача, час гри, 5 кращих гравців за тиждень.
Складність: середня.
Примітки: **В**, **Т**.

51. Створення веб-сайту для масової гри у «морський бій» онлайн, який буде надавати можливість грати багатьом користувачам окремо та між собою. Частина 1. Створення веб-сайту із авторизацією користувачів. Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «морський бій» по парам онлайн. У даній частині сайту відбувається авторизація користувача, перевірка хешованого пароля, захист через введення текст з картинок (CAPTHA), можливість зареєструватись новому користувачу і відновити пароль. На сайті суперник вибирається автоматично (той, хто зайшов відразу після даного користувача, який шукає пару). Обов’язкове використання Unity 3D.
Вхідні дані: Ім’я користувача і пароль, зареєстровані онлайн користувачі.
Вихідні дані: Результат правильності введення.
Складність: низька.
Примітки: **В**, **Т**.

52. Створення веб-сайту для масової гри у «морський бій» онлайн, який буде надавати можливість грати багатьом користувачам окремо та між собою. Частина 2. Створення графічного інтерфейсу гри у «морський бій». Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «морський бій» по парам онлайн. У даній частині сайту виконується надання графічного інтерфейсу, меню, підготовка поля для розташування кораблів, відображається процес битви і виведення повідомлення про перемогу чи поразку відповідному користувачу. Обов’язкове використання Unity 3D.
Вхідні дані: Суперники (пара користувачів), натискання стрілок для розташування кораблів, координати пострілів.

Вихідні дані: Координати цілих та потоплених (підбитих) кораблів, переможець.
Складність: середня.
Примітки: **В**, **Г**, **Т**.

53. Створення веб-сайту для масової гри у «морський бій» онлайн, який буде надавати можливість грати багатьом користувачам окремо та між собою. Частина 3. Створення штучного інтелекту для гри у «морський бій». Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «морський бій» по парам онлайн. У даній частині сайту виконується надання можливості гри без іншого користувача (зі штучним інтелектом) з використання графічного інтерфейсу від розробника частини 2. Необхідно зробити 3 рівні складності штучного інтелекту. Відображається процес битви і виведення повідомлення про перемогу чи поразку відповідному користувачу. Обов'язкове використання Unity 3D.
Вхідні дані: Натискання стрілок для розташування кораблів, координати пострілів.
Вихідні дані: Координати цілих та потоплених (підбитих) кораблів, переможець.
Складність: висока.
Примітки: **В**, **М**, **Т**.

54. Створення веб-сайту для масової гри у «морський бій» онлайн, який буде надавати можливість грати багатьом користувачам окремо та між собою. Частина 4. Створення інтерфейсу та функцій для ведення статистики на сайті та записом у БД.
Необхідно написати веб-сайт для масової гри (багатьма користувачами паралельно) у «морський бій» по парам онлайн. У даній частині сайту виконується створення інтерфейсу та функцій для ведення статистики на сайті за кожним користувачем та всіма користувачами разом, визначення кращих гравців, відбір у таблиці чемпіонату (турнір на протязі дня). Вся статистика записується у базу даних. Кожен користувач бачить власну статистику та статистику сайту в цілому. Вхідні дані: Ім'я користувача і пароль, зареєстровані онлайн користувачі, їх попередні успіхи.
Вихідні дані: Кількість перемог і поразок, середній час боїв, успішність (%) кожного користувача, 10 кращих гравців, ступінь у чемпіонаті.
Складність: середня.
Примітки: **В**, **Б**.

55. Реалізація гри «морський бій» із автоматичним встановленням програми (інсталятор). Частина 1. Реалізація інтерфейсу із авторизацією користувачів та можливістю обміну повідомленнями по локальній мережі. Програма повинна реалізовувати програмний інтерфейс, який надає можливість введення користувача, його авторизація (перевірка паролю користувача, завантаження його даних і статистики-для цього можна використати модулі розробника частини 2, 3), а також під'єднання по локальній мережі або інтернет, вставлення з'єднання, доставка та відображення повідомлень, відправлених іншим користувачем. Програма повинна відправляти інформацію про користувача супернику та отримувати інформацію про суперника, в т.ч. в процесі гри (координати і успішність пострілу).
Вхідні дані: ім'я користувача, пароль, адреса в мережі (ім'я хоста, IP-адреса), номер порту, текст повідомлень для відправлення супернику.
Вихідні дані: ім'я суперника, текст повідомлень, отриманих від суперника.
Складність: середня.
Примітки: **К**.

56. Реалізація гри «морський бій» із автоматичним встановленням програми (інсталятор). Частина 2. Реалізація графічного інтерфейсу гри у «морський бій». Програма повинна реалізовувати графічний інтерфейс користувача (GUI), який надає можливість введення користувача, його авторизація (перевірка паролю користувача, завантаження його даних і статистики), а також інтерфейс (форми для введення даних) для під'єднання по локальній мережі. Також програма повинна графічно представляти статистику користувача на основі отриманих даних від модуля з частини 3. Також програма реалізовує графічно відображає власне ігровий процес для «морського бою». Обов'язкове використання кількох власних спрайтів для гри.
Вхідні дані: ім'я користувача, пароль, адреса в мережі (ім'я хоста, IP-адреса), номер порту, текст повідомлень для відправлення супернику.
Складність: середня. Примітки: **К**, **Г**.

57. Реалізація гри «морський бій» із автоматичним встановленням програми (інсталятор). Частина 3. Реалізація штучного інтелекту для гри у «морський бій». Програма повинна реалізовувати на основі модулів частини 1, 2 штучний інтелект, який дозволить грати користувачу при відсутності іншого гравця (суперника). Штучний інтелект повинен самостійно розставляти кораблі, вибирати стратегію(та змінювати залежно від успішності), координати для власних пострілів. Стратегії поведінки та оцінка виграшу описуються у розділі математики «теорія ігор».
Вхідні дані: координат пострілів суперника та успішність власних постріл-

лів.
Вихідні дані: координати розташування кораблів, координати власних пострілів.
Складність: середня.
Примітки: **К**, **М**.

58. Реалізація гри «морський бій» із автоматичним встановленням програми (інсталятор). Частина 4. Реалізація ведення статистики та запису у БД Cloudscape. Програма повинна реалізовувати програмний інтерфейс, який надає інформацію (статистику боїв) про користувача(для цього можна використати модулі розробника частини 2), а також про суперника користувача. Також даний модуль формує таблицю кращих гравців за всю історію («дошка слави»). Після завершення бою програма повинна внести нову інформацію у базу даних та перерахувати дані користувачів. Обов'язкове використання для реалізації програми бази даних Cloudscape (дозволяється змінити, але конкретна БД має бути наперед визначена).
Вхідні дані: результати і дані поточного бою, історія і статистика ігор користувача.
Вихідні дані: успішність, статистика і рейтинг користувача та суперника, «дошка слави» кращих гравців.
Складність: середня.
Примітки: **К**, **Б**, **Т**.

59. Реалізація гри «морський бій» із автоматичним встановленням програми (інсталятор). Частина 5. Реалізація програми встановлення гри із визначенням платформи та перевірки можливості встановлення гри. Програма, отримавши завершені модулі від розробників частин 1,2,3,4 повинна реалізовувати визначенням платформи користувача, виконати перевірку можливості встановлення усієї програми гри на комп'ютер користувача, у випадку неможливості зазначити причину та запропонувати шляхи вирішення проблеми. Програма повинна мати механізм додання оновлень та патчів при вже встановленій грі, та здійснювати контроль і сумісність версій. Також повинен бути передбачений контроль умовної ліцензії та отримання від користувача згоди на цю ліцензію.
Вхідні дані: платформа користувача, версія гри.
Вихідні дані: допустимість встановлення на платформа користувача, або причина неможливості встановлення програми та шляхи виправлення.
Складність: середня.
Примітки: **К**.

60. Реалізація гри «пятнашки» для мобільного пристрою.
Програма повинна реалізовувати для мобільного пристрою (телефона, планшета) гру «пятнашки», реалізувати графічний інтерфейс користувача (GUI), можливість управління клавішами/сенсорним екраном, вітати користувача при виграші та записувати результати (час і кількість виграшів). Обов'язкове використання Unity (або його окремих елементів) не нижче 5-ї версії.
Вихідні дані: Час гри.
Складність: середня.
Примітки: П, **Г**, **Т**.

61. Реалізація гри «хрестики-нулики» для мобільного пристрою на Android.
Програма повинна реалізовувати для мобільного пристрою (телефона, планшета) гру «хрестики-нулики», реалізувати графічний інтерфейс користувача (GUI), можливість управління клавішами/сенсорним екраном, вітати користувача при виграші та записувати результати (час і кількість виграшів). Обов'язкове використання OpenGL. Допускається реалізація не лише на Android.
Вихідні дані: Час гри.
Складність: середня.
Примітки: **П**, **Г**, **Т**.

62. Реалізація гри «монополія».
Програма повинна реалізовувати гру «монополія», реалізувати графічний інтерфейс користувача (GUI), можливість управління клавішами/мишею, передбачити можливість або грати двом гравцям, або одному гравцю зі штучним інтелектом. Програма повинна вітати користувача при виграші та записувати результати (час, параметри і кількість виграшів). Обов'язкове використання Unity 2D.
Вихідні дані: Час і параметри гри.
Складність: середня.
Примітки: **Г**, **М**, **Т**.

63. Реалізація програми для гри у пінг-понг по локальній мережі. Частина 1. Реалізація графічного інтерфейсу та штучного інтелекту способом 1. Програма повинна реалізовувати гру «пінг-понг», реалізувати графічний інтерфейс користувача (GUI) із формами для введення даних для інтернет-з'єднання (використовувати дані форми введення і обробляти дані буде розробник для «частини 2»), а також власне ігрового процесу (відображен-

ня гравця, суперника, м’ячика), можливість управління клавішами/мишею, передбачити можливість або грати двом гравцям, або одному гравцю зі штучним інтелектом, а також змагатися даному штучному інтелекту із штучним інтелектом іншого розробника (із частини 2). Програма повинна вітати користувача при виграші та записувати результати (час, параметри і кількість виграшів). Обов’язкове використання WebGL.
Вхідні дані: координати суперника і м’ячика.
Вихідні дані: власні координати, час гри, кількість пропущених «м’ячиків» кожного гравця.
Складність: середня.
Примітки: **М**, **К**, **Г**.

64. Реалізація програми для гри у пінг-понг по локальній мережі. Частина 2. Реалізація з’єднання по Internet та штучного інтелекту способом 2. Програма повинна реалізовувати програмний інтерфейс, який надає можливість введення користувача, завантаження його даних і статистики, а також під’єднання по локальній мережі або інтернет, вставлення з’єднання, доставка повідомлень із координатами, відправлених іншим користувачем. Також потрібно реалізувати штучний інтелект, який буде змагатися із штучним інтелектом іншого розробника (з частини 1). Вхідні дані: ім’я користувача, пароль, адреса в мережі (ім’я хоста, IP-адреса), номер порта, координати суперника і м’ячика.
Вихідні дані: власні координати, час гри, кількість пропущених «м’ячиків» кожного гравця.
Складність: середня.
Примітки: **М**, **К**.

65. Реалізація гри «лучник». Частина 1. Реалізація інтерфейсу користувача. Програма повинна реалізовувати графічний інтерфейс користувача (GUI), який реалізує ігровий процес: фігуру гравця-лучника, який може змінювати силу та кут нахилу; стрілу у польоті (розрахунок координат, траєкторії, вплив вітру виконує розробник частини 2; малювання перешкод (властивості перешкод та їх вплив на політ стріли виконує розробник частини 3; надає можливість введення користувача, завантаження його даних і статистики. Обов’язкове використання OpenGL або його компонентів SFML/WebGL/ін.
Вхідні дані: ім’я користувача, кількість пройдених рівнів, сила та кут натягу лука, швидкість і напрям вітру.
Складність: висока.
Примітки: **К**, **Г**, **Т**.

66. Реалізація гри «лучник». Частина 2. Реалізація модуля для прораховування траєкторії стріли залежно від кута, сили запуску, напряму та сили вітру.
Програма повинна реалізовувати прораховування траєкторії стріли і координати в будь-який момент часу, якщо відомо силу та кут нахилу лука. Швидкість вітру може бути як детермінована, так і випадкова величина, яка завжди відома і змінює траєкторію стріли у польоті. Програма повинна передавати дані модулям із частини 1 та частини 3. Обов'язкове використання розрахунку фізики з модуля Unity.
Вхідні дані: сила та кут натягу лука, швидкість і напрям вітру.
Вихідні дані: поточні координати і траєкторія стріли.
Складність: середня.
Примітки: **К**, **М**, **Т**.

67. Реалізація гри «лучник». Частина 3. Реалізація модуля для створення перешкод (непробивні блоки, відбивачі, важелі, механізми).
Програма повинна реалізовувати перешкоди для стріли (непробивні блоки, відбивачі, важелі, механізми), задавати властивості та розмір перешкод кожного типу, та прораховування вплив на траєкторію стріли даних перешкод (поточні координати і траєкторія отримуються від модуля частини 2). Обов'язкове використання розрахунку для фізики з модуля Unity.
Вхідні дані: ім'я користувача, кількість пройдених рівнів, сила та кут натягу лука, швидкість і напрям вітру, кількість, параметри і властивості перешкод.
Вихідні дані: поточні координати і траєкторія стріли з урахуванням перешкод.
Складність: середня.
Примітки: **К**, **М**, **Т**.

68. Реалізація гри «лучник». Частина 4. Реалізація набору рівнів гри та етапу навчання.
Програма повинна реалізовувати створення рівнів гри із наростанням складності, в яких відбувається розташування перешкод. Також потрібно передбачити навчальну кампанію для гравця, який не має елементарних знань з фізики та уявлення про техніку стрільби із лука із поясненнями.
Вхідні дані: кількість рівнів, параметри і властивості перешкод.
Вихідні дані: файли рівнів із розставленими перешкодами, файл опису (допомоги).
Складність: середня.
Примітки: **К**, **Г**.

69. Реалізація гри «мушкетер». Частина 1. Реалізація інтерфейсу користувача (GUI).
Програма повинна реалізовувати графічний інтерфейс користувача(GUI), який реалізує графічно власне ігровий процес: фігуру гравця-стрільця, який може змінювати кількість пороху, тип зброї та кут нахилу; куля у польоті (розрахунок координат, траєкторії, вплив вітру виконує розробник частини 2); малювання різних перешкод (властивості перешкод та їх вплив на політ кулі залежно від кута влучання виконує розробник частини 3), а також можливість рикошету, роздроблення кулі, появу осколків перешкод; надає можливість введення користувача, завантаження його даних і статистики; відображає положення і дію ворогів (розрахунок їх позиції, поведінки, стратегії виконує розробник частини 4). Завдання розробника частини 1 полягає у створенні графічної бібліотеки, яку повинні використовувати розробники інших модулів. Обов'язкове використання хоча б однієї власної текстури/спрайта.
Вхідні дані: ім'я користувача, кількість пройдених рівнів, тип зброї, кількість і тип ворогів, перелік перешкод і їх типу (матеріалу), швидкість і напрям вітру.
Складність: висока.
Примітки: **К**, **Г**.

70. Реалізація гри «мушкетер». Частина 2. Реалізація модуля для прораховування траєкторії кулі залежно від кута, кількості пороху, напряму та сили вітру.
Програма повинна реалізовувати прораховування траєкторії кулі і координати в будь-який момент часу, якщо відомо кількість зарядженого пороху та кут нахилу зброї, а також випадковий параметр дрижання зброї для недосвідченого гравця. Швидкість вітру може бути як детермінована, так і випадкова величина, яка завжди відома і змінює траєкторію стріли у польоті. Програма повинна передавати дані модулям із частини 1, 3, 4. Обов'язкове використання одного з існуючих ігрових рушіїв (або його елементів), наприклад Unreal Engine.
Вхідні дані: тип зброї, кількість заряду пороху, кут нахилу зброї, напрям та величина коливання зброї, швидкість і напрям вітру.
Вихідні дані: поточні координати і траєкторія кулі.
Складність: середня.
Примітки: **К**, **М**, **Т**.

71. Реалізація гри «мушкетер». Частина 3. Реалізація модуля властивостей матеріалів поверхонь (міцність, крихкість, можливість рикошету та роздроблення) та розрахунок імпульсу кулі до і після влучання.
Програма повинна реалізовувати перешкоди для кулі (непробивні блоки, відбивачі, пробивні блоки, блоки, що розсипаються), задавати властивості та розмір перешкод кожного типу, та прораховування вплив на траєкторію кулі даних перешкод (поточні координати і траєкторія отримуються від модуля частини 2). Також повинна бути розрахована імовірність рикошету та роздроблення кулі, швидкість, напрям та імпульс уламків, ступінь знищення та імовірність пробиття перешкоди. Обов'язкове використання розрахунку фізики з модуля Unity.
Вхідні дані: тип зброї, кількість заряду пороху, кут нахилу зброї, імпульс та швидкість кулі, тип (матеріал) та розмір перешкоди, ступінь знищення перешкоди.
Вихідні дані: поточні координати і траєкторія кулі з урахуванням перешкод, імовірність рикошету та роздроблення кулі, швидкість, напрям та імпульс уламків, ступінь знищення та імовірність пробиття перешкоди.
Складність: середня.
Примітки: **К**, **М**, **Т**.

72. Реалізація гри «мушкетер». Частина 4. Реалізація модуля штучного інтелекту для ворогів (різних типів).
Програма повинна реалізовувати створення штучного інтелекту, який задає тактику і стратегію для ворогів різних типів. Штучний інтелект повинен вибирати поведінку залежно від типу зброї гравця і власної зброї, наявності, розміру і типу перешкод. У кожного ворога є свій тип (стратегія) поведінки. Бажано додати можливість самонавчання «сильних» ворогів. Обов'язкове використання одного з існуючих ігрових рушіїв (наприклад з модуля Unity).
Вхідні дані: поточний рівень, статистика успішності гравця і ворогів, наявність та тип перешкод, тип зброї гравця і ворогів.
Вихідні дані: координати положення ворогів (куди йти), час пострілів, кут нахилу зброї для кожного з ворогів.
Складність: висока.
Примітки: **К**, **М**.

73. Реалізація гри «мушкетер». Частина 5. Реалізація модуля вдосконалень для зброї гравця та ворогів.
Програма повинна реалізовувати створення можливості вдосконалення зброї гравця, ворогів залежно або незалежно від попередніх успіхів.

Повинен враховуватись баланс сили гравця та ворогів. Розробник задає параметри зброї, які повинні використовувати для своїх розрахунків розробники всіх інших частин (1-5).
Вхідні дані: кількість рівнів, статистика успішності гравця і ворогів.
Вихідні дані: параметри вдосконаленої зброї (імовірність дрижання, кількість пороху, початкова швидкість кулі, час перезарядки).
Складність: середня.
Примітки: **К**.

74. Реалізація програми з віртуальною реальність (VR).
Програма повинна реалізовувати на створення віртуальною реальності на камері комп'ютера, мобільного телефона, планшета. Зображення зробити самостійно в 3D чи VR-редакторі. Зображення повинно змінюватись залежно від положення та повороту камери.
Складність: середня.
Примітки: **Г**.

75. Реалізація програми з доповненою реальність (AR) на мобільному пристрої.
Програма повинна реалізовувати на мобільному телефоні або планшеті програму для створення доповненої реальності. Зображення, яке виникає при наведенні на опорну точку, зробити самостійно в 3D-редакторі. Зображення повинно бути рухомим. Дозволяється використовувати будь-які засоби (в т.ч. набори AR Studio), окрім модуля Unity 3D. Корисною буде робота з бібліотекою Vuforia.
Складність: середня.
Примітки: **Г**.

76. Реалізація програми з доповненою реальність (AR) з допомогою Unity 3D
Програма повинна реалізовувати на створення доповненої реальності при наведенні на опорну точку камерою комп'ютера, мобільного телефона, планшета. Зображення зробити самостійно в 3D-редакторі. Зображення повинно бути інтерактивним (взаємодіяти з користувачем через пристрій введення чи екранну клавіатури). Обов'язкове використання Unity 3D. Корисною буде робота з бібліотекою Vuforia та фізикою в Unity.
Складність: середня.
Примітки: **Г**, **Т**.

77. Реалізація шпигунської програми, замасковану під гру з доповненою реальність (AR).
Програма повинна реалізовувати гру, принцип якої аналогічний Pokemon Go, тільки в менших масштабах: збираєш деякі елементи, створені на основі доповненої реальності і отримуєш за це віртуальну винагороду. Елементи генеруються випадково чи за заданим алгоритмом. В процесі гри має бути передбачена можливість таємного передавання фото/відео з камери за вказаною адресою. У адміністратора програми повинна бути можливість «зйомки» потрібних місць карти (через камери користувачів з цією програмою, які грають у вказаних межах), редагування положення і кількості ігрових елементів, можливість надсилання повідомлення користувачам через гру. Корисною буде робота з бібліотекою Vuforia та фізикою в Unity.
Складність: висока.
Примітки: **Г**, **З**.

78. Реалізація гри «симулятор каменю».
Програма повинна реалізовувати вигляд для спостерігача відносно деякої нерухомої точки та відображати на екрані зображення природних чи штучних об’єктів з камери/спостерігача (якість не має значення, можна використовувати підхід низько полігональних моделей LowPoly). Наприклад, вид відносно «каменю». Усі необхідні зображення, текстури, спрайти, моделі зробити самостійно в 3D-редакторі. У грі повинні бути як рухомі моделі, так і нерухомі (положення спостерігача (каменя) незмінне). Передбачити можливість зміни швидкості часу та запис «історії життя каменя».
Складність: середня.
Примітки: **Г**.

79. Реалізація гри «лабіринт».
Програма повинна гру, де головний герой ходить по лабіринту у шукає вихід. У лабіринті кожна стіна має свою текстуру/спрайт (якщо зустрічаються 2 протилежні стіни однакової довжини, на одній зображення повинно бути створене векторним редактором, на протилежній-растровим). Усі необхідні зображення, текстури, спрайти, моделі зробити самостійно. Гра повинна мати вигляд 3D (дозволяється використовувати підхід «псевдо 3D»). Вихід з лабіринту для головного героя повинен бути анімованим.
Складність: низька.
Примітки: **Г**.

80. Реалізація гри на основі ігрового рушія MineCraft.
Програма повинна гру, де головний герой ходить по території, яка може редагуватись. Рельєф може складатися з окремих моделей та мати різні текстури/спрайти. Гра повинна мати вигляд 3D (дозволяється використовувати підхід «псевдо 3D»). Для гри можна використовувати підхід низько полігональних моделей LowPoly та ігровий рушій MineCraft або аналогічний, наприклад з Unity.
Складність: низька.
Примітки: **Г**.

Примітки

С – Завдання від спеціаліста в певній галузі. Для виконання даної роботи може знадобитись консультація спеціаліста у певній галузі (забезпечує викладач).
М – Завдання з математичною складовою. Для виконання даного завдання потрібні знання у певній галузі математики (у рамках університетського курсу «вищої математики»).
К – Командне завдання. Виконання даного завдання передбачає розподіл завдання між кількома виконавцями. Усім виконавцям потрібно домовитися про використання сумісних між собою технологій, способів реалізацій та узгодити спосіб обміну даними між програмними модулями. Оцінювання кожного виконавця буде здійснюватись залежно від якості виконаного власного завдання та незалежно від якості виконаних інших частин завдання, проте за умови поганого виконання завдань іншими учасниками усе завдання разом не функціонуватиме належним чином.
Т – Завдання із використанням конкретних технологій. При виконання даного завдання використання даних технологій є обов'язковим. Студент у технічному завданні може запропонувати альтернативну реалізацію даної технології, або запропонувати використання подібних та інших технологій.
П – Завдання для реалізації на портативних пристроях (планшети, мобільні телефони). Вибір мобільної системи/платформи/пристрою описується у пояснювальній записці (якщо завданням не передбачено використання чітко вказаних технологій мобільної системи/платформи/пристрою)
Г – Завдання із графікою. Завдання потребує приділення великої уваги використанню технологій роботи з графікою або графічному інтерфейсу користувача (GUI).
З – Завдання із захистом інформації. Завдання передбачає обов'язковий захист даних користувача від стороннього доступу шляхом використання

криптографії, стенографії, аутентифікації користувача, цифрового підпису та інших технологій захисту інформації.

В – Завдання передбачає створення веб-сайту. Призначення, основні функції та контент веб-сайту повинні описуватися у технічному завданні.

Б – Завдання із базою даних. Завдання передбачає обов'язкове використання бази даних. Вибір бази даних та супутніх технологій (сервер, протокол та мова запитів, інше) повинен описуватися у пояснювальній записці.

Під складністю виконання завдання мається на увазі очікуваний рівень знань, які потрібні студенту для написання курсової роботи.

Електронне навчальне видання

Костянтин Вячеславович Овчинников,
Володимир Григорович Сторчак

Методичні вказівки до виконання курсових робіт з дисципліни «Програмування» зі спеціальності «Інформаційні системи і технології»

Рукопис оформив К. Овчинников

Редактор Н. Кравчук

Оригінал-макет виготовлено в РВВ ВНТУ

Підписано до видання 04.02.2026 р.
Гарнітура Times New Roman.
Зам. № Р2026-014

Видавець та виготовлювач
Вінницький національний технічний університет,
Редакційно-видавничий відділ,
ВНТУ, ГНК, к. 114.
Хмельницьке шосе, 95,
м. Вінниця, 21021.
press.vntu.edu.ua;
Email: rvv.vntu@gmail.com.
Свідоцтво суб'єкта видавничої справи
серія ДК № 3516 від 01.07.2009 р.