ДЕРЖАВНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
СИСТЕМА УПРАВЛІННЯ ЯКІСТЮ
Система забезпечення якості освітньої діяльності та якості вищої освіти
сертифікована на відповідність ДСТУ ISO 9001:2015 / ISO 9001:2015
Кафедра інженерії програмного забезпечення та кібербезпеки

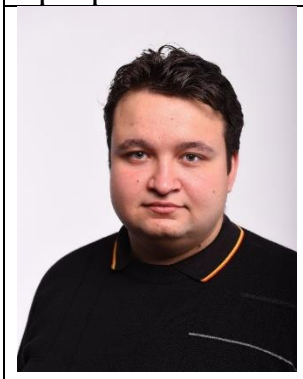
СИЛАБУС

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ/ OBJECT-ORIENTED PROGRAMMING

SYLLABUS

ЗАТВЕРДЖЕНО
засіданням кафедри
(протокол №.1
від «04» серпня 2024 р.)
завідувач кафедри
_____Олена КРИВОРУЧКО

Київ 2024

Назва освітньої компоненти	**ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ/ OBJECT-ORIENTED PROGRAMMING**
Спеціальність	121 «Інженерія програмного забезпечення»
Освітній ступінь	Перший (бакалаврський)
Освітньо-професійна програма	**ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**
	Лектор: **Цюцюра Микола Ігорович** -професор кафедри інженерії програмного забезпечення та кібербезпеки, -доктор технічних наук -пофесор Резюме викладача: https://knute.edu.ua/blog/read/?pid=46090&uk Науковий профіль: https://knute.edu.ua/blog/read/?pid=46717 e-пошта: mitsutsura@dteu.edu.ua
Консультації	https://knute.edu.ua/blog/read/?pid=47103&uk
ОПП освітньої компоненти	https://knute.edu.ua/blog/read/?pid=48212

1. **Дисципліна**: «**Об'єктно-орієнтовне програмування**»,
   - рік навчання: II;
   - семестр навчання: 3-4;
   - кількість кредитів: 6;
   - кількість годин за семестр: 180 год.
     - лекційних: 28/34 год.
     - практичних: 56/68 год.
     - на самостійне опрацювання: 96/78 год.
   - кількість аудиторних годин на тиждень:
     - лекційних: 2 год.
     - практичних: 4 год.
2. **Час та місце проведення:**
   - аудиторні заняття або онлайн заняття - відповідно до розкладу ДТЕУ з врахуванням специфіки дисципліни проведення останньої передбачено в аудиторіях: 505, 510, 510а, 514 або MS Teams;
   - поза аудиторна робота - самостійна робота студента, результат виконання якої висвітлено засобами Office 365;
   - всі практичні завдання виконуються на основі інтерактивних методів навчання у електронному середовищі. Передбачається можливість проведення лабораторних та лекційних занять на базах підприємств-партнерів.
3. **Пререквізити та постреквізити навчальної дисципліни:**
   - **пререквізити:** дисципліна базується на знаннях та компетентностях, що набуває здобувач вищої освіти під час вивчення дисциплін «Економічна інформатика», «Іноземна мова за професійним спрямуванням», «Інформаційні системи і технології».

– **постреквізити:** Дисципліна надає студентам необхідні знання та навички, які будуть корисні при проходженні виробничої практики, підготовці до випускного кваліфікаційного проекту та у подальшій професійній діяльності.

4. Характеристика дисципліни:

4.1. Призначення навчальної дисципліни: Дисципліна «Об'єктно-орієнтоване програмування» є важливою складовою підготовки сучасних фахівців ІТ-сфери. Вона є багатогранною та досить широкою, але з її допомогою можна суттєво підвищити свій рівень знань.

4.2. Мета вивчення дисципліни: метою вивчення дисципліни Об'єктно-орієнтоване програмування» є формування у студентів теоретичних знань та набуття ними практичних навичок в галузі розробки та програмування мовою C#/.

4.3. Задачі вивчення дисципліни (відповідно до ОП): Основними завданнями вивчення дисципліни «Об'єктно-орієнтоване програмування» є формування у студентів компетентностей та програмних результатів навчання:

Номер в освітній програмі	Зміст компетентності	Номер теми, що розкриває зміст компетентності
Загальні компетентності		
К02	Здатність застосовувати знання у практичних ситуаціях.	1-20
К05	Здатність вчитися і оволодівати сучасними знаннями.	1-20
К06	Здатність до пошуку, оброблення та аналізу інформації з різних джерел.	1-20
К07	Здатність працювати в команді.	1-20
Спеціальні (фахові, предметні) компетентності		
К14	Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення.	1-20
К15	Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.	1-20
К23	Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя.	1-20
К27	Здатність до алгоритмічного та логічного мислення.	1-20
Програмні результати навчання за освітньою програмою		
ПР01	Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.	1-20
ПР02	Знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії програмного забезпечення і дотримуватись їх в професійній діяльності.	1-20
ПР03	Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення.	1-20
ПР05	Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.	1-20

ПР07	Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.	1-20
ПР12	Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення	1-20
ПР13	Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.	1-20
ПР15	Мотивовано обирати мови програмування та технології розробки для розв'язання завдань створення і супроводження програмного забезпечення.	1-20
ПР16	Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації.	1-20
ПР17	Вміти застосовувати методи компонентної розробки програмного забезпечення.	1-20
ПР19	Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.	1-20

4.4. **Зміст навчальної дисципліни:** відповідає навчальній та робочій програмі, яка відповідає запитам стейкхолдерів.

4.5. План вивчення дисципліни та оцінювання:

ТЕОРЕТИЧНИЙ БЛОК:

Навчальна діяльність	Робочий час студента (год.)
1	2
ТЕМА 1. ОСНОВИ ОБ'ЄКТНО-ОРІЄНТОВАНОГО АНАЛІЗУ ТА ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ***План:*** 1. Еволюція технології розробки програмного забезпечення. 2. Структурний та об'єктно-орієнтований підходи. 3. Поняття об'єктно-орієнтованого аналізу предметного середовища. 4. Загальні принципи побудови об'єктної модель предметного середовища. 5. Поняття об'єктно-орієнтованого проектування програмного забезпечення. ***Список рекомендованих джерел:*** *Основний: 1[с. 5-13], 2 [с. 6-14].* *Додатковий: 5 [с. 12-31], 6 [с. 5-13].* *Інтернет-ресурси: 8-11*	4
ТЕМА 2. ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ: СУЧАСНІ ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ВІЗУАЛЬНОГО ПРОГРАМУВАННЯ ***План:*** 1. Середовища візуального програмування. 2. Основні компоненти графічного інтерфейсу користувача. 3. Використання бібліотеки візуальних компонентів (меню, панелі	4

1	2
інструментів, шаблони діалогових вікон тощо). 4. Побудова та опис діаграм прецедентів. ***Список рекомендованих джерел:*** *Основний: 1 [с. 13-17], 2 [с. 14-20], 3 [с. 2-13].* *Додатковий: 5 [с. 31-36], 6 [с. 14-29].* *Інтернет-ресурси: 8-11*	
ТЕМА 3. ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ: ОСНОВИ ПРОГРАМУВАННЯ КЕРОВАНОГО ПОДІЯМИ ***План*** 1. Архітектура програм, орієнтованих на події. 2. Обробники подій від миші, клавіатури, команд меню, елементів управління тощо. 3. Побудова та опис діаграм взаємодії та компонентів. ***Список рекомендованих джерел:*** *Основний: 1 [с. 17-26], 2 [с. 20-33], 3 [с. 13-23].* *Додатковий: 5 [с. 36-50], 6 [с. 30-55].* *Інтернет-ресурси: 8-11*	4
ТЕМА 4. КОНЦЕПЦІЯ КЛАСІВ ТА ОБ'ЄКТІВ. ДІАГРАМИ КЛАСІВ. РОЗРОБКА КЛАСІВ ТА ОБ'ЄКТІВ МОВОЮ C# ***План*** 1. Опис класу. 2. Конструктори, деструктори класів. 3. Створення екземпляру класу. 4. Масиви об'єктів. 5. Статичні, константні члени класів. ***Список рекомендованих джерел:*** *Основний: 1 [с. 26-31], 2 [с. 34-69], 3 [с. 23-36, 48-61, 102-110].* *Додатковий: 5 [с. 50-59], 6 [с. 56-108].* *Інтернет-ресурси: 8-11*	4
ТЕМА 5. ОСНОВНІ ПОНЯТТЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ: ІНКАПСУЛЯЦІЯ ***План*** 1. Поняття інкапсуляції. 2. Директиви видимості класів C# за замовчанням. 3. Організація доступу до захищених полів класу. 4. Дружні функції та класи. ***Список рекомендованих джерел:*** *Основний: 1 [с. 32-39], 2 [с. 69-119], 3 [с.37-42, 111-113].* *Додатковий: 5 [с. 60-63], 6 [с. 175-182].* *Інтернет-ресурси: 8-11*	4
ТЕМА 6. ОСНОВНІ ПОНЯТТЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ: СПАДКОВІСТЬ, ПРОСТЕ ТА МНОЖИННЕ УСПАДКУВАННЯ ***План:*** 1. Поняття спадковості. 2. Опис класів-нащадків в C# при одиничній спадковості. 3. Директиви видимості при спадкуванні в C#. 4. Явний виклик конструкторів-предків. 5. Послідовність викликів конструкторів/деструкторів для ієрархії класів.	4

1	2
6. Множинна спадковість в C#. 7. Проблеми ромбовидної спадковості, віртуальні предки. ***Список рекомендованих джерел:*** *Основний: 1 [с. 39-45], 2 [с. 119-129], 3 [с.43-49, 113-120].* *Додатковий: 5 [с. 63-77], 6 [с. 184-187].* *Інтернет-ресурси: 8-11*	
ТЕМА 7. ОСНОВНІ ПОНЯТТЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ: ПОЛІМОРФІЗМ ***План*** 1. Поняття раннього та пізнього зв'язування. 2. Поняття перекриття та заміщення методів. 3. Механізм виклику заміщуваних методів, таблиці віртуальних методів. 4. Використання покажчиків на класи, як спосіб застосування поліморфізму. 5. Виклик заміщуваних методів нащадків з класів предків. ***Список рекомендованих джерел:*** *Основний: 1 [с. 46-57], 2 [с. 129-132], 3 [с. 120-125].* *Додатковий: 5 [с. 84-96], 6 [с. 187-191].* *Інтернет-ресурси: 8-11*	4
ТЕМА 8. ПОКАЖЧИКИ НА ОБ'ЄКТИ. ПЕРЕДАЧА ОБ'ЄКТІВ ЯК ПАРАМЕТРІВ ФУНКЦІЙ ***План*** 1. Операції динамічного виділення пам'яті new та delete мови C#. 2. Особливості передачі параметрів-об'єктів за значенням мови C#. 3. Використання покажчиків та посилань на об'єкти. 4. Використання конструктору копії об'єкту. 5. Повернення об'єкту в якості результату роботи функції. ***Список рекомендованих джерел:*** *Основний: 2 [с. 132-138].* *Додатковий: 6 [с. 191-231].* *Інтернет-ресурси: 8-11*	2
ТЕМА 9. КОНЦЕПЦІЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ: АБСТРАКЦІЯ. АБСТРАКТНІ КЛАСИ ТА МЕТОДИ ***План:*** 1. Поняття абстракції як аспекту об'єктно-орієнтованого програмування. 2. Абстрактні класи. 3. Абстрактні методи. 4. Використання абстрактних методів та класів. ***Список рекомендованих джерел:*** *Основний: 2 [с. 138-142].* *Додатковий: 5 [с. 73-78], 6 [с. 223-231].* *Інтернет-ресурси: 8-11*	4
ТЕМА 10. ПЕРЕВАНТАЖЕННЯ ФУНКЦІЙ ТА ПЕРЕВАНТАЖЕННЯ ОПЕРАЦІЙ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C# ***План:*** 1. Неоднозначність при перевантаженні 2. Сумісність типів параметрів. 3. Параметри-посилання. 4. Параметри за замовчуванням.	2

1	2
5. Перевантаження методів класу та конструкторів. 6. Перевантаження бінарних та унарних операцій. 7. Перевантаження операції індексування. 8. Перевантаження операції присвоєння. 9. Особливості присвоєння об'єктів мови C#. 10. Перевантаження операцій потокового введення-виведення. ***Список рекомендованих джерел:*** *Основний: 2 [с. 161-165], 3[с. 126-127].* *Додатковий: 5 [с. 80-84, 108-113], 6 [с. 231-265].* *Інтернет-ресурси: 8-11*	
ТЕМА 11. ІНТЕРФЕЙСИ ТА ЇХ РЕАЛІЗАЦІЯ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C#. КОНЦЕПЦІЯ ПРОЕКТУВАННЯ ЧЕРЕЗ ІНТЕРФЕЙСИ ***План:*** 1. Поняття інтерфейсу 2. Інтерфейс в сучасних умовах ООП 3. Задачі інтерфейсу при розробці програмного забезпечення 4. Типи інтерфейсів 5. Реалізація інтерфейсів 6. Прототипування шляхом використання інтерфейсів ***Список рекомендованих джерел:*** *Основний: 1 [с. 58-73], 2 [с. 149-152].* *Додатковий: 5 [с. 96-107], 6 [с. 5-13, 182-185].* *Інтернет-ресурси: 8-11*	4
ТЕМА 12. ДЕЛЕГАТИ; ПОНЯТТЯ ДЕЛЕГАТІВ; ПРИЗНАЧЕННЯ ТА ВИКОРИСТАННЯ ДЕЛЕГАТІВ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C# ***План:*** 1. Поняття: Делегати. 2. Область застосування делегатів. 3. Реалізація делегатів. 4. Комбіновані (групові) делегати. 5. Анонімні методи. 6. Оператор лямбди і лямбда виразів. ***Список рекомендованих джерел:*** *Основний: 2 [с. 165-168].* *Додатковий: 6 [с. 401-423].* *Інтернет-ресурси: 8-11*	4
ТЕМА 13. ПОДІЇ. ПОНЯТТЯ ПОДІЙ В ОБ'ЄКТНО-ОРІЄНТОВАНОМУ ПРОГРАМУВАННІ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C#. КОНЦЕПЦІЯ ВИКОРИСТАННЯ ПОДІЙ ТА ДЕЛЕГАТІВ ***План:*** 1. Поняття подій 2. Поняття подійно-орієнтованого програмування 3. Область застосування подій 4. Реалізація подій 5. Властивості подій ***Список рекомендованих джерел:*** *Основний: 2 [с. 168-170].* *Додатковий: 6 [с. 191-199, 359-401].* *Інтернет-ресурси: 8-11*	4
ТЕМА 14. УНІВЕРСАЛЬНІ ШАБЛОНИ ФУНКЦІЙ І КЛАСІВ НА	2

1	2
ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C# ***План:*** 1. Поняття шаблону функції. 2. Опис шаблону. 3. Створення екземплярів шаблону функції. 4. Відмінності перевантаження функції та шаблонів функції. 5. Опис шаблону-класу. 6. Створення екземплярів шаблону класу. ***Список рекомендованих джерел:*** *Основний: 3 [с. 81-90].* *Додатковий: 5 [с. 128-138], 6 [с. 113-145].* *Інтернет-ресурси: 8-11*	
ТЕМА 15. УНІВЕРСАЛЬНІ ШАБЛОНИ: ОБМЕЖЕННЯ УНІВЕРСАЛЬНИХ ШАБЛОНІВ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C# ***План:*** 1. Поняття обмеження універсальних шаблонів 2. Обмеження універсальних методів 3. Обмеження універсальних шаблонів типів 4. Типи обмежень та стандартні обмеження 5. Використання множинних універсальних параметрів ***Список рекомендованих джерел:*** *Основний: 3 [с. 91-101].* *Додатковий: 5 [с. 138-166], 6 [с. 315-355].* *Інтернет-ресурси: 8-11*	2
ТЕМА 16. ПОТОКИ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C#. БАГАТОПОТОЧНІСТЬ ***План:*** 1. Введення в поняття потоків 2. Огляд бібліотеки System.Threading та класу Thread 3. Створення потоків. Делегат ThreadStart 4. Потоки з параметрами ParameterizedThread ***Список рекомендованих джерел:*** *Основний: 2 [с. 170-174].* *Додатковий: 7 [с. 338-432].* *Інтернет-ресурси: 8-11*	2
ТЕМА 17. БАГАТОПОТОЧНІСТЬ: ОСНОВНІ КОНЦЕПЦІЇ СИНХРОНІЗАЦІЇ ПОТОКІВ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C# ***План:*** 1. Поняття синхронізації потоків 2. Методи синхронізації потоків 3. Монітори 4. Клас AutoResetEvent 5. Мютекси 6. Семафори ***Список рекомендованих джерел:*** *Основний: 2 [с. 174-179].* *Додатковий: 7 [с. 433-518].* *Інтернет-ресурси: 8-11*	2
ТЕМА 18. МЕТОДОЛОГІЯ ДЕБАГУ ПРОЄКТУ. ОБРОБЛЕННЯ ВИНЯТКОВИХ СИТУАЦІЙ НА ПРИКЛАДІ МОВИ ПРОГРАМУВАННЯ C#	2

1	2
План: 1. Розгляд питання обробки винятків. 2. Практична обробка винятків. 3. Ключове слово try. 4. Ключове слово catch. 5. Ключове слово finally. 6. Ключове слово throw. 7. Клас Exception. ***Список рекомендованих джерел:*** *Основний: 3 [с. 157-161].* *Додатковий: 5 [с. 177-185], 6 [с. 145-175].* *Інтернет-ресурси: 8-11*	
ТЕМА 19. ШАБЛОНИ ПРОЄКТУВАННЯ: ІСТОРІЯ ШАБЛОНІВ ПРОЕКТУВАННЯ, КОРИСТЬ ТА ПЕРЕВАГИ ШАБЛОНІВ ПРОЕКТУВАННЯ ***План:*** 1. Вступ до поняття шаблонів проектування (паттернів) 2. Історія виникнення патернів 3. Проблема яку вирішує патерн 4. Особливості реалізації в різних контекстах 5. Мотивація щодо вирішення проблеми способом, який пропонує патерн ***Список рекомендованих джерел:*** *Основний: 4 [с. 1-385].* *Інтернет-ресурси: 8-11*	2
ТЕМА 20. ШАБЛОНИ ПРОЄКТУВАННЯ: КЛАСИФІКАЦІЯ ШАБЛОНІВ ПРОЕКТУВАННЯ, НЕДОЛІКИ ШАБЛОНІВ ПРОЕКТУВАННЯ ***План:*** 1. Класифікація: породжуючі патерни 2. Класифікація: структурні патерни 3. Класифікація: поведінкові патерни 4. Недоліки шаблонів проєктування ***Список рекомендованих джерел:*** *Основний: 4 [с. 386-611].* *Інтернет-ресурси: 8-11*	
Всього	62

ПРАКТИЧНІ ЗАНЯТТЯ:

Навчальна діяльність	**Робочий час студента (год.)**	**Оцінювання (бал)**

1	2	3
Практична робота 1. ***Механізми проектування мовою UML*** **Мета:** Опанувати навички побудови UML діаграм. **Завдання:** Провести об'єктно-орієнтований аналіз та проектування мовою UML майбутнього програмного продукту.	8	5
Практична робота 2.	8	5

1	2	3
Інструментальні засоби візуального програмування **Мета:** Опанувати інструментальні засоби візуального програмування **Завдання:** Мотивація створення ООП, альтернативні підходи, тонкощі використання. Реалізація інкапсуляції та принципу DRY з та без ООП в C#.		
Практична робота 3. ***Визначення основ програмування керованого подіями.*** **Мета:** Закріпити знання та навички роботи з системними дерiктивами **Завдання:** Перетворити команди проміжкової мови Intermediate Language (IL) в команди процесора у відповідності з індивідуальним завданням.	8	5
Практична робота 4. ***Робота з класами та об'єктами мовою C#*** **Мета:** Закріпити основні знання та навички роботи з класами та конструкторами. **Завдання:** 1) Створити клас із ім'ям Address. У тілі класу потрібно створити поля: index, country, city, street, house, apartment. Для кожного поля створити властивість з двома методами доступу. Створити екземпляр класу Address. У поля екземпляра записати інформацію про поштову адресу. Виведіть на екран значення полів, що описують адресу. 2) Створити клас із ім'ям Rectangle. У тілі класу створити два поля, що описують довжини сторін double side1, side2. Створити власний конструктор Rectangle(double side1, double side2), в тілі якого поля side1 та side2 ініціалізуються значеннями аргументів. Створити два методи, що обчислюють площу прямокутника - double AreaCalculator() та периметр прямокутника - double PerimeterCalculator(). Створити дві властивості double Area та double Perimeter з одним методом доступу get. Написати програму, яка приймає від користувача довжини двох сторін прямокутника та виводить на екран периметр та площу. 3) Створити класи Point та Figure. Клас Point повинен містити два числових поля і одне рядкове поле. Створити три властивості з методом доступу get. Створити власний конструктор, у тілі якого проініціалізуйте поля значеннями аргументів. Клас Figure повинен містити конструктори, які приймають від 3 до 5 аргументів типу Point. Створити два методи: double LengthSide(Point A, Point B), що розраховує довжину сторони багатокутника; void PerimeterCalculator(), що розраховує периметр багатокутника. Написати програму, яка виводить на екран назву та периметр багатокутника.	8	5
Практична робота 5. ***Застосування принципу інкапсуляції*** **Мета:** Закріпити основні знання принципів використання	8	5

1	2	3
базових особливостей інкапсуляції. **Завдання:** Реалізувати інкапсуляцію в C#, використовуючи модифікатори доступу public, protected, internal, private, protected internal.		
Практична робота 6. ***Спадковість, просте та множинне успадкування*** **Мета:** Закріпити основні знання та навички використання та реалізації спадкування. **Завдання:** 1) Створіть клас Printer. У тілі класу створіть метод void Print(string value), що виводить на екран значення аргументу. Реалізуйте можливість того, щоб у разі успадкування від даного класу інших класів, та виклику відповідного методу їх екземпляра, рядки, передані як аргументи методів, виводилися різними кольорами. Обов'язково використовуйте типи. 2) Створити клас, який представляє навчальний клас ClassRoom. Створити клас учень - Pupil. У тілі класу створіть методи void Study(), void Read(), void Write(), void Relax(). Створити 3 похідні класу ExcelentPupil, GoodPupil, BadPupil від класу базового класу Pupil і перевизначити кожен із методів, залежно від успішності учня. Конструктор класу ClassRoom приймає аргументи типу Pupil, клас має складатися із 4 учнів. Необхідно передбачити можливість, що користувач може передати 2 або 3 аргументи. Виведіть інформацію про те, як усі учні екземпляра класу ClassRoom вміють вчитися, читати, писати, відпочивати. 3) Створіть клас DocumentWorker. У тілі класу створіть три методи OpenDocument(), EditDocument(), SaveDocument(). У тілі кожного з методів додайте виведення на екран відповідних рядків: "Документ відкритий", "Правка документа доступна у версії Про", "Збереження документа доступне у версії Про". Створіть похідний клас ProDocumentWorker. Перевизначте відповідні методи, при перевизначенні методів виводьте такі рядки: "Документ відредаговано", "Документ збережено у старому форматі, збереження в інших форматах доступне у версії Експерт". Створити похідний клас ExpertDocumentWorker від базового класу ProDocumentWorker. Перевизначте відповідний метод. При виклику даного методу необхідно виводити на екран "Документ збережений в новому форматі". У тілі методу Main() реалізуйте можливість прийому від користувача ключа доступу pro і exp. Якщо користувач не вводить ключ, він може користуватися тільки безкоштовною версією (створюється екземпляр базового класу), якщо користувач ввів номери ключа доступу pro і exp, то повинен створити екземпляр відповідної версії класу, наведений до базового – DocumentWorker.	8	5
Практична робота 7. ***Прикладне використання поліморфізму*** **Мета:** Закріпити основні знання та навички використання	8	5

1	2	3
поліморфізму. **Завдання:** Використовуючи можливості поліморфізму напишіть більш абстрактні, розширювані програми, один і той же код використовується для об'єктів різних класів, поліпшується читабельність коду. Поліморфізм має позбавити розробника від написання, читання і налагодження безлічі if-else / switch-case конструкцій.		
Практична робота 8 ***Операції NEW та DELETE*** **Мета:** Закріпити основні знання пов'язані з роботою з об'єктами. **Завдання:** 1. В індивідуальному завданні об'єднати всі дії, необхідні для створення об'єкта, в одну операцію, яку виконує оператор new. 2. За допомогою оператора delete оптимізуйте команду, яка знищує динамічно створений об'єкт MyType і звільняє займану їм пам'ять.	4	5
Практична робота 9 ***Абстрактні класи та методи*** **Мета:** Закріпити основні знання і навички реалізації та використання абстрактних методів та класів. **Завдання:** 1) Створіть клас AbstractHandler. У тілі класу створити методи void Open(), void Create(), void Chenge(), void Save(). Створити похідні класи XMLHandler, TXTHandler, DOCHandler від базового класу AbstractHandler. Написати програму, яка виконуватиме визначення документа і для кожного формату мають бути методи відкриття, створення, редагування, збереження певного формату документа. 2) **Приведене завдання необхідно вирішити використовуючи абстрактні методи.** Створіть клас DocumentWorker. У тілі класу створіть три методи OpenDocument(), EditDocument(), SaveDocument(). У тілі кожного з методів додайте виведення на екран відповідних рядків: "Документ відкритий", "Правка документа доступна у версії Про", "Збереження документа доступне у версії Про". Створіть похідний клас ProDocumentWorker. Перевизначте відповідні методи, при перевизначенні методів виводьте такі рядки: "Документ відредаговано", "Документ збережено у старому форматі, збереження в інших форматах доступне у версії Експерт". Створити похідний клас ExpertDocumentWorker від базового класу ProDocumentWorker. Перевизначте відповідний метод. При виклику даного методу необхідно виводити на екран "Документ збережений в новому форматі". У тілі методу Main() реалізуйте можливість прийому від користувача ключа доступу pro і exp. Якщо користувач не вводить ключ, він може користуватися тільки безкоштовною версією (створюється екземпляр базового класу), якщо користувач ввів номери ключа доступу pro і exp, то повинен створити екземпляр відповідної версії класу, наведений до базового –	8	5

1	2	3
DocumentWorker.		
Практична робота 10 **Мета:** Закріпити основні знання і навички перевантаження функція та операцій. **Завдання:** 1) Скласти й реалізувати алгоритми та програми мовою C#, які використовують функції, створені користувачем. У відповідності з індивідуальним завданням. 2) Перевизначити (перевантажити) звичні знаки операцій таким чином, щоб вони стосувались екземплярів класів. Наприклад, нехай визначений класс **complex**, що реалізує комплексне число. Також можна реалізувати дії з екземплярами цього класу як звертання до певних функцій класу, проте значно зручніше було б оперувати звичними для сприйняття виразами на кшталт: **complex a, b; complex c = a + b; c = a*b;** Завдання виконати у відповідності з індивідуальним завданням.	4	5
Практична робота 11 **Мета:** Закріпити основні знання і навички реалізації інтерфейсів. **Завдання:** 1) Створіть 2 інтерфейси IPlayable та IRecodable. У кожному з інтерфейсів створіть по 3 методи void Play() / void Pause() / void Stop() та void Record() / void Pause() / void Stop() відповідно. Створіть похідний клас Player від базових інтерфейсів IPlayable та IRecodable. Написати програму, яка виконує програвання та запис.	8	5
Практична робота 12 ***Використання делегатів та лямбда виразів*** **Мета:** Закріпити основні знання і навички реалізації та використання делегатів та лямбда виразів. **Завдання:** 1) Створіть чотири лямбда оператора для виконання арифметичних дій: (Add – додавання, Sub – віднімання, Mul – множення, Div – поділ). Кожен лямбда оператор повинен приймати два аргументи та повертати результат обчислення. Лямбда оператор поділу повинен перевірити поділ на нуль. Написати програму, яка виконуватиме арифметичні дії, вказані користувачем. 2) Створіть анонімний метод, який приймає як аргумент масив делегатів і повертає середнє арифметичне значення значень методів, повідомлених з делегатами в масиві. Методи, повідомлені з делегатами із масиву, повертають випадкове значення типу int. 3) Створіть анонімний метод, який приймає як параметри три цілих аргументи і повертає середнє арифметичне цих аргументів.	8	5
Практична робота 13 ***Реалізація та використання подій.*** **Мета:** Закріпити основні знання і навички використання	8	5

1	2	3
подій. **Завдання:** 1) Створіть програму секундоміру. Потрібно виводити показання секундоміра у вікні. Вікно має кнопки запуску, зупинки та скидання секундоміра. 2) Створіть калькулятор на чотири арифметичні дії (складання, віднімання, множення та поділ).		
Практична робота 14 ***Реалізація та використання універсальних шаблонів*** **Мета:** Закріпити основні знання і навички реалізації та використання універсальних шаблонів методів та класів. **Завдання:** 1) Створіть клас MyList<T>. Реалізуйте у найпростішому наближенні можливість використання його екземпляра аналогічно екземпляру класу List<T>. Мінімально необхідний інтерфейс взаємодії з екземпляром повинен включати метод додавання елемента, індексатор для отримання значення елемента за вказаним індексом і властивість тільки для читання для отримання загальної кількості елементів. 2) Створіть клас MyDictionary<TKey,TValue>. Реалізуйте у найпростішому наближенні можливість використання його екземпляра аналогічно екземпляру класу Dictionary. Мінімально необхідний інтерфейс взаємодії з екземпляром повинен включати метод додавання пар елементів, індексатор для отримання значення елемента за вказаним індексом і властивість тільки для читання для отримання загальної кількості пар елементів. 3) Створіть метод, що розширює: public static T[] GetArray<T>(this MyList<T> list). Застосуйте розширюючий метод до екземпляра типу MyList<T>, розробленого в завданні 2. Виведіть на екран значення елементів масиву, який повернув розширюючий метод GetArray( ).	4	5
Практична робота 15 ***Обмеження універсальних шаблонів*** **Мета:** Закріпити основні знання і навички реалізації та використання обмежень універсальних. **Завдання:** 1) У колекцію ArrayList, через виклик методу Add додайте елементи структурного та посилального типу, переберіть цю колекцію за допомогою циклу for – вирішіть виникшу проблему. 2) Створіть клас CarCollection<T>. Реалізуйте у найпростішому наближенні можливість використання екземпляра для створення парку машин. Мінімально необхідний інтерфейс взаємодії з екземпляром повинен включати метод додавання машин з назвою машини і року її випуску, індексатор для отримання значення елемента за вказаним індексом і властивість тільки для читання для	4	5

1	2	3
отримання загальної кількості елементів. Створіть спосіб видалення всіх машин автопарку. Обов'язково використовуйте обмеження. 3) Створіть клас Dictionary <TKey, TValue>. Реалізуйте у найпростішому наближенні можливість використання його екземпляра аналогічно екземпляру класу Dictionary із простору імен System.Collections.Generic. Мінімально необхідний інтерфейс взаємодії з екземпляром повинен включати метод додавання пар елементів, індексатор для отримання значення елемента за вказаним індексом і властивість тільки для читання для отримання загальної кількості пар елементів. Обов'язково використовуйте обмеження.		
Практична робота 16 ***Прикладне використання потоків*** **Мета:** Закріпити основні знання і навички реалізації та використання потоків. **Завдання:** Напишіть програму, в якій метод буде викликатись рекурсивною. Кожен новий виклик методу виконується окремому потоці.	4	5
Практична робота 17 ***Використання багатопоточності та синхронізація потоків*** **Мета:** Закріпити основні знання і навички реалізації та використання та синхронізації потоків. **Завдання:** 1) Створіть програму, яка виводить на екран ланцюжки падаючих символів. Довжина кожного ланцюжка визначається випадково. Перший символ ланцюжка – білий, другий символ – світло-зелений, решта символів темно-зелені. Під час падіння ланцюжка на кожному кроці всі символи змінюють своє значення. Дійшовши до кінця, ланцюжок зникає і зверху формується новий ланцюжок. 2) Розширте завдання 1 так, щоб в одному стовпці одночасно могло бути два ланцюжки символів.	4	5
Практична робота 18 ***Оброблення виняткових ситуацій*** **Мета:** Закріпити основні знання і навички реалізації та використання та синхронізації потоків. **Завдання:** 1) Створіть клас Calculator. У тілі класу створіть чотири методи для арифметичних дій: (Add – додавання, Sub – віднімання, Mul – множення, Div – поділ). Метод поділу повинен перевірити поділ на нуль, якщо перевірка не проходить - згенерувати виняток. Користувач вводить значення, над якими хоче зробити операцію та вибрати саму операцію. У разі виникнення помилок повинні викидатися винятки. 2) Описати структуру з ім'ям Worker, що містить такі поля: • прізвище та ініціали працівника;	4	5

1	2	3
• назва займаної посади; • рік початку роботи. Написати програму, яка виконує такі дії: • введення з клавіатури даних до масиву, що складається з п'яти елементів типу Worker (записи повинні бути впорядковані за абеткою); • якщо значення року введено не у відповідному форматі видає виняток. • виведення на екран прізвища працівника, стаж роботи якого перевищує введене значення. 3) Описати структуру з іменем Price, що містить такі поля: • Назва товару; • назва магазину, де продається товар; • вартість товару у гривнях. Написати програму, яка виконує такі дії: • введення з клавіатури даних до масиву, що складається з двох елементів типу Price (записи повинні бути впорядковані в алфавітному порядку за назвами магазинів); • виведення на екран інформації про товари, що продаються в магазині, назва якого введена з клавіатури (якщо такого магазину немає, вивести виняток).		
Практична робота 19 ***Вступ до шаблонів проектування*** **Мета:** Закріпити основні знання і базові навички використання шаблонів проектування. **Завдання:** Створити калькулятор (що має базові можливості: додавання, віднімання, ділення, множення), використовуючи шаблон проектування "Одинак".	4	5
Практична робота 20 ***Шаблони проектування та їх реалізація*** **Мета:** Закріпити основні знання і навички реалізації шаблонів проектування. **Завдання:** 1) Подайте звіт, що містить діаграми класів для 23 шаблонів проектування GoF і відповідний код для прикладу реалізації шаблону з результатами компіляції та виконання. Використовуючи Reverse Engineering, показати точну відповідність шаблону діаграми класів кожного прикладу. 2) Реалізувати шаблон "Абстрактна фабрика". Проект "Заводи з виробництва автомобілів". У проекті має бути реалізована можливість створювати автомобілі різних типів різних заводах. 3) Реалізувати шаблон "Адаптер". Проект "Годинник". У проекті має бути реалізований адаптер, який дає можливість користуватися годинником зі стрілками так само, як і цифровим годинником. У класі "Годинник зі стрілками" зберігаються повороти стрілок.	4	5

** всі практичні завдання виконуються на основі інтерактивних методів навчання у*

комп'ютерному середовищі

САМОСТІЙНА РОБОТА:

Навчальна діяльність	Робочий час студента (год.)	Оцінювання (бал)
1	2	3
Тема 1. Основи об'єктно-орієнтованого аналізу та проектування програмного забезпечення **Самостійна робота студентів.** Проаналізувати та закріпити лекційний матеріал. Розкрити наступні теми: 1. Визначення об'єктно-орієнтованого аналізу, проектування та програмування. 2. Основи об'єктно-орієнтованого проектування програмного забезпечення мовою UML. 3. Основні діаграми UML. 4. Правила побудови UML-діаграми класів ***Список рекомендованих джерел:*** *Основний: 1[с. 5-13], 2 [с. 6-14].* *Додатковий: 5 [с. 12-31], 6 [с. 5-13].* *Інтернет-ресурси: 8-11*	8	5
Тема 2. Об'єктно-орієнтоване програмування: сучасні інструментальні засоби візуального програмування **Самостійна робота студентів**. Опанувати дистанційно безкоштовний курс «SoloLearn C#». https://www.sololearn.com/learning/1080 ***Список рекомендованих джерел:*** *Основний: 1 [с. 13-17], 2 [с. 14-20], 3 [с. 2-13].* *Додатковий: 5 [с. 31-36], 6 [с. 14-29].* *Інтернет-ресурси: 8-11*	8	5

1	2	3
Тема 3. Об'єктно-орієнтоване програмування: основи програмування керованого подіями **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. За рахунок чого досягається міжмовна інтеграція у .Net Framework? 2. За рахунок чого досягається незалежність від платформи у .Net Framework? 3. Які основні можливості надає середовище Common Language Runtime? 4. Яка мова програмування є основною мовою програмування технології .Net Framework? 5. Що собою являє технологія .Net Framework? ***Список рекомендованих джерел:*** *Основний: 1 [с. 17-26], 2 [с. 20-33], 3 [с. 13-23].* *Додатковий: 5 [с. 36-50], 6 [с. 30-55].* *Інтернет-ресурси: 8-11*	8	5
Тема 4. Концепція класів та об'єктів. Діаграми класів. Розробка класів та об'єктів мовою c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке клас? 2. Що таке об'єкт? 3. Що таке екземпляр класу? 4. Що таке властивість? 5. Які види властивостей бувають? 6. Що таке модифікатори доступу та де їх використовують? 7. Які типи конструкторів ви знаєте? 8. Навіщо використовувати конструктори за замовчуванням? ***Список рекомендованих джерел:*** *Основний: 1 [с. 26-31], 2 [с. 34-69], 3 [с. 23-36, 48-61, 102-110].* *Додатковий: 5 [с. 50-59], 6 [с. 56-108].* *Інтернет-ресурси: 8-11*	8	5
Тема 5. Основні поняття об'єктно-орієнтованого програмування: інкапсуляція **Самостійна робота студентів**. Провести порівняльний аналіз модифікаторів доступу в C#, а саме public, protected, internal, private, protected internal. Порівняння підкріпити прикладами. ***Список рекомендованих джерел:*** *Основний: 1 [с. 32-39], 2 [с. 69-119], 3 [с.37-42, 111-113].* *Додатковий: 5 [с. 60-63], 6 [с. 175-182].* *Інтернет-ресурси: 8-11*	8	5
Тема 6. Основні поняття об'єктно-орієнтованого програмування: спадковість, просте та множинне успадкування	8	5

1	2	3
Самостійна робота студентів. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке спадкування? 2. Які недоліки наслідування ви знаєте? 3. Що таке модифікатори доступу та де їх використовують? 4. Для чого використовується ключове слово virtual? 5. Що таке Cast, Upcast, Downcast? 6. Чи поясніть призначення ключового слова sealed? 7. Доступ до членів базового класу з класу послідовника. 8. Доступ до членів базового класу, які оголошені як public, protected, internal та protected internal. 9. Члени базового класу з модифікатором доступу privat. ***Список рекомендованих джерел:*** *Основний: 1 [с. 39-45], 2 [с. 119-129], 3 [с.43-49, 113-120].* *Додатковий: 5 [с. 63-77], 6 [с. 184-187].* *Інтернет-ресурси: 8-11*		

1	2	3
Тема 7. Основні поняття об'єктно-орієнтованого програмування: поліморфізм **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Віртуальний метод – як, метод, який МОЖЕ бути перевизначений в класі-спадкоємця. 2. Абстрактний метод – як, метод, який повинен бути реалізований в класі-спадкоємця. 3. Перевизначення методу – як, зміна реалізації методу, встановленого як віртуальний. ***Список рекомендованих джерел:*** *Основний: 1 [с. 46-57], 2 [с. 129-132], 3 [с. 120-125].* *Додатковий: 5 [с. 84-96], 6 [с. 187-191].* *Інтернет-ресурси: 8-11*	8	5
Тема 8. Покажчики на об'єкти. Передача об'єктів як параметрів функцій **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Перелік бібліотечних файлів, які належать бібліотеці часу виконання C# 2. Перелік бібліотечних файлів, які належать до стандартної бібліотеки C# 3. Функції бібліотеки CRT. ***Список рекомендованих джерел:*** *Основний: 2 [с. 132-138].* *Додатковий: 6 [с. 191-231].* *Інтернет-ресурси: 8-11*	8	5
Тема 9. Концепція об'єктно-орієнтованого програмування: абстракція. Абстрактні класи та методи **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке абстрактний клас? 2. Що таке множинне спадкування? 3. Чим абстрактний клас відрізняється від конкретного? 4. Які члени можуть бути абстрактними? ***Список рекомендованих джерел:*** *Основний: 2 [с. 138-142].* *Додатковий: 5 [с. 73-78], 6 [с. 223-231].* *Інтернет-ресурси: 8-11*	8	5
Тема 10. Перевантаження функцій та перевантаження операцій на прикладі мови програмування c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Розкрити наступні запитання: 1. Перевантаження функцій - простий поліморфізм. 2. Поняття сигнатур**.** 3. Операторні функції 4. Бінарні операції 5. Декларація функції-члену класу для перевантаження бінарної операції.	8	5

1	2	3
Список рекомендованих джерел: *Основний: 2 [с. 161-165], 3[с. 126-127].* *Додатковий: 5 [с. 80-84, 108-113], 6 [с. 231-265].* *Інтернет-ресурси: 8-11*		

1	2	3
Тема 11. Інтерфейси та їх реалізація на прикладі мови програмування c#. Концепція проектування через інтерфейси **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке інтерфейс? 2. Чим абстрактний клас відрізняється від інтерфейсу? 3. Що таке множинне спадкування? ***Список рекомендованих джерел:*** *Основний: 1 [с. 58-73], 2 [с. 149-152].* *Додатковий: 5 [с. 96-107], 6 [с. 5-13, 182-185].* *Інтернет-ресурси: 8-11*	8	5
Тема 12. Делегати; поняття делегатів; призначення та використання делегатів на прикладі мови програмування c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке делегат? 2. Що таке комбінований делегат? 3. Що таке анонімний метод? 4. Які правила при виклику методів пов'язаних із делегатами? 5. Що таке лямбда вираз? 6. Чим лямбда вираз відрізняється від лямбда оператора? ***Список рекомендованих джерел:*** *Основний: 2 [с. 165-168].* *Додатковий: 6 [с. 401-423].* *Інтернет-ресурси: 8-11*	8	5
Тема 13. Події. Поняття подій в об'єктно-орієнтованому програмуванні на прикладі мови програмування c#. Концепція використання подій та делегатів **Самостійна робота студентів.** Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке подія? 2. Чи можуть бути події статичними? 3. Чи можуть бути події віртуальними? 4. Чи можуть бути події абстрактними? 5. Чи можуть бути події перевизначеними? 6. Опишіть сигнатуру методу обробника події, згідно з угодою щодо створення методів обробників подій? 7. Ключове слово event використовується для оголошення події у класі видавця або підписника? ***Список рекомендованих джерел:*** *Основний: 2 [с. 168-170].* *Додатковий: 6 [с. 191-199, 359-401].* *Інтернет-ресурси: 8-11*	8	5
Тема 14. Універсальні шаблони функцій і класів на прикладі мови програмування c# **Самостійна робота студентів**.	10	5

1	2	3
Проаналізувати та закріпити лекційний матеріал. Розкрийте тему у власних прикладах: Що відбувається при виклику функції, визначеної шаблоном? Перевантаження шаблонів може призвести до складних і неоднозначних виборів функцій-кандидатів. Дайте відповіді на наступні питання: 1. Що таке узагальнення? 2. Що таке закритий тип? 3. Що таке відкритий тип? 4. Поясніть поняття коваріантності та контраваріантності узагальнень. 5. Які переваги використання узагальнень? ***Список рекомендованих джерел:*** *Основний: 3 [с. 81-90].* *Додатковий: 5 [с. 128-138], 6 [с. 113-145].* *Інтернет-ресурси: 8-11*		
Тема 15. Універсальні шаблони: обмеження універсальних шаблонів на прикладі мови програмування c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дайте відповіді на наступні питання: 1. Які недоліки використання узагальнень? 2. Що таке тип Nullable? 3. Що таке операція поглинання? 4. Які види обмежень узагальнень ви знаєте? 5. Які ви знаєте типи обмежень для узагальнення? ***Список рекомендованих джерел:*** *Основний: 3 [с. 91-101].* *Додатковий: 5 [с. 138-166], 6 [с. 315-355].* *Інтернет-ресурси: 8-11*	10	5
Тема 16. Потоки на прикладі мови програмування c#. Багатопоточність **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке потік? 2. Який клас потрібно використовувати для створення екземпляра потоку? 3. Який делегат потрібно використовувати для передачі методу до потоку? 4. Яким чином можна передати параметри в потік? ***Список рекомендованих джерел:*** *Основний: 2 [с. 170-174].* *Додатковий: 7 [с. 338-432].* *Інтернет-ресурси: 8-11*	10	5

1	2	3
Тема 17. Багатопоточність: основні концепції синхронізації потоків на прикладі мови програмування c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке критична секція? 2. Що таке ресурс, що розділяється? 3. Що таке об'єкт синхронізації доступу до ресурсу, що розділяється? 4. Чим відрізняється використання оператора lock від класу Monitor? ***Список рекомендованих джерел:*** *Основний: 2 [с. 174-179].* *Додатковий: 7 [с. 433-518].* *Інтернет-ресурси: 8-11*	10	5
Тема 18. Методологія дебагу проєкту. Оброблення виняткових ситуацій на прикладі мови програмування c# **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке виняток? 2. Що таке конструкція try-catch? 3. Що таке конструкція try-catch-finally? 4. Як створити виняток користувача? 5. У яких випадках не спрацьовує блок? ***Список рекомендованих джерел:*** *Основний: 3 [с. 157-161].* *Додатковий: 5 [с. 177-185], 6 [с. 145-175].* *Інтернет-ресурси: 8-11*	10	5
Тема 19. Шаблони проєктування: історія шаблонів проектування, користь та переваги шаблонів проектування **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке патерн? 2. Для чого були створені патерни? 3. Які переваги використання патернів в прикладному програмуванні? ***Список рекомендованих джерел:*** *Основний: 4 [с. 1-385].* *Інтернет-ресурси: 8-11*	10	5
Тема 20. Шаблони проєктування: класифікація шаблонів проектування, недоліки шаблонів проектування **Самостійна робота студентів**. Проаналізувати та закріпити лекційний матеріал. Дати відповіді на наступні запитання: 1. Що таке породжуючі патерни? 2. Які породжуючі патерни ви знаєте? Для чого вони використовуються? 3. Що таке структурні патерни? 4. Які структурні патерни ви знаєте? Для чого вони використовуються?	10	5

1	2	3
5. Що таке поведінкові патерни? 6. Які поведінкові патерни ви знаєте? Для чого вони використовуються? 7. Які основні недоліки використання патернів? З чим вони пов'язані? ***Список рекомендованих джерел:*** *Основний: 4 [с. 386-611].* *Інтернет-ресурси: 8-11*		

5. Список рекомендованих джерел

Основний

1. Настенко, Д. В. Об'єктно-орієнтоване програмування. Частина 1. Основи об'єктно-орієнтованого програмування на мові C#: навчальний посібник для бакалаврів напряму підготовки 6.050701 «Електротехніка та електротехнології» програми професійного спрямування «Системи управління виробництвом та розподілом електроенергії»– Київ : НТУУ «КПІ», 2016. – 76 с.

2. Дібрівний О.А., Гребенюк В.В. Вступ до об'єктно орієнтованого програмування C#: Навчальний посібник. – К.: Державний університет телекомунікацій, 2018, - 190с.

3. Asaad, Renas. (2020). Object Oriented Programming C-Sharp Programming C# provides full support for object-oriented programming including encapsulation, inheritance, and polymorphism.

4. Eric Freeman, Elisabeth Robson. Head First Design Patterns, 2nd Edition, December 2020, O'Reilly Media, Inc., ISBN: 9781492078005

Додатковий

5. Simon Kendal. Object Oriented Programming using C#, 2011, Simon Kendal & Ventus Publishing ApS. ISBN 978-87-7681-814-2

6. Benjamin Perkins, Jon D. Reid. Beginning C# and .NET, 2021 Edition ISBN: 978-1-119-79578-0

7. Rishabh Verma, Neha Shrivastava, Ravindra Akella. Parallel Programming with C# and .NET Core: Developing Multithreaded Applications Using C# and .NET Core 3.1 from Scratch (English Edition). BPB Publications (June 26, 2020). ISBN-13: 978-9389423327

Інтернет-джерела

8. Microsoft Visual Studio. URL: https://visualstudio.microsoft.com/vs/

9. Object-Oriented programming (C#). URL: https://docs.microsoft.com/en-us/dotnet/csharp/fundamentals/tutorials/oop

10. C# Tutorial. URL: https://www.w3schools.com/cs/cs_oop.php

11. C# Corner. URL: https://www.c-sharpcorner.com/

**Курсивом зазначені джерела, що є в наявності в бібліотеці ДТЕУ*

6. Контроль та оцінювання результатів навчання:

Положення про оцінювання результатів навчання студентів і аспірантів наказ ДТЕУ №2891 від 16.09.2019р. (Електронний ресурс. Точка доступу: https://knute.edu.ua/file/NzU4MQ==/69da3a261374f213990591e6e9a812cd.pdf)

Під час вивчення дисципліни викладачем здійснюється поточний та підсумковий контроль. Поточний контроль та оцінювання передбачає:

- перевірку рівня засвоєння теоретичного матеріалу (тестування за матеріалами лекції, який здійснюється з використанням 365 Office);
- захист лабораторних робіт (проходить під час кожної лабораторної роботи);
- перевірка ходу виконання індивідуального завдання (фінальний проект);
- перевірка засвоєння матеріалу, що винесений на самостійне опрацювання під час фронтального опитування на лекції та заслуховування доповідей на обрані студентами теми.
-

ОЦІНЮВАННЯ ЗНАНЬ ЗДОБУВАЧІВ ОСВІТИ		
Сума балів, накопичених здобувачем вищої освіти за виконання всіх видів поточних навчальних завдань (робіт) на лабораторних/практичних заняттях, свідчить про ступінь оволодіння ним програмою освітньої компоненти на конкретному етапі її вивчення. Протягом семестру здобувачі освіти можуть набрати від 0 до 100 балів, що переводяться у національну шкалу оцінювання і відповідно у шкалу ЄКТС. Кількість балів відповідає певному рівню засвоєння дисципліни		
Довідник з розподілу оцінок ДТЕУ (Шкала ЄКТС):		
Бали ДТЕУ	Відсоток балів відносно загальної кількості одержаних прохідних балів	Кумулятивний відсоток отриманих прохідних балів
90-100	20	20
82-89	10	зо
75-81	20	50
69-74	10	60
60-68	40	100
Вимоги до критеріїв оцінювання самостійної роботи студента (оцінювання одного завдання у відсотковому еквіваленті)		
40%	Детальний розгляд сутності та вмісту основних джерел. Подання фактів, ідей і результатів досліджень у логічній послідовності. Правильно проаналізовано поточний стан дослідження проблеми та зроблено огляд перспектив подальшого розвитку даного питання.	
40%	Обґрунтованість аргументів, підтвердження особистого ставлення, пропозиції стосовно вирішення завдання, встановлення напрямків аналізу.	
20%	Оформлення звіту у відповідності вимог	
Критерії оцінювання самостійної роботи студента (оцінювання одного завдання у відсотковому еквіваленті)		
100%	В повному обсязі володіє навчальним матеріалом, вільно самостійно та аргументовано його викладає під час усних виступів та письмових відповідей, глибоко та всебічно розкриває зміст теоретичних питань та лабораторних завдань,	

	використовуючи при цьому обов'язкову та додаткову літературу. Правильно вирішив усі тестові завдання.
80%	Достатньо повно володіє навчальним матеріалом, обґрунтовано його викладає під час усних виступів та письмових відповідей, в основному розкриває зміст теоретичних питань та лабораторних завдань, використовуючи при цьому обов'язкову літературу. Але при викладанні деяких питань не вистачає достатньої глибини та аргументації, допускаються при цьому окремі несуттєві неточності та незначні помилки. Правильно вирішив більшість тестових завдань
60%	В цілому володіє навчальним матеріалом викладає його основний зміст під час усних виступів та письмових відповідей, але без глибокого всебічного аналізу, обґрунтування та аргументації, без використання необхідної літератури допускаючи при цьому окремі суттєві неточності та помилки. Правильно вирішив половину тестових завдань.
40%	Не в повному обсязі володіє навчальним матеріалом. Фрагментарно, поверхово (без аргументації та обґрунтування) викладає його під час усних виступів та письмових відповідей, недостатньо розкриває зміст теоретичних питань та лабораторних завдань, допускаючи при цьому суттєві неточності, правильно вирішив меншість тестових завдань.
20%	Частково володіє навчальним матеріалом не в змозі викласти зміст більшості питань теми під час усних виступів та письмових відповідей, допускаючи при цьому суттєві помилки. Правильно вирішив окремі тестові завдання.
0%	Не володіє навчальним матеріалом та не в змозі його викласти, не розуміє змісту теоретичних питань та практичних завдань. Не вирішив жодного тестового завдання.
ОСНОВНІ ПОЛОЖЕННЯ, ЩО РЕГЛАМЕНТУЮТЬ ОСВІТНІЙ ПРОЦЕС	
діючі положення	https://knute.edu.ua/blog/read/?pid=44402
нормативно-правова база організації освітнього процесу	https://knute.edu.ua/blog/read/?pid=7330&uk
студенту	https://knute.edu.ua/#forstudent
НЕФОРМАЛЬНА ОСВІТА Рекомендовані сертифікаційні програми, курси, посібники користувача	
GCP	https://cloud.google.com/docs/
AWS	https://www.alibabacloud.com/en/free
MS AZURE	https://learn.microsoft.com/uk-ua/training/azure/
Cloud Native Computing Foundation	https://denovo.ua/kaas?gad_source
Isaca	https://www.isaca.org/training-and-events
CSA (Cloud security alliance)	https://cloudsecurityalliance.org/research/artifacts

ПОЛІТИКА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ:	
Відвідування лекційних та лабораторних занять: відвідування	Відвідування лекційних та лабораторних занять є обов'язковим. Допускаються пропуски занять з таких поважних причин, як хвороба (викладачу надається копія довідки від медичного закладу), участь в олімпіаді, творчому конкурсі тощо за попередньою домовленістю та згодою викладача за умови дозволу деканату (надаються документи чи інші матеріали, які підтверджують заявлену участь у діяльності студента).
Відпрацювання пропущених занять:	відпрацювання пропущених занять є обов'язковим незалежно від причини пропущеного заняття. Лекційне заняття має бути відпрацьоване до наступної лекції на консультації викладача з використанням ПЗ 365 Office Teams. Відпрацювання лекційного матеріалу передбачає вивчення пропущеного теоретичного матеріалу та складання тесту за цим матеріалом. Лабораторне заняття відпрацьовується під час консультації викладача (розклад консультацій на сайті).
Правила поведінки під час занять	обов'язковим є дотримання техніки безпеки в комп'ютерних лабораторіях. Студенти повинні приймати активну участь в обговоренні навчально матеріалу ознайомившись з ним напередодні (навчальний матеріал надається викладачем). Мобільні пристрої дозволяється використовувати лише під час он-лайн тестування та підготовки практичних завдань в процесі заняття. Задля зручності, дозволяється використання ноутбуків та інших електронних пристроїв під час навчання в комп'ютерних аудиторіях (за взаємною згодою всіх учасників освітнього процесу)
Політика академічної доброчесності ДТЕУ	https://knute.edu.ua/blog/read/?pid=38987&uk