ЗАКЛАД ВИЩОЇ ОСВІТИ
«УНІВЕРСИТЕТ КОРОЛЯ ДАНИЛА»

Факультет суспільних і прикладних наук

Кафедра інформаційних технологій

ЗАТВЕРДЖУЮ
Проректор з методичної роботи
_____ Ярослав ШТАНЬКО
“___” _____ 2024 р.

ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

СИЛАБУС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Галузь знань:	12 «Інформаційні технології»
Спеціальність:	121 «Інженерія програмного забезпечення»
Освітньо-професійна (освітньо-наукова) програма:	Розробка та тестування програмного забезпечення
Освітній рівень:	перший (бакалаврський)
Статус дисципліни:	обов’язкова
Мова викладання, навчання та оцінювання:	українська

Івано-Франківськ
2024

2

Розробник:
професор кафедри ІТ
к.т.н., доц. Олег ПАШКЕВИЧ

ЗАТВЕРДЖЕНО
на засіданні кафедри інформаційних технологій
протокол № 5 від 19.12.2024 р.

завідувач кафедри Сергій ВАЩИШАК

УЗГОДЖЕНО:

Гарант ОПП Олександр ІВАНОВ
на засіданні кафедри інформаційних технологій
протокол № 5 від 19.12.2024 р.

СХВАЛЕНО:

на засіданні Науково-методичної ради, протокол №5 від 20.12 2024 р.

e-mail	oleh.pashkevych@ukd.edu.ua
Номер аудиторії чи кафедри	Кафедра інформаційних технологій, ауд 206
Посилання на сайт УКД	https://ukd.edu.ua
Сторінка курсу в СДО	Об'єктно-орієнтоване програмування

ВСТУП

Анотація навчальної дисципліни

Навчальна дисципліна “Об’єктно-орієнтоване програмування” є складовою освітньо-професійної програми підготовки фахівців за освітнім ступенем “бакалавр” галузі знань 12 “Інформаційні технології” спеціальності 121 “Інженерія програмного забезпечення”, освітньої програми “Розробка та тестування програмного забезпечення”.

Слухачі дисципліни повинні одержати теоретичні знання та практичні навички з використання об’єктно-орієнтованої парадигми при створенні програмного забезпечення в систематизованому вигляді та застосування принципів та шаблонів ООП під час розробки програмних систем на основі сучасних мов програмування.

Мета вивчення дисципліни – надання студентам знань в галузі розробки програмних продуктів. Необхідним навиком для успішної професійної кар’єри є вміння працювати з об’єктно-орієнтованими мовами, які є найбільш популярними на даний час. Дана дисципліна закладає ґрунтовну базу для подальшого розвитку в сфері розробки програмних продуктів.

Для досягнення мети поставлені такі основні **завдання:**

- створювати системне та прикладне програмне забезпечення комп’ютерних систем та мереж.
- оформляти отримані робочі результати у вигляді презентацій, науково-технічних звітів, статей і доповідей на науково-технічних конференціях.

У результаті засвоєння курсу студент повинен **знати:**

- засоби та задачі системного програмування, архітектуру та систему команд базового процесору, програмування з використанням актуальних мов програмування;
- принципи побудови об’єктно-орієнтованих програм;
- читати та будувати UML діаграми;
- текстові формати JSON та XML;
- основні шаблони об’єктно-орієнтованого проектування;
- SOLID принцип об’єктно-орієнтованого програмування.

студент повинен **вміти:**

- застосовувати знання для розв’язування задач аналізу та синтезу засобів, характерних для спеціальності;
- правильно вибрати структуру даних для конкретної задачі;
- інтегрувати необхідні структури даних в свої програми для вирішення поставлених задач.

Професійні компетентності та результати навчання, яких набувають здобувачі внаслідок вивчення навчальної дисципліни «Об’єктно-орієнтоване програмування» (шифри та зміст компетентностей та програмних результатів вказані відповідно до освітньої програми “Розробка та тестування програмного забезпечення”, введеної в дію ЗВО “Університет Короля Данила”)

Шифр та назва компетентності	Шифр та назва результату навчання
ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.	ПРН5. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об’єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення. ПРН6. Уміння вибирати та використовувати відповідні задачі методологію створення програмного забезпечення. ПРН12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.
ЗК2. Здатність застосовувати знання у практичних ситуаціях.	
ЗК7. Здатність працювати в команді.	
ФК2. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.	
ФК3. Здатність розробляти архітектури, модулі та компоненти програмних систем.	
ФК8.Здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв’язання завдань інженерії програмного забезпечення.	
ФК11. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення.	
ФК14. Здатність до алгоритмічного та логічного мислення	

ОПИС НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Курс	**2**		
Семестр	**4**		
Кількість кредитів ЄКТС	**6**		
Аудиторні навчальні заняття		**денна форма**	**заочна форма**
	лекції	**42 (в годинах)**	**8 (в годинах)**
	семінари, практичні	**42 (в годинах)**	**8 (в годинах)**
Самостійна робота		**96 (в годинах)**	**164 (в годинах)**
Форма підсумкового контролю	Екзамен	Екзамен	

Структурно-логічна схема вивчення навчальної дисципліни:

Пререквізити	**Постреквізити**
Основи програмування	Паралельні та розподілені обчислення
Алгоритми та структури даних	

ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Перелік тем лекційного матеріалу

Змістовий модуль 1

Тема 1. Вступ. ООП і якість програмного забезпечення (2 год.)

Вступ. Загальна інформація про проектування програмного забезпечення з використанням об’єктно-орієнтованого підходу, аналіз існуючих систем. Порівняння з іншими парадигмами програмування (функціональне, процедурне). Роль ООП у розробці програмного забезпечення.

Самостійне вивчення: ознайомитися із моделлю якості програмного забезпечення та вивчити її основні характеристики: Функціональність, Надійність, Зручність використання, Ефективність, Зручність супроводу, Переносимість.

Тема 2. Складові частини об’єктного підходу (2 год.)

Знайомство з парадигмою об’єктно-орієнтованого програмування та її основними концепціями: інкапсуляція, успадкування, поліморфізм, абстракція. Інші концепції: сувора типізація, паралелізм, збереженість. Принцип абстракції в ООП.

Самостійне вивчення: ознайомитися із іншими парадигмами програмування: структурною, функціональною, логічною.

Тема 3. Методологія ООП. Поняття об’єкта і класу (4 год.)

Огляд механізму який дозволяє реалізовувати об’єктно-орієнтований підхід. Створення класів та об’єктів. Поля (атрибути) та методи (функції класу). Конструктори та деструктори. Приховування реалізації та доступ до полів і методів: модифікатори доступу (public, private, protected). Основи успадкування, механізми і правила, множинне успадкування. Поняття поліморфізму.

Самостійне вивчення: ознайомитися із механізмами реалізації об’єктно-орієнтованої моделі, які пропонують різні мови програмування.

Тема 4. Уніфікована мова моделювання UML (4 год.)

Знайомство з мовою графічного опису систем. Нотація та метамодель мови UML. Типи UML-діаграм: структурні (діаграми варіантів використання, діаграма класів, об'єктів) та поведінкові (діаграма послідовностей, станів, активностей).

Самостійне вивчення: опрацювати книгу Martin Fowler. UML Distilled. A Brief Guide to the Standard Object Modeling Language. та ознайомитися всіма діаграмами, що пропонує UML. Ознайомитися із інструментами автоматичного створення коду на основі наданих UML діаграм. Використання UML-інструментів (наприклад, Visual Paradigm, StarUML або Lucidchart).

Тема 5. SOLID принципи об'єктно-орієнтованого програмування та дизайну (4 год.)

Знайомство з п'ятьма базовими принципами об'єктно-орієнтованого програмування та дизайну запропонованих Робертом Мартіном: принцип єдиного обов'язку; принцип відкритості/закритості; принцип підстановки Лісков, принцип розділення інтерфейсу, принцип інверсії залежностей.

Самостійне вивчення: ознайомитися із іншими принципами (KISS, DRY, YAGNI тощо).

Тема 6. Шаблони об'єктно-орієнтованого проєктування (8 год.)

Знайомство з шаблонами проектування: історія створення, призначення, способи та структура опису шаблонів, переваги та недоліки їх використання. Твірні шаблони проектування (Creational patterns). Шаблони поведінки (Behavioral patterns). Структурні шаблони (Structural patterns).

Самостійне вивчення: ознайомитися із анти-шаблонами: Copy and Paste Programming, Spaghetti code, Golden hammer, Magic numbers, Reinventing the wheel, God Object та ін. Ознайомитися із шаблонами, що не були подані на лекційному занятті. Визначити їх особливості застосування та зв'язок із іншими шаблонами.

Тема 7. Системні шаблони (System patterns) (2 год.).

Знайомство з системними шаблонами, які призначені для проектування структур з урахуванням системних вимог та обмежень.

Самостійне вивчення: ознайомитися із шаблонами, що не були подані на лекційному занятті. Визначити їх особливості застосування та зв'язок із іншими шаблонами.

Змістовий модуль 2

Тема 8. Синтаксис мови JAVA (4 год.)

Знайомство з мовою програмування JAVA, її особливостями, синтаксисом та допоміжними інструментами, які необхідні для роботи з даною мовою.

Самостійне вивчення: ознайомитися із синтаксисом інших об'єктно-орієнтованих мов програмування: C++, C#, Kotlin, Python.

Тема 9. Колекції об'єктів. Параметризація колекцій (6 год.)

Знайомство з колекціями, їх використання на практиці, розуміння відмінностей між ними та ефективне їх застосування відповідно до потреб. Використання параметризації для досягнення строгої типізації, використання параметризації в класах, методах та колекціях.

Самостійне вивчення: ознайомитися із різними способами перебору всіх елементів колекції: звичайний цикл із індексом, цикл foreach, ітератор.

Тема 10. Основи вводу/виводу, засоби обробки виключних подій (4 год.)

Знайомство з механізмом вводу та виводу інформації, використання його для рішення практичних задач. Огляд механізму обробки помилок та його використання в реальних проектах. Використання обробки виключень для підвищення надійності програм.

Самостійне вивчення: ознайомитися із особливостями послідовного та рандомізованого вводу/виводу.

Тема 11. Об’єктна модель пам’яті (2 години)

Пам’ять об’єктів у стеку та купі (heap, stack). Garbage Collection: автоматичне управління пам’яттю в мовах високого рівня.

Самостійне вивчення: Особливості роботи з пам’яттю в C++ (видалення вручну).

Зміст практичних занять

Змістовий модуль 1

Тема 1. Створення простих класів та об'єктів (4 год.)

Створення класів для моделювання реальних об'єктів.

Самостійне вивчення: ознайомитися із іншими парадигмами програмування: структурною, функціональною, логічною.

Тема 2. Створення класів із прихованими полями (4 год.)

Використання модифікаторів доступу для приховування полів класу. Робота з гетерами та сетерами.

Самостійне вивчення: ознайомитися із механізмами реалізації інкапсуляції, які пропонують різні мови програмування.

Тема 3. Конструктори та деструктори (4 год.)

Робота з конструкторами (стандартним, параметризованим, копіювальним) та деструктором для управління життєвим циклом об'єктів.

Самостійне вивчення: ознайомитися із механізмами створення об’єктів, які пропонують різні мови програмування.

Тема 4. Побудова ієрархії класів (4 год.)

Робота з ієрархією класів: використання базового та похідних класів, перевизначення методів.

Самостійне вивчення: ознайомитися із механізмами реалізації успадкування, які пропонують різні мови програмування. Множинне успадкування: його обмеження.

Тема 5. Побудова ієрархії класів на основі агрегації (композиції) (4 год.)

Використання агрегації та композиції для побудови ієрархії класів, які моделюють взаємозалежні об’єкти.

*Самостійне вивчення: приклади з реального життя, які ілюструють кожен із цих типів зв’язків (*агрегацію та композицію*). Проаналізуйте, чому використання композиції забезпечує сильнішу залежність між об'єктами порівняно з агрегацією. Які потенційні переваги та недоліки такої залежності у великих програмних системах?*

Змістовий модуль 2

Тема 6. Побудова складної ієрархії класів (4 год.)

Використання агрегації та композиції для побудови ієрархії класів, які моделюють взаємозалежні об’єкти.

*Самостійне вивчення: приклади з реального життя, які ілюструють кожен із цих типів зв’язків (*агрегацію та композицію*). Проаналізуйте, чому використання композиції забезпечує сильнішу залежність між об'єктами порівняно з агрегацією. Які потенційні переваги та недоліки такої залежності у великих програмних системах?*

Тема 7. Розробка UML діаграм (4 год.)

Створення UML-діаграм різних типів, використання системного підходу до аналізу та проектування програмних систем.

Самостійне вивчення: які сучасні інструменти для роботи з UML (наприклад, StarUML, Lucidchart, PlantUML) є найбільш зручними для створення діаграм?

Тема 8. Дотримання принципів SOLID (4 год.)

Вивчення принципів об’єктно-орієнтованого проєктування SOLID. Виявити переваги SOLID-архітектури через порівняння двох підходів, а також попрактикуватися в їх реалізації.

Самостійне вивчення: Як принципи SOLID поєднуються з іншими підходами до проєктування, такими як GRASP або дизайн-патерни? Які є альтернативи SOLID і в яких випадках вони можуть бути більш доцільними?

Тема 9. Структурні шаблони проєктування (6 год.)

Вивчення основ шаблону “Composite”, його застосування для роботи з ієрархічними структурами та переваги в уніфікованій обробці складних об’єктів.

Самостійне вивчення: Структурні шаблони ООП, їх особливості, переваги та недоліки, відносини з іншими шаблонами. Інтеграція структурних шаблонів із принципами SOLID?

Тема 10. Поведінкові шаблони проєктування (6 год.)

Вивчення основ шаблону “Observer”, його реалізацію та застосування в реальних сценаріях.

Самостійне вивчення: Поведінкові шаблони ООП, їх особливості, переваги та недоліки, відносини з іншими шаблонами. Інтеграція поведінкових шаблонів із принципами SOLID?

Зміст самостійної роботи здобувачів

Розподіл годин, виділених на вивчення дисципліни:

Найменування видів робіт	**Розподіл годин за формами навчання**	
	денна	**заочна**
Самостійна робота, год, у т.ч.:	**96**	**164**
Опрацювання матеріалу, викладеного на лекціях	30	10
Підготовка до практичних занять та контрольних заходів	30	20
Підготовка звітів з практичних робіт	16	20
Опрацювання матеріалу, винесеного на самостійне вивчення	20	114

ПОЛІТИКА КУРСУ

1) щодо системи поточного і підсумкового контролю

Організація поточного та підсумкового семестрового контролю знань студентів, проведення практик та атестації, переведення показників академічної успішності за 100-бальною шкалою в систему оцінок за національною шкалою здійснюється згідно з “Положенням про систему поточного і підсумкового контролю, оцінювання знань та визначення рейтингу здобувачів освіти”. Ознайомитись з документом можна за покликанням.

2) щодо оскарження результатів контрольних заходів

Здобувачі вищої освіти мають право на оскарження оцінки з дисципліни отриманої під час контрольних заходів. Апеляція здійснюється відповідно до «Положення про політику та врегулювання конфліктних ситуацій». Ознайомитись з документом можна за покликанням.

3) щодо відпрацювання пропущених занять

Згідно "Положення про організацію освітнього процесу" здобувач допускається до семестрового контролю з конкретної навчальної дисципліни (семестрового екзамену, диференційованого заліку), якщо він виконав усі види робіт, передбачені на семестр навчальним планом та силабусом/робочою програмою навчальної дисципліни, підтвердив опанування на мінімальному рівні результатів навчання (отримав ≥35 бали), відпрацював визначені індивідуальним навчальним планом всі лекційні, практичні, семінарські та лабораторні заняття, на яких він був відсутній. Ознайомитись з документом можна за покликанням.

4) щодо дотримання академічної доброчесності

"Положення про академічну доброчесність" закріплює моральні принципи, норми та правила етичної поведінки, позитивного, сприятливого, доброчесного освітнього і наукового середовища, професійної діяльності та професійного спілкування спільноти Університету, викладання та провадження наукової (творчої) діяльності з метою забезпечення довіри до результатів навчання. Ознайомитись з документом можна за покликанням.

5) щодо використання штучного інтелекту

"Положення про академічну доброчесність" визначає політику щодо використання технічних засобів на основі штучного інтелекту в освітньому процесі. Ознайомитись з документом можна за покликанням. "Положення про систему запобігання та виявлення академічного плагіату, самоплагіату, фабрикації та фальсифікації академічних творів" містить рекомендації щодо використання в академічних текстах генераторів на основі штучного інтелекту. Ознайомитись з документом можна за покликанням.

6) щодо використання технічних засобів в аудиторії та правила комунікації

Використання мобільних телефонів, планшетів та інших гаджетів під час лекційних та практичних занять дозволяється виключно у навчальних цілях (для уточнення певних даних, перевірки правопису, отримання довідкової інформації тощо). На гаджетах повинен бути активований режим «без звуку» до початку заняття. Під час занять заборонено надсилання текстових повідомлень, прослуховування музики, перевірка електронної пошти, соціальних мереж тощо, окрім виробничої необхідності. Під час виконання заходів контролю використання гаджетів заборонено (за винятком, коли це передбачено умовами його проведення). У разі

порушення цієї заборони результат анульовується без права перескладання. Комунікація відбувається через електронну пошту і сторінку дисципліни в Moodle.

7) щодо зарахування результатів навчання, здобутих шляхом формальної/інформальної освіти

Процедури визнання результатів навчання, здобутих шляхом формальної/інформальної освіти визначаються «Положенням про порядок визнання результатів навчання, здобутих шляхом неформальної та / або інформальної освіти». Ознайомитись з документом можна за покликанням.

Під час вивчення навчальної дисципліни "Об'єктно-орієнтоване програмування" студентам надається можливість перерахування неформальної освіти. До прикладу, із запропонованого переліку можна пройти сертифіковані (безкоштовні) курси на освітніх платформах, відтак сертифікат, який отримали під час навчання, – є підтвердженням засвоєння студентом окремих тем, що включені у зміст дисципліни.

МЕТОДИ НАВЧАННЯ

Програмний результат навчання	Метод навчання	Метод оцінювання
ПРН5. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.	Лекція, розповідь-пояснення, бесіда, комп'ютерні і мультимедійні методи, практичні роботи, дедуктивний, порівняння, проблемно-пошуковий, дослідницький, робота під керівництвом викладача,бесіда-діалог.	Екзамен, поточний контроль, усний контроль,поточний контроль, тестовий контроль.
ПРН6. Уміння вибирати та використовувати відповідні задачі методологію створення програмного забезпечення.	Лекція, розповідь-пояснення, бесіда, комп'ютерні і мультимедійні методи, практичні роботи, дедуктивний, порівняння, проблемно-пошуковий, дослідницький, робота під керівництвом викладача,бесіда-діалог.	Екзамен, поточний контроль, усний контроль,поточний контроль, тестовий контроль.
ПРН12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.	Лекція, розповідь-пояснення, демонстрування, комп'ютерні і мультимедійні методи, індуктивний метод, інтерактивні методи (дискусія, диспут, мозковий штурм), виокремлення основного	Поточний контроль, тестовий контроль, екзамен

ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Вид	Зміст	% від загальної оцінки	Бал	
			min	max
Поточні контрольні заходи	всього	60	35	60
Підсумкові контрольні заходи	екзамен	40	24	40
Всього:	-	100	60	100

Процедура проведення контрольних заходів, а саме поточного контролю знань протягом семестру та підсумкового семестрового контролю (екзамена) – регулюється «Положенням про систему поточного та підсумкового контролю оцінювання знань та визначення рейтингу студентів», яке розміщено на сайті університету в розділі «Публічна інформація».

Фіксація **поточного** контролю здійснюється в "Електронному журналі обліку успішності академічної групи" на підставі чотирибальної шкали - "2"; "3"; "4"; "5". У разі відсутності студента на занятті виставляється "н". За результатами поточного контролю у Журналі, автоматично визначається підсумкова оцінка, здійснюється підрахунок пропущених занять.

Усі пропущені заняття, а також негативні оцінки студенти зобов'язані відпрацювати впродовж трьох наступних тижнів. У випадку недотримання цієї норми, замість "н" в журналі буде виставлено "0" (нуль балів), без права перездачі. Відпрацьоване лекційне заняття в електронному журналі позначається літерою «в».

Критерії оцінювання

До підсумкового контролю допускаються студенти які за результатами поточного контролю отримали не менше 35 балів. Усі студенти, що отримали 34 балів і менше, не допускаються до складання підсумкового контролю і на підставі укладання додаткового договору, здійснюють повторне вивчення дисципліни впродовж наступного навчального семестру. За результатами підсумкового контролю (диференційований залік/екзамен) студент може отримати 40 балів. Студенти, які під час підсумкового контролю отримали 24 бали і менше, вважаються такими, що не здали екзамен/диференційований залік і повинні йти на перездачу.

Загальна семестрова оцінка з дисципліни, яка виставляється в екзаменаційних відомостях оцінюється в балах (згідно з **Шкалою оцінювання знань за ЄКТС**) і є сумою балів отриманих під час поточного та підсумкового контролю.

Шкала оцінювання знань за ЄКТС:

Оцінка за національною шкалою	Рівень досягнень, %	Шкала ECTS
Національна диференційована шкала		
Відмінно	90 – 100	A
Добре	83 – 89	B
	75 – 82	C
Задовільно	67 – 74	D
	60 – 66	E
Незадовільно	35 – 59	FX
	0 – 34	F
Національна недиференційована шкала		
Зараховано	60 – 100	-
Не зараховано	0 – 59	-

Студенти, які не з’явилися на заліки/екзамени без поважних причин, вважаються такими, що одержали незадовільну оцінку.

РЕКОМЕНДОВАНІ ДЖЕРЕЛА ІНФОРМАЦІЇ

1. Олександр Швець. Занурення в патерни проектування. Refactoring.Guru, 2021. 396 с.
2. Cay Horstmann. Core Java Volume 1 Fundamentals: 11th Edition. Pearson, 2018. 928 p.
3. Herbert Schildt. Java: The Complete Reference, Eleventh Edition. – McGraw Hill, 2018. 1248 p.
4. Андрій Будай. Дизайн-патерни просто, як двері. : URL: https://sites.google.com/site/designpatternseasy/
5. Ковалюк Т. В. Основи програмування [Розд. 6.3: Основні концепції об’єктно-орієнтованої методології програмування] : підручник. К.: Вид. гр. BHV, 2005. 384 с.: іл.
6. Жуковський С.С., Вакалюк Т.А. Об’єктно-орієнтоване програмування мовою С++ [Електронний ресурс]: навч.-метод. посіб. – Житомир: Вид-во ЖДУ, 2016. – 100 с. – Режим доступу: https://core.ac.uk/download/pdf/84273987.pdf. (дата звернення: 18.08. 2022).
7. Конспект лекцій з навчальної дисципліни «Об’єктно-орієнтоване програмування” [Електронний ресурс] / уклад. І. Б. Шевчук; ЛНУ ім. Івана Франка. – Львів, 2020. – 192 с. – Режим доступу:

https://financial.lnu.edu.ua/wp-content/uploads/2017/09/OOP_konspekt-lektsiy_CH_1.pdf (дата звернення: 18.08. 2022).

8. Бублик В. В. Об’єктно-орієнтоване програмування [Електронний ресурс]: підручник. – К.: ІТ-книга, 2015. – 640 с.: іл. – Режим доступу: https://cutt.ly/EMRkGQg (дата звернення: 18.08. 2022).

9. Scott Tilley. Systems Analysis and Design (MindTap Course List): 12th Edition. Cengage Learning: 2019. – 576 p.

10. Grady Booch. Unified Modeling Language User Guide: 2nd Edition. Addison-Wesley Professional: 2017. – 504 p.

11. Eric Freeman. Head First Design Patterns: Building Extensible and Maintainable Object-Oriented Software: 2nd Edition / Eric Freeman, Elisabeth Robson. – O’Reilly Media: 2021. – 669 p.

12. David Hay. UML and Data Modeling: A Reconciliation. – Technics Publications: 2015. – 243 p.