Код та назва дисципліни	**1-ф05-06 Методи та парадигми декларативного програмування / Methods and paradigms of declarative programming**
Рекомендується для галузі знань	F (12) Інформаційні технології
Кафедра	Кафедра інженерії програмного забезпечення та інформаційних технологій
П.І.П. НПП (за можливості)	
Рівень ВО	Перший (бакалаврський)
Курс, семестр *(в якому буде викладатись)*	2-3 курси
Мова викладання	українська
Пререквізити (передумови вивчення дисципліни)	Для успішного опанування курсу здобувач **повинен володіти:** - основами програмування (бажано імперативні мови — Python, C++, Java); - базовими уявленнями про життєвий цикл ПЗ (SDLC); - дискретною математикою (логіка, булева алгебра, теорія відношень); - алгоритмами і структурами даних; - фундаментами системного аналізу. **Бажано:** - базове розуміння теорії програмування; - досвід написання програм середньої складності.
Чому це цікаво/треба вивчати	Розглянуто **сукупність методів, що охоплюють різні етапи життєвого циклу ПЗ**: - аналіз вимог → формалізація знань (логічне програмування); - проєктування → абстрактні моделі (функціональне програмування); - реалізація → декларативні підходи; - тестування → формальні методи верифікації. **Ключова ідея:** підвищення ефективності розробки за рахунок використання декларативних моделей замість покрокових алгоритмів. **Prolog (логічна парадигма):** - працює з фактами і правилами; - природний для задач штучного інтелекту, експертних систем; - ефективний на стадії аналізу та формалізації знань. **Lisp (функціональна парадигма):** - гнучка обробка символів і структурування коду; - сприяє швидкому створенню прототипів; - використовується на етапах проєктування та реалізації. **Переваги:** скорочення часу розробки; підвищення надійності; краща підтримуваність коду; зручність моделювання складних систем.
Перелік тем з дисципліни	**1. Основи методів розроблення ПЗ** Життєвий цикл програмного забезпечення (SDLC). Моделі розробки (Waterfall, Agile). Роль декларативного програмування у SDLC. **2. Теоретичні основи декларативного програмування** Парадигми програмування. Імперативний vs декларативний підхід. Формальні методи і специфікації. **3. Логічне програмування (Prolog)** Предикатна логіка. Факти, правила, запити. Уніфікація та механізм виводу. Побудова баз знань. Застосування у експертних системах, системах підтримки прийняття рішень. **4. Функціональне програмування (Lisp)** Основи Lisp і S-вирази. Рекурсія та обробка списків. Функції вищого порядку. Макроси та метапрограмування. Швидке прототипування

	систем. **5. Методи на різних стадіях життєвого циклу** Формалізація вимог (Prolog). Проєктування абстрактних моделей (Lisp). Декларативна реалізація. Тестування та верифікація. **6. Інтеграція і практичне застосування** Поєднання декларативного та імперативного підходів. Використання у сучасних системах (AI, data processing). Метрики ефективності розробки.
Як можна користуватися набутими знаннями і уміннями *(компетентності)*	Здобувач набуває здатностей: **Загальні компетентності:** абстрактне та логічне мислення; формалізація задач; системний підхід до розробки ПЗ. **Фахові компетентності:** застосування методів декларативного програмування на різних етапах SDLC; побудова баз знань і логічних моделей (Prolog); розробка функціональних програм і прототипів (Lisp); аналіз та вибір оптимальної парадигми; інтеграція різних підходів програмування. **Практичні навички:** створення експертних систем; розробка інтелектуальних застосунків; оптимізація програмних рішень; використання декларативних мов у задачах AI і Data Science.
Очікувані результати навчання	**Знати:** принципи декларативного програмування; роль різних парадигм у життєвому циклі ПЗ; основи Prolog і Lisp; методи формалізації знань. **Вміти:** застосовувати Prolog для представлення знань і логічного виводу; використовувати Lisp для створення гнучких програмних рішень; проєктувати системи з використанням декларативних підходів; аналізувати ефективність методів розробки. **Володіти:** навичками створення декларативних програм; методами підвищення ефективності розробки; інструментами моделювання та прототипування; підходами до формального опису задач.
Інформаційне забезпечення	**Основна література**: 1. Clocksin W., Mellish C. Programming in Prolog 2. Winston P., Horn B. Lisp 3. Norvig P. Paradigms of Artificial Intelligence Programming **Додаткова література**: 4. Abelson H., Sussman G. Structure and Interpretation of Computer Programs 5. Sterling L., Shapiro E. The Art of Prolog **Інтернет-ресурси**: https://www.swi-prolog.org (Prolog), https://lisp-lang.org (Lisp), документація GNU CLISP, SBCL, навчальні платформи (Coursera, edX) **Програмне забезпечення**: SWI-Prolog, GNU CLISP/SBCL, Emacs/VS Code (з підтримкою Lisp і Prolog), інтерактивні середовища розробки
Види навчальних занять *(лекції, практичні, семінарські, лабораторні заняття тощо)*	Лекції Лабораторні заняття
Вид семестрового контролю	диференційований залік
Максимальна кількість здобувачів	90
Мінімальна кількість здобувачів *(тільки для мовних та творчих дисциплін)*	

Декан факультету _____ Олена КІСЕЛЬОВА