



# СТРУКТУРА ТА ІНТЕРПРЕТАЦІЯ КОМП'ЮТЕРНИХ ПРОГРАМ

## Робоча програма навчальної дисципліни (Силабус)

### Реквізити навчальної дисципліни

Рівень вищої освіти	<i>Перший (бакалаврський)</i>
Галузь знань	<i>11 Математика та статистика</i>
Спеціальність	<i>113 Прикладна математика</i>
Освітня програма	<i>Наука про дані та математичне моделювання</i>
Статус дисципліни	<i>Вибіркова</i>
Форма навчання	<i>очна(денна)</i>
Рік підготовки, семестр	<i>2 курс, весняний семестр</i>
Обсяг дисципліни	<i>4 кредити</i>
Семестровий контроль/ контрольні заходи	<i>Залік/Контрольні (2) та лабораторні (5) роботи</i>
Розклад занять	<i>Лекції — 1 раз на тиждень (18 лекцій), лабораторні — 1 раз на тиждень (18 занять)</i>
Мова викладання	<i>Українська/Англійська</i>
Інформація про керівників курсу / викладачів	Лектор: <i>Борисенко Павло Борисович, <a href="mailto:pavlo.borysenko@gmail.com">pavlo.borysenko@gmail.com</a></i> Практичні: <i>Борисенко Павло Борисович, <a href="mailto:pavlo.borysenko@gmail.com">pavlo.borysenko@gmail.com</a>;</i> <i>Громова Вікторія Вікторівна, <a href="mailto:vikvikgrom@gmail.com">vikvikgrom@gmail.com</a></i>
Розміщення курсу	<i>Канали #fp-course та #fp-labs у кафедральному Slack</i>

### Програма навчальної дисципліни

#### 1. Опис навчальної дисципліни, її мета, предмет вивчення та результати навчання

*Функційне програмування є однією з двох — поруч із об'єктно-орієнтованим програмуванням — домінуючих парадигм у сучасних підходах до розробки програмного забезпечення. Розуміння принципів функційного програмування дозволяє проектувати стабільні та підтримувані застосунки за менший проміжок часу. Концепції функційного програмування — імутабельність даних, функції вищих порядків, монади — дедалі частіше зустрічаються навіть у традиційно нефункційних мовах та виборюють все більше прихильників через лаконічність, декларативність та верифікованість, яку вони привносять у код.*

*В рамках цього курсу ми розглянемо основні концепції функціонального програмування, що дозволять вам легше орієнтуватися у сучасних трендах та краще зрозуміти зв'язок між математичними концепціями та програмуванням, а також розглянемо базові принципи побудови комп'ютерних програм та їх відображення у мовах програмування.*

*На лабораторних роботах ви зможете з нуля опанувати мову програмування Scheme — функційну мову, діалект Lisp — від простих структур даних до мета-циклічного обчислювача.*

*Курс «Структура та інтерпретація комп'ютерних програм» є просунутим вибіркоким курсом підготовки спеціалістів у сфері науки про дані та математичного моделювання, що базується на основі однойменного класичного підручника з MIT та дозволяє краще осягнути математичні засновки концепцій програмування.*

Метою курсу є вивчення:

- основних понять і концепцій функційного програмування,
- базових конструкцій та синтаксису мови Scheme,
- принципів та підходів до функційного та декларативного вирішення алгоритмічних задач;
- принципів роботи з імутабельними та рекурсивними структурами даних;
- принципів побудови комп'ютерних програм та їх відображення в структурі самих мов.

Предметом вивчення є програмний код мовою Scheme та його синтаксичні примітиви; імутабельні та рекурсивні структури даних; математичні основи теорії типів.

Після засвоєння дисципліни студенти матимуть наступні:

- компетентності:
  - здатність до абстрактного мислення, аналізу та синтезу;
  - знання та розуміння предметної області та розуміння професійної діяльності;
  - здатність розробляти алгоритми та структури даних, програмні засоби та програмну документацію;
  - здатність використовувати сучасні технології програмування та тестування програмного забезпечення;
  - здатність зрозуміти постановку завдання, сформульовану мовою певної предметної галузі, здійснювати пошук та збір необхідних вихідних даних;
- знання:
  - синтаксису мови Scheme;
  - про рекурсивні та алгебраїчні типи даних, їх представлення у пам'яті комп'ютера;
  - принципів рекурсії і рекурсивної обробки даних;
  - принципів роботи «лінивих обчислень»;
  - принципів побудови та ефективної обробки імутабельних даних;
  - принципів роботи компілятора та систем виведення типів;
  - принципів побудови комп'ютерних програм на основі мовних примітивів;
- уміння:
  - представляти задачу у вигляді композиції перетворень вихідних даних;
  - конструювати та працювати із імутабельними типами даних;
  - використовувати функції вищих порядків для структурування програм та контролю за перетворенням даних;
  - читати та розуміти функційний код;
  - використовувати функційні патерни та підходи разом з іншими парадигмами;
- навички:
  - представлення циклічних процесів рекурсивно;
  - роботи з рекурсивними структурами даних (деревами, списками тощо);
  - аналізу процесу виконання програм;
  - роботи з мета-циклічними обчисленнями;
  - поєднання об'єктно-орієнтованого та функційного підходів до вирішення задач;
- досвід:
  - програмування функційних примітивів мовою Scheme;
  - використання функційних патернів;
  - верифікації програм компілятором;
  - роботи із рекурсивними структурами даних.

## **2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)**

*Дисципліна вивчається у весняному семестрі 2 курсу та базується на результатах навчання з дисциплін:*

- Алгоритми і структури даних
- Дискретна математика
- Математична логіка та теорія алгоритмів
- Програмування
- Програмування мовою Python
- Об'єктно-орієнтоване програмування.

*Успішне проходження курсу допоможе студентам у вивченні наступних дисциплін з навчального плану підготовки бакалаврів:*

- Аналіз даних
- Криптографічні методи захисту інформації
- Алгоритми і системи комп'ютерної математики.

## **3. Зміст навчальної дисципліни**

*Розділ 1. Побудова абстракцій.*

*Тема 1.1. Елементи програмування.*

*Тема 1.2. Функції як головний примітив.*

*Тема 1.3. Абстракція даних, рекурсивні типи.*

*Тема 1.4. Патерн-матчинг та генералізовані операції.*

*Розділ 2. Модулярність, об'єкти і стан.*

*Тема 2.1. Присвоєння, стан і побічні ефекти.*

*Тема 2.2. Потоки.*

*Розділ 3. Металінгвістична абстракція.*

*Тема 3.1. Мета-циклічний обчислювач.*

*Тема 3.2. Логічне програмування.*

*Тема 3.3. Регістрові машини.*

*Тема 3.4. Компіляція.*

## **4. Навчальні матеріали та ресурси**

### **Підручники**

*Оскільки цей курс базується на конкретному підручнику, ми рекомендуємо саме його:*

1. Harold Abelson, Gerald Jay Sussman, Julie Sussman, *Structure and Interpretation of Computer Programs*, 2 ed., 1996.

*Англійська версія: <https://mitpress.mit.edu/sites/default/files/sicp/full-text/book/book.html>*

*Відеокурс від авторів книги:*

*[https://ocw.mit.edu/courses/6-001-structure-and-interpretation-of-computer-programs-spring-2005/video\\_galleries/video-lectures/](https://ocw.mit.edu/courses/6-001-structure-and-interpretation-of-computer-programs-spring-2005/video_galleries/video-lectures/)*

*Це класичний підручник для CS-курсів у MIT, так звана «книга чарівника», який оглядає велику кількість питань що стосуються абстрактних елементів побудови програм та їх вплив на мову програмування.*

*Розділи 1-4 — обов'язкові.*

*Розділ 5 за бажанням.*

## Додаткові матеріали

1. Документація Scheme. *The Revised [6] Report on the Algorithmic Language Scheme*.  
Посилання: <http://www.r6rs.org/>
2. R. Kent Dybvig, *The Scheme Programming Language*.  
Посилання: <https://www.scheme.com/tspl3/>  
Це хороший огляд основ мови програмування Scheme.
3. Daniel P. Friedman and Matthias Felleisen, *The Little Schemer*, 4th ed., 1995.  
Це допоміжний підручник зі Scheme. Він написаний простою мовою і не концентрується на великих концепціях, надаючи перевагу практичним прийомам.

## Навчальний контент

### 5. Методика опанування навчальної дисципліни (освітнього компонента)

#### Лекції

##### Розділ 1. Побудова абстракцій

Лекція 1. Елементи програмування: обчислення виразів, редукція виразів, функції, умовні вирази та предикати. Синтаксис Scheme.

Лекція 2. Функції як головний примітив. Функції вищих порядків. Складені дані.

Лекція 3. Функції і дані. Іммутабельність даних. Ешерівська схема Гендерсона. Символічна диференціація.

Лекція 4. Співставлення патернів. Заміна за правилами. Генералізовані операції.

##### Розділ 2. Модулярність, об'єкти і стан

Лекція 5. Присвоєння, стан і побічні ефекти. Обчислювальні об'єкти. Змінні локального стану. Модель середовища обчислень.

Лекція 6. Потоки. Потоки як відкладені списки. Парадигма потоків. Паралельні обчислення.

##### Розділ 3. Металінгвістична абстракція

Лекція 7. Метациклічний обчислювач. Дані як програма. Відокремлення синтаксичного аналізу від виконання. Лінійні обчислення.

Лекція 8. Логічне програмування. Система запитів. Отримання дедуктивної інформації.

Лекція 9. Регістрові машини. Обчислювач із безпосереднім контролем. Стек як модель рекурсії. Асемблер.

Лекція 10. Компіляція. Виділення пам'яті та збір сміття.

#### Лабораторні роботи

##### Розділ 1. Побудова абстракцій

Лабораторна 0. Функційна та імперативна парадигми програмування.

Завдання: реалізувати просту програму маніпуляції списком на Scheme та C. Порівняти підходи до вирішення типових задач у функційній та імперативній парадигмах.

Лабораторна 1. Прості процедури і типи даних.

Завдання: написати функцію маніпуляції гетероморфним списком. Підготувати алгебраїчну рекурсивну структуру даних, що описує дерево заданого типу. Вирішення простих фізичних задач.

Лабораторна 2. Прості алгоритми на АД. Патерн-матчінг та рекурсія. Функції вищих порядків. Складні структури даних.

*Завдання: вирішити задачі на роботу із матрицями, реалізувати власні функції вищих порядків.*

## **Розділ 2. Модулярність, об'єкти і стан**

*Лабораторна 3. Дерева і їх обхід.*

*Завдання: написати функції для роботи зі структурою дерева, використовуючи функції вищих порядків. Вивести приклади роботи цих функцій. Написати простий веб-кродлер, використовуючи дерево як базову структуру даних.*

*Лабораторна 4. Об'єкти і стани.*

*Завдання: побудувати просту гру-пригоду, використовуючи власну абстракцію об'єктів.*

### **6. Самостійна робота студента**

*До самостійного опрацювання виносяться:*

- підготовка до аудиторних та практичних занять — до 2 годин на тиждень;
- підготовка до контрольних робіт та заліку — до 10 годин за семестр;
- виконання лабораторних робіт — до 50 годин;
- виконання домашньої практичної роботи — до 20 годин;
- вивчення наступних тем:
  - Основні синтаксичні одиниці Scheme.
  - Стандартні функції вищих порядків (*map, filter, foldl/foldr*).
  - Об'єкти та робота із ними.

## **Політика та контроль**

### **7. Політика навчальної дисципліни**

#### **Відвідування**

*Відвідування лекцій необов'язкове, але ми заохочуємо студентів не пропускати лекційні заняття через можливість ставити уточнюючі питання та брати участь у живому обговоренні. Записи лекцій (та оглядової частини лабораторних занять) поточного чи минулих років будуть доступні онлайн.*

*Виконання і захист лабораторних робіт обов'язкове. Активність на лабораторних заняттях становить 50% семестрового рейтингу, тому не варто нехтувати нею.*

*Пропущені контрольні роботи можна перездати за погодженням із викладачем.*

#### **Оцінювання**

*Оцінювання контрольних та практичних робіт відбувається з точністю до десятих, округлення за звичними правилами.*

#### **Дедлайни**

*Контрольні роботи мають бути здані у рамках часу, відведеного на їх проведення.*

*Лабораторні роботи та домашня практична робота мають окремі визначені терміни (дедлайни). Роботи, здані після цих термінів, будуть оцінюватися з модифікатором:*

- здані після **софт дедлайну** (окремі для кожної роботи, але не раніше ніж за 2 тижні після отримання завдання) — 0,5, тобто отримують половину балів;
- здані після **хард дедлайну** (тиждень до заліку) — не отримують балів.

#### **Додаткові бали**

*Активність на лекціях та лабораторних заняттях — відповіді на запитання викладача, знаходження помилок у лекційних чи лабораторних матеріалах; питання, що свідчать про*

вдумливу роботу з навчальним матеріалом, надання оригінальних рішень у лабораторних чи домашній практичній роботах тощо — заохочується додатковими балами на розсуд викладача.

Крім того, заохочується додатковими балами підтверджене сертифікатами проходження курсів (онлайн чи офлайн), що стосуються тем дисципліни.

Протягом курсу можна отримати не більше 10 додаткових балів.

### **Академічна доброчесність**

Ми підтримуємо принципи академічної доброчесності і рівності всіх студентів. У випадку виявлення випадків списування (у контрольних, лабораторних, домашніх роботах) чи плагіату — бали за відповідні роботи будуть анульовані. Повторні порушення принципів академічної доброчесності можуть призвести до недопуску до складання заліку.

Викладачі можуть перевіряти роботи, виконані у рамках курсу, за допомогою систем виявлення плагіату Unicheck та MOSS.

Якщо не зазначено іншого, усі контрольні заходи проводяться у форматі «відкритої книги». Це означає, що ви маєте право користуватися будь-якими ресурсами, окрім допомоги сторонніх осіб. Ми довіряємо нашим студентам і покладаємо надію на те, що вони не порушать цю довіру.

## **8. Види контролю та рейтингова система оцінювання результатів навчання (PCO)**

**Поточний контроль:**

- **модульні контрольні роботи (20%):**

10 балів x 2 роботи = **20** балів

Модульні контрольні роботи представляють собою практичне завдання за темою модуля. На виконання кожної модульної контрольної роботи виділяється 2 години. Ці роботи проводяться замість лекційних занять. Вас буде заздалегідь попереджено про проведення контрольної роботи.

- **захист лабораторних робіт (50%):**

10 балів x 5 робіт = **50** балів

Захист лабораторних робіт складається із демонстрації коду та дієздатності програми, а також короткої співбесіди. На кожну лабораторну роботу виділяється кілька занять.

Лабораторні роботи мають дедлайни як описано вище.

- **домашня практична робота (30%):**

30 балів x 1 робота = **30** балів

Домашня практична робота складається із 10 комплексних завдань на написання чи доповнення/редагування коду, кожне з яких оцінюється у 1-5 балів. Після перевірки роботи, у студента є можливість один раз виправити помилки для підвищення оцінки.

Домашня практична робота має дедлайни як описано вище.

**Календарний контроль:** проводиться двічі на семестр як моніторинг поточного стану виконання вимог силабусу. Календарний контроль проводиться за результатами лабораторних занять, проведених на момент початку контролю.

**Семестровий контроль:** залік.

Залікова оцінка виставляється на основі семестрового рейтингу, або — за бажанням студента чи при семестровому рейтингу менше 60 балів — за результатами написання залікової контрольної роботи.

Залікова контрольна робота складається із 10 практичних питань 3 рівнів складності (по 3 питання 1 та 3 рівнів та 4 питання рівня 2), що оцінюються у 5, 10 і 15 балів за кожне відповідно.

Умови допуску до семестрового контролю: мінімальний рейтинг не нижче 25 балів та виконання усіх лабораторних робіт.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою:

<i>Кількість балів</i>	<i>Оцінка</i>
100-95	Відмінно
94-85	Дуже добре
84-75	Добре
74-65	Задовільно
64-60	Достатньо
Менше 60	Незадовільно
Не виконані умови допуску	Не допущено

#### **9. Додаткова інформація з дисципліни (освітнього компонента)**

- *додатки до силабусу — приклади залікової контрольної роботи, положення про PCO;*
- *зазвичай залік проходить на останньому занятті з дисципліни, відповідно, хард дедлайн буде за тиждень до цього;*
- *у випадку проведення курсу дистанційно, результати контролю (контрольні та залікові роботи) мають бути виконані в цифровому вигляді: як текстові файли, чи файли з кодом, або за неможливості — у вигляді розбірливих фото. Усі такі матеріали мають бути завантажені на указаний викладачем ресурс у терміни відведені під відповідний тип контролю. Ми надаватимемо буферні 5 хвилин на випадок форс-мажорних подій. У окремих випадках залікова контрольна може бути проведена у формі співбесіди. Аналогічно, окремі форми захисту можуть проводитися заочно, через листування чи месенджери.*

#### **Робочу програму навчальної дисципліни (силабус):**

**Складено** асистентом Борисенком Павлом Борисовичем

**Ухвалено** кафедрою ПМА (протокол № 7 від 09.02.2022)

**Погоджено** Методичною комісією факультету прикладної математики (протокол № 6 від 25.03.2022)