Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Кафедра прикладної математики

ФУНКЦІЙНЕ ПРОГРАМУВАННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни	
Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*11 Математика та статистика*
Спеціальність	*113 Прикладна математика*
Освітня програма	*Наука про дані та математичне моделювання*
Статус дисципліни	*Вибіркова*
Форма навчання	*очна(денна)*
Рік підготовки, семестр	*2 курс, весняний семестр*
Обсяг дисципліни	*4 кредити*
Семестровий контроль/ контрольні заходи	*Залік/Контрольні (2) та лабораторні (4) роботи*
Розклад занять	*Лекції — 1 раз на тиждень (18 лекцій),* *лабораторні — 1 раз на тиждень (18 занять)*
Мова викладання	*Українська/Англійська*
Інформація про керівників курсу / викладачів	Лектор: *Борисенко Павло Борисович, pavlo.borysenko@gmail.com* Практичні: *Борисенко Павло Борисович, pavlo.borysenko@gmail.com;* *Громова Вікторія Вікторівна, vikvikgrom@gmail.com*
Розміщення курсу	Канали #fp-course та #fp-labs у кафедральному Slack Лекції: *https://www.youtube.com/playlist?list=PLjBFEyMhtyMtNPE7clj9Vh64puM0sag6J*

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 — іммутабельність 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 на лабораторних роботах ви зможете з нуля опанувати мову програмування Haskell — класичну чисто функційну мову, — від простих структур даних до монад, та навіть автоматичного доведення теорем.

Курс «Функційне програмування» є просунутим вибірковим курсом підготовки спеціалістів у сфері науки про дані та математичного моделювання, що дозволяє краще осягнути математичні засновки концепцій програмування.

Метою курсу є вивчення:

- *основних понять і концепцій функційного програмування,*
- *базових конструкцій та синтаксису мови Haskell,*

- *принципів та підходів до функційного та декларативного вирішення алгоритмічних задач;*
- *принципів роботи з іммутабельними та рекурсивними структурами даних.*

Предметом вивчення є програмний код мовою Haskell та його синтаксичні примітиви; іммутабельні 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 Haskell;*
  - *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 іммутабельних даних;*
  - *функціональних патернів (монад, станів, ROP тощо);*
  - *про принципи автоматичного доведення теорем;*
  - *принципів роботи компілятора ти системи виведення типів Хіндлі-Міллера;*
- *уміння:*
  - *представляти задачу у вигляді композиції перетворень вихідних даних;*
  - *конструювати та працювати із іммутабельними 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);*
  - *роботи з типами-сумами та типами-добутками;*
  - *роботи зі строго типізованою мовою програмування;*
- *досвід:*
  - *програмування функційних примітивів мовою Haskell;*
  - *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. Принципи функційного програмування.

Тема 1.1. Лямбда-числення.

Тема 1.2. Функції як головний примітив.

Тема 1.3. Алгебраїчні типи даних, рекурсивні типи.

Тема 1.4. Рекурсія та ліниві обчислення.

Тема 1.5. Тайпкласи і конструктори типів.

Розділ 2. Монади.

Тема 2.1. Основи теорії категорій для програмістів.

Тема 2.2. Монади Maybe та IO.

Тема 2.3. Управління виконанням програми.

Розділ 3. Функціональні патерни і спеціальні теми.

Тема 3.1. Лямбда-куб і системи типів.

Тема 3.2. Монада async і абстракція паралельних обчислень.

Тема 3.3. Теорія типів та автоматичне доведення теорем.

4. Навчальні матеріали та ресурси

Підручники

Ви можете обрати один з наступних підручників:

1. *Miran Lipovača, Learn You a Haskell for Great Good. / Міран Ліповача, Вивчіть собі Хаскела на велике щастя*
   Українська версія: https://haskell.trygub.com/
   Англійська версія: http://learnyouahaskell.com/chapters
   Це найбільш розмірений і збалансований підручник.
   Розділ 14 — не обов’язковий.
2. *Bryan O’Sulllivan et al., Real World Haskell, 2008.*
   Посилання: http://book.realworldhaskell.org/read/
   Цей підручник більше концентрується на практичних застосуваннях Гаскела.
   Розділи 1-4, 6-7, 9, 13-15, 18-20, 23-24 — обов’язкові.
   Решта — на ваш вибір, вони покривають великий масив цікавих прикладних задач.
3. *Антон Холомьёв, Учебник по Haskell.*
   Посилання: https://anton-k.github.io/ru-haskell-book/book/home.html

Додаткові матеріали

1. *Документація Haskell.*
   Посилання: https://hoogle.haskell.org/
2. *Bartosz Milewski, Category Theory for Programmers.*

Посилання:
https://github.com/hmemcpy/milewski-ctfp-pdf/releases/download/v1.3.0/category-theory-for-programmers.pdf
Це хороший огляд теорії категорій для програміста, однак він вимагає хорошої математичної підготовки.

3. *F# for Fun and Profit.*
   Посилання: https://fsharpforfunandprofit.com/site-contents/
   Це сайт присвячений програмуванню на F#, але тут є багато корисних статей загальної тематики. Рекомендуємо звернути увагу на дві:
   - *Functional Programming Design Patterns.*
   - *Railway Oriented Programming.*

Рекомендовані онлайн-курси

1. *Graham Hutton. Functional Programming*
   https://www.youtube.com/playlist?list=PLF1Z-APd9zK7usPMx3LGMZEHrECUGodd3
2. *EdX. Introduction to Functional Programming*
   https://www.edx.org/course/introduction-to-functional-programming
3. *Future Learn. Functional Programming in Haskell*
   https://www.futurelearn.com/courses/functional-programming-haskell

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Лекції

Розділ 1. Принципи функціонального програмування

Лекція 1. Лямбда-числення: обчислення виразів, лямбда-функції, бета-редукція та ета-розширення.

Лекція 2. Функції вищих порядків і карування. Чисті функції і прозорість посилань. Функції як об'єкти. Синтаксис Haskell.

Лекція 3. Функції і дані. Іммутабельність даних. Алгебраїчні типи даних, типи-суми і типи-добутки. Рекурсивні типи. Maybe.

Лекція 4. Співставлення патернів. Відкладені обчислення. Рекурсія та оптимізація хвостової рекурсії. Let-вирази та синтаксичне зв'язування.

Лекція 5. Тайпкласи: Num, Functor, Applicative, Monad. Власні тайпкласи.

Розділ 2. Монади

Лекція 6. Поняття групи, моноїда, функтора та монади. Список як монада.

Лекція 7. Maybe як монада. do-нотація.

Лекція 8. Програма як перетворення станів. Стан як композиція функцій. Монада State.

Лекція 9. Монада IO та інкапсуляція сторонніх ефектів. Зв'язування, ліфти.

Лекція 10. Railway-oriented programming.

Розділ 3. Функціональні патерни та спеціальні теми

Лекція 11. Лямбда-куб, системи і виведення типів. Система типів Хіндлі-Міллера.

Лекція 12. Монада async. Абстракція паралельних обчислень. Композиція Кляйслі.

Лекція 13. Теорія типів та автоматичне доведення теорем. Аксіома унівалентності.

Лабораторні роботи

Розділ 1. Принципи функціонального програмування

Лабораторна 0. Функційна та імперативна парадигми програмування.

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

Лабораторна 1. Алгебраїчні типи даних.

Завдання: написати функцію маніпуляції гетероморфним списком. Підготувати алгебраїчну рекурсивну структуру даних, що описує дерево заданого типу. Написати приклад валідного дерева та реалізувати його виведення.

Розділ 2. Монади

Лабораторна 2. Прості алгоритми на АТД. Патерн-матчінг та рекурсія. Функції вищих порядків. Карування та композиція.

Завдання: написати функції для роботи зі структурою дерева, описаного у минулій лабораторній роботі, використовуючи функції вищих порядків. Вивести приклади роботи цих функцій. Реалізувати власні функції вищих порядків.

Лабораторна 3. Тайпкласи

Завдання: побудувати тайпклас для опису властивостей дерева, описаного у попередніх роботах. Реалізувати стандартні тайпкласи для цього дерева, якщо потрібно.

Розділ 3. Функціональні патерни та спеціальні теми

Лабораторна 4 (опціональна). Автоматичне доведення теорем.

Завдання: довести кілька примітивних теорем щодо натуральних чисел, використовуючи систему типів та компілятор Haskell в якості верифікатора.

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:*
  - *Синтаксис роботи зі списками у Lisp.*
  - *Основні синтаксичні одиниці Haskell.*
  - *Гетероморфні списки.*
  - *Стандартні функції вищих порядків (map, filter, foldl/foldr) та тайпкласи.*

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 40% 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 отримають половину балів;*
- *здані після* **хард дедлайну** *(тиждень до заліку) — не отримають 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 (РСО)

Поточний контроль:

- ***модульні контрольні роботи*** *(30%):*
  15 балів х 2 роботи = ***30*** *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*** *(40%):*
  10 балів х 4 робіт = ***40*** *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 х 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 питань 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:

Кількість балів	*Оцінка*
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 РСО;*
- *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)