Навчальна дисципліна

Парадигма функціонального програмування

Галузі знань: *10 Природничі науки, 11 Математика та статистика, 12 Інформаційні технології, 15 Автоматизація та приладобудування, 16 Хімічна та біоінженерія, 17 Електроніка та телекомунікації, 19 Архітектура та будівництво, 27 Транспорт*

Рівень вищої освіти	*перший (бакалаврський)*
Статус дисципліни	*вибіркова (Дисципліна індивідуального вибору 3)*
Обсяг дисципліни	150 годин / 5 кредитів ЄКТС
Мова викладання	*українська*
Що буде вивчатися (предмет вивчення)	Функціональне програмування є парадигмою програмування, у якій основна увага приділяється функціям як основним будівельним блокам програми. Мови програмування **F# та Scala (і не тільки)** є потужним інструментом для функціонального програмування та мають багатий набір функціональних можливостей. *1.* ***Вступ до функціонального програмування*****:** історія та еволюція функціонального програмування, мотивація та основні концепції, порівняння з іншими парадигмами програмування (імперативна, об'єктно-орієнтована). *2.* ***Чисті функції:*** визначення та важливість чистих функцій, переваги чистих функцій для тестування та налагодження, приклади чистих функцій в різних мовах програмування. *3.* ***Імутабельність:*** принципи незмінності даних, переваги імутабельності для паралельних обчислень та безпеки коду, використання імутабельних структур даних. *4.* ***Функції вищого порядку:*** функції, що приймають інші функції як аргументи або повертають функції як результат, приклади функцій вищого порядку та їх застосування, каррінг (currying) та часткове застосування функцій. *5.* ***Лямбда-вирази та анонімні функції*****:** визначення та використання лямбда-виразів, приклади анонімних функцій у різних мовах, переваги та недоліки використання анонімних функцій. *6.* ***Рекурсія:*** основи рекурсії та її відмінності від ітерації, приклади рекурсивних алгоритмів, оптимізація рекурсії за допомогою хвостової рекурсії. *7.* ***Функціональні структури даних****:* списки, кортежі, множини та інші структури даних, використання функціональних структур даних для обробки інформації, порівняння з імперативними структурами даних. *8.* ***Зіставлення зі зразком:*** основи співставлення з шаблоном (pattern matching), приклади використання співставлення з шаблоном для обробки складних структур даних, переваги співставлення з шаблоном для чистоти та зрозумілості коду. *9.* ***Ледаче обчислення:*** принципи та механізми ледачого обчислення, переваги та недоліки ледачого обчислення, приклади застосування ледачого обчислення для підвищення продуктивності. *10.* ***Композиція функцій:*** основи композиції функцій, приклади створення складних функцій з простих, переваги композиції функцій для модульності та повторного використання коду. *11.* ***Монадична обробка даних:*** вступ до монад та їх використання, приклади монад в різних мовах програмування, використання монад для обробки побічних ефектів та асинхронних операцій
Як можна користуватися набутими знаннями і уміннями (компетентності)	***Розширення набору навичок.*** Вивчення функціонального програмування дозволяє програмісту розширити свій набір навичок та підходів до розробки. Це дозволяє вирішувати завдання ефективніше, а також забезпечує нові інструменти та техніки, які можна застосовувати у своїй роботі.

<table>
<tr><td></td><td colspan="3">Розробка якіснішого коду. Функціональне програмування звертає увагу на чистоту, ясність та безпеку коду. Вивчення функціонального програмування допомагає програмісту писати більш читаний, модульний та надійний код.
Розробка більш гнучких рішень. Функціональне програмування дозволяє створювати гнучкі та абстрактні рішення, використовуючи концепції, такі як функції вищих порядків та карування. Це дозволяє програмісту створювати код, який легко адаптується до змін вимог чи умов.
Підвищення продуктивності та ефективності. Функціональне програмування передбачає використання незмінних даних та уникнення побічних ефектів, що може сприяти підвищенню продуктивності та спрощенню тестування програмного забезпечення.
Робота з паралельними та асинхронними операціями. Функціональне програмування пропонує ефективні інструменти для роботи з паралельними та асинхронними операціями. Вивчення функціонального програмування дозволяє програмісту освоїти ці інструменти та створювати масштабовані та чуйні програми.
Розуміння альтернативних підходів. Вивчення функціонального програмування дозволяє програмісту отримати розуміння альтернативних підходів до програмування та розширити своє мислення у розробці програмного забезпечення</td></tr>
<tr><td>Пререквізити</td><td colspan="3">Бажано «Основи програмування» та «Комп'ютерна дискретна математика»</td></tr>
<tr><td>Кафедра</td><td colspan="3">Інженерії програмного забезпечення (603)</td></tr>
<tr><td>Факультет</td><td colspan="3">Програмної інженерії та бізнесу</td></tr>
<tr><td>Викладач</td><td>  </td><td>ПІБ</td><td>Кузнецова
Юлія Анатоліївна</td></tr>
<tr><td></td><td></td><td>Посада</td><td>доцент кафедри 603</td></tr>
<tr><td></td><td></td><td>Вчене звання</td><td>доцент</td></tr>
<tr><td></td><td></td><td>Науковий ступінь</td><td>кандидат технічних наук</td></tr>
<tr><td></td><td></td><td>e-mail</td><td>y.kuznetsova@khai.edu</td></tr>
<tr><td></td><td></td><td>Веб-сторінка</td><td>https://se.khai.edu/kuznecova/</td></tr>
<tr><td>Посилання на електронні матеріали курсу</td><td colspan="3">https://mentor.khai.edu/course/view.php?id=8443</td></tr>
<tr><td>Посилання на силабус</td><td colspan="3"></td></tr>
</table>