КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Кафедра теорії та технології програмування

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи
_____ Олена КАШПУР
«07» травня 2021 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ФОРМАЛЬНІ МЕТОДИ РОЗРОБКИ ПРОГРАМНИХ СИСТЕМ

для студентів

галузь знань	**12 «Інформаційні технології»**
спеціальність	**122 «Комп'ютерні науки»**
освітній рівень	**магістр**
освітня програма	**«Інформатика»**
вид дисципліни	**обов'язкова**

Форма навчання	**денна**
Навчальний рік	**2021/2022**
Семестр	**2**
Кількість кредитів ECTS	**4**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**іспит**

Викладачі: **к.т.н., доцент Олексій ТКАЧЕНКО**

Пролонговано: на 20__/20__ н.р. _____(_____) «__»___ 20__р.

на 20__/20__ н.р. _____(_____) «__»___ 20__р.

КИЇВ – 2021

Розробник: Олексій ТКАЧЕНКО, к. т. н., доцент, доцент кафедри теорії та технології програмування.

ЗАТВЕРДЖЕНО

Завідувач кафедри
теорії та технології програмування
_____ Микола НІКІТЧЕНКО

Протокол № 10 від « 27 » квітня 2021 року

Схвалено гарантом освітньо-наукової програми «Інформатика»

_____ Степан ШКІЛЬНЯК

« 6 » травня 2021 року

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики

Протокол від « 6 » травня 2021 року № 10

Голова науково-методичної комісії _____ Людмила ОМЕЛЬЧУК

« 6 » травня 2021 року

1. Мета дисципліни – засвоєння основних концепцій, принципів, методів та інструментів розробки програмних систем з використанням формальних підходів для створення сучасних програмних високої якості.

2. Попередні вимоги до опанування або вибору навчальної дисципліни

1. Знати: основні поняття, засоби і методи математичної логіки, їх застосування в інформатиці й програмуванні; знати мови пропозиційної логіки та логіки 1-го порядку, їх можливості для опису предметних областей; основні методи пошуку доведень та засоби логічного виведення.

2. Вміти: описувати на формальних мовах 1-го порядку твердження для різних предметних областей; встановлювати істинність пропозиційних формул, безквантворних формул, формул 1-го порядку; встановлювати наявність логічного наслідку; встановлювати виразність та невиразність предикатів у моделях мови.

3. Володіти елементарними навичками: програмування в сучасних мовах, перевірки виконуваності формул.

3. Анотація навчальної дисципліни:

Навчальна дисципліна «Формальні методи розробки програмних систем» є складовою освітньо-наукової програми підготовки спеціалістів за освітньо-кваліфікаційним рівнем «магістр» галузі 12 «Інформаційні технології» зі *спеціальності 122* «Комп'ютерні науки», *освітньо-наукової програми* **–** «Інформатика».

Дана дисципліна є обов'язковою навчальною дисципліною за ***програмою «Інформатика»***.

Викладається в 2 семестрі 1 курсу магістратури в обсязі **120** годин **(*4 кредити ECTS*), *)** зокрема: *лекції – 38 год., консультації – 2 год., самостійна робота – 80 год.* У курсі передбачено **2** частини та 2 контрольні роботи.

Завершується дисципліна **іспитом у 2-му семестрі.**

4. Завдання (навчальні цілі):

Опанування курсу покликане забезпечити формування компетентностей:

– ЗК1. Здатність до абстрактного мислення, аналізу та синтезу.

– ЗК5. Здатність спілкуватися іноземною мовою.

– СК16. Здатність і готовність до проектування інформаційної системи визначеного прикладного застосування шляхом аналізу та синтезу складу та структури системи або окремих їх складових, розробка функціональних і нефункціональних вимог до системи, що проектується.

5. Результати навчання за дисципліною

Результат навчання (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		Форми (та/або методи і технології) викладання і навчання	Методи оцінювання та пороговий критерій оцінювання (за необхідності)	Відсоток у підсумковій оцінці з дисципліни
Код	**Результат навчання**			
РН1.1	*Знати основні поняття програмування та логічні числення.*	*Лекція*	*Контрольна робота 60% правильних відповідей, іспит*	20%
РН1.2	*Знати методи формалізації мов програмування та мов специфікацій.*	*Лекція*	*Контрольна робота 60% правильних відповідей, іспит*	20%
РН1.3	*Знати методи моделювання предметних областей.*	*Лекція*	*Контрольна робота 60% правильних відповідей, іспит*	20%
РН2.1	*Вміти формалізувати мови специфікацій та програм, моделювати предметні області за допомогою відповідних мов, застосувати програмні засоби аналізу специфікацій.*	*Лекція, самостійна робота*	*Захист проекту, іспит*	20%
РН3.1	*Обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань проектування та розробки специфікацій програм.*	*Лекція, самостійна робота*	*Захист проекту, іспит*	10%
РН4.1	*Організовувати свою самостійну роботу для досягнення результату.*	*Іспит, самостійна робота*	*Захист проекту, іспит*	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни **Програмні результати навчання**	**РН 1.1**	**РН 1.2**	**РН 1.3**	**РН 2.1**	**РН 3.1**	**РН 4.1**
(з опису освітньої програми)						
ПРН 13. Використовувати знання з комп'ютерних наук та інформаційних технологій й уміння критичного мислення, аналізу та синтезу в професійних цілях.	+	+	+		+	+
ПРН 15. Володіти методами розробки та впровадження заходів, спрямованих на підвищення ефективності інформаційних систем.				+	+	+

7. Схема формування оцінки

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

1. Контрольна робота 1 (до 8-го тижня): РН 1.1., РН 1.2 — 25 балів/15 балів.

2. Контрольна робота 2 (до 16-го тижня): РН1.3 – 25 балів/15 балів.

3. Захист проекту: РН2.1, РН3.1, РН4.1 – 10 балів/6 балів

Прострочення виконання роботи тягне за собою зниження балів.

- підсумкове оцінювання (у формі іспиту)*:*

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;

- результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН1.3, РН2.1, РН3.1, РН4.1;

- форма проведення і види завдань: письмова.

Види завдань: 4 письмових питання.

- 1 питання: *РН1.1, РН3.1, РН4.1;*
- 2 питання: *РН1.2, РН3.1, РН4.1;*
- 3 питання: *РН1.3, РН3.1, РН4.1;*
- 4 питання: *РН2.1, РН3.1, РН4.1;*

За розгорнуту відповідь на кожне завдання студент може отримати від 1 до 10 балів.

Критерії оцінювання відповіді студента на питання:

- повнота розкриття питання 1-4 бали;
- логіка викладення 2 бал;
- аналітичні міркування 1-4 бали.

Типове завдання контрольної роботи 1:

Побудувати моделі та специфікації програмних систем відповідно до варіанту.

Теоретичні питання до контрольної роботи 1:

1. Властивості основної пентади програмування.
2. Властивості програмної пентади.
3. Що таке часткова коректність програм? Яким чином доводиться часткова коректність програм?
4. Що таке повна коректність програм? Яким чином доводиться повна коректність програм?
5. Розкрийте зміст формалізації поняття програми.
6. Визначте різні класи функцій.
7. Визначте програмні системи різного рівня абстракції.
8. Дайте визначення класу номінативних даних.
9. Повний клас обчислюваних функцій над номінативними даними.

Контрольні запитання до частини 1:

1. Які принципи є основними методологічними принципами теорії програмування?
2. Як формулюються принципи розвитку та гносеологічності?
3. Як визначається пентода основних понять програмування?
4. Як визначаються поняття користувача, проблеми, програми, обчислюваності, програмування?

5. Які властивості основних понять програмування?
6. Як аспекти програм є головними?
7. На підставі яких принципів відбувається формалізація програмних понять?
8. Як визначаються інтенсіональні та екстенсіональні аспекти програмних понять?
9. Як визначаються системи різних рівнів абстракції?
10. Як визначаються мови специфікацій та програмування?
11. Які є формальні моделі обчислюваних функцій?
12. Як визначається обчислюваність над складними структурами даних?
13. Як визначається обчислюваність над номінативними даними?
14. Що таке натуральна обчислюваність?
15. Як визначається обчислюваність композицій програм?
16. Повні класи обчислюваних функцій.
17. Якими методами описують предметні області?
18. Як методи використовують для специфікації вимог до програмних систем?
19. Які засади RAISE-методу розробки програм?
20. Які логічні формалізми використовують для специфікацій програм?
21. Як використовується класична та некласична логіка для специфікацій програм?
22. Як визначаються темпоральні та модальні логіки?
23. Як визначаються аксіоматичні методи специфікації програм?
24. Як визначається логіка Флойда-Хоара та які властивості вона має?
25. Які особливості має семантико-синтаксична технологія розробки програм?
26. Які особливості має метод послідовних уточнень?
27. Які особливості у розробку програм вносять об’єктно-орієнтовані методи?
28. Яка мета стандартів програмування?
29. Як використовують технологічні та інструментальні засоби специфікації та розробки програм?

Рекомендована література: [1–5].

Типове завдання контрольної роботи 2:

Верифікувати з використання програмних засобів моделі програмних систем відповідно до варіанту.

Теоретичні питання до частини 2:

1. Навести приклади станів транзиційних систем.
2. Визначити логіки Флойда-Хоара. Сформулювати їх аксіоматику.
3. Довести коректність числення та відносну повноту логіки Флойда-Хоара.
4. Засоби специфікації та розробки програм за допомогою логік Флойда-Хоара.
5. Основні засади методу TLA та побудувати приклади.
6. Сформулювати властивості досяжності і безпеки.
7. Надати визначення транзиційного переходу.
8. Принципи верифікації в TLA.
9. Принципи перевірки моделей, заданих в TLA, за допомогою SPIN.
10. Принципи перевірки моделей, заданих в Z.
11. Властивості Z-методу.
12. Принципи перевірки моделей, заданих в B.
13. Властивості B-методу.
14. Принципи перевірки моделей, заданих в RAISE.
15. Властивості RAISE-методу.

Рекомендована література: [1–5, 7–13].

7.2 Неформальна освіта.

Студенти мають можливість замінити підготовку та захист теоретичної частини контрольних робіт 1 і 2 сертифікатом успішного проходження навчальних курсів на базі платформ ММОС або на базі тренінгових центрів. Тематика неформального навчання має бути відповідною дисципліні та попередньо погоджена з викладачем. Максимальна кількість отриманих балів за результатами неформального навчання становить 10.

Запитання для підготовки до іспиту

1. Які принципи є основними методологічними принципами теорії програмування?
2. Як формулюються принципи розвитку та гносеологічності?
3. Як визначається пентада основних понять програмування?
4. Як визначаються поняття користувача, проблеми, програми, обчислюваності, програмування?
5. Які властивості основних понять програмування?
6. Властивості основної пентади програмування.
7. Властивості програмної пентади.
8. Що таке часткова коректність програм? Яким чином доводиться часткова коректність програм?
9. Що таке повна коректність програм? Яким чином доводиться повна коректність програм?
10. Розкрийте зміст формалізації поняття програми.
11. Визначте різні класи функцій.
12. Визначте програмні системи різного рівня абстракції.
13. Дайте визначення класу номінативних даних.
14. Повний клас обчислюваних функцій над номінативними даними.
15. Які аспекти програм є головними?
16. На підставі яких принципів відбувається формалізація програмних понять?
17. Класи функцій, що використовуються для формалізації програм.
18. Як визначаються інтенсіональні та екстенсіональні аспекти програмних понять?
19. Поняття композиційно-номінативної системи.
20. Як визначаються системи різних рівнів абстракції?
21. Як визначаються мови специфікацій та програмування?
22. Якими методами описують предметні області?
23. Які методи використовують для специфікації вимог до програмних систем?
24. Які засади RAISE-методу розробки програм?
25. Які засади B-методу розробки програм?
26. Які засади Z-методу розробки програм?
27. Які засади TLA-методу розробки програм?
28. Які логічні формалізми використовують для специфікацій програм?
29. Як використовується класична та некласична логіка для специфікацій програм?
30. Як визначаються темпоральні та модальні логіки?
31. Як визначаються аксіоматичні методи специфікації програм?
32. Як визначається логіка Флойда-Хоара та які властивості вона має?
33. Повнота логіки Флойда-Хоара.
34. Які особливості має семантико-синтаксична технологія розробки програм?
35. Які особливості має метод послідовних уточнень?
36. Які особливості у розробку програм вносять об’єктно-орієнтовані методи?
37. Яка мета стандартів програмування?
38. Як використовують технологічні та інструментальні засоби специфікації та розробки програм?

Для отримання загальної позитивної оцінки з дисципліни оцінка за іспит не може бути меншою 24 балів.

Студент не допускається до іспиту, якщо під час семестру набрав менше ніж 36 балів.

7.3. Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота 1: до 10 тижня семестру.

2. Контрольна робота 2: до 18 тижня семестру.

Студент має право на одне перескладання кожної контрольної роботи із можливістю отримання максимально 80% початково визначених за цю контрольну роботу балів. Термін перескладання визначається викладачем.

7.3 Шкала відповідності оцінок

Відмінно / Excellent	90-100
Добре / Good	75-89
Задовільно / Satisfactory	60-74
Незадовільно / Fail	0-59

8. Структура навчальної дисципліни. Тематичний план лекцій

№ лекції	Назва лекції	Кількість годин		
		Лекції	Конс.	Самост. роб.
	Частина 1. Формальні моделі програм та методи їх специфікації			
1.	**Тема 1.** Вступ до предмету. Основні методологічні принципи побудови формальних моделей програм. *Самостійна робота: Основні поняття програмування. Формальні моделі програм. Формалізація основних понять. Методи формалізації мов програмування та специфікацій. Приклади специфікацій простих систем*	2		4
2.	**Тема 2**. Принципи формалізації програмних понять. Інтенсіонал та екстенсіонал понять множини, функції та програми. *Самостійна робота: Теоретико-функціональна та теоретико-номінатна платформи формалізації програмних понять. Інтенсіонал поняття, екстенсіонал поняття. Класи недетермінованих функцій. Формалізація композицій. Рівні абстракції в розгляді основних понять програмування. Властивості основних понять програмування. Головні аспекти програм. Формалізми подання синтаксис та семантика програм*	2		4
3.	**Тема 3.** Мови специфікації та програмування *Самостійна робота: Особливості мов специфікацій у порівнянні з мовами програмування. Визначення денотативних композицій у мовах специфікацій. Логічні композиції, які використовуються у мовах специфікацій*	2		4
4.	**Тема 4.** Формальні моделі обчислюваних функцій *Самостійна робота: Традиційні формальні моделі обчислюваних функцій та їх обмеженість. Моделі абстрактної обчислюваності. Аксіоматичні визначення обчислюваних функцій.* *Традиційні формальні моделі обчислюваних функцій, випадки їх обмеженості. Основні моделі абстрактної обчислюваності. Аксіоматичні визначення обчислюваних функцій*	2		4
5.	**Тема 5.** Предметні області та методи їх опису. Методи розробки програм. *Самостійна робота: Мови опису предметних областей. Специфікація трансляторів і мов програмування. DSL*	2		4
6.	**Тема 6.** Структури даних та класи функцій у мовах специфікацій *Самостійна робота: Номінативні дані та їх класифікація. Базові функції. Моделювання натуральних чисел номінативними даними.*	2		4
7.	**Тема 7.** Класи композицій у мовах специфікацій *Самостійна робота: Композиції номінативних функцій*	2		4

8.	**Тема 8.** Методи уточнення даних, функцій та композицій. *Самостійна робота: Дані скінченної структури. Функції натуралізації та денатуралізації. Схема натуральної обчислюваності. Доведення теореми про повні класи обчислюваних функцій*	2		4
9.	**Тема 9.** Приклади специфікацій програмних систем *Самостійна робота: Специфікація фрагменту програмної системи згідно варіанту.*	1		6
Контрольна робота 1		1		
Всього по частині 1		18		38
	Частина 2. Методи верифікації програм			
10.	**Тема 10.** Верифікація в логіці Флойда-Гоара. Коректність та повнота логіки. *Самостійна робота: Уточнення даних на підставі схеми Гоара. Приклади. Доведення властивості уточнених даних*	2		4
11.	**Тема 11.** Технологічні та інструментальні засоби специфікації та розробки програм за допомогою логік Флойда-Гоара. *Самостійна робота: Класифікація класів функцій у мовах специфікацій. Приклади використання однозначних та багатозначних функцій. Способи подання та перетворення функцій*	2		4
12.	**Тема 12.** Верифікація систем за допомогою Code Contracts та Spec# *Самостійна робота: Специфікаціяі верифікація фрагменту програмної системи згідно варіанту з використанням Code Contracts*	2		4
13.	**Тема 13.** Верифікація систем за допомогою VDM *Самостійна робота: Специфікація і верифікація фрагменту програмної системи згідно варіанту з використанням VDM*	2		4
14.	**Тема 14.** Верифікація систем за допомогою Dafny *Самостійна робота: Специфікація і верифікація фрагменту програмної системи згідно варіанту з використанням Dafny*	2		4
15.	**Тема 15.** Верифікація систем в TLA. *Самостійна робота: Специфікація і верифікація фрагменту програмної системи згідно варіанту з використанням TLA*	2		4
16.	**Тема 16.** Перевірка моделей за допомогою SPIN. *Самостійна робота: Робота з інструментом SPIN для перевірки коректності розподіленої програми.*	2		4
17–18.	**Тема 17.** Верифікація систем в Z та B. Верифікація систем в RAISE. *Самостійна робота: Специфікація і верифікація фрагменту програмної системи згідно варіанту з використанням Z та/або B. Специфікація і верифікація фрагменту програмної системи згідно варіанту з використанням RAISE*	3		8

19.	**Тема 18.** Програмні композиційно-номінативні логіки та їх застосування. *Самостійна робота: Класифікація класів композицій у мовах специфікацій. Визначення конотативних та денотативних композицій, їх властивості. Визначення основних методів подання та перетворення композицій*	2		6
Контрольна робота 2		1		
Всього по частині 2		20		42
Консультація			2	
ВСЬОГО		**38**	**2**	**80**

Загальний обсяг *120 год.,* в тому числі:
Лекцій **– 38** *год.*
Консультації – ***2** год.*
Самостійна робота **- *80** год.*

Теми, винесені на самостійне вивчення:

Основні поняття програмування. Формальні моделі програм. Формалізація основних понять. Методи формалізації мов програмування та специфікацій. Приклади специфікацій простих систем [1,4,5].

Теоретико-функціональна та теоретико-номінатна платформи формалізації програмних понять. Інтенсіонал поняття, екстенсіонал поняття. Класи недетермінованих функцій. Формалізація композицій [1,2,4,5].

Рівні абстракції в розгляді основних понять програмування. Властивості основних понять програмування. Головні аспекти програм. Формалізми подання синтаксис та семантика програм [1,2,4,5].

Відмінності між теоретико-функціональною та теоретико-номінативною платформою формалізації програмних понять. Приклади. Визначення інтенсіоналу поняття та екстенсіоналу. Приклади. Визначення класів недетермінованих функцій. Методи формалізації композицій [1, 2, 4, 5].

Особливості мов специфікацій у порівнянні з мовами програмування. Визначення денотативних композицій у мовах специфікацій. Логічні композиції, які використовуються у мовах специфікацій [1–5].

Традиційні формальні моделі обчислюваних функцій та їх обмеженість. Моделі абстрактної обчислюваності. Аксіоматичні визначення обчислюваних функцій [1,4,5].

Традиційні формальні моделі обчислюваних функцій, випадки їх обмеженості. Основні моделі абстрактної обчислюваності. Аксіоматичні визначення обчислюваних функцій. [1, 4, 5].

Номінативні дані та їх класифікація. Базові функції. Моделювання натуральних чисел номінативними даними. Композиції номінативних функцій [1, 4, 5].

Дані скінченої структури. Функції натуралізації та денатуралізації. Схема натуральної обчислюваності. Доведення теореми про повні класи обчислюваних функцій [1,4,5].

Уточнення даних на підставі схеми Хоара. Приклади. Доведення властивості уточнених даних [1, 6].

Класифікація класів функцій у мовах специфікацій. Приклади використання однозначних та багатозначних функцій. Способи подання та перетворення функцій [1–3].

Класифікація класів композицій у мовах специфікацій. Визначення коннотативних та денотативних композицій, їх властивості. Визначення основних методів подання та перетворення композицій [1–3].

Побудова моделей та специфікацій обраних програмних систем.

9. Рекомендовані джерела

Основні

1. М.С. Нікітченко, Теорія програмування: Частина 1.– Ніжин: Видавництво НДУ імені Миколи Гоголя, 2010.– 119 с.
2. М.С. Нікітченко, С.С. Шкільняк. Математична логіка та теорія алгоритмів. – К., 2008.
3. И.А. Басараб, Н.С.Никитченко, В.Н. Редько. Композиционные базы данных. - К., Либідь, 1992.– 182 с.
4. The RAISE specification language. Prentice Hall Int.– 1992.– 397 p.
5. Schneider K.: Verification of Reactive Systems. Formal Methods and Algorithms. Springer-Verlag Berlin Heidelberg (2004).
6. С. Лавров. Программирование. Математические основы, средства, теория.– СПб.: БХВ-Петербург, 2001.– 320 с.
7. Бабенко Л.П., Лавріщева К.М. Основи програмної інженерії: Навч. посіб.–К.: Т-во "Знання", 2001.– 269 с.

Додаткові

8. Hoare C.A.R., Jifeng He. Unifying Theories of Programming.– London: Prentice Hall Europe, 1998.– 298 p.
9. Clarke E.M., Grumberg O., Peled D.: Model Checking. MIT Press (1999)
10. Mike Spivey. The Z Notation: A Reference Manual, 2nd edition. Prentice Hall International Series in Computer Science, 1992.
11. Jim Davies and Jim Woodcock. Using Z: Specification, Refinement and Proof. Prentice Hall International Series in Computer Science, 1996.
12. Jean-Raymond Abrial. Assigning Programs to Meanings, Cambridge University Press, 1996. ISBN 0-521-49619-5.
13. Steve Schneider. The B-Method: An Introduction, Cornerstones of Computing series, 2001. ISBN 0-333-79284-X.
14. Leslie Lamport. Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers, 2002 Pearson Education Publ.