КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Кафедра теорії та технології програмування

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи
_____ Кашпур О.Ф.
«_28_» ____08_____2020 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ФОРМАЛЬНІ МЕТОДИ РОЗРОБКИ ПРОГРАМНИХ СИСТЕМ/ FORMAL METHODS IN SOFTWARE DEVELOPMENT
для студентів / for students

галузь знань	**12 – Інформаційні технології / Information Technologies**
спеціальність	**122 – Комп'ютерні науки / Computer Science**
освітній рівень	**магістр / Master**
освітня програма	**Штучний інтелект / Artificial Intelligence**
вид дисципліни	**обов'язкова / mandatory**

Форма навчання	**денна**
Навчальний рік	**2020/2021**
Семестр	**2**
Кількість кредитів ECTS	**5**
Мова викладання, навчання та оцінювання	**англійська, українська/ Ukrainian, English**
Форма заключного контролю	**іспит**

Викладачі: **д.ф.-м.н., проф. Нікітченко М.С.** (лекції)

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

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

КИЇВ – 2020

Розробник: **Нікітченко Микола Степанович**, д.ф.-м.н., професор, завідувач кафедри «Теорії та технології програмування»

ЗАТВЕРДЖЕНО
Зав. кафедри теорії та технології програмування
_____ (Нікітченко М.С.)
(підпис) (прізвище та ініціали)
Протокол №1 від «28» 08 2020 р.

Схвалено Гарантом освітньо-наукової програми «Штучний інтелект»

_____ (Крак Ю.В.)

«28» 08 2020 р.

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

Протокол від «28» 08 2020 року № 1

Голова науково-методичної комісії _____ (Омельчук Л.Л.)
(підпис) (прізвище та ініціали)

«28» вересня 2020 року

1. Мета дисципліни – засвоєння основних концепцій, принципів та понять сучасних методів розробки програмних систем та їх застосування для адекватного моделювання мов специфікацій і програмування та використання побудованих моделей для створення сучасних програмних та інформаційних систем високої якості.

/

Discipline aim. The aim of the discipline is to master the basic concepts, principles and concepts of modern methods of software development and their application for adequate modeling of specification languages and programming and use of built models to create modern software and information systems of high quality.

2. Попередні вимоги до опанування або вибору навчальної дисципліни (*за наявності*):

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 стосовно тих чи інших 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.

/

Preliminary requirements to master or choice of the discipline (if any):

1. To know: basic concepts, tools and methods of mathematical logic, their application in computer science and programming; to know the languages of propositional logic and first-order logic, their possibilities for describing subject areas; basic methods of finding proofs and means of logical inference.
2. To be able to: describe in formal languages first-order statements relevant to certain subject areas; to establish the truth of propositional formulas, quantifier-free formulas, first-order formulas; establish of a logical consequence; establish the definability and indefinability of predicates in language models.
3. To have basic skills: programming in modern languages, checking the satisfiability of formulas.

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* **–** „Штучний інтелект”.

Дана дисципліна є обов’язковою навчальною дисципліною за ***програмою “Штучний інтелект”***.

Викладається в 2 семестрі 1 курсу магістратури в обсязі 150 годин.

(5 кредитів ECTS)) зокрема: *лекції – 36 год., консультації – 2 год., самостійна робота – 112 год.* У курсі передбачено **2** *частини* та 2 *контрольні роботи*. Завершується дисципліна – **екзаменом в 2 семестрі.**

В результаті вивчення навчальної дисципліни студент повинен:

знати: основні поняття програмування, методи формалізації мов програмування та мов специфікацій, методи моделювання предметних областей; логічні числення.

вміти: формалізувати мови програмування та специфікацій, моделювати предметні області за допомогою відповідних мов, застосувати програмні засоби аналізу специфікацій.

/

The discipline "Formal methods in software development" is a part of the educational-scientific program of training specialists at the educational-qualification level "Master" in the field 12 "Information Technologies" in the specialty 122 "Computer Science", educational-scientific program "Artificial Intelligence".

This discipline is mandatory in the specialty 122 "Computer Science", educational-scientific program "Artificial Intelligence".

It is taught in the 2nd semester of the 1st year of master's studies in the amount of 150 hours.

(5 ECTS credits)) in particular: lectures - 36 hours, consultations - 2 hours, independent work - 112 hours. The course includes 2 parts and 2 tests. The discipline ends with an exam in the 2nd semester.

As a result of studying the discipline the student must:

to know: basic concepts of programming, methods of formalization of programming and specification languages, methods of modeling subject domains; logical calculi.

be able to: formalize programming and specifications languages, model subject domains with the help of appropriate languages, apply software tools for specification analysis.

4. Завдання (навчальні цілі) / Objectives of study:

набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень у програмуванні, відповідно освітньої кваліфікації «Магістр з комп'ютерних наук».

Зокрема, розвивати:

- здатність до абстрактного мислення, аналізу та синтезу;
- здатність спілкуватися іноземною мовою;
- 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.

/

Objectives of study: acquiring knowledge, skills and abilities (competencies) at the level of the latest achievements in programming, according to the educational qualification "Master in Computer Science".

In particular, to develop:

- ability to abstract thinking, analysis and synthesis;
- ability to communicate in a foreign language;
- ability and readiness to design of an information system of a specified application by analyzing and synthesizing the composition and structure of the system or their individual components, development of functional and non-functional requirements for the projected system.

5. Результати навчання за дисципліною / Learning outcomes:

Результат навчання (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)/ Learning outcomes to know; 2. to be able; 3. ability to communicate; 4. autonomy and responsibility		Форми (та/або методи і технології) викладання і навчання / Forms (and / or methods and technologies) of teaching and learning	Методи оцінювання та пороговий критерій оцінювання (за необхідності)/ Evaluation methods and evaluation threshold (if applicable)	Відсоток у підсумковій оцінці з дисципліни/ Percentage in the final grade of the discipline
Код / Code	**Результат навчання/ Learning outcomes**			
РН1.1 / LO1.1	*Знати основні поняття програмування та логічні числення.* / *To know the basic concepts of programming and logical calculi*	*Лекції / lectures*	*Контрольна робота 1, 60% правильних відповідей, Іспит / Test 1, 60% of correct answers, exam*	20%
РН1.2 / LO1.2	*Знати методи формалізації мов програмування та мов специфікацій.* / *To know the methods of formalization of programming and specification languages.*	*Лекції / lectures*	*Контрольна робота1, 60% правильних відповідей, Іспит / Test 1, 60% of correct answers, exam*	20%
РН1.3 / LO1.3	*Знати методи моделювання предметних областей.* / *To know methods of subject domain modelling.*	*Лекції / lectures*	*Контрольна робота 2, 60% правильних відповідей, Іспит / Test 2, 60% of correct answers, exam*	20%
РН2.1 / LO2.1	*Вміти формалізувати мови специфікацій та програм, моделювати предметні області за допомогою відповідних мов, застосувати програмні засоби аналізу специфікацій.* / *To be able to formalize specifications and programing languages, model subject domains using appropriate languages, apply software tools for specification analysis.*	*Лекції, самостійна робота / Lectures, independent work*	*Контрольна робота 2, 60% правильних відповідей, поточне оцінювання, Іспит / Test 2, 60% of correct answers, exam*	20%
РН3.1 / LO3.1	*Обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань проектування та розробки специфікацій програм.* / *Justify your own view on the task, communicate with colleagues on the design and development of program specifications.*	*Лекції / lectures*	*Поточне оцінювання, Іспит / current evaluation, exam*	10%

РН4.1 / LO4.1	*Організовувати свою самостійну роботу для досягнення результату.* / *Organize your independent work to achieve results.*	*Самостійна робота / independent work*	*Поточне оцінювання, Іспит / current evaluation, exam*	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання / Correlation of learning outcomes of the discipline with program learning outcomes

Результати навчання дисципліни / Learning outcomes of the discipline **Програмні результати навчання / Program learning outcomes** *(з опису освітньої програми) (from the description of the educational program)*	**РН 1.1 / LO 1.1**	**РН 1.2 / LO 1.2**	**РН 1.3 / LO 1.3**	**РН 2.1 / LO 2.1**	**РН 3.1 / LO 3.1**	**РН 4.1 / LO 4.1**
ПРН4. Аналізувати великі дані та моделювати високорівневі абстракції у великих наборах даних різної природи, проектувати сховища великих даних, для видобутку даних і знань, візуалізувати великі дані, будувати і оцінювати регресивні моделі, що генеруються на основі великих даних. / PLO4 To analyze large data and simulate high-level abstractions in large data sets of different nature, design large data warehouses, extract data and knowledge, visualize large data, build and evaluate regressive models generated on the basis of large data		+	+			
ПРН13. Використовувати знання з комп'ютерних наук та інформаційних технологій й уміння критичного мислення, аналізу та синтезу в професійних цілях. / PLO13 To use computer science and information technology and critical thinking skills, analysis and synthesis for professional purposes.	+					+
ПРН15. Володіти методами розробки та впровадження заходів, спрямованих на підвищення ефективності інформаційних систем. / PLO15. To master the methods of development and implementation of measures aimed at increasing the efficiency of information systems				+	+	

7. Схема формування оцінки / Evaluation scheme.

7.1 Форми оцінювання студентів / Forms of student assessment:

- семестрове оцінювання / semester assessment:

1. Контрольна робота 1: РН 1.1., РН 1.2 — **25 балів/15 балів.**
2. Контрольна робота 2: РН1.3, РН2.1 – **25 балів/15 балів.**
3. Поточне оцінювання: РН2.1, РН3.1, РН4.1 – **10 балів /6 балів.**

Здобувач освіти може бути недопущений до підсумкового оцінювання, якщо під час семестру він: 1) не досяг мінімального порогового рівня (60%) оцінки тих результатів навчання, які не можуть бути оцінені під час підсумкового контролю; 2) набрав кількість балів, що є недостатньою для отримання позитивної оцінки навіть у випадку досягнення ним на підсумковому контролі максимально можливого результату.

Рекомендований мінімум – 36 балів

/

1. Test 1: LO 1.1, LO 1.2 - 25 points / 15 points.
2. Test 2: LO1.3, LO2.1 - 25 points / 15 points.
3. Current evaluation: LO2.1, LO3.1, LO4.1 - 10 points / 6 points.

An applicant may not be admitted to the final assessment if during the semester he: 1) has not reached the minimum threshold level (60%) of the assessment of those learning outcomes that cannot be assessed during the final control; 2) scored the number of points, which is insufficient to obtain a positive assessment, even if he achieves the maximum possible result in the final control.

The recommended minimum is 36 points.

- підсумкове оцінювання: іспит

- 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 робота.

Рекомендований мінімум – 24 балів.

/

- final assessment: exam.

- the maximum number of points that can be obtained: 40 points;
- learning outcomes that will be evaluated: LO1.1, LO1.2, LO1.3, LO2.1, LO3.1, LO4.1;
- form and types of tasks: written work.

The recommended minimum is 24 points.

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 – 1-2 бали;
- аналітичні міркування – 1-4 бали.

/

Types of tasks: 4 written questions.

- 1th question: LO1.1, LO3.1, LO4.1;
- 2nd question: LO1.2, LO3.1, LO4.1;
- 3rd question: LO1.3, LO3.1, LO4.1;
- 4th question: LO2.1, LO3.1, LO4.1.

For a detailed answer to each task, a student can receive from 1 to 10 points.
Criteria for evaluating the student's answer to the question:

- completeness of the question – 1-4 points;
- logic of presentation – 1-2 points;
- analytical considerations – 1-4 points.

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.

Завдання для самостійної роботи. Побудувати моделі та специфікації наступних програмних систем:

1. Банкомат
2. Кавовий автомат
3. Склад
4. Е-магазин
5. Мікрохвильова піч
6. Телефон
7. Електроплита

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. Які 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].

/

Test questions for part 1:

1. What principles are the main methodological principles of programming theory?
2. How are the principles of development and epistemology formulated?
3. How is the pentad of the basic concepts of programming defined?
4. How are the concepts of user, problem, program, computing, and programming defined?
5. What are the properties of the basic concepts of programming?
6. What are the main aspects of programs?
7. On the basis of what principles are program concepts formalized?
8. How are the intentional and extensional aspects of program concepts defined?
9. How are systems of different levels of abstraction defined?
10. How are specification and programming languages defined?
11. What are the formal models of computable functions?
12. How is computability determined over complex data structures?
13. How is computability determined over nominative data?
14. What is natural computing?
15. How is computability of program compositions determined?
16. Complete classes of computable functions.
17. What methods describe subject domains?
18. What methods are used to specify requirements for software systems?
19. What are the principles of RAISE-method of software development?
20. What logical formalisms are used for program specifications?
21. How are classical and non-classical logic used for program specifications?
22. How are temporal and modal logics defined?
23. How are axiomatic methods of program specification determined?
24. How is the Floyd-Hoare logic defined and what properties does it have?
25. What are the features of semantic-syntactic technology of program development?
26. What are the features of the method of sequential refinements?
27. What are the features of object-oriented methods in software development?
28. What is the purpose of programming standards?
29. How to use technologies and tools for specification and program development?

Recommended References: [1–5].

Завдання для самостійної роботи. Побудувати моделі та специфікації наступних програмних систем:

1. Банкомат
2. Кавовий автомат
3. Склад
4. Е-магазин
5. Мікрохвильова піч
6. Телефон
7. Електроплита

/

Tasks for independent work. Build models and specifications of the following systems:

1. ATM
2. Coffee machine
3. Storage
4. E-shop
5. Microwave oven
6. Phone
7. Electric stove

Типове завдання контрольної роботи 2:

Верифікувати з використання програмних засобів моделі наступних програмних систем:

1. Банкомат
2. Кавовий автомат
3. Склад
4. Е-магазин
5. Мікрохвильова піч
6. Телефон
7. Електроплита

/

Typical test task 2:

Verify with software tools models of the following systems:

1. ATM
2. Coffee machine
3. Storage
4. E-shop
5. Microwave oven
6. Phone
7. Electric stove

Контрольні запитання 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].

Test questions for part 2:

1. Give examples of states of transition systems.
2. Define the Floyd-Hoare logic. Formulate its axiomatics.
3. Prove correctness of calculus and the relative completeness of Floyd-Hoare logic.
4. Software tools for specification and program development using Floyd-Hoare logic.
5. Describe basic principles of TLA method and build examples.
6. Formulate the properties of reachability and safety.
7. Provide a definition of the transition.
8. Principles of verification in TLA.
9. Principles of checking TLA models using SPIN.
10. Principles of verification of models specified in Z.
11. Properties of Z-method.
12. Principles of verification of models specified in B.
13. Properties of B-method.
14. Principles of verification of models specified in RAISE.
15. Properties of RAISE method.

Recommended literature: [1–5, 7–13].

Запитання для підготовки до екзамену

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. Як 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?

/

Questions to prepare for the exam

1. What principles are the main methodological principles of programming theory?
2. How are the principles of development and gnoseology formulated?
3. How is the pentad of basic programming concepts defined?
4. How are the concepts of user, problem, program, computing, and programming defined?
5. What are the properties of the basic concepts of programming?
6. Properties of the main programming pentad.
7. Properties of the software pentad.
8. What is the partial correctness of programs? How is partial correctness of programs proved?
9. What is complete correctness of programs? How is complete correctness of programs proved?
10. Explain the meaning of formalizing the concept of program.
11. Identify different classes of functions.
12. Identify software systems at different levels of abstraction.
13. Define the class of nominative data.
14. Complete class of computable functions over nominative data.
15. What aspects of programs are the main ones?
16. On the basis of what principles are program concept formalized?
17. Classes of functions used to formalize programs.
18. How are the intentional and extensional aspects of program concepts defined?
19. The concept of compositional-nominative system.
20. How are systems at different levels of abstraction defined?
21. How are specification and programming languages defined?
22. What methods describe the subject domains?
23. What are methods used to specify software system requirements?
24. What are the principles of RAISE-method of software development?
25. What are the principles of B-method of program development?
26. What are the principles of Z-method of software development?
27. What are the principles of TLA-method of software development?
28. What logical formalisms are used for program specifications?
29. How are classical and non-classical logic used for program specifications?
30. How are temporal and modal logics defined?
31. How are axiomatic methods of program specification defined?
32. How is the Floyd-Hoare logic defined and what properties does it have?
33. Completeness of Floyd-Hoare logic.
34. What are the features of semantic-syntactic technology of program development?

35. What are the features of the method of successive refinement?
36. What are the features of object-oriented methods in software development?
37. What is the purpose of programming standards?
38. How are technologies and tools used to specify and develop programs?

7.2. Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота 1: до 7 тижня семестру.
2. Контрольна робота 2: до 14 тижня семестру.
3. Поточне оцінювання: протягом 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.

У випадку відсутності студента з поважних причин відпрацювання та перездачі контрольних робіт здійснюються у відповідності до „Положення про порядок оцінювання знань студентів при кредитно-модульній системі організації навчального процесу” від 1 жовтня 2010 року.

/

Terms of evaluation forms:

1. Test 1: up to 7 weeks of the semester.
2. Test 2: up to 14 weeks of the semester.
3. Current assessment: during the semester.

The student has the right to one retake of each test with the possibility of obtaining a maximum of 80% of the points initially determined for this test. The term of reassembly is determined by the teacher.

In case of absence of a student for valid reasons working off and transfer of tests are carried out according to "Regulations on the order of an estimation of knowledge of students at the credit-modular system of the organization of educational process" from October 1, 2010.

7.3 Шкала відповідності оцінок / Rating scale:

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

8. Структура навчальної дисципліни. Тематичний план лекцій
/
The structure of the discipline. Thematic plan of lectures and independent work

№ лекції / № of lectrure	Назва лекції / Title of a lecture	Кількість годин / Number of hours		
		Лекції / Lectures	Практ. зан. / practical training	Сам. р-та / Independent work
	Частина 1. Формальні моделі програм та методи їх			

	специфікації / Part 1. Formal models of programs and methods of their specification			
1	**Тема 1.** Вступ до предмету. Основні методологічні принципи побудови формальних моделей програм. **Самостійна робота:** Основні поняття програмування. Формальні моделі програм. [1,4,5]. / **Topic 1.** Introduction to the subject. Basic methodological principles of construction of formal programs models. **Independent work:** Basic concepts of programming. Formal models of programs. [1,4,5].	2		6
2	**Тема 2**. Принципи формалізації програмних понять. Інтенсіонал та екстенсіонал понять множини, функції та програми. **Самостійна робота:** Формалізація основних понять. Методи формалізації мов програмування та специфікацій. Приклади специфікацій простих систем [1, 4, 5]. / **Topic 2.** Formalization principles of program concepts. Concepts of intensions and extensions of sets, functions and programs. **Independent work:** Formalization of basic concepts. Methods of formalizing programming and specifications languages. Examples of specifications of simple systems [1, 4, 5].	2		6
3	**Тема 3.** Мови специфікації та програмування **Самостійна робота:** Рівні абстракції в розгляді основних понять програмування. Властивості основних понять програмування. [1,2,4,5]. / **Topic 3.** Specification and programming languages **Independent work:** Levels of abstraction in the consideration of basic concepts of programming. Properties of basic programming concepts. [1,2,4,5].	2		6
4	**Тема 4.** Формальні моделі обчислюваних функцій **Самостійна робота:** Головні аспекти програм. Формалізми подання синтаксис та семантика програм [1,2,4,5]. / **Topic 4.** Formal models of computable functions **Independent work:** The main aspects of the program. Formal representations of syntax and semantics of programs [1,2,4,5].	2		6
5	**Тема 5.** Предметні області та методи їх опису. **Самостійна робота:** Теоретико-функціональна та теоретико-номінатна платформи формалізації програмних понять. Інтенсіонал поняття, екстенсіонал поняття. Класи не детермінованих функцій. Формалізація композицій [1,2,4,5]. / **Topic 5**. Subject domains and methods of their description.	2		6

	Independent work: Function-theoretic and nominative-theoretic platforms of formalization of program concepts. Intension and extension of notions. Classes of non-deterministic functions. Formalization of compositions [1,2,4,5].			
6	**Тема 6.** Загальні схеми розробки програм. **Самостійна робота:** Сформулювати відмінності між теоретико-функціональною та теоретико-номінативною платформою формалізації програмних понять. Навести приклади. [1, 2, 4, 5]. / **Topic 6**. General schemes of program development. **Independent work:** Formulate the differences between the function-theoretic and nominative-theoretic platforms of formalization of program concepts. Give examples. [1, 2, 4, 5].	2		6
7	**Тема 7.** Структури даних та класи функцій у мовах специфікацій **Самостійна робота:** Надати визначення інтенсіоналу поняття та екстенсіоналу. Навести приклади. Визначити класи не детермінованих функцій. Розглянути методи формалізація композицій [1, 2, 4, 5]. / **Topic 7.** Data structures and function classes in specification languages **Independent work:** Provide a definition of the concepts of intension and extension. Give examples. Identify classes of non-deterministic functions. Consider methods of formalization of compositions [1, 2, 4, 5].	2		6
8	**Тема 8.** Класи композицій у мовах специфікацій **Самостійна робота:** Сформулювати особливості мов специфікацій у порівнянні з мовами програмування. Визначити денотативні композиції в мовах специфікацій. Які логічні композиції використовують у мовах специфікацій [1–5]? / **Topic 8.** Classes of compositions in specification languages. **Independent work:** Formulate features of specification languages in comparison with programming languages. Identify denotative compositions in specification languages. What logical compositions are used in specification languages [1–5]?	2		6
9	**Тема 9.** Методи уточнення даних, функцій та композицій. **Самостійна робота:** Традиційні формальні моделі обчислюваних функцій та їх обмеженість. Моделі абстрактної обчислюваності. Аксіоматичні визначення обчислюваних функцій [1,4,5]. / **Topic 9.** Methods of refining data, functions and compositions. **Independent work:** Traditional formal models of computable functions and their limitations. Models of abstract	2		6

	computability. Axiomatic definitions of computable functions [1,4,5].			
Контрольна робота 1 / Test 1				
Всього по частині 1 / Total for part 1		18		54
	Частина 2. Методи верифікації програм / Part 2. Methods of program verification			
10	**Тема 10.** Верифікація в логіка Флойда-Хоара. **Самостійна робота:** Перерахувати основні моделі абстрактної обчислюваності. Розглянути аксіоматичні визначення обчислюваних функцій. [1, 4, 5]. / **Topic 10.** Verification in Floyd-Hoare logic. **Independent work:** Specify the basic models of abstract computability. Consider axiomatic definitions of computable functions. [1, 4, 5].	2		6
11	**Тема 11.** Приклади застосування. Коректність та повнота логіки. **Самостійна робота:** Номінативні дані та їх класифікація. Базові функції. Моделювання натуральних чисел номінативними даними. Композиції номінативних функцій [1, 4, 5]. / **Topic 11.** Examples of application. Correctness and completeness of logic. **Independent work:** Nominative data and their classification. Basic functions. Modeling of natural numbers by nominative data. Compositions of nominative functions [1, 4, 5].	2		6
12	**Тема 12.** Технологічні та інструментальні засоби специфікації та розробки програм за допомогою логік Флойда-Хоара. **Самостійна робота:** Дати визначення даним скінченої структури. Сформулювати визначення функцій натуралізації та денатуралізації. [1,4,5]. / **Topic 12.** Technological and instrumental tools for specification and program development using Floyd-Hoare logic. **Independent work:** Define the data of a finite structure. Formulate the definition of the functions of naturalization and denaturalization. [1,4,5].	2		6
13	**Тема 13.** Верифікація систем в TLA. Загальні положення. **Самостійна робота:** Розглянути схему натуральної обчислюваності. Довести теореми про повні класи обчислюваних функцій [1,4,5]. / **Topic 13**. System verification in TLA. General statements. **Independent work:** Consider the scheme of natural computability. Prove theorems on complete classes of computable functions [1,4,5].	2		6
14	**Тема 14.** Верифікація систем в TLA. Приклади. **Самостійна робота:** Розглянути уточнення даних на	2		6

	підставі схеми Хоара. Навести приклади. Довести властивості уточнених даних [1, 6]. / **Topic 14.** System verification in TLA. Examples. **Independent work:** Consider data refinement on the basis of the Hoare scheme. Give examples. Prove the properties of refined data [1, 6].			
15	**Тема 15.** Перевірка моделей за допомогою SPIN. **Самостійна робота:** Класифікувати класи функцій у мовах специфікацій. [1–3]. / **Topic 15.** Checking models using SPIN. **Independent work:** Classify function classes in specification languages. [1–3].	2		6
16	**Тема 16.** Верифікація систем в Z. **Самостійна робота:** Навести приклади використання однозначних та багатозначних функцій. Розглянути способи подання та перетворення функцій [1–3]. / **Topic 16.** System verification in Z. **Independent work:** Give examples of the use of unambiguous and multivalued functions. Consider ways to represent and transform functions [1–3].	2		6
17	**Тема 17.** Верифікація систем в B. **Самостійна робота:** Класифікувати класи композицій у мовах специфікацій. [1–3]. / **Topic 17.** System verification in B. **Independent work:** Classify composition classes in specification languages. [1–3].	2		6
18	**Тема 18.** Програмні композиційно-номінативні логіки та їх застосування **Самостійна робота:** Побудувати моделі та специфікації обраних програмних систем. / **Topic 18.** Composition-nominative program logics and their application **Independent work:** Build models and specifications of selected software systems.	2		6
Контрольна робота 2 / Task 2				
Всього по частині 2 / Total for part 2		18		56
	Консультація / Consultations		2	
	Екзамен / Exam			
ВСЬОГО / TOTAL		36	2	112

Загальний обсяг *150 год.,* в тому числі:
Лекцій **– *36 год.***
Консультації – ***2 год.***
Самостійна робота **- *112 год.***
/
The total amount is 150 hours, including:
Lectures - 36 hours.

Consultations - 2 hours.
Independent work - 112 hours.

9. Рекомендовані джерела / Recommended references::

Основні / Basic

1. С.Л. Кривий. Вступ до методів створення програмних систем. Київ, НаУКМА,2018.–449 с.
2. Дорошенко А.Ю., Жереб К.А., Иванов Е.В., Никитченко Н.С., Яценко Е.А. Формальные методы построения параллельных программ, Кропивницький, 2016.– 440 с.
3. М.С. Нікітченко, С.С. Шкільняк. Математична логіка та теорія алгоритмів. – К., 2008.
4. М.С. Нікітченко, С.С. Шкільняк. Прикладна логіка. – К., 2013.– 277 с.

Додаткові / Additional

5. И.А. Басараб, Н.С.Никитченко, В.Н. Редько. Композиционные базы данных. - К., Либідь, 1992.– 182 с.
6. The RAISE specification language. Prentice Hall Int.– 1992.– 397 p.
7. С. Лавров. Программирование. Математические основы, средства, теория.– СПб.: БХВ-Петербург, 2001.– 320 с.
8. Бабенко Л.П., Лавріщева К.М. Основи програмної інженерії: Навч. посіб.–К.: Т-во "Знання", 2001.– 269 с.
9. E.C.R.Hehner: A practical Theory of Programming. Springer, New York, 1993.– 244 pages.
10. Hoare C.A.R., Jifeng He. Unifying Theories of Programming.– London: Prentice Hall Europe, 1998.– 298 p.
11. Schneider K.: Verification of Reactive Systems. Formal Methods and Algorithms. Springer-Verlag Berlin Heidelberg (2004)
12. Clarke E.M., Grumberg O., Peled D.: Model Checking. MIT Press (1999)
13. Mike Spivey. The Z Notation: A Reference Manual, 2nd edition. Prentice Hall International Series in Computer Science, 1992.
14. Jim Davies and Jim Woodcock. Using Z: Specification, Refinement and Proof. Prentice Hall International Series in Computer Science, 1996.
15. Jean-Raymond Abrial. Assigning Programs to Meanings, Cambridge University Press, 1996. ISBN 0-521-49619-5.
16. Steve Schneider. The B-Method: An Introduction, Cornerstones of Computing series, 2001. ISBN 0-333-79284-X.
17. Leslie Lamport. Specifying Systems: The TLA+ Language and Tools for Hardware and Software Engineers, 2002 Pearson Education Publ.