КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Кафедра теоретичної кібернетики

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи
_____ Олена КАШПУР
«_12_» ___жовтня___ 2021 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Розробка програмного забезпечення

для студентів

галузь знань	**12 «Інформаційні технології»** *(шифр і назва)*
спеціальність	**122 «Комп'ютерні науки»** *(шифр і назва спеціальності)*
освітній рівень	**бакалавр** *(молодший бакалавр, бакалавр, магістр)*
освітня програма	**«Інформатика»** *(назва освітньої програми)*
вибірковий блок	**«Інформаційні технології та системи»** *(назва спеціалізації)*
вид дисципліни	вибіркова

Форма навчання	**денна**
Навчальний рік	**2022/2023**
Семестр	**7**
Кількість кредитів ECTS	**4**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**іспит**

Викладачі: **Тетяна КАРНАУХ, к.ф.-м.н., доц.** (лекції)

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

на 20__/20__ н.р. _____(_____) «__»___ 20__р.
(підпис, ПІБ, дата)

КИЇВ – 2021

Розробник: Тетяна КАРНАУХ, к.ф.-м.н., доцент, доцент кафедри теоретичної кібернетики

ЗАТВЕРДЖЕНО
Зав. кафедри теоретичної кібернетики
_____ Юрій КРАК
(підпис)

Протокол № 7 від « 8 » 02 20 21 р.

Схвалено Гарантом освітньо-професійної програми «Інформатика»

_____ Людмила ОМЕЛЬЧУК
(підпис)

« 11 » 02 20 21 р.

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

Протокол від « 11 » 02 20 21 року № 7

Голова науково-методичної комісії _____ Людмила ОМЕЛЬЧУК
(підпис)

« 11 » 02 20 21 року

1. Мета дисципліни — ознайомити студентів з процесом розроблення програмного забезпечення (ПЗ), його основними діяльностями, сучасними методологіями розроблення програмного забезпечення та найбільш уживаними інструментальними засобами підтримки процесу розроблення.

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

1. Знати основні поняття програмування, орієнтуватись у парадигмах програмування.
2. Вміти проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей.
3. Вміти програмувати мовами високого рівня.

3. Анотація навчальної дисципліни:

Розглядаються основні методології, діяльності та робочі процеси розроблення ПЗ; критерії якості ПЗ, їх взаємозв'язок з робочими процесами; інструментальні засоби підтримки процесу розроблення; деякі шаблони, що сприяють побудові "чистого" коду та використовуються під час реалізації робочих процесів. Прикладна складова включає створення застосунків під Django.

Знання та практичний досвід, отримані в процесі вивчення курсу, значно розширять можливості студентів у практичному програмуванні та при написанні курсових та дипломних проектів.

Викладається в **7** семестрі **4** курсу в **обсязі – 120 год.** ***(4 кредити ECTS)***, зокрема: *лекції – 42 год., консультації – 2 год., самостійна робота – 76 год*.

4. Завдання (навчальні цілі):

Розвивати програмні компетентності (подальший перелік наведено згідно освітньої програми):

- СК8. Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.
- СК10. Здатність застосовувати методології, технології та інструментальні засоби для управління процесами життєвого циклу інформаційних і програмних систем, продуктів і сервісів інформаційних технологій відповідно до вимог замовника.
- СК17.3. Здатність реалізовувати фази та ітерації життєвого циклу створення програмних систем на основі моделей та методів розробки програмного забезпечення.
- СК19.3. Здатність застосовувати математичний апарат та принципи програмування в процесі розробки програмних систем.

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 методології розробки ПЗ*	*Лекція, самостійна робота*	*Контрольна робота, практичне завдання, іспит*	30%
РН1.2	*Знати технології та інструментальні засоби які використовуються на етапах розробки ПЗ*	*Лекція, самостійна робота*	*Контрольна робота, практичне завдання, іспит*	20%
РН2.1	*Вміти застосовувати різні методології розробки ПЗ*	*Самостійна робота*	*Контрольна робота, практичне завдання, іспит*	20%

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**			
РН2.2	*Вміти використовувати інструментальні засоби під час розробки ПЗ*	*Самостійна робота*	*Практичне завдання, іспит*	10%
РН3.1	*Здатність обґрунтовувати власний погляд на задачу, спілкуватись з колегами щодо конкретних питань проектування, складати доповіді у письмовій формі та виступати з результатами власної роботи.*	*Самостійна робота*	*Практичне завдання, іспит*	20%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни / Програмні результати навчання	РН 1.1	РН 1.2	РН 2.1	РН 2.2	РН 3.1
(з опису освітньої програми)					
ПРН11. Володіти навичками використання методології управління життєвим циклом програмного забезпечення, продуктів і сервісів інформаційних технологій відповідно до вимог і обмежень замовника, вміти готувати проектну документацію (техніко-економічне обґрунтування, технічне завдання, бізнес-план, креативний бриф, угоду, договір, контракт та ін.).			+		+
ПРН14. Застосовувати знання методології та CASE-засобів проектування складних систем, методів структурного аналізу систем, об'єктно-орієнтованої методології проектування в процесі побудови і практичного застосування функціональних моделей організаційно-економічних і виробничо-технічних систем.			+	+	+
ПРН17.3. Знати математичний апарат та принципи програмування та вміти застосовувати їх у створенні програмних систем.		+	+		
ПРН18.3. Знати фази та ітерації життєвого циклу програмних систем.	+	+	+	+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

1. Контрольна робота: РН1.1, РН1.2, РН2.1 — 25 балів/15 балів.
2. Практичне завдання 1: РН1.1, РН2.1, РН3.1 – 15 балів/9 балів.
3. Практичне завдання 2: РН1.2, РН2.1, РН2.2, РН3.1 – 20 балів/10 балів.

Типова контрольна робота складається з теоретичних та практичних завдань (з відкритими та закритими відповідями) за матеріалом частин 1 та 2.

Матеріал, що виноситься на контрольну роботу:

див. запитання 1-30 для підготовки до оцінювання.

Практичне завдання 1 на тему "Межі застосовності методологій розроблення ПЗ": дати розгорнутий аналіз проблеми та/або розгорнуту відповідь на питання, при цьому висвітлити переваги, недоліки, нюанси різних можливих підходів; по варіантах.

Деталізовані умови завдань на початку семестру розміщуються за посиланням: https://drive.google.com/drive/u/1/folders/0B2BX3uya5IRKflZFZzJ1TGdNS212aGFJWHl1WGRKR3IwREttdVN6WnRhaWs3OEpraXA1Z28

Практичне завдання 1 має бути вчасно представлене повнотекстовим електронним документом.

У разі неякісної підготовки практичного завдання (наприклад, відсутності всебічного аналізу поставленої проблеми) або нездатності студента аргументовано відстоювати наведену в роботі думку викладач має право не зарахувати його або знизити за нього бали відповідно до ступеня розкриття проблеми.

Практичне завдання 2 на тему "Розроблення веб-застосунку з використанням фреймворку Django", по варіантах.

Деталізовані умови завдань, вимоги до них та схема оцінювання визначається окремо з урахуванням специфіки задачі та на початку семестру розміщуються за посиланням: https://drive.google.com/drive/u/1/folders/0B2BX3uya5IRKflZFZzJ1TGdNS212aGFJWHl1WGRKR3IwREttdVN6WnRhaWs3OEpraXA1Z28

У разі неякісного виконання практичного завдання 2, або неуспішного захисту проекту, або нездатності студента презентувати власний проект чи пояснити зміст та призначення використаних у ньому елементів викладач має право не зарахувати завдання або знизити за нього бали.

У разі виявлення критичного обсягу запозичень у практичних завданнях різних студентів такі роботи оцінюються в 0 (нуль) балів. У разі виникнення підозри щодо несамостійного виконання практичного завдання або контрольної роботи викладач має право виставити бали згідно проведеної із студентом співбесіди (за самим завданням та/або за відповідним теоретичним матеріалом), а також має право запропонувати інше завдання для розв'язання під контролем викладача. У разі відмови від спростування підозри в запропонований викладачем спосіб відповідна робота оцінюється в 0 (нуль) 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, РН2.1, РН2.2, РН3.1;
- форма проведення: письмова;
- види завдань: 10 тестових запитань (із закритими та відкритими відповідями) по 4 бали кожне.

Матеріал, що виноситься на іспит:

див. запитання 1-31 для підготовки до оцінювання.

Студент допускається до іспиту, якщо в семестрі набрав не менше 20 балів. Для отримання загальної позитивної оцінки з дисципліни оцінка за іспит має бути не менше 24 балів.

Запитання для підготовки до оцінювання

1. Проблеми розробки сучасних програмних комплексів
2. Трикутники балансування
3. Якість програмного забезпечення (ПЗ)
4. Характеристики, що впливають на якість ПЗ
5. Використання шаблонів у проектуванні
6. Надлишкове та недостатнє проектування
7. Ознаки поганого коду
8. Основні робочі процеси розробки ПЗ
9. Водоспадна модель розробки ПЗ та її модифікації
10. Модель хаосу
11. Ітеративна та інкрементна розробка, властивості, переваги, недоліки
12. Прогнозні та гнучкі методи розробки, їх порівняльний аналіз
13. Спіральна модель розробки
14. RAD
15. Гнучкі методи розробки ПЗ, властивості, переваги, недоліки
16. Методи DSDM, FDD, TDD
17. Екстремальне програмування (XP)
18. Раціональний уніфікований процес розробки ПЗ (RUP)

19. Scrum
20. Scrum для великих проектів
21. Процес аналізу вимог
22. Моделі та метрики якості
23. Верифікація та інспекція коду
24. Верифікація, валідація, тестування
25. Типи тестування
26. Реалізація тестування на основі вибору вхідних даних
27. Артефакти тестування, їх побудова
28. Методи зменшення залежностей
29. Модифікація та підтримка успадкованого коду
30. Шаблони поведінки Strategy, State, Template Method
31. Шаблони MVC, MVVM, MVP, Facade, Observer, Composite.

7.2 Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота: на останній лекції семестру.

2. Практичне завдання 1: має бути здано на перевірку не пізніше як за 4 тижні до кінця теоретичного навчання в семестрі.

3. Практичне завдання 2: має бути здано на перевірку не пізніше передостаннього тижня теоретичного навчання в семестрі.

4. Захист практичного завдання 2: останній тиждень теоретичного навчання в семестрі.

Перескладання контрольної роботи та практичних завдань не передбачаються. Невчасно подані на перевірку практичні завдання не розглядаються.

7.3 Шкала відповідності оцінок

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

8. Структура навчальної дисципліни. Тематичний план лекцій

№ п/п	Номер і назва теми	Кількість годин	
		лекції	самостійна робота
Частина 1 Моделі розроблення програмного забезпечення			
1	**Тема 1**. Проблеми розроблення сучасних програмних комплексів *Самостійна робота*: Різновиди трикутників балансування. Внутрішні характеристики коду. Поняття "чистого" коду. Код, що сам себе документує. Ознаки поганого коду. Функціональне проектування. Зв'язки в програмуванні.	4	10
2	**Тема 2**. Моделі розробки програмного забезпечення, що не відносяться до гнучких *Самостійна робота*: Класифікація методологій розроблення ПЗ. Водоспадна модель (її фази та робочі процеси) та її модифікації (V-модель, sashimi). BDUF. Спіральна модель. Модель хаосу. Загальна характеристика ітеративних та інкрементних моделей. Ризики та керована ризиками розробка. Cleanroom. RUP, його фази та робочі процеси. RAD. Переваги та недоліки, застосовність та порівняльний аналіз методологій.	6	10
3	**Тема 3**. Гнучкі методології. *Самостійна робота*: Agile Manifesto, принципи гнучких методологій. DSDM, FDD. Основні принципи, правила та практики XP. TDD. Використання рефакторінгу в гнучких методологіях. Переваги та недоліки, застосовність та порівняльний аналіз методологій.	4	4
4	**Тема 4**. Методологія Scrum та її масштабування *Самостійна робота*: Основні цінності, принципи, ролі, діяльності та артефакти методології Scrum. Оцінювання розміру задач. Масштабування Scrum на великі проекти: SoS, SAFe, LeSS. Scrum та Lean. Переваги та недоліки, застосовність та порівняльний аналіз методологій.	4	8
5	**Тема 5**. Межі застосовності методологій розроблення ПЗ. *Самостійна робота*: Виконання практичного завдання 1.		8
	Всього по частині 1	18	40
Частина 2 Робочі процеси розроблення програмного забезпечення			
6	**Тема 1**. Вимоги до програмного забезпечення та його якість. *Самостійна робота*: Формування вимог до ПЗ. Модель предметної області. Бізнес-модель. Уточнення та аналіз вимог. Якість програмного забезпечення. Визначення якості через дефекти та через атрибути якості. Внутрішня та зовнішня якість. Якість під час використання. Моделі якості продукту. Моделі якості під час використання. Моделі якості даних. Визначення вимог та побудова показників якості на основі атрибутів.	4	6

№ п/п	Номер і назва теми	Кількість годин	
		лекції	самостійна робота
7	**Тема 2**. Забезпечення якості. *Самостійна робота*: Верифікація та валідація вимог. Верифікація проектної документації, коду, системи, планів. Експертиза, аудит та інспекція технічних артефактів. Оцінка ПЗ за Фаганом. Тестування та його типи. Тестування на основі вибору вхідних даних. Артефакти тестування. Побудова тестових прикладів та тестових переліків.	4	6
8	**Тема 3**. Модифікація коду. *Самостійна робота*: Стратегії модифікації коду. Виконання рефакторінгу. Використання шаблонів поведінки Strategy, State, Template Method під час модифікації коду. Прийоми, що дозволяють розбивати залежності. Покриття коду блочними тестами, зокрема з використанням "підробних" (fake, mock) об'єктів.	4	8
	Контрольна робота	2	
	Всього по частині 2	14	20
Частина 3 Створення застосунків під Django			
9	**Тема 1**. Системи керування версіями. *Самостійна робота*: Використання git та GitHub.	2	
10	**Тема 2**. Фреймворк Django. *Самостійна робота*: Огляд мови Python. Шаблони MVC, MVVM, MVP, Facade, Observer, Composite. Архітектура фреймворку Django. Шаблони (template). Розроблення веб-застосунку з використанням фреймворку Django. Тестування та його автоматизація. Виконання практичного завдання 2.	8	15
	Захист практичного завдання 2.		1
	Всього по частині 3	10	16
	ВСЬОГО	42	76

Загальний обсяг 120 год., у тому числі:
лекції – **42** год.
консультації – **2** год.
самостійна робота - **76** год.

9. Рекомендовані джерела:

Основні:

1. Larman, Basili. Iterative and Incremental Development: A Brief History.
2. www.martinfowler.com.
3. Мартін, Р. С. Чистий код. / Роберт С. Мартін. – Х.:Фабула, 2019.
4. Гамма, Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – СПб: Питер, 2004.
5. Wake W.C. Extreme programming explored. – Addison-Wesley, 2002.
6. Якобсон, А. Унифицированный процесс разработки программного обеспечения./ А. Якобсон, Г. Буч, Дж. Рамбо. – СПб.:Питер, 2002.

7. Schwaber, K. The 2020 Scrum Guide./ K. Schwaber, J. Sutherland. – https://www.scrumguides.org
8. Куликов, С. Тестирование програмного обеспечения. – 2020. – http://svyatoslav.biz/software_testing_book/.
9. Feathers, M. Working Effectively with Legacy Code./ Michael C. Feathers. – Pearson Education, 2009.
10. https://git-scm.com/
11. https://www.djangoproject.com/

Додаткові:

12. Wikipedia — www.wikipedia.org.
13. Фаулер, М. Рефакторинг: улучшение существующего кода./ М. Фаулер. – СПб: Символ-Плюс, 2003.
14. Kerievsky, J. Refactoring ro patterns. – Addison-Wesley, 2005.
15. Веклич, Р.А. Вступ до програмування мовою С++. Структури даних / Р. А. Веклич, Т. О. Карнаух, А. Б. Ставровський. – К.: ВПЦ "Київський університет", 2018.
16. Основы инженерии качества программных систем./ Ф.И. Андон [и др.]. – К. Академпериодика, 2007.
17. Брауде. Э. Технология разработки программного обеспечения. – СПб.: Питер, 2005.
18. Raccoon. The Chaos Model and the Chaos Life Cycle. // ACM Software Engineering Notes. – V. 20. – № 1. – 1995. – P. 55-66.
19. Boehm, B. W. A Spiral Model of Software Development and Enhancement // ACM Software Engineering Notes. – 8, 1986. – P. 14-24.
20.Agile Manifesto. – https://agilemanifesto.org.

10. Додаткові ресурси:

https://classroom.google.com/u/1/c/MjQ3MzMxNjI1MTM3
https://drive.google.com/drive/u/1/folders/0B2BX3uya5IRKflZFZzJ1TGdNS212aGFJWHl1WGRKR3IwREttdVN6WnRhaWs3OEpraXA1Z28