КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ
КАФЕДРА ІНТЕЛЕКТУАЛЬНИХ ПРОГРАМНИХ СИСТЕМ

«ЗАТВЕРДЖУЮ»

Заступник декана
з навчальної роботи

_____Кашпур О.Ф.

«_ ___»_ _____2019 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

для студентів

галузь знань	**12 Інформаційні технології**
спеціальність	**121 Інженерія програмного забезпечення**
освітній рівень	**бакалавр**
освітня програма	**Програмна інженерія**
вид дисципліни	**вибіркова**

Форма навчання	**денна**
Навчальний рік	**2023/2024**
Семестр	**8**
Кількість кредитів ECTS	**5**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**іспит**

Викладач: **к. т. н., доцент Демківський Є.О.** (лекції, лабораторні заняття).

Пролонговано: на 20__/20__н. р._____(_____) «__»_ __20__р.

на 20__/20__н. р._____(_____) «__»____20__р.

КИЇВ – 2023

Розробник: Демківський Євген Олександрович, к.т.н., доцент кафедри інтелектуальних програмних систем.

ЗАТВЕРДЖЕНО

Завідувач кафедри інтелектуальних програмних систем

_____О.І.Провотар

Протокол №___від «__»_____2023 р.

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

Протокол від «____»_____2023 року №____

Голова науково-методичної комісії_____Л.Л. Омельчук

«_____»_____2023 року

Затверджено вченою радою факультету комп'ютерних наук та кібернетики

Протокол від «____»_____2023 року №____

Голова вченої ради факультету_____А.В. Анісімов

1. Мета дисципліни «Тестування програмного забезпечення» – засвоєння студентами базових знань щодо місії й задач тестування програмного забезпечення в процесі його конструювання, рівнів і видів динамічного тестування, відповідних їм ефективних методів статичного й динамічного тестування та підходів до їх автоматизації, а також актуальних технологічних моделей запровадження й сталого вдосконалення ресурсно-ефективного процесу тестування в проектній і лінійно-матричній організації-розробнику – для успішної діяльності в ролях програмістів і тестувальників, а також членів груп гарантування якості й удосконалення процесу розроблення програмних продуктів.

2. Попередні вимоги до опанування або вибору навчальної дисципліни. Для успішного вивчення дисципліни «Тестування програмного забезпечення» студенти повинні:

1. Знати:
    1. Основні моделі життєвого циклу програмних систем;
    2. Основні підходи та базові шаблони проектування програмних систем;
    3. Основні концепції процедурного та об’єктно-орієнтованого програмування.
2. Вміти:
    1. Вилучати, балансувати та документувати вимоги до програмної системи;
    2. Безпечно працювати з програмним кодом у системах контролю версій – автономно та в складі команди.
3. Володіти:
    1. Базовими навичками застосування поширених інтегрованих середовищ розробки програмного забезпечення, зокрема Microsoft Visual Studio, J2EE;
    2. Базовими навичками застосування систем контролю версій, зокрема Git;
    3. Англійською мовою на базовому рівні.
4. Успішно опанувати курси освітньо-професійної програми «Програмна інженерія»:
    1. Об’єктно-орієнтоване програмування;
    2. Організація баз даних та знань;
    3. Розробка WEB-орієнтованих систем:
    4. Програмна інженерія.

3. **Анотація навчальної дисципліни.** Навчальна дисципліна «Тестування програмного забезпечення» є складовою освітньо-наукової програми підготовки фахівців за першим (бакалаврським) рівнем вищої освіти у галузі знань 12 Інформаційні технології за спеціальністю 121 Інженерія програмного забезпечення в рамках освітньо-професійної програми «Програмна інженерія».

Дана дисципліна є вибірковою дисципліною блоку спеціалізації «Програмна інженерія». Викладається у **8** семестрі в **обсязі – 150 год. (5 кредитів ECTS),** зокрема: лекції – 28 год., лабораторні – 14 год., консультації – 8 год., самостійна робота – 100 год. У курсі передбачено 2 змістовні частини, 2 контрольні роботи та 5 лабораторних робіт. Завершується дисципліна – **іспитом.**

В результаті вивчення навчальної дисципліни студенти повинні:

Знати:

1. Техніки автоматизованої формальної інспекції специфікації вимог до програмної системи, її архітектури й тесту коду та інструментальні засоби підтримки цих технік.
2. Методи динамічного тестування основних видів і рівнів з доступом до тестованого коду й без нього та стратегії їх поєднання для локальних і Веб-застосунків у типових ситуаціях в процесі конструювання програмних продуктів.
3. Засади, вбудовані засоби та автономні каркаси автоматизованого тестування коду різних рівнів і видів.

4. Засади розроблення програмного забезпечення, керованого тестами (TDD).
5. Техніки й процедури документування результатів тестування, задачі й методи їх аналізу та засоби автоматизованої підтримки процедур і методів.
6. Моделі ефективного процесу тестування та ступеню його зрілості, а також підходи до розгортання цього процесу згідно з моделями в проектній і лінійно-матричній організації-розробнику.

Вміти:

1. Планувати й виконувати автоматизовану формальну інспекцію вимог, проекту архітектури та тексту коду.
2. Планувати, проектувати, виконувати й документувати тести основних рівнів для локальних і Веб-застосунків з ефективним застосуванням методів чорної, сірої й білої скриньки та засобів автоматизації тестування.
3. Ефективно застосовувати в процесі конструювання програмних продуктів елементи технології розроблення, керованого тестами (TDD).
4. Автоматизовано аналізувати результати тестування й формувати рекомендації щодо заходів з удосконалення процесу тестування.
5. Обґрунтовано вибирати моделі процесу тестування в організації-розробнику та рівня його зрілості, розгортати і вдосконалювати його згідно з вибраними моделями.

4. Завдання (навчальні цілі). Основними завданнями дисципліни «Тестування програмного забезпечення» є набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень в області тестування програмного забезпечення відповідно до кваліфікації фахівець з інформаційних технологій. Зокрема, розвивати:

- Здатність до абстрактного мислення, аналізу та синтезу (ЗК01).
- Здатність застосовувати знання у практичних ситуаціях (ЗК02).
- Здатність спілкуватися державною мовою як усно, так і письмово (ЗК03).
- Здатність вчитися і оволодівати сучасними знаннями (ЗК05).
- Здатність до пошуку, оброблення та аналізу інформації з різних джерел (ЗК06).
- Здатність працювати в команді (ЗК07).
- Здатність дотримуватися специфікацій, стандартів, правил і рекомендацій в професійній галузі при реалізації процесів життєвого циклу (СК05).
- Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя (СК10).
- Здатність обґрунтовано обирати та освоювати інструментарій з розробки та супроводження програмного забезпечення (СК13).
- .Здатність до алгоритмічного та логічного мислення (СК14).
- Здатність застосовувати дискретні структури і сучасні методи дискретної математики під час аналізу, синтезу та проектування інформаційних систем різної природи (СК15.2).

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 об’єкти та техніки формальної інспекції.	Лекції, самостійна робота.	Контрольна робота №1, іспит.	5%
РН1.2	Знати рівні й види тестування коду та відповідні їм методи проектування, виконання й документування результатів тестів для локальних і веб-застосунків.	Лекції, лабораторні заняття, самостійна робота.	Контрольна робота №1, захист лабораторних робіт, іспит.	20%
РН1.3	Знати засоби автоматизованого тестування коду різних рівнів і видів.	Лекції, лабораторні заняття, самостійна робота.	Контрольна робота №1, захист лабораторних робіт, іспит.	5%
РН1.4	Знати основні моделі процесу тестування та його зрілості й умови їх застосування.	Лекції, самостійна робота.	Контрольна робота №1, іспит.	5%
РН1.5	Знати засади й передумови застосування розроблення, керованого тестами.	Лекції, самостійна робота.	Контрольна робота №2, іспит.	15%
РН1.6	Знати задачі й методи аналізу результатів тестування.	Лекції, самостійна робота.	Контрольна робота №2, іспит.	5%
РН2.1	Вміти автоматизовано виконувати формальну інспекцію й документувати її результати	Лекції, самостійна робота.	Контрольна робота №1, іспит.	15%
РН2.2	Вміти проектувати й автоматизовано виконувати набори динамічних тестів чорної, сірої й білої скриньки для локальних і Веб-застосунків у типових ситуаціях розроблення та документувати їх результати.	Лекції, лабораторні заняття, самостійна робота.	Контрольні роботи №1, №2, захист лабораторних робіт, іспит.	5%
РН2.3	Вміти визначати доцільні моделі процесу тестування й рівня його зрілості та заходи з удосконалення процесу.	Лекції, самостійна робота.	Контрольна робота №2, іспит.	5%
РН3.1	Зрозуміло формулювати проблеми вивчення	Лекції, лабораторні заняття, години	Захист лабораторних	5%

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**			
	теоретичного матеріалу й виконання лабораторних робіт у питаннях до викладача й колег та обґрунтовано відповідати на такі питання з їх боку.	консультацій.	робіт.	
РН3.2	Послідовно й зрозуміло обґрунтовувати власні рішення в лабораторних роботах.	Лабораторні заняття.	Захист лабораторних робіт.	5%
РН4.1	Самостійно виконувати тестування коду та документувати прояви дефектів.	Лабораторні заняття.	Захист лабораторних робіт.	5%
РН4.2	Самостійно аналізувати теоретичний матеріал і практичний досвід для обґрунтування лабораторних і контрольних робіт.	Лабораторні заняття, опрацювання рекомендованих інформаційних джерел.	Контрольні роботи №1, №2, захист лабораторних робіт.	5%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання.

Програмні результати навчання \ **Результати навчання дисципліни**	**РН1.1**	**РН1.2**	**РН1.3**	**РН1.4**	**РН1.5**	**РН1.6**	**РН2.1**	**РН2.2**	**РН2.3**	**РН3.1**	**РН3.2**	**РН4.1**	**РН4.2**
ПРН01. Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.			+	+					+	+			
ПРН07. Знати і застосовувати на практиці фундаментальні концепції, парадигми і основні принципи функціонування мовних, інструментальних і обчислювальних засобів інженерії програмного забезпечення.	+	+		+	+	+	+	+					
ПРН14. Застосовувати на практиці			+				+	+				+	

Програмні результати навчання / Результати навчання дисципліни	РН1.1	РН1.2	РН1.3	РН1.4	РН1.5	РН1.6	РН2.1	РН2.2	РН2.3	РН3.1	РН3.2	РН4.1	РН4.2
інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.													
ПРН19. Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.	+						+				+		
ПРН20. Знати підходи щодо оцінки та забезпечення якості програмного забезпечення.	+	+		+	+	+				+		+	+
ПРН25.2. Аналізувати, оцінювати і вибирати інструментальні та обчислювальні засоби, технології, алгоритмічні і програмні рішення для розв'язання завдань інженерії програмного забезпечення.		+	+	+					+		+	+	+
ПРН29.2. Знати та вміти застосовувати методи тестування програмних систем.	+	+			+	+	+	+			+	+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів.

Семестрове оцінювання:

1. Контрольна робота 1: РН1.1, РН1.2, РН1.3, РН1.4, РН2.1, РН2.2, РН4.2 – **20 балів/12 балів**.
2. Контрольна робота 2: РН1.5, РН1.6, РН2.2, РН2.3, РН4.2 – **20 балів/12 балів**.
3. Лабораторна робота 1: РН2.2, РН3.1, РН3.2, РН4.1, РН4.2 – **4 бали/2 бали**.
4. Лабораторна робота 2: РН2.2, РН3.1, РН3.2, РН4.1, РН4.2 – **3 балів/3 бали**.
5. Лабораторна робота 3: РН2.2, РН3.1, РН3.2, РН4.1, РН4.2 – **5 балів/3 бали**.
6. Лабораторна робота 4: РН2.2, РН3.1, РН3.2, РН4.2, РН4.2 – **5 балів/3 бали**.
7. Лабораторна робота 5: РН2.2, РН3.1, РН3.2, РН4.1 – **3 бали/2 бали**.

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, РН1.4, РН1.5, РН1.6, РН2.1, РН2.2, РН2.3.
- Форма проведення і види завдань: письмова робота.
- Види завдань: 4 письмових завдання (тестове завдання, аналітичне питання, теоретичне питання та задача на проектування тестів методами чорної скриньки.
- Для отримання загальної позитивної оцінки з дисципліни оцінка за іспит має бути не меншою, ніж 24 бали.
- Студенти не допускаються до іспиту, якщо протягом семестру вони набрали менше ніж 36 балів.

– Студент не допускається до іспиту, якщо протягом семестру він не виконав і не здав 100 % лабораторних робіт, передбачених планом.

Критерії оцінювання на іспиті.

Завдання	Тема завдання	Максимальний відсоток від 40 балів	Всього відсотків
Завдання 1	Тестове питання за теоретичним матеріалом курсу.	10%	10%
Завдання 2	Аналітичне питання, що потребує зіставлення теоретичного матеріалу курсу, лабораторних робіт, власного досвіду.	25%	25%
Завдання 3	Теоретичне питання.	27%	25%
Завдання 4	Задача з проектування тестів методами чорної скриньки.	38%	40%
			100%

Питання для підготовки до іспиту.

Тестові питання з тестування:

1. Динамічне тестування програми полягає у …?
    a. відлагодженні програми;
    b. перевірці відповідності результатів виконання програми вимогам до неї;
    c. перевірці дотримання стандартів кодування;
    d. виправленні помилок у коді програми.
2. Коли тестувальник може лише запустити тестований код на виконання (текст коду і специфікація недоступні), застосовним є:
    a. дослідницьке тестування;
    b. стохастичне тестування;
    c. метод функціональних діаграм;
    d. статистичне тестування.
3. Метод спрямованого пошуку помилок застосовують на рівні тестування:
    a. модульному;
    b. інтеграційному;
    c. системному;
    d. на всіх рівнях.
4. Нехай функція обчислює площу трикутника на підставі трійки довжин його сторін або видає діагностичне повідомлення про некоректність вхідних даних. Тестові дані (2,4,7) і (5,6,8) належать до:
    a. одного припустимого класу еквівалентності;
    b. одного неприпустимого класу еквівалентності;
    c. різних припустимих класів еквівалентності;
    d. одного припустимого та одного неприпустимого класу еквівалентності.
5. До необхідних для тестування властивостей вимог належить:
    a. незалежність;
    b. взаємозалежність;

c. формулювання в термінах користувача;
d. відстежуваність.

5. Під час тестування інтерфейсу користувача програмної системи слід виконати:
   a. тестування конфігурації;
   b. тестування продуктивності;
   c. димове тестування;
   d. модульне тестування

6. Під час системного тестування програмної системи слід виконати такі види тестів:
   a. тестування циклів;
   b. тестування класів вхідних даних;
   c. стресове тестування;
   d. тестування функцій програмної системи.

7. Безпосереднім результатом динамічного тестування є опис:
   a. помилок;
   b. дефектів;
   c. відмов;
   d. оцінок якості тестованої програми.

8. Аналіз граничних значень застосовний на рівні тестування?
   a. модулів;
   b. інтеграції;
   c. системи;
   d. на всіх рівнях.

9. Уникнути автономного тестування модулів складної програмної системи можна за допомогою:
   a. модифікованого низхідного тестування;
   b. висхідного тестування;
   c. санітарного тестування;
   d. тестування з постійною інтеграцією.

10. Що таке тест-драйвер?
   a. функція, яка викликає тестований модуль (метод);
   b. тестований модуль;
   c. функція, яка викликається з тестованого модуля;
   d. функція, яка протоколює поведінку тестованого модуля(методу) за тестування.

11. Тестувати зручність застосування інтерфейсу користувача можна за допомогою:
   a. аналізу граничних значень;
   b. переходів між станами;
   c. підсівання помилок;
   d. спрямованого пошуку помилок.

12. Для функціонального тестування Веб-застосунку застосовують техніки:
   a. функціональних діаграм;
   b. тестування розгалужень;
   c. аналізу граничних значень;
   d. підсівання помилок.

13. Для тестування зручності використання інтерфейсу користувача можна застосувати техніку:
   a. аналізу граничних значень;
   b. переходів між станами;
   c. підсівання помилок;

d. формальної інспекції

14. Який вид тестування необхідний і для Веб-застосунку, і для Windows-форми?
    a. тестування безпеки;
    b. модульне тестування;
    c. тестування html-посилань;
    d. тестування інтерфейсу користувача.
15. Для функціонального тестування програми з великою кількістю взаємозалежних параметрів найбільше придатна техніка:
    a. покриття гілок та умов;
    b. розбиття вхідного простору на категорії;
    c. попарного тестування;
    d. аналізу граничних значень.

Теоретичні питання:

1. Які властивості вимог до програмної системи необхідні для її тестування? Які документи регламентують ці властивості?
2. Основні структурні критерії тестового покриття та їх співвідношення за силою.
3. Основні рівні тестування відповідно до об'єктів тестування.
4. Основні види динамічного тестування.
5. Сутність, переваги та обмеження стратегій інтеграційного тестування, які не потребують заглушок.
6. Які компоненти складають тестовий набір? Для чого призначений кожний з них?
7. Яке спільне обмеження мають методи аналізу граничних значень та еквівалентного розбиття? Як можна опрацювати це обмеження?
8. Які техніки насамперед застосовні для тестування інтерфейсу користувача?
9. У чому полягає відмінність між функціональним і структурним тестуванням?
10. Види тестування, необхідні для Web-застосунку.
11. На яких рівнях тестування та для чого одночасно застосовні і драйвери, і заглушки?
12. Які підходи застосовують для організації середовища модульного тестування об'єктно-орієнтованих програм ?
13. Які стратегії доцільні для інтеграційного тестування великих ПС складної структури ? Які з них потребують застосування драйверів?
14. Склад, сутність та умови застосування основних стратегій динамічного тестування.
15. Які евристичні критерії застосовують для тестування зручності використання інтерфейсу користувача?
16. Сутність, умови застосування, переваги й обмеження технології розроблення, керованого тестами (TDD).
17. Сутність, умови застосування, переваги й обмеження технології розроблення, керованого приймальними тестами (ATDD).

Задачі з тестування:

Розробити тести для програми, яка реалізує наведені далі алгоритми з обробленням довільних некоректних вхідних даних і наданням користувачу відповідних діагностичних повідомлень з описом некоректності та дій з її усунення.

1. Розширений алгоритм Евкліда.
2. Найменше спільне кратне двох невід'ємних цілих чисел.
3. Нехай прямі задані рівняннями $a_1x + b_1y + c_1 = 0$ і $a_2x + b_2y + c_2 = 0$, a_i, b_i, c_i – цілі числа. Знайти координати точки (x_0, y_0) перетину цих прямих або видати повідомлення про їх взаємне розміщення, якщо вони не перетинаються.
4. Визначити кількість точок перетину двох кіл, заданих ціло-чисельними координатами їх центрів і радіусами: (x_1, y_1, r_1); (x_2, y_2, r_2).
5. Визначити координати точок перетину кола, заданого ціло-чисельними координатами центру і радіусом (x_1, y_1, r_1), і прямої, що проходить через точки з ціло-чисельними координатами (x_2, y_2), (x_3, y_3).
6. Для трикутника, заданого невпорядкованою трійкою координат вершин (x_1, y_1), (x_2, y_2), (x_3, y_3), і прямої, що проходить через точки (x_4, y_4), (x_5, y_4), визначити кількість точок їх перетину. Всі координати є цілими числами.
7. Обчислення знижки на квиток для пасажира на підставі відомостей щодо його віку та статі: діти до 7 років включно – 100%; пенсіонери (чол. з 65 р., жін. з 60 р. включно) – 50%, решта – 0 % (вік – ціле число, стать – одно-символьний рядок, в якому символ F позначає жіночу, M – чоловічу стать).
8. Видати повідомлення «Точка належить прямокутнику» або «Точка не належить прямокутнику» для точки з ціло-чисельними координатами (x_0, y_0) і прямокутника, заданого ціло-чисельними координатами своїх вершин (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4).
9. Видати повідомлення «Точка в перетині» або «Точка не в перетині» для точки з ціло-чисельними координатами (x_0, y_0) і двох кіл, заданих ціло-чисельними координатами їх центрів і радіусами: (x_1, y_1, r_1); (x_2, y_2, r_2).
10. Для трикутника, заданого невпорядкованою трійкою координат вершин (x_1, y_1), (x_2, y_2), (x_3, y_3), які є дійсними числами, видати повідомлення «Є тупий кут» або «Немає тупого кута», якщо в трикутнику дійсно є тупий кут або його немає.
11. Нехай вхідні параметри програми – впорядковані пари цілих чисел (x_i, y_i), $i=1,\ldots,4$, де пари (x_1, y_1), (x_2, y_2), задають координати кінців одного відрізку на площині, а пари (x_3, y_3), (x_4, y_4), – відповідно, кінців другого. Програма має обчислювати координати точки перетину відрізків або видавати повідомлення, що вони не перетинаються або один належить іншому. Вважають, що відрізки перетинаються, якщо вони мають одну спільну точку.
12. Розв'язання рівняння $ax + by = 1$ для невід'ємних цілих чисел a і b, $a \geq b$ згідно з наданим описом алгоритму.
13. Нехай r – ціло-чисельний радіус кола, a,b – ціло-чисельні довжини сторін прямокутника, центр якого збігається з центром кола. Видати повідомлення «Прямокутник вписано», якщо він розміщений в колі, і «Прямокутник не вписано» в іншому випадку.
14. Визначити точки(у) перетину кола з центром у точці (x,y) і радіусом r та прямокутника з центром у точці (0;0) і довжинами a,b сторін (a – сторона, паралельна осі абсцис, b – осі ординат) або видати повідомлення «Прямокутник і коло не

перетинаються» в разі відсутності точок перетину. Вхідними даними є: впорядкова пара цілих чисел (a,b) і впорядкована трійка цілих чисел (x, y, r).

15. Знайти координати (x_4, y_4) четвертої вершини прямокутника, заданого цілочисельними координатами своїх вершин (x_1, y_1), (x_2, y_2), (x_3, y_3).

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. Лабораторні роботи 1-3: до 7 тижня семестру.
4. Лабораторні роботи 4, 5: до 14 тижня семестру.

Студенти мають право на одне перескладання кожної контрольної роботи у визначений викладачем термін.

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

Студенти мають право захищати лабораторні роботи протягом усього навчального семестру.

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 і лабораторних занять.

№ лекції	Назва лекції	Кількість годин		
		Лекції	**Лабораторні заняття**	**Самостійна робота**
Частина 1. Цілі та базові методи тестування програмного забезпечення.				
1	**Тема 1.** Місія, задачі та термінологія в галузі тестування програмного забезпечення.	2		4
2	**Тема 2.** Система базових методів динамічного тестування – з доступом до коду та без нього.	4		7
3	**Тема 3.** Середовище та рамкова стратегія модульного тестування. Особливості тестування об’єктно-орієнтованих програм	2	4	4
4	**Тема 4.** Цілі та рамкові стратегії інтеграційного й системного тестування.	2		
5	**Тема 5.** Цілі та рамкова стратегія тестування інтерфейсу користувача. Евристики Я.Нільсена	2		6

№ лекції	Назва лекції	Кількість годин		
		Лекції	Лабораторні заняття	Самостійна робота
6	**Тема 6.** Цілі та рамкова стратегія тестування веб-застосунку. Тестування на проникнення.	2	4	
7	**Тема 7.** Формальна інспекція робочих продуктів програмного проекту та засоби її автоматизації.	2		
Контрольна робота 1				2
Всього по частині 1		16	8	40
Частина 2. Моделі та методи розгортання процесу тестування в організації-розробнику				
8	**Тема 8.** Документування та аналіз результатів тестування. Життєвий цикл і рамкові класифікації дефектів.	2	2	4
9	**Тема 9.** Умови застосування та методи регресійного тестування	2		
10	**Тема 10.** Умови застосування та основні підходи дослідницького тестування	2	4	
11	**Тема 11.** .Засади технологій розроблення програмного забезпечення, керованого модульними та приймальними тестами. Підробні об’єкти (mocks)	2		
12	**Тема 12.** Основні моделі процесу тестування та його зрілості. Вбудовані та автономні засоби автоматизації тестування	4		
Контрольна робота 2				2
Всього по частині 2		12	6	40
Консультація			8	
ВСЬОГО		**28**	**22**	**100**

Загальний обсяг – **150** год, в тому числі:
Лекції **– 28** год.
Лабораторні заняття **– 14** год.
Консультації – **8** год.
Самостійна робота **– 100** год.

Лабораторні роботи:

Лабораторна робота 1: Розроблення, відлагодження та модульне тестування самим студентом-розробником консольної програми, яка реалізує певний алгоритм дискретної

математики з числа алгоритмів, охоплених навчальною. дисципліною ННД10 «Дискретна математика», та збереження результатів обчислень у файлі.

Лабораторна робота 2: Модульне **т**естування заданого класу, що реалізує деякий алгоритм дискретної математики з числа алгоритмів, охоплених навчальною. дисципліною ННД10 «Дискретна математика» і містить відомі викладачу дефекти, у вибраному студентом інтегрованому середовищі розроблення (MS Visual Studio тощо) за допомогою вбудованих засобів тестування.

Лабораторна робота 3: Тестування функцій і системне тестування заданого застосунку-форми.

Лабораторна робота 4: Виконання та документування дослідницького тестування заданих динамічних сайтів на підставі накопичуваного досвіду з використанням on-line ресурсів.

Лабораторна робота 5: Документування дефектів, виявлених у лаб. роботах №1-4, у баг-трекерах Mantis і RedMine.

9. Рекомендовані джерела.

Основні:

1. Erik van Veenendaal, Rex Black. Foundations of Software Testing ISTQB Certification, 4th edition. – Cengage Learning EMEA; 4th edition (August 9, 2019)
2. Kristin Jackvony. The Complete Software Tester: Concepts, Skills, and Strategies for High-Quality Testing. – December 10, 2021.
3. Panagiotis Leloudas. Introduction to Software Testing: A Practical Guide to Testing, Design, Automation, and Execution. – Apress; 1st ed. edition (May 17, 2023).
4. Paul C. Jorgensen, Byron DeVries. Software Testing: A Craftsman's Approach, Fifth Edition. – Auerbach Publications; 5th edition (August 15, 2022).
5. Chhavi Raj Dosaj. The Self-Taught Software Tester A Step By Step Guide to Learn Software Testing Using Real-Life Project. – Independently published (April 21, 2020).
6. Jaime Mantilla. Software Testing Explained: A Comprehensive Guide for IT and Non-IT Professionals to Thrive in a High-Demand Field, Drive Business Value, Boost Efficiency, and Maximize Savings for Big Corporations. – Independently published (August 18, 2023).
7. Simon Amey. Software Test Design: Write comprehensive test plans to uncover critical bugs in web, desktop, and mobile apps. – Packt Publishing (December 2, 2022).

Додаткові:

1. Дідковська М.В. Тестування: Основні визначення, аксіоми та принципи. Текст лекцій. Частина I / М.В. Дідковська, Ю.О.Тимошенко – МОН України. ННК НТУУ «КПІ». Кафедра математичних методів системного аналізу, 2010 – 62 с.
2. Дідковська М.В. Тестування: Критерії та методи. Текст лекцій. Частина II / М.В. Дідковська – МОН України. ННК НТУУ «КПІ». Кафедра математичних методів системного аналізу, 2010 – 90 с.
3. Roman A. A Study Guide to the ISTQB® Foundation Level 2018 Syllabus: Test Techniques and Sample Mock Exams / A.Roman [1st ed.] – Springer International Publishing, 2018. – 251 p.
4. Spillner A. H. Software Testing Foundations: A Study Guide for the Certified Tester Exam /

A. Spillner, T. Linz, H. Schaefer – 4th ed., Rocky Nook, 2014 – 305 p.

5. Copeland L. A practitioner Guide to Software Test Design / L. Copeland – STQE Publishing. 2004 – 238 p.
6. Tarlinder A. Developer Testing. Building Quality into Software / A.Tarlinder – Pearson Education, Inc.,2017. – 279 p.
7. Grindal M. Combination Testing Strategies – A Survey / M. Grindal, J. Offutt, S.F. Andler // Softw. Test. Verif. Reliab. – 2005. – N 15. – P. 167–199.
8. Офіційний сайт проєкту TestDriven.NET. [Електронний ресурс]. – Режим доступу: http://www.testdriven.net/.
9. Офіційний сайт проєкту EasyMock. [Електронний ресурс]. – Режим доступу: http://www.easymock.org/.
10. Офіційний сайт проєкту Rhino.Mocks [Електронний ресурс]. – Режим доступу: http://www.ayende.com/projects/rhino-mocks.aspx.
11. Swinkels R. A comparison of TMM and other Test Process Improvement Models. Draft. / R. Swinkels // Frits Philips Institute, 2000. – 53 p. [Електронний ресурс]. – Режим доступу: http://wwwbruegge.informatik.tu-muenchen.de/static/contribute/Lehrstuhl/documents/12-4-1-FPdef.pdf.
12. Koomen T. TMap® Next for result-driven testing / T. Koomen et al. – Sogeti, 2007 – 106 p.