ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет інформаційних технологій

Кафедри інженерії програмного забезпечення

ЗАТВЕРДЖУЮ
Декан ФІТ _____ Говорущенко Т. О.
1 вересня 2024 р.

СИЛАБУС

Навчальна дисципліна **Конструювання програмного забезпечення**
Освітньо-наукова програма **Інженерія програмного забезпечення**

Рівень вищої освіти **Перший (бакалаврат)**

Позиція	Зміст інформації
Викладач(і)	**Праворська Наталія Іванівна**
Профайл викладача	http://ipz.khnu.km.ua/праворська-н-і/
E-mail викладача	margana2000007@gmail.com
Контактний телефон	Заповнюється за домовленістю
Сторінка дисципліни в ІСУ	https://msn.khnu.km.ua/course/view.php?id=6688#section-0
Навчальний рік	2024-2025
Консультації	Очні: вівторок 6 пара, 1-201 Он-лайн: за необхідністю і попередньою домовленістю

Загальна характеристика дисципліни

Форма здобуття освіти	Курс	Семестр	Загальне навантаження		Кількість годин					Курсовий проект	Курсова робота	Залік	Іспит
					Аудиторні заняття				Самостійна робота				
			Європ. кредит	Години	Всього	Лекції	Лабораторні роботи	Практичні заняття					
Денна /очна	**4**	**7**	**6**	**180**	**68**	**34**	**34**		**112**				**+**
Разом ДФН			**6**	**180**	**68**	**34**	**34**		**112**				**+**

Анотація навчальної дисципліни

Дисципліна «Конструювання програмного забезпечення» є дисципліною з циклу професійної та практичної підготовки в галузі інженерії програмного забезпечення. Для успішного засвоєння даного курсу здобувач першого (бакалаврського) рівня вищої освіти, повинен мати навички програмування на мовах високого рівня, вміти працювати з базами даних та вміти узагальнювати інформацію отриману з різних джерел, вміти представляти результати своїх досліджень.

Дисципліна викладається для здобувачів першого (бакалаврського) рівня вищої освіти денної форми навчання спеціальностей галузі інформаційних технологій. При викладанні дисципліни використовуються активні і творчі форми проведення занять, зокрема, методи проблемного навчання.

Мета і завдання дисципліни

Мета дисципліни. Забезпечення базової профілюючої підготовки за фахом, формування у майбутніх інженерів-програмістів теоретичних знань та практичних навичок у галузі конструювання та розробки програмного забезпечення, формування сучасного рівня інформаційної та програмістської культури, оволодіння основними принципами конструювання програмного забезпечення; набуття практичних навичок самостійного написання якісного коду для розв'язання різноманітних задач у практичній діяльності. Також дисципліна «Конструювання програмного забезпечення» націлена на формування у здобувачів навичок збору, обробки та представлення початкових даних для прийняття проектних рішень; розробка концептуальних, інформаційно-логічних і функціональних моделей програмного забезпечення; об'єктно-орієнтованого аналізу та проектування.

Завдання дисципліни. Ознайомлення здобувачів з сучасними методами та технологіями конструктивного програмного забезпечення; вивчення способів конструювання програмного забезпечення з використанням мови моделювання UML; формування вмінь та навиків виробітки конструкторських рішень; формування навичок роботи у сучасних інструментальних середовищах підтримки процесу конструювання програмних систем

У рамках дисципліни «Конструювання програмного забезпечення» вивчається процес конструювання програмних систем, розповсюдженні методики і практики побудови надійного програмного забезпечення.

Очікувані результати навчання

Відповідно до Стандарту вищої освіти та освітньої програми дисципліна має забезпечити: **компетентності**: Здатність застосовувати знання у практичних ситуаціях. Здатність працювати в команді. Здатність брати участь у проєктуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя. Здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розробки програмного забезпечення. Здатність здійснювати процес інтеграції системи, застосовувати стандарти і процедури управління змінами для підтримки цілісності, загальної функціональності і надійності програмного забезпечення.

програмні результати навчання: Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки. Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення. Вміти розробляти людино-машинний інтерфейс. Проводити передпроєктне обстеження предметної області, системний аналіз об'єкта проектування. Вибирати вихідні дані для проектування, керуючись формальними методами опису вимог та моделювання. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань. Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення. Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації. Вміти застосовувати методи компонентної розробки програмного забезпечення. Знати та вміти застосовувати методи верифікації та валідації програмного забезпечення.

Здобувач, який успішно завершив вивчення дисципліни, повинен: вміло оцінювати і вибирати методи і моделі розроблення, впровадження, супроводу програмного забезпечення та управління відповідними процесами на всіх етапах життєвого циклу; знати і застосовувати сучасні професійні стандарти і інші нормативно-правові документи з інженерії програмного забезпечення; аргументувати вибір методів формування вимог; розробляти, аналізувати та систематизувати вимоги до програмного забезпечення; розробляти і оцінювати стратегії проектування програмних засобів; обґрунтовувати, аналізувати і оцінювати прийняті проектні рішення з точки зору якості кінцевого програмного продукту; застосовувати моделі і методи оцінювання та забезпечення якості на всіх стадіях життєвого циклу програмного забезпечення.

Тематичний і календарний план вивчення дисципліни

№ тижня	**Тема лекції**	**Тема лабораторного заняття**	**Самостійна робота студента**		
			Зміст	Год.	Література
1	**2**	**3**	**4**	**5**	**6**
1	Лекція 1. Проблеми розробки ПЗ та шляхи їх вирішення. Інженерія ПЗ. Розв’язання професійних задач за допомогою сучасних досягнень науки і техніки.	Планування розробки ПЗ. Етапи розробки програмного забезпечення при структурному підході до програмування. Стадія «Технічне завдання». Передпроєктне обстеження предметної області, системний аналіз об'єкта проектування.	Введення в конструювання Програмного забезпечення. Історія розвитку. Базові терміни. Елементи конструювання ПЗ. Ключові принципи конструювання.	8	1-3; 7; 12
2	Лекція2. Якість ПЗ та фактори, які на нього впливають. Методи компонентної розробки програмного забезпечення.		Технологія розробки ПЗ і якість ПЗ. Характеристики якості ПЗ, які є важливими для користувача. Фактори, що впливають на якість ПЗ. Підготовка до захисту лабораторної роботи №1	8	2; 5; 13
3	Лекція 3. Системний підхід до розробки ПЗ. Каскадна модель життєвого циклу ПЗ. Конструювання ПЗ. Критерії якості ПЗ розробника. Стандарти та розробка ПЗ.	Структурний підхід до програмування. Стадія «Ескізний проект». Сітковий графік виконання робіт. Вибір методології розробки ПЗ. Опис вимог до ПЗ. Робота в команді.	Важливість виконання попередніх умов. Визначення типу ПЗ. Попередні умові, пов’язані з визначенням проблеми. Попередні умови, пов’язані з розробкою умов.	8	3-4; 11, 14
1	**2**	**3**	**4**	**5**	**6**

4	Лекція 4. Верифікація ПЗ. V-образна схема життєвого циклу ПЗ. спіральна модель ЖЦ ПЗ. Технології розробки ПЗ: «важкі і легкі», методологія Agile, XP-програмування, DevOps (синхронне виконання етапів ЖЦ).		Попередні умови, пов'язані з розробкою архітектури. Час, який слід присвятити виконанню попередніх умов. Основи моделювання при конструюванні. Підготовка до захисту лабораторної роботи №2	8	4; 7; 8-10, 15
5	Лекція 5. Три види програмних розробок з точки зору конструювання, технології створення і експлуатації.	Стадія розробки програмного забезпечення «Технічний проект». Структурний підхід до програмування. Проектування інтерфейсу користувача. Розробка прототипу ПЗ.	Проблеми, пов'язані з проектуванням ПЗ. Основні концепції проектування.	8	2; 4-6; 16
6	Лекція 6. Проектування архітектури ПЗ. Ітеративний характер проектування системи і ПЗ. Ціна помилок проектування. CASE технології розробки ПЗ.		Принципи класифікації ПЗ. Три види програмних розробок з точки зору конструювання, технології створення і експлуатації. Види документів, що випускаються на ПЗ, по етапах розробки системи. Підготовка до захисту лабораторної роботи №3	8	1-3; 7; 17
7	Лекція 7. Структурна схема ПЗ. СТС, як засіб конструювання ПЗ. Ієрархічна структура ПЗ СТС. Циклічність (періодичність) у часі вирішення завдань ПЗ. Співпраця між виконавцями різних етапів ЖЦ ПЗ.	Етапи розробки програмного забезпечення Стадія «Реалізація». Проектування та реалізація простого проекту ПЗ з інтерфейсом користувача.	Основи класів: абстрактні типи даних. Якісні інтерфейси класів. Питання проектування та реалізації.	8	4-5; 6
1	**2**	**3**	**4**	**5**	**6**

8	Лекція 8. Тимчасова діаграма роботи системи, як засіб конструювання ПЗ, яке забезпечує паралельні фізичні процеси. Багатозадачна робота ПЗ і її причини.		Причини створення класів. Аспекти, специфічні для язиків. Наступний рівень: пакети класів. . Підготовка до захисту лабораторної роботи №4	8	1; 4; 13
9	Лекція 9. Захисне конструювання ПЗ з паралельними процесами. Задачі синхронізації процесів.	Виконання оцінки проекту на основі loc- і fp-метрик.	Характеристики якості ПЗ. Методики підвищення якості ПЗ. Важність вибору підходу до інтеграції.	8	3-4; 6; 14
10	Лекція 10. Конструювання ПЗ з мінімізацією його складності. Основні поняття структурного та об'єктно-орієнтованого підходу до конструювання ПЗ. Проектування «зверху вниз» і «знизу вгору».		Особливості конструювання програм для вбудованих ЦОМ критичних систем. Фіксований розподіл пам'яті. Проектування знизу-вгору і проектування зверху-вниз. Підготовка до захисту лабораторної роботи №5	8	3; 5; 7
11	Лекція 11. Евристичні принципи конструювання програм. Конструювання ПЗ, пристосованого до змін (готового до проведення змін).	Аналіз чутливості програмного проекту на основі моделі COCOMO II	Компоненти проектування: евристичні принципи. Методики проектування. Коментарі з приводу популярних методологій.	8	1-2; 7; 12
12	Лекція 12. Конструювання програм із захистом від помилок (стійких до помилок програм).		Стратегії рефакторингу. Підготовка до захисту лабораторної роботи №6	8	5-6, 12, 16
13	Лекція 13. Методи обробки помилок у вхідних і вихідних даних. Твердження і загальні принципи їх використання.	Метрики об’єктно-орієнтованих програмних систем.	Поняття оптимізації коду, оцінка продуктивності коду.	8	3, 4, 16
1	**2**	**3**	**4**	**5**	**6**

	Конструювання аварійного захисту в ПЗ для мінімізації збитку від проявилися помилок.				
14	Лекція 14. Технологія налагодження ПЗ. Помилки ПЗ. Статична, динамічна, структурна, функціональна відладки.		Логіка, цикли, зміни типів даних, вирази та методи, переробка коду на низькому рівні. Використання відлагоджувальних Засобів. Підготовка до захисту. лабораторної роботи №7	8	4, 7, 12, 18
15	Лекція 15. Структурна динамічна відладка. Автономна відладка (АВ) і комплексна відладка (КВ) ПЗ.	Організація таблиць у трансляторах і робота з ними. Рефакторинг. Оптимізація коду.	Інструменти для проектування, інструменті для роботи з початковим кодом. Драйвери і заглушки при автономній відладці.	8	4, 13, 16
16	Лекція 16. Деякі проектні моделі оцінки числа маршрутів при відладці ПЗ. Контроль відладки ПЗ в процесі налагодження.		Інструменти для роботи с виконавчим кодом, інструменти та середовища. Підготовка до захисту лабораторної роботи №8	8	4, 14, 17
17	Лекція17. Моделювання роботи СТС з метою генерації даних для проведення комплексної відладки ПЗ. Успадковане ПЗ. Мобільність ПЗ і реінжиніринг ПЗ.	Захист лабораторної роботи №8	Основні поняття і принципи тестування ПЗ. Особливості тестування «білого ящика». Спосіб тестування базового шляху. Способи тестування умов. Спосіб тестування потоків даних. Тестування циклів. Методика тестування програмних систем. Тестування елементів. Тестування інтеграції. Тестування правильності. Системне тестування. Мистецтво налагодження.	14	5, 7, 16

Примітка:* лекції, лабораторні роботи проводяться раз в 2 тижні або по дві або чотири години відповідно.

МЕТОДИ НАВЧАННЯ

Процес навчання з дисципліни ґрунтується на використанні традиційних та сучасних методів: методи проблемного викладання, словесні, наочні (лекції); пояснювально-ілюстративні, проблемного викладання, дослідницькі, частково-пошукові (лабораторні заняття), проблемного викладання, дослідницькі, частково-пошукові (самостійна робота: індивідуальні завдання). Всі заняття проводяться з використанням інформаційних технологій і мають за мету – набуття здобувачами першого (бакалаврського) рівня вищої освіти практичних навичок з конструювання програмного забезпечення, тестування та налагодження відладки, використовувати метрики для оцінки розробленого ПЗ.

ФОРМИ І МЕТОДИ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Поточний контроль здійснюється під час лекційних та лабораторних занять. Семестровий контроль проводиться у формі іспиту. При виведенні остаточної оцінки враховуються результати поточного контролю та письмового іспиту.

Процес оцінювання підготовленості здобувача першого (бакалаврського) рівня вищої освіти можна розділити на етапи:

Перший етап оцінювання направлений на визначення знань інформаційного мінімуму. Якщо здобувач твердо засвоїв визначену навчальним планом суму формальних знань, то це означає, що він вміє використати їх при вирішенні різних питань при проектуванні програмних систем, вміє розширити їх.

Перед вивченням дисципліни, як правило, проводиться вхідний контроль знань з дисциплін, що їй передують і забезпечують. При цьому необхідно встановити рівні та критерії сформованості знань щодо змісту навчальних елементів. Такими рівнями є:

Ознайомчо-орієнтовний (ОО) – особа має орієнтовне уявлення щодо понять, які вивчаються, здатна: програмувати основні елементи програмних систем різними мовами програмування, обирати сучасні методології та технології проектування програмного забезпечення, обґрунтовано використовувати сучасні середовища розроблення програмного забезпечення для розроблення програмних систем.

Понятійно-аналітичний (ПА) – особа має чітке уявлення щодо навчального об'єкту, здатна перенести раніше засвоєнні знання на типові ситуації.

Продуктивно-синтетичний (ПС) – особа має глибоке розуміння щодо навчального об'єкту, здатна здійснювати синтез, генерувати нові ідеї та уявлення, переносити раніше засвоєні знання на нетипові, нестандартні ситуації.

Кожний вид роботи з дисципліни оцінюється за *чотирибальною* шкалою. Семестрова підсумкова оцінка визначається як середньозважена з усіх видів навчальної роботи, виконаних і зданих *позитивно* з врахуванням коефіцієнта вагомості. Вагові коефіцієнти змінюються залежно від структури дисципліни і важливості окремих її видів робіт. Здобувач, який набрав позитивний середньозважений бал за поточну роботу і не здав контрольний захід (іспит), вважається невстигаючим.

При оцінюванні знань здобувачів першого (бакалаврського) рівня вищої освіти використовуються різні засоби контролю, зокрема: усне опитування перед допуском до виконання лабораторної роботи – здійснюється на її початку; засвоєння теоретичного матеріалу з тем перевіряється тестовим контролем; якість виконання, набуття теоретичних знань і практичних навичок перевіряється шляхом захисту кожної лабораторної роботи згідно з робочою програмою дисципліни і робочим навчальним планом.

Оцінка, яка виставляється за *лабораторне заняття*, складається з таких елементів: усне опитування здобувачів перед допуском до виконання лабораторної роботи; знання теоретичного матеріалу з теми; якість оформлення протоколу і графічної частини; вміння здобувача обґрунтувати прийняті конструктивні рішення; своєчасний захист лабораторної роботи. Для виконання програми дисципліни здобувач повинен отримати 7 оцінок за лабораторні роботи.

Термін захисту лабораторної роботи вважається своєчасним, якщо здобувач захистив її на наступному після виконання роботи занятті.

Пропущене лабораторне заняття здобувач повинен відпрацювати не пізніше, ніж за два тижні до кінця теоретичних занять у семестрі.

При *оцінюванні знань* здобувачів першого (бакалаврського) рівня вищої освіти викладач керується такими критеріями.

Оцінку «відмінно» отримує здобувач за глибоке і повне опанування змісту навчального матеріалу, в якому він легко орієнтується, понятійного апарату, за уміння зв'язувати теорію з практикою, вирішувати практичні завдання, висловлювати і обґрунтовувати свої судження. Відмінна оцінка передбачає грамотний, логічний виклад відповіді (як в усній, так і в письмовій формі), якісне зовнішнє оформлення. Здобувач повинен набути практичних навичок із проектування та програмної реалізації програмних систем.

Оцінка «відмінно» виставляється здобувачу, який глибоко засвоїв основні принципи проектування програмних систем та вміє їх раціонально застосувати, знає методики та вміє ними користуватися при розробленні програмного забезпечення. Здобувач не повинен вагатися при видозміні запитання, повинен робити детальні та узагальнюючі висновки.

Оцінку «добре» отримує здобувач за повне засвоєння навчального матеріалу, володіння понятійним апаратом, орієнтування у вивченому матеріалі, свідоме використання знань для вирішення практичних завдань, грамотний виклад відповіді, але у змісті і формі відповіді мали місце окремі неточності (похибки), нечіткі формулювання закономірностей тощо. Відповідь здобувача повинна будуватись на основі самостійного мислення.

Оцінку «добре» отримує здобувач за правильну відповідь з однією-двома суттєвими помилками.

Оцінки «задовільно» заслуговує здобувач, який виявив знання основного навчально-програмного матеріалу в обсязі, необхідному для подальшого навчання та практичної діяльності за професією, що справляється з виконанням практичних завдань, передбачених програмою. Як правило, відповідь здобувача будується на рівні репродуктивного мислення, здобувач слабо знає структуру курсу, допускає помилки у відповіді, засвоїв і набув практичних навичок у проектуванні та реалізації програмних систем, але припустився неточностей. Вагається при відповіді на видозмінене запитання, разом з тим здобувач володіє знаннями, що дозволяють йому під керівництвом викладача усунути неточності у відповіді.

Оцінки «задовільно» заслуговує здобувач за неповне опанування програмного матеріалу, але отримані знання і набуті практичні навички із проектування та розроблення програмного забезпечення.

Оцінка «незадовільно» виставляється, коли здобувач має розрізнені, безсистемні знання, не вміє виділяти головне і другорядне, допускається помилок у визначенні понять, перекручує їх зміст, хаотично і невпевнено викладає матеріал, не може використовувати знання при вирішенні практичних завдань. Як правило, оцінка "незадовільно" виставляється здобувачу, який не може продовжити навчання без додаткових знань з курсу.

На основі результатів поточного контролю і іспиту виставляється підсумкова семестрова оцінка.

Залік виставляється при отриманні здобувачем з дисципліни від 3,00 до 5,00 балів. При цьому за вітчизняною шкалою ставиться «зараховано», а за шкалою ECTS – набрана здобувачем кількість балів та відповідна їй оцінка.

При викладанні дисципліни використовуються такі види навчальних занять, як лекції, лабораторні роботи, індивідуальне консультування і керівництво самостійною роботою здобувача, в т.ч. за індивідуальним завданням.

При оцінюванні знань здобувачів використовуються різні засоби контролю, зокрема: допуск до виконання лабораторної роботи здійснюється на її початку усним опитуванням кожного здобувача; засвоєння теоретичного матеріалу змістових модулів перевіряється змістовим контролем; якість виконання, набуття теоретичних знань і практичних навичок перевіряється шляхом захисту кожної лабораторної роботи та індивідуального завдання згідно з робочим планом.

Оцінка, яка виставляється за лабораторне заняття, складається з таких елементів: усне опитування здобувачів перед допуском до виконання лабораторної роботи; знання теоретичного матеріалу з теми; якість оформлення протоколу і графічної частини; вміння здобувача обґрунтувати прийняті конструктивні рішення; своєчасний захист лабораторної роботи.

Термін захисту лабораторної роботи вважається своєчасним, якщо здобувач захистив її на наступному після виконання роботи занятті.

Пропущене з поважної причини лабораторне заняття здобувач повинен відпрацювати в лабораторіях кафедри у встановлений викладачем термін.

Система поточного контролю знань, умінь, навичок

Формою поточного контролю, що проводиться на кожному лабораторному занятті, є коротке усне та письмове опитування викладеного лекційного матеріалу, а також письмове виконання прикладів розв'язування задач. Формою підсумкового контролю є письмова контрольна робота, яка включає одне питання з переліку питань для підсумкового контролю і задачу.

Структурування дисципліни за видами робіт і оцінювання результатів навчання здобувачів у семестрі за ваговими коефіцієнтами

<table>
<tr><td colspan="9">Аудиторна робота</td><td rowspan="2">Семестровий контроль, іспит (І)</td><td rowspan="2">Підсумковий бал</td></tr>
<tr><td colspan="8">Лабораторні роботи (ЛР)</td><td>Тест</td></tr>
<tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>Т</td><td rowspan="2">0,4</td><td rowspan="2">ЛР*0,3+Т*0,3+І*0,4</td></tr>
<tr><td colspan="8">ВК = 0,3</td><td>ВК = 0,3</td></tr>
</table>

Умовні позначення: ВК – ваговий коефіцієнт, ЛР – лабораторна робота, Т – тест, І – іспит.

Оцінювання тестових завдань. Тематичний тест для кожного здобувача складається з двадцяти тестових завдань, кожне з яких оцінюється одним балом. Максимальна сума балів, яку може набрати здобувач, складає 20.

Оцінювання здійснюється за чотирибальною шкалою.

Відповідність набраних балів за тестове завдання оцінці, що виставляється здобувачу, представлена у нижченаведеній таблиці.

Сума балів за тестове завдання	1–11	12–14	15–18	19-20
Оцінка	2	3	4	5

На тестування відводиться 20 хвилин. Тестування проводиться з використанням модульного середовища для навчання MOODLE. Правильні відповіді здобувач реєструє в он-лайн режимі в модульному середовищі MOODLE. Через 30 хвилин здобувачі завершують тестування та надсилають свої відповіді на сервер. Викладач оголошує результати тестування згідно журналу оцінок модульного середовища MOODLE.

Якщо здобувач отримав негативну оцінку, то він має перездати її в установленому порядку, але обов'язково до терміну наступного контролю.

Підсумкова семестрова оцінка за національною шкалою і шкалою ЄКТС встановлюється в автоматизованому режимі після внесення усіх оцінок до електронного журналу. Співвідношення вітчизняної шкали оцінювання і шкали оцінювання ЄКТС наведені у наступній таблиці.

Співвідношення вітчизняної шкали оцінювання і шкали оцінювання ЄКТС

Оцінка ЄКТС	Інституційна інтервальна шкала балів	Вітчизняна оцінка, критерії		
A	4,75–5,00	5	***Відмінно*** – глибоке і повне опанування навчального матеріалу і виявлення відповідних умінь та навиків	Зараховано
B	4,25–4,74	4	***Добре*** – повне знання навчального матеріалу з кількома незначними помилками	
C	3,75–4,24	4	***Добре*** – в загальному правильна відповідь з двома-трьома суттєвими помилками	
D	3,25–3,74	3	***Задовільно*** – неповне опанування програмного матеріалу, але достатнє для практичної діяльності за професією	
E	3,00–3,24	3	***Задовільно*** – неповне опанування програмного матеріалу, що задовольняє мінімальні критерії оцінювання	

FX	2,00–2,99	2	***Незадовільно*** – безсистемність одержаних знань і неможливість продовжити навчання без додаткових знань з дисципліни	Незараховано
F	0,00–1,99	2	***Незадовільно*** – необхідна серйозна подальша робота і повторне вивчення дисципліни	

Залік виставляється, якщо середньозважений бал, який отримав здобувач з дисципліни, знаходиться у межах від 3,00 до 5,00 балів. При цьому за вітчизняною шкалою ставиться оцінка «зараховано», а за шкалою ЄКТС – буквене позначення оцінки, що відповідає набраній здобувачем кількості балів відповідно до таблиці Співвідношення.

8. Питання для самоперевірки та іспиту з дисципліни “Конструювання програмного забезпечення”

1. Каскадні моделі життєвого циклу розробки програмних систем
2. Діаграми пакетів в UML
3. Спіралевидні моделі життєвого циклу розробки програмних систем
4. Способи задання обмежень в UML діаграмах.
5. Спіралевидні моделі життєвого циклу розробки об’єктно-орієнтованих програмних систем
6. Типи відношень, які використовуються в класових діаграмах
7. Роботи, що проводяться на етапі аналізу для життєвого циклу розробки програмних систем.
8. Способи відображення активних об’єктів в UML діаграмах.
9. Роботи, що виконуються при специфікації вимог до програмних систем.
10. Відношення спадкування в діаграмах класів.
11. Роботи, що виконуються на етапі системного аналізу
12. Відношення агрегації в класових діаграмах.
13. Роботи, що виконуються на етапі об’єктного аналізу.
14. Відношення асоціації в діаграмах класів та способи їх реалізації в об’єктно-орієнтованих мовах програмування
15. Роботи, що виконуються на етапі проектування.
16. Способи представлення багатошарових архітектур засобами UML.
17. Роботи, що виконуються на етапі проектування механізмів.
18. Подібності і відмінності автоматних моделей і діаграм Харела, що використовуються в діаграмах станів в UML.
19. Роботи, що виконуються на етапі детального проектування.
20. Параметризація класів в UML.
21. Роботи, що виконуються на етапі проектування архітектури.
22. Типи обмежень у вбудованих системах та системах реального часу.
23. Роботи, що виконуються на етапі кодування.
24. Використання UML-діаграм на різних етапах життєвого циклу розробки програмної системи.
25. Роботи, що виконуються на етапі тестування.
26. Мови специфікацій вимог до програмного забезпечення.
27. Діаграми прецедентів (UseCase діаграми) в UML.
28. Поняття мета класу.
29. Об’єктні діаграми (діаграми взаємодії) в UML.
30. Типи відношень в UseCase діаграмах.
31. Діаграми класів в UML.
32. Інструментальні засоби проектування, які використовують UML в якості вхідної мови.
33. Діаграми станів в UML.
34. Сценарії для UseCase діаграм та способи їх представлення.
35. Діаграми послідовностей в UML.
36. Відношення асоціації та його використання при генерації програм.
37. Діаграми розгортання в UML.
38. Способи задавання обмежень в діаграмах послідовностей в UML.
39. Діаграми компонентів в UML.

40. Класові діаграми та їх роль в діаграмах розгортання.
41. Проведення оцінки проекту на основі loc- і fp-метрик.
42. Аналіз чутливості програмного проекту на основі моделі COCOMO II
43. Принцип дії рефакторингу.
44. Метрики об'єктно-орієнтованих програмних систем.
45. Оптимізація коду, принцип дії.

9. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

ОСНОВНА:

1. Роберт Мартін. Чистий код: Чистий код: створення, аналіз, рефакторинг. - Фабула. 2019 – 416 с.
2. Постіл. С.Д. UML. Уніфікована мова моделювання інформаційних систем: Навч. посіб., 2019. - 321 с.
3. Piotr Sliż. Organizacja procesowo-projektowa. Istota, modelowanie, pomiar dojrzałości. Difin – 2021 – 292 str.
4. Ian Sommerville. Software Engineering, 10th edition. Published by Pearson . July 14th 2021 – 816 p.
5. Philippe Kruchten. The Rational Unified Process: An Introduction (5rd Edition) (Addison-wesley Object Technology Series). Addison-Wesley Professional; 5rd edition. 2018 – 407 p.
6. Pinto J. Project Management: Achieving Competitive Advantage, 5th Edition. – 2019. – 592 p.
7. Martin Fowler. Refactoring: ImprovingtheDesignofExistingCode (WebEdition), 2nd Edition/ WebEdition.- 2018
8. Маєвський Я. І., Праворська Н.І. Підвищення ефективності автоматизації масштабування мікросервісів у системі керування контейнеризованими застосунками KUBERNETES. - Вісник ХНУ, серія Технічні науки, N 5 , 2022
9. Philipp Hohl, Jil Klünder, Arie van Bennekum, Ryan Lockard, James Gifford, Jürgen Münch, Michael Stupperich and Kurt Schneider. Back to the future: origins and directions of the "Agile Manifesto" – views of the originators Citation:Journal of Software Engineering Research and Development, 2018
10. Paulo Sérgio Medeiros dos Santos, Alessandro Caetano Beltrão, Bruno Pedraça de Souza and Guilherme Horta Travassos. On the benefits and challenges of using kanban in software engineering a structured synthesis study – Citation:Journal of Software Engineering Research and Development, 2018
11. Seblewongel Esseynew Biable, Nuno Manuel Garcia, Dida Midekso, Nuno Pombo. Ethical Issues in Software Requirements Engineering. Software 2022, 1(1), 31- 52; https://doi.org/10.3390/software1010003

ДОПОМІЖНА:

12. Shyam R Chidamber,Chris F Kemerer (Author), Sloan School of Management. A Metrics Suite for Object Oriented Design. Franklin Classics. - 2018 – 44 p.
13. D.Spinellis.Programcodeanalysis -AddisonWesley, 2004
14. ErichGamma, RichardHelm, RalphJohnson, andJohnVlissides. DesignPatterns: ElementsofReusable Object-Oriented Software.Addison-Wesley - 2009 - 417p.
15. Beck Kent. Extreme Programming Explained: EmbraceChange (The XP Series). Addison-WesleyProfessional. – 2004 - 224
16. Dave Nicolette. Software Development Metrics 1st Edition - Manning; 1st edition – 2015 – 192 p.
17. Eric J. Braude; Michael E. Bernstein. Software Engineering. Waveland Press, 2016 – 802 p.
18. Dijkstra, E. W. (Aug 1972). "The Humble Programmer". Communications of the ACM 15 (10): 859–866. doi:10.1145/355604.361591.
19. http://www.cs.utexas.edu/~EWD/transcriptions/EWD03xx/EWD340.html. (EWD340) PDF, 1972 ACM TuringAwardlecture
20. RussellGold, ThomasHammell, TomSnyder. TestDrivenDevelopment: A J2EE Example.- Apress, 2005.- 296 pages.

10. ІНФОРМАЦІЙНІ РЕСУРСИ

1. Модульне середовище для навчання. Доступ до ресурсу: https://msn.khnu.km.ua.

2. Електронна бібліотека університету . Доступ до ресурсу: http://lib.khnu.km.ua/asp/php_f/p1age_lib.php.
3. Репозитарій ХНУ. Доступ до ресурсу: http://elar.khnu.km.ua/jspui/?locale=uk.

Розробник: Праворська Н.І.
Погоджено:
Зав. каф. ІПЗ Бедратюк Л.П.
Гарант ОНП Бедратюк Л.П.