ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ЗАТВЕРДЖУЮ
Декан факультету
інформаційних технологій
_____ Олег САВЕНКО
_____ 2023 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Основи інженерії програмного забезпечення

Галузь знань 12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Рівень вищої освіти – Перший бакалаврський
Освітньо-професійна програма – Інженерія програмного забезпечення
Обсяг дисципліни –5 кредитів ЄКТС, ***Шифр дисципліни*** – ОПП.04
Статус дисципліни: обов'язкова, ***Мова навчання*** Англійська, українська
Факультет – Інформаційних технологій
Кафедра – Інженерії програмного забезпечення

Форма навчання	Курс	Семестр	Загальне навантаження		Кількість годин						Курсовий проект	Курсова робота	Форма семестрового контролю	
					Аудиторні заняття				Індивідуальна робота студента	Самостійна робота студента в т.ч. ІРС				
			Кредити ЄКТС	Години	Разом	Лекції	Лабораторні роботи	Практичні заняття					Залік	Іспит
Очна (денна)	1	2	5	150	72	36	36			78			+	
Разом			**5**	**150**	**72**	**36**	**36**			**78**			**1**	

Робоча програма складена на основі Стандарту вищої освіти, освітньої програми підготовки бакалаврів 2023 року та навчального плану.

Програма складена _____ Єлизавета ГНАТЧУК

Схвалено на засіданні кафедри ІПЗ
протокол № 1 від 31 08 2023 р.

Зав. кафедри інженерії програмного забезпечення _____ Леонід БЕДРАТЮК

Робоча програма розглянута та схвалена Вченою радою факультету інформаційних технологій

Голова вченої ради _____ Олег САВЕНКО
Ініціали, прізвище

Хмельницький 2023

ОСНОВИ ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Тип дисципліни	Обов’язкова
Рівень вищої освіти	Перший (бакалаврський)
Мова викладання	Українська
Семестр	Другий
Обсяг кредитів ЄКТС	5
Форма здобуття освіти	Очна (денна)

Результати навчання. Студент, який успішно завершив вивчення дисципліни, має: аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки; знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії ПЗ і дотримуватись їх у професійній діяльності; знати основні процеси, фази та ітерації життєвого циклу ПЗ; знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії ПЗ; проводити передпроєктне обстеження предметної області, системний аналіз об'єкта проєктування; вибирати вихідні дані для проєктування, керуючись формальними методами опису вимог та моделювання; застосовувати на практиці ефективні підходи щодо проєктування ПЗ; застосовувати на практиці інструментальні програмні засоби доменного аналізу, проєктування, тестування, візуалізації, вимірювань та документування ПЗ; мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації; вміти документувати та презентувати результати розроблення ПЗ.

Пререквізити – Вихідна.

Кореквізити – Моделювання та оцінка програмного забезпечення, Аналіз вимог та якість програмного забезпечення.

Зміст навчальної дисципліни. Інженерні основи ПЗ. Основи моделювання. Основи інженерії вимог до ПЗ. Технології розроблення ПЗ. Етичні та культурні аспекти інженерії програмного забезпечення.

Запланована навчальна діяльність: лекції – 36 год., лабораторні заняття – 36 год., самостійна робота – 78 год., разом – 150 год.

Форми (методи) навчання: лекції (з використанням методів проблемного навчання і візуалізації); лабораторні заняття (з використанням методів інформаційних технологій та сучасних Case-засобів моделювання і проектування, майстер-класів, практикумів), самостійна робота.

Форми оцінювання результатів навчання: захист лабораторних робіт; тематичне тестування.

Форма семестрового контролю: залік.

Навчальні ресурси:

1. Бородкіна І., Бородкін Г. Інженерія програмного забезпечення : посіб. для студ. вищих навч. закладів. К : Вид-тво "Центр навчальної літератури", 2018. 204 с.
2. Левус Є., Мельник Н.. Вступ до інженерії програмного забезпечення. Л. : Львівська політехніка, 2018. 248 с.
3. Rod Stephens. Beginning Software Engineering: Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.
4. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 P.
5. Модульне середовище для навчання MOODLE. Доступ до ресурсу: https://msn.khmnu.edu.ua/
6. Електронна бібліотека університету. Доступ до ресурсу: http://lib.khmnu.edu.ua/asp/php_f/p1age_lib.php

Викладачі: кандидат технічних наук, доцент Єлизавета ГНАТЧУК, асистент Анастасія ДЬОМІНА

3 ПОЯСНЮВАЛЬНА ЗАПИСКА

Дисципліна «Основи інженерії програмного забезпечення», як одна з фахових дисциплін, займає провідне місце у підготовці фахівців освітнього рівня «бакалавр» зі спеціальності 121 «Інженерія програмного забезпечення» за освітньо-професійною програмою «Інженерія програмного забезпечення» і є теоретичним та практичним фундаментом для подальшого вивчення студентами циклу професійно-орієнтованих дисциплін.

Пререквізити – Вихідна

Кореквізити – Моделювання та оцінка програмного забезпечення, Аналіз вимог та якість програмного забезпечення

Мета дисципліни. Метою дисципліни є: навчити студентів фундаментальним основам інженерії програмного забезпечення (ПЗ) відповідно до змісту міжнародного професійного стандарту Guide to the Software Engineering Body of Knowledge (SWEBOK); сформувати в них загальне уявлення про задачі, процеси, методології, методи та засоби інформаційних технологій, які використовуються в інженерії ПЗ; ознайомити їх з основними положеннями технологій розроблення ПЗ, включаючи розгляд питань, пов'язаних з аналізом, проєктуванням, реалізацією, тестуванням, документуванням та супроводом програмних продуктів.

Предмет дисципліни. Методи та засоби інженерії ПЗ з акцентом на ключові аспекти теоретичного й практичного навчання процесам організації розробки ПЗ з урахуванням базового ядра знань SWEBOK.

Завдання дисципліни. Формування у здобувачів вищої освіти базових теоретичнх знань та практичних навичок у галузі інженерії ПЗ на всіх етапах життєвого циклу ПЗ, опанування ними необхідного апарату для подальшого вивчення циклу професійно-орієнтованих дисциплін.

Відповідно до ***Стандарту вищої освіти*** із та освітньої програми дисципліна сприяє забезпеченню:

компетентностей: ФК02 – здатність брати участь у проєктуванні ПЗ, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування; ФК04 – здатність формулювати та забезпечувати вимоги щодо якості ПЗ у відповідності з вимогами замовника, технічним завданням та стандартами; ФК05 – здатність дотримуватися специфікацій, стандартів, правил і рекомендацій у професійній галузі при реалізації процесів життєвого циклу; ФК08 – здатність застосовувати фундаментальні і міждисциплінарні знання для успішного розв’язання завдань інженерії програмного забезпечення; ФК10 – здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження ПЗ та визнання важливості навчання протягом всього життя; ФК11 – здатність реалізовувати фази та ітерації життєвого циклу програмних систем та інформаційних технологій на основі відповідних моделей і підходів розроблення ПЗ.

програмних результатів навчання: ПРН01 – аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки; ПРН02 – знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії ПЗ і дотримуватись їх у професійній діяльності; ПРН03 – знати основні процеси, фази та ітерації життєвого циклу ПЗ; ПРН04 – знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії ПЗ; ПРН10 – проводити передпроєктне обстеження предметної області, системний аналіз об'єкта проєктування; ПРН11 – вибирати вихідні дані для проєктування, керуючись формальними методами опису вимог та моделювання; ПРН12 – застосовувати на практиці ефективні підходи щодо проєктування ПЗ; ПРН14 – застосовувати на практиці інструментальні програмні засоби доменного аналізу, проєктування, тестування, візуалізації, вимірювань та документування ПЗ;

ПРН16 – мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації; ПРН23 – вміти документувати та презентувати результати розроблення ПЗ.

Результати навчання. Студент, який успішно завершив вивчення дисципліни, має: аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки; знати кодекс професійної етики, розуміти соціальну значимість та культурні аспекти інженерії ПЗ і дотримуватись їх у професійній діяльності; знати основні процеси, фази та ітерації життєвого циклу ПЗ; знати і застосовувати професійні стандарти і інші нормативно-правові документи в галузі інженерії ПЗ; проводити передпроєктне обстеження предметної області, системний аналіз об'єкта проєктування; вибирати вихідні дані для проєктування, керуючись формальними методами опису вимог та моделювання; застосовувати на практиці ефективні підходи щодо проєктування ПЗ; застосовувати на практиці інструментальні програмні засоби доменного аналізу, проєктування, тестування, візуалізації, вимірювань та документування ПЗ; мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації; вміти документувати та презентувати результати розроблення ПЗ.

Політика дисципліни Організація освітнього процесу з дисципліни відповідає вимогам положень про організаційне і навчально-методичне забезпечення освітнього процесу, освітній програмі та навчальному плану. Студент зобов’язаний відвідувати лекції, практичні заняття, лабораторні роботи, тощо, згідно з розкладом, не запізнюватися на заняття, виконувати усі завдання та контрольні точки відповідно до графіка. Пропущені практичні заняття і лабораторні роботи студент зобов’язаний опрацювати самостійно у повному обсязі і відзвітувати перед викладачем не пізніше, ніж за тиждень до чергової атестації. До практичних занять і лабораторних робіт студент має підготуватися за відповідною темою і проявляти активність. Набутті особою знання з дисципліни або її окремих розділів у неформальній освіті зараховуються відповідно до Положення про порядок перезарахування результатів навчання та визначення академічної різниці у ХНУ.

4 СТРУКТУРА ЗАЛІКОВИХ КРЕДИТІВ ДИСЦИПЛІНИ

Назва розділу (теми)	**Кількість годин, відведених на:**		
	лекції	лабораторні роботи	СРС
Тема 1. Інженерні основи програмного забезпечення	6		13
Тема 2. Основи моделювання	12	16	27
Тема 3. Основи інженерії вимог до програмного забезпечення	6	12	12
Тема 4. Технології розроблення програмного забезпечення	10	8	22
Тема 5. Етичні та культурні аспекти інженерії програмного забезпечення	2		4
Разом:	**36**	**36**	**78**

5 ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

5.1 Зміст лекційного курсу

Номер лекції	Перелік тем лекцій, їх анотації	Кількість годин
1	*2*	*3*
1	**Лекція 1. Вступ до інженерії програмного забезпечення** Програма як формалізований опис процесу обробки даних. Програмний засіб, програмний продукт, програмне забезпечення (ПЗ). Види ПЗ за різними ознаками. Складові вартості ПЗ. Інженерія ПЗ як сукупність інженерних методів і засобів створення ПЗ. Предмет, зміст та практичне застосування предмету інженерії ПЗ. Інженери та інші фахівці в інженерії ПЗ. Стандартизація в інженерії ПЗ. Ядро знань Software Engineering Body of Knowledge (SWEBOK). Загальна характеристика областей знань SWEBOK. Літ.: [1], [2], [6]	2
2	**Лекція 2. Життєвий цикл програмного забезпечення** Поняття життєвого циклу (ЖЦ) ПЗ. Основні, допоміжні та організаційні процеси ЖЦ ПЗ. Зв'язок між процесами. Класичні моделі ЖЦ (каскадна, спіральна, ітераційна, інкрементна), їх переваги та недоліки. Вибір моделі ЖЦ. Стандарти ЖЦ ПЗ. Літ.: [1], [2], [5], [6]	2
3	**Лекція 3. Моделі життєвого циклу в сучасних методологіях розроблення програмного забезпечення** Методології швидкої розробки ПЗ. Методологія RUP (Rational Unified Process). Методології MSF (Microsoft Solution Framework) та MOF (Microsoft Operations Framework). Гнучкі методології. XP, Scrum, сімейство методологій Crystal, DSDM (Dynamic Systems Development Method), FDD (Feature Driven Development), адаптивне розроблення (ASD). Місце гнучких методологій серед сучасних технологій розроблення ПЗ. Літ.: [1], [2], [5]	2
4	**Лекція 4. Структурно-функціональний аналіз і моделювання. Методологія SADT** Поняття системи, моделі, моделювання. Основні підходи до аналізу і моделювання в інженерії ПЗ (структурний, об'єктно-орієнтований). Сутність та основні принципи структурно-функціонального аналізу та моделювання. Основні положення методології Structured Analysis and Design Technique (SADT). Стандарти функціонального моделювання IDEF0 та IDEF3. Загальна методика побудови IDEF0- та IDEF3 діаграм. Case-засоби автоматизації структурно-функціонального моделювання. Літ.: [3]	2
5	**Лекція 5. Аналіз і моделювання потоків даних** Поняття діаграми потоків даних (**D**ata **F**low **D**iagrams, DFD). Синтаксис і семантика DFD-діаграм. Нотації Йордана-Де Марко і Гейна-Сарсона. Декомпозиція і відповідні розширення діаграм потоків даних. Побудова DFD-діаграм. Побудова ієрархії потоків даних. Case-засоби автоматизації моделювання потоків даних. Порівняльний аналіз SADT- та DFD-моделей. Літ.: [3], [5], [6]	2

1	*2*	*3*
6	**Лекція 6. Інфологічне моделювання** Поняття предметної області. Інфологічне моделювання та його мета. Концептуальний, логічний та фізичний рівні моделювання. Модель "сутність-зв'язок" (Entity-Relationship Diagrams, ERD). Конструктивні елементи інфологічних моделей. Фундаментальні типи зв'язків. Нормалізація даних. Огляд ER-нотацій. Загальна методика побудови ER-діаграм. Case-засоби автоматизації інфологічного моделювання.. Літ.: [3], [6]	2
7	**Лекція 7. Уніфікована мова моделювання UML. Діаграми варіантів використання** Загальні поняття та сутність об'єктно-орієнтованого підходу (ООП) до розробки ПЗ. Принципи ООП. Методологія RUP (Rational Unified Process). Характеристика та концептуальні основи UML (Unified Modeling Language). Типи діаграм UML. Case-засоби автоматизації об'єктно-орієнтованого моделювання. Діаграми варіантів використання. Літ.: [4 – 6]	2
8	**Лекція 8. Уніфікована мова моделювання UML. Деталізація варіантів використання** Способи опису варіантів використання. Потоки подій (основний, альтернативні). Текстовий опис потоків подій. Діаграми станів та діаграми діяльності, їх конструктивні елементи. Особливості побудови діаграм станів та діяльності. Літ.: [4 – 6]	2
9	**Лекція 9. Уніфікована мова моделювання UML. Документування сценаріїв використання за допомогою діаграм взаємодії. Діаграми класів** Поняття сценарію використання. Основний та альтернативні сценарії. Форми представлення діаграм взаємодії. Діаграма послідовності та діаграма кооперації, їх елементи. Особливості побудови діаграм взаємодії. Діаграма класів, її елементи. Особливості побудови діаграм класів. Літ.: [4 – 6]	2
10	**Лекція 10. Процес розроблення вимог до програмного забезпечення** Поняття вимоги до ПЗ. Класифікація вимог. Методології і стандарти, що регламентують роботу з вимогами до ПЗ (SWEBOK, ДСТУ ISO/IEC/IEEE 12207:2018, ISO/IEC/IEEE 29148-2018, ДСТУ ISO/IEC TR 24766:2016). Властивості вимог до ПЗ. Учасники розроблення вимог до ПЗ. Загальний процес розроблення вимог до ПЗ. Літ.: [1], [2], [5]	2
11	**Лекція 11. Джерела та стратегії виявлення вимог до програмного забезпечення** Джерела вимог до ПЗ. Основні стратегії виявлення вимог. Особливості виявлення вимог до програмного продукту «під замовлення» і для відкритого ринку. Аналіз здійснимості. Прототипування. Моделювання. Комунікації проєктної команди і замовника в процесі розроблення вимог до ПЗ. Літ.: [1], [2], [5]	2
12	**Лекція 12. Аналіз, систематизація та специфікація вимог до програмного забезпечення** Мета аналізу вимог до ПЗ. Основні процедури верифікації та валідації вимог до ПЗ. Специфікація вимог до ПЗ. Специфікація функціональних вимог до ПЗ за допомогою варіантів використання. Формалізація вимог. Технічне завдання (ТЗ) на програмний продукт. Правила складання ТЗ. Літ.: [1], [2], [5]	2

1	*2*	*3*
13	**Лекція 13. Основи проєктування програмного забезпечення** Базові концепції проєктування ПЗ. Проєктування ПЗ за SWEBOK, за стандартами ДСТУ ISO/IEC/IEEE 12207:2018, ДСТУ ISO/IEC/IEEE 42010:2018. Архітектурне та детальне проектування. Поняття структури та архітектури ПЗ. Основні види архітектур ПЗ. Архітектурні стилі та шаблони (патерни). Фізичне представлення архітектурних моделей систем у мові UML (діаграми реалізації). Методична, технологічна, інструментальна та організаційна підтримка процесу проєктування ПЗ. Структурно-функціональне проєктування ПЗ. Проєктування ПЗ при об'єктному підході. Проєктування інтерфейсу користувача. Літ.: [1], [2], [5], [6]	2
14	**Лекція 14. Основи конструювання програмного забезпечення** Конструювання ПЗ за SWEBOK. Управління конструюванням ПЗ. Моделі конструювання. Вимірювання в конструюванні. Огляд парадигм програмування. Класифікація мов програмування. Прикладні та теоретичні методи програмування. Інструментальні засоби та інтегровані середовища програмування. Транслятори. Кодування. Правила оформлення модулів. Інтеграція. Оптимізація програм. Способи економії пам'яті. Способи скорочення часу виконання. Стандарти конструювання ПЗ. Літ.: [1], [2], [5]	2
15	**Лекція 15. Основи тестування та якості програмного забезпечення** Поняття та мета тестування ПЗ. Основні види тестування. Рівні тестування. Стратегії тестування. Локалізація та виправлення програмних помилок. Особливості тестування об'єктно-орієнтованих модулів та об'єктно-орієнтованої інтеграції. Тестування інтерфейсу користувача. Верифікація та валідація (атестація) ПЗ. Розробка планів тестування. Стратегія розроблення тестових наборів даних. Основні поняття в області якості ПЗ. Метрики і моделі якості програмних систем. Стандарти якості ПЗ. Літ.: [1], [2], [5]	2
16	**Лекція 16. Документування програмного забезпечення** Загальні принципи розроблення документації, пов'язаної з життєвим циклом ПЗ. Передпроєктна та проєктна документація. Документація, призначена для користувача ПЗ. Електронна документація. Інтерактивні електронні керівництва. Стандарти, що регламентують розроблення документації ПЗ. Технічні письменники. Автоматизація розроблення програмної документації. Літ.: [1]	2
17	**Лекція 17. Управління конфігурацією в життєвому циклі ПЗ. Впровадження, супровід та моніторинг ПЗ** Процеси управління конфігурацією програмних засобів. Етапи і процедури при управлінні конфігурацією ПЗ. Технологічне забезпечення при управлінні конфігурацією ПЗ. Основні етапи готовності програмного продукту (альфа-версія, бета-версія, реліз). Організація, методи і процедури супроводу ПЗ. Техніки супроводу. Реінжинірінг, зворотній реінжинірінг. Рефакторинг. Реверсна інженерія. Літ.: [1], [5]	2
18	**Лекція 18. Етичні та культурні аспекти інженерії програмного забезпечення** Поняття етики і моралі. Категорії етики. Професійна та корпоративна етика. Комп'ютерна етика. Принципи і правила ділової поведінки. Роль професійних співтовариств у розробці етичних обов'язків розробників ПЗ. Кодекс этики і професійної практики IEEE-CS/ACM та його принципи. Корпоративна культура розробки ПЗ (на прикладі компанії Microsoft). Літ.: [2], [5]	2
	Разом за 2-й семестр:	**36**

5.2 Зміст лабораторних занять

№ з/п	Тема лабораторного заняття	Кількість годин
	Другий семестр	
1	Моделювання на основі електронних таблиць. Аналіз моделей Літ.: [7]	4
2	Структурно-функціональне моделювання за методологією SADT (IDEF0) Літ.: [7]	4
3	Аналіз і моделювання потоків даних. Декомпозиція процесів у нотації IDEF3 Літ.: [7]	4
4	Інфологічне моделювання. Побудова діаграм «сутність-зв'язок» Літ.: [7]	4
5	Аналіз вимог до ПЗ. Побудова UML-діаграм варіантів використання Літ.: [7]	4
6	Аналіз вимог до ПЗ. Побудова UML-діаграм станів та діяльності Літ.: [7]	4
7	Аналіз вимог до ПЗ. Побудова UML-діаграм взаємодії (послідовності та кооперації) Літ.: [7]	4
8	Проєктування класів. Розроблення UML-діаграм реалізації. Літ.: [7]	8
	Разом за 2-й семестр:	**36**

5.3 Зміст самостійної (у т. ч. індивідуальної) роботи

Самостійна робота студентів полягає у систематичному опрацюванні програмного матеріалу з відповідних джерел інформації, ідготовці до виконання та захисту лабораторних робіт, підготовці до тестування з теоретичного та практичного матеріалу.

Керівництво самостійною роботою студентів здійснюється викладачем згідно з розкладом консультацій у позаурочний час.

Номер тижня	Вид самостійної роботи	Кількість годин
	Другий семестр	
1	*2*	*3*
1	Опрацювання теоретичного матеріалу. Підготовка до виконання ЛР1.	4
2	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР1. Підготовка до виконання ЛР2.	5
3	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР2. Підготовка до виконання ЛР3.	4
4	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР2. Підготовка до виконання ЛР3.	5
5	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР3. Підготовка до виконання ЛР4.	4
6	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР3. Підготовка до виконання ЛР4.	4
7	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР4. Підготовка до виконання ЛР5. Підготовка до тестового контролю з тем 1-2.	5

1	*2*	*3*
8	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР4. Підготовка до виконання ЛР5. Підготовка до тестового контролю з тем 1-2.	5
9	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР5. Підготовка до виконання ЛР6.	4
10	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР5. Підготовка до виконання ЛР6.	4
11	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР6. Підготовка до виконання ЛР7.	4
12	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР6. Підготовка до виконання ЛР7.	4
13	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР7. Підготовка до виконання ЛР8.	4
14	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР7. Підготовка до виконання ЛР8.	4
15	Опрацювання теоретичного матеріалу. Підготовка до тестового контролю з тем 3-5.	5
16	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР8. Підготовка до тестового контролю з тем 3-5.	5
17	Опрацювання теоретичного матеріалу. Підготовка до захисту ЛР8.	4
18	Опрацювання теоретичного матеріалу.	4
	Разом за 2-й семестр:	**78**

6 ТЕХНОЛОГІЇ ТА МЕТОДИ НАВЧАННЯ

Процес навчання з дисципліни ґрунтується на використанні традиційних та сучасних технологій, зокрема: лекції (з використанням методів проблемного навчання і візуалізації); лабораторні заняття (з використанням методів інформаційних технологій та сучасних Case-засобів моделювання і проєктування, майстер-класів, практикумів), самостійна робота, і мають за мету – оволодіння студентами спеціальною термінологією; вивчення студентами тих аспектів діяльності, які складають суть професії розробника ПЗ; набуття ними практичних навичок з моделювання та проєктування ПЗ; формування у студентів базових компетентностей, необхідних для подальшого вивчення циклу професійно-орієнтованих дисциплін.

7 ФОРМИ І МЕТОДИ ОЦІНЮВАННЯ РЕЗУЛЬТАТІВ НАВЧАННЯ

Поточний контроль здійснюється під час лабораторних занять, а також у дні проведення контрольних заходів, встановлених робочою програмою та графіком навчального процесу. При цьому використовуються наступні методи поточного контролю: усне опитування перед допуском до лабораторного заняття; захист лабораторних робіт; тестовий контроль теоретичного матеріалу з теми; презентація виконаних завдань.

Семестровий контроль проводиться у формі заліку.

Студент, який не набрав позитивний середньозважений бал за поточну роботу у семестрі, вважається невстигаючим.

Оцінювання академічних досягнень здобувача вищої освіти здійснюється відповідно до «Положення про контроль і оцінювання результатів навчання здобувачів вищої освіти у ХНУ». Кожний вид роботи з дисципліни оцінюється за інституційною ***чотирибальною*** шкалою. Семестрова підсумкова оцінка визначається як середньозважена з усіх видів навчальної роботи,

виконаних і зданих ***позитивно*** з урахуванням коефіцієнта вагомості. Вагові коефіцієнти змінюються залежно від структури дисципліни і важливості окремих видів її робіт.

Оцінка, яка виставляється за лабораторну роботу, складається з таких елементів: усне опитування студентів перед допуском до виконання лабораторної роботи; знання теоретичного матеріалу з теми; якість оформлення звіту з лабораторної роботи; вільне володіння студентом спеціальною термінологією і уміння професійно обґрунтувати прийняті конструктивні рішення; своєчасний захист лабораторної роботи.

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

Пропущене лабораторне заняття студент зобов'язаний відпрацювати не пізніше, ніж за два тижні до кінця теоретичних занять у семестрі.

Засвоєння студентом теоретичного матеріалу з дисципліни оцінюється тестуванням.

Структурування дисципліни за видами робіт і оцінювання результатів навчання студентів денної форми навчання у семестрі за ваговими коефіцієнтами

Аудиторна робота			**Семестровий контроль**
Лабораторні роботи	Тестовий контроль		
№ 1 – 8	ТК1 (Т1-2)	ТК2 (Т3-5)	Залік
ВК: 0,5	**0,5**		

Умовні позначення: *ВК – ваговий коефіцієнт, ТК – тестовий контроль, Т – тема дисципліни.*

Оцінювання тестових завдань

Тематичний тест для кожного студента складається з 20 тестових завдань, кожне з яких оцінюється одним балом. Максимальна сума балів, яку може набрати студент, складає 20. Тестові завдання для кожного студента генеруються випадковим чином із загального банку питань у середовищі для навчання Moodle. Оцінювання відповідей студента здійснюється в автоматичному режимі. Оцінювання здійснюється за чотирибальною шкалою. Сума балів пропорційна кількості правильних відповідей. Відповідність набраних балів за тестове завдання оцінці, що виставляється студенту, представлена у нижченаведеній таблиці.

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

На тестування відводиться 25 хвилин. При отриманні негативної оцінки тест слід перездати до терміну наступного контролю.

Підсумкова семестрова оцінка за інституційною шкалою і шкалою ЄКТС встановлюється в автоматизованому режимі після внесення викладачем усіх оцінок до електронного журналу.

Залік виставляється, якщо середньозважений бал, який отримав студент з дисципліни, знаходиться у межах від 3,00 до 5,00 балів. При цьому за інституційною шкалою ставиться оцінка «зараховано», а за шкалою ЄКТС – буквене позначення оцінки, що відповідає набраній студентом кількості балів. Співвідношення інституційної шкали оцінювання і шкали оцінювання ЄКТС наведені у таблиці.

Співвідношення інституційної шкали оцінювання і шкали оцінювання ЄКТС

Оцінка ЄКТС	**Інституційна інтервальна шкала балів**	**Вітчизняна оцінка, критерії**	
A	4,75-5,00	Зараховано	ВІДМІННО – глибоке і повне опанування навчального матеріалу і виявлення відповідних умінь та навиків
B	4,25-4,74		ДОБРЕ – повне знання навчального матеріалу з кількома незначними помилками

C	3,75-4,24		ДОБРЕ – в загальному правильна відповідь з двома-трьома суттєвими помилками
D	3,25-3,74		ЗАДОВІЛЬНО – неповне опанування програмного матеріалу, але достатнє для практичної діяльності за професією
E	3,00-3,24		ЗАДОВІЛЬНО – неповне опанування програмного матеріалу, що задовольняє мінімальні критерії оцінювання
FX	2,00 -2,99	Незараховано	НЕЗАДОВІЛЬНО – безсистемність одержаних знань і неможливість продовжити навчання без додаткових знань з дисципліни
F	0,00-1, 99		НЕЗАДОВІЛЬНО – необхідна серйозна подальша робота і повторне вивчення дисципліни

8. ПИТАННЯ ДЛЯ САМОКОНТРОЛЮ РЕЗУЛЬТАТІВ НАВЧАННЯ

1. Поняття програми, програмного засобу, програмного забезпечення. Види ПЗ за типами ліцензій.
2. Причини та передумови виникнення дисципліни «Інженерія ПЗ». Визначення інженерії ПЗ.
3. Інженерія ПЗ як інженерна та наукова дисципліна.
4. Стандартизація в інженерії ПЗ.
5. Основні області знань SWEBOK.
6. Життєвий цикл програмного забезпечення.
7. Загальна характеристика каскадної моделі життєвого циклу ПЗ; її переваги і недоліки.
8. Загальна характеристика спіральної моделі життєвого циклу ПЗ та її переваги і недоліки.
9. Загальна характеристика ітераційної моделі життєвого циклу ПЗ та її переваги і недоліки.
10. Загальна характеристика інкрементної (покрокової) моделі життєвого циклу ПЗ та її переваги і недоліки.
11. Загальна характеристика методології швидкої розробки додатків RAD (Rapid Application Development) та її переваги і недоліки.
12. Методологія RUP (Rational Unified Process). Загальна характеристика процесу розробки ПЗ.
13. Модель проектної групи MSF (Microsoft Solutions Framework).
14. Модель процесів MSF (Microsoft Solutions Framework).
15. Гучкі методології розробки ПЗ. Загальна характеристика.
16. Екстремальне програмування (XP).
17. Методологія Scrum.
18. Міжнародні та національні стандарти життєвого циклу ПЗ.
19. Сутність та методи структурного аналізу і моделювання.
20. Загальна характеристика методологій структурного аналізу і моделювання.
21. Основні принципи структурного аналізу і моделювання.
22. Нотація IDEF0 для графічного представлення SADT-моделі.
23. Синтаксис і семантика DFD-діаграм.
24. Базові поняття інформаційного моделювання «сутність-зв’язок» (ERD).
25. Огляд нотацій, що використовуються при побудові ER-діаграм.
26. Методика побудови ER-діаграм.
27. Сутність об’єктно-орієнтованого аналізу і моделювання; його переваги і недоліки.
28. Основні принципи об’єктно-орієнтованого аналізу і моделювання.
29. Загальна характеристика мови UML. Діаграми UML.
30. Загальний процес розробки вимог до ПЗ.
31. Властивості вимог до ПЗ.
32. Джерела та стратегії виявлення вимог до ПЗ.
33. Аналіз, систематизація та перевірка вимог до ПЗ.
34. Специфікація та формалізація вимог до ПЗ.
35. Технічне завдання на ПЗ.
36. Поняття архітектури ПЗ. Основні класи архітектур.
37. Основні принципи проектування архітектури ПЗ.
38. Модульна структура програмного продукту. Основні характеристики модулів.
39. Типи програмних модулів та їх структура.
40. Низхідне («зверху-донизу») та висхідне («знизу-доверху») проектування ПЗ.
41. Розробка структури ПЗ при об'єктному підході.
42. Принципи, правила та етапи проектування інтерфейсу ПЗ.
43. Парадигми програмування.
44. Поняття мови програмування. Етапи розвитку мов програмування.
45. Класифікація мов програмування.
46. Інструментальні засоби та інтегровані середовища програмування. Транслятори.
47. Основи конструювання ПЗ. Стандарти в конструюванні. Управління конструюванням.

48. Класифікація програмних помилок. Методи відладки ПЗ. Загальна методика відладки ПЗ.
49. Основні види тестування.
50. Рівні тестування (модульне, інтеграційне та системне тестування).
51. Тести. Класифікація тестів. Валідаційні тести. Вимірювання результатів тестування.
52. Особливості об'єктно-орієнтованого тестування.
53. Особливості тестування об'єктно-орієнтованих модулів.
54. Верифікація та атестація ПЗ.
55. Загальні принципи складання та оформлення документації ПЗ.
56. Основні види програмної документації.
57. Характеристика документації, призначеної для користувача ПЗ.
58. Електронна довідкова система.
59. Процеси управління конфігурацією програмних засобів.
60. Етапи і процедури при управлінні конфігурацією ПЗ.
61. Технологічне забезпечення при управлінні конфігурацією ПЗ.
62. Основні етапи готовності програмного продукту. Альфа-версія. Бета-версія. Реліз.
63. Організація і методи супроводу ПЗ. Етапи і процедури при супроводі ПЗ.
64. Техніки супроводу. Реінжинірінг, зворотній реінжинірінг.
65. Кодекс етичних норм професіонала в області інженерії ПЗ (IEEE-CS/ACM).
66. Корпоративна етика та культура.

9 МЕТОДИЧНЕ ЗАБЕЗПЕЧЕННЯ

Навчальний процес з дисципліни повністю і в достатній кількості забезпечений необхідними навчально-методичними розробками у Модульному навчальному середовищі. Доступ до ресурсу: https://msn.khmnu.edu.ua/course/view.php?id=3403).

10 РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Бородкіна І., Бородкін Г. Інженерія програмного забезпечення : посіб. для студ. вищих навч. закладів. К : Вид-тво "Центр навчальної літератури", 2018. 204 с.

2. Левус Є., Мельник Н.. Вступ до інженерії програмного забезпечення. Л. : Львівська політехніка, 2018. 248 с.

3. Моделювання бізнес-процесів та управління ІТ-проектами : навч. посіб. / Крижановський Є. М., Ящолт А. Р., Жуков С. О., Козачко О. М. Вінниця: ВНТУ, 2018. 91 с.

4. Постіл С. Д. UML. Уніфікована мова моделювання інформаційних систем : навч. посіб. Ірпінь : УДФСУ, 2019. 322 с.

5. Rod Stephens. Beginning Software Engineering: Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.

6. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 P.

Додаткова

7. Електронні версії методичних вказівок до лабораторних робіт.

8. Ronald J. Leach. Introduction to Software Engineering: Second Edition. Howard University, Washington, DC, USA. CRC Press, Taylor & Francis Group. 2018. 420 P.

9. Автоматизоване проектування інформаційних систем. URL: https://posibniki.com.ua/catalog-avtomatizovane-proektuvannya-informaciynih-sistem---denisova-o-o

10. Левус Є. В., Марусенкова Т. А. Вступ до інженерії програмного забезпечення : 1024 завдання для підготовки до контрольних заходів. Львів : Львівська політехніка, 2021. 188 с.

11. Мартін Роберт. Чистий Agile: назад до основ; пер. з англ. В. Луненко. Харків : Вид-во «Ранок»: Фабула, 2021. 224 с.

12. Software Engineering Body of Knowledge (SWEBOK). URL: https://www.computer.org/education/bodies-of-knowledge/software-engineering

13. Introduction to the Microsoft Solutions Framework. URL: https://learn.microsoft.com/en-us/previous-versions/tn-archive/bb497060(v=technet.10)?redirectedfrom=MSDN

14. Manifesto for Agile Software Development. URL: http://agilemanifesto.org/

15. The Software Engineering Code of Ethics and Professional Practice. URL: https://ethics.acm.org/code-of-ethics/software-engineering-code/

16. ACM Code of Ethics and Professional Conduct. URL: https://ethics.acm.org/code-of-ethics/

17. ДСТУ ISO/IEC/IEEE 24765:2018 (ISO/IEC/IEEE 24765:2017, IDT). Інженерія систем і програмних засобів. Словник термінів.

18. ДСТУ ISO/IEC/IEEE 12207:2018. Інженерія систем і програмних засобів. Процеси життєвого циклу програмних засобів (ISO/IEC/IEEE 12207:2017, IDT).

19. ДСТУ ISO/IEC TR 24774:2016 (ISO/IEC TR 24774:2010, IDT). Інженерія систем і програмних засобів. Керування життєвим циклом. Настанови щодо опису процесу.

20. ДСТУ ISO/IEC TS 24748-1:2018 (ISO/IEC TS 24748-1:2016, IDT). Інженерія систем і програмних засобів. Керування життєвим циклом. Частина 1. Настанови щодо керування життєвим циклом.

21. ДСТУ ISO/IEC TR 24748-3:2016 (ISO/IEC TR 24748-3:2011, IDT). Інженерія систем і програмного забезпечення. Керування життєвим циклом. Частина 3. Настанова щодо застосування ISO/IEC 12207 (Процеси життєвого циклу програмного забезпечення).

22. ISO/IEC/IEEE 29148-2018. Systems and software engineering – Life cycle processes – Requirements engineering.

23. ДСТУ ISO/IEC TR 24766:2016 (ISO/IEC 24766:2009, IDT). Інформаційні технології. Інженерія систем і програмних засобів. Настанови щодо розроблення технічних вимог до програмних засобів.

24. ДСТУ ISO/IEC/IEEE 42010:2018 (ISO/IEC/IEEE 42010:2011, IDT). Інженерія систем і програмних засобів. Опис архітектури.

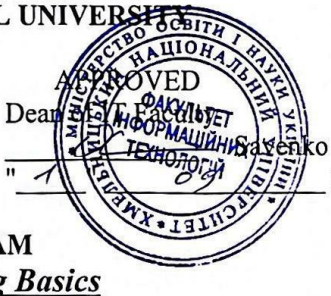
25. ДСТУ ISO/IEC/IEEE 15289:2019 (ISO/IEC/IEEE 15289:2017, IDT). Інженерія систем і програмних засобів. Уміст інформаційних об'єктів життєвого циклу (документації).

26. ДСТУ ISO/IEC/IEEE 26515:2018 (ISO/IEC/IEEE 26515:2011, IDT). Інженерія систем і програмних засобів. Розроблення документації користувача в гнучкому середовищі.

11 ІНФОРМАЦІЙНІ РЕСУРСИ

1. Модульне середовище для навчання. Доступ до ресурсу: https://msn.khmnu.edu.ua/
2. Електронна бібліотека університету.
   Доступ до ресурсу: http://lib.khmnu.edu.ua/asp/php_f/p1age_lib.php
3. Репозитарій ХНУ. Доступ до ресурсу: http://elar.khmnu.edu.ua/jspui/

KHMELNYTSKYI NATIONAL UNIVERSITY



APPROVED
Dean of IT Faculty
_____ Savenko O.S.
"___" _____ 2023.

COURSE PROGRAM
Software Engineering Basics

Field of study: 12 - Information Technologies
Major: 121 – Software Engineering
Level of Higher Education: First Level (Bachelor)
Educational program: Software Engineering
Discipline status: Compulsory
Faculty: Information Technologies
Department: Software Engineering

Study mode	Year	Semester	Total Credits	Number of hours						Course project	Coursework	Semester control form	
				Classwork hours				Seminar classes	Independent work, including individual				
			ECTS credits	Total	Lectures	Laboratory works	Practical classes					pass/ fail test	Exam
Full-time (Daytime)	1	2	5	150	36	36			78			+	1
Total			**5**	**150**	**36**	**36**			**78**				

The course program is based on the Higher Education Standard, the 2023 Bachelor's degree educational program, and the curriculum.

Program's author _____ Y.H. Hnatchuk

Approved at the staff meeting of the Department of Software Engineering
Minutes from 31 08 2023 No. 1

Chief Department of Software Engineering _____ L.P. Bedratyuk

The course program is approved by the Academic Board of the Faculty of Information technologies

Head of the Academic Board _____ O.S. Savenko

Khmelnytskyi 2023

SOFTWARE ENGINEERING BASICS

Type of discipline	Compulsory
Level of higher education	First (Bachelor's)
Language of teaching	Ukrainian, English
Semester	Second
The amount of ECTS loans	5
Form of education	Full-time(Daytime)

Learning outcomes. According to the Standard of higher education in the specified specialty and educational program, the discipline must ensure:

– ***competences*** : the ability to solve complex specialized tasks or practical problems of software engineering, characterized by complexity and uncertainty of conditions, using theories and methods of information technologies (integral competence) ; the ability to participate in software design , including modeling (formal description) of its structure, behavior and functioning processes ; the ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards; the ability to adhere to specifications, standards, rules and recommendations in the professional field when implementing life cycle processes; the ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering tasks; the ability to accumulate, process and systematize professional knowledge regarding the creation and maintenance of software and recognition of the importance of lifelong learning; the ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate models and approaches to software development.

– ***program learning outcomes*** : analyze, purposefully search for and choose information and reference resources and knowledge necessary for solving professional tasks, taking into account modern achievements of science and technology; know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities; know the main processes, phases and iterations of the software life cycle; know and apply professional standards and other legal documents in the field of software engineering; conduct a pre-project survey of the subject area, system analysis of the design object; select initial data for design, guided by formal requirements description and modeling methods ; apply effective approaches to software design in practice ; apply in practice instrumental software tools for domain analysis, design, testing, visualization, measurement and documentation of software ; have skills in team development, approval, design and release of all types of software documentation; be able to document and present the results of software development.

Content of the academic discipline. Engineering basics of software. About modeling dreams. Fundamentals of software requirements engineering. Software development technologies . Ethical and cultural aspects of software engineering.

Planned educational activity : lectures - 36 hours, laboratory classes - 36 hours, independent work - 78 hours, together - 150 hours.

Forms (methods) of education : lectures (using methods of problem-based learning and visualization); laboratory classes (using information technology methods and modern Case modeling and design tools, master classes, workshops), independent work.

Forms of evaluation of learning results : defense of laboratory works; thematic online testing.

Semester control form: pass\fail test

Educational resources:

1. Borodkina I., Borodkin G. Software engineering: manual. for students higher education institutions K: Publishing House "Center for Educational Literature", 2018. 204 p.
2. Levus E., Melnyk N.. Introduction to software engineering. L.: Lviv Polytechnic, 2018. 248 p.
3. Rod Stephens. Beginning Software Engineering : Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.
4. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 p.
5. MOODLE modular learning environment . Access to the resource: https :// msn . khmnu _ edu . u /
6. University electronic library . Access to the resource: http://lib.khmnu.edu.ua/asp/php_f/p1age_lib.php

Teacheri: associate professor, PhD Hnatchuk E.H., assistant Dyomina A.I.

3 EXPLANATORY NOTE

Fundamentals of software engineering " discipline, as one of the professional disciplines, occupies a leading place in the training of "bachelor" level specialists in the specialty 121 "Software engineering" under the educational and professional program "Software engineering" and is a theoretical and practical - foundation for further study by students of the cycle of professionally oriented disciplines.

According to the Standard of higher education in the specified specialty and educational program, the discipline must ensure:

– ***competences*** : IC – the ability to solve complex specialized tasks or practical problems of software engineering, characterized by complexity and uncertainty of conditions, using theories and methods of information technologies ; PC02 – the ability to participate in software design, including modeling (formal description) of its structure, behavior and functioning processes ; PC04 – the ability to formulate and ensure software quality requirements in accordance with customer requirements, specifications and standards; PC05 – the ability to follow specifications, standards, rules and recommendations in the professional field when implementing life cycle processes; PC08 – the ability to apply fundamental and interdisciplinary knowledge to successfully solve software engineering tasks; PC10 – the ability to accumulate, process and systematize professional knowledge regarding the creation and maintenance of software and recognition of the importance of lifelong learning; PC11 – the ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate models and software development approaches.

– ***program learning outcomes*** : PLON01 – to analyze, purposefully search for and choose the necessary information and reference resources and knowledge for solving professional tasks, taking into account modern achievements of science and technology; PLO02 – know the code of professional ethics, understand the social significance and cultural aspects of software engineering and adhere to them in professional activities; PLO03 – to know the main processes, phases and iterations of the software life cycle; PLO04 – know and apply professional standards and other legal documents in the field of software engineering; PLO10 – conduct a pre-project survey of the subject area, system analysis of the project - object ; PLO11 – select initial data for design, guided by formal methods of requirements description and modeling ; PLO12 – apply effective approaches to software design in practice ; PLO14 – apply in practice instrumental software tools of domain analysis, design, testing, visualization, measurement and documentation of software ; PLO6 – to have skills in team development, approval, design and release of all types of technical documentation; PLO23 – to be able to document and present the results of software development.

The purpose of the discipline . The purpose of the discipline is: to teach students the fundamentals of software engineering in accordance with the content of the international professional standard Guide to the Software Engineering Body of Knowledge (SWEBOK); to form in them a general idea of tasks, processes, methodologies, methods and means of information technologies used in software engineering; familiarize them with the basic provisions of software development technologies, including consideration of issues related to analysis, design, implementation, testing, documentation and support of software products.

Subject of discipline . Methods and means of software engineering with an emphasis on key aspects of theoretical and practical training in the processes of software development organization, taking into account the SWEBOK core of knowledge .

Tasks of the discipline . Formation of students of higher education basic theoretical knowledge and practical skills in the field of software engineering at all stages of the life cycle of software, mastering the necessary equipment for further study of the cycle of professionally oriented disciplines.

Learning outcomes **.** After studying the discipline, the student should: have professional terminology and basic concepts of software engineering; describe areas of software engineering knowledge according to SWEBOK; characterize models and processes of the software life cycle based on

current standards; to navigate modern software development methodologies and technologies; use methods, notations and tools for software modeling and design; build structural-functional , infological and object-oriented software models; evaluate the degree of adequacy of the developed models; document the decisions made based directly on the models; demonstrate a culture of thinking when developing models; analyze the subject area and determine the functional requirements for the software; analyze the problems and trends in the development of software engineering; apply the methods and means of software engineering in the study of the cycle of professionally oriented disciplines.

Discipline Policy. The organization of the educational process for the discipline complies with the requirements of the provisions on organizational and instructional-methodological support of the educational process, the educational program, and the curriculum. Students are required to attend lectures, practical classes, laboratory work, etc., according to the schedule, not to be late for classes, and to complete all tasks and checkpoints according to the schedule. Missed practical classes and laboratory work must be independently completed by the student in full and reported to the instructor no later than one week before the next assessment. For practical classes and laboratory work, students must prepare on the relevant topic and demonstrate active participation. Knowledge acquired by an individual in the discipline or its specific sections through informal education is credited according to the Regulation on the procedure for transferring learning outcomes and determining academic differences at KhNU

4 STRUCTURE OF CREDIT CREDITS OF THE COURSE

Name of the section (topics)	The number of hours allocated to:		
	lectures	laboratory work	SRS
Topic 1. Engineering basics of software	6		13
Topic 2. Basics of modeling	12	16	27
Topic 3. Fundamentals and engineering of software requirements	6	12	12
Topic 4. Development technologies Software	10	8	22
Topic 5. Ethical and cultural aspects of software engineering	2		4
Together:	**36**	**36**	**78**

5 PROGRAM OF EDUCATIONAL DISCIPLINE

5.1 Content of the lecture course

Number lectures	List of topics of lectures, their annotations	Number of hours
1	*2*	*3*
1	**Lecture 1. Introduction to software engineering** The program as a formalized description of the data processing process. Software, software product, software (software). Types of software according to different characteristics. Software cost components. And software engineering as a set of engineering methods and means of software creation. The subject, content and practical application of the software engineering subject . Engineers and other specialists in software engineering. Standardization in software engineering. Software Engineering Body of Knowledge (SWEBOK). General characteristics of SWEBOK knowledge areas. Lit.: [1], [2] , [6]	2
2	**Lecture 2. Life cycle of software** The concept of the software life cycle. Basic, auxiliary and organizational processes of the residential complex PZ. Communication between processes. Classic models of LCD (cascade, spiral, iterative, incremental), their advantages and disadvantages. Choosing a model of residential housing. Standards of ZhC PZ. Letters: [1], [2], [5], [6]	2
3	**Lecture 3. Life cycle models in modern software development methodologies** Methodologies for rapid software development. RUP (Rational Unified Process) methodology . MSF (Microsoft Solution Framework) and MOF (Microsoft Operations Framework) methodologies. Flexible methodologies. XP , Scrum , Crystal family of methodologies, DSDM (Dynamic Systems Development Method), FDD (Feature Driven Development), adaptive development (ASD). The place of flexible methodologies among modern software development technologies . Letters: [1], [2], [5]	2
4	**Lecture 4. Structural and functional analysis and modeling. SADT methodology** Concept of system, model, simulation. Basic approaches to analysis and modeling in software engineering (structural, object-oriented). The essence and basic principles of structural and functional analysis and modeling . Basic principles of the Structured methodology Analysis and Design Technique (SADT). The state of the functional modeling I D EF0 and I D EF3 . For the general method of constructing I D EF0 and I D EF3 diagrams. Case - from the person of automation of structural and functional modeling. Lit.: [3]	2
5	**Lecture 5. Analysis and modeling of data flows** Concept of data flow diagram (**Data** Flow **Diagrams ,** DFD) **.** Syntax and semantics of DFD diagrams. Jordan-De Marco and Heine-Sarson notations . De composition and corresponding extensions of data flow diagrams. Construction of DFD diagrams. Building a hierarchy of data flows. Case - on	2

	behalf of the automation of data flow modeling . Comparative analysis of SADT and DFD models. Lit.: [3], [5], [6]	

1	*2*	*3*
6	**Lecture 6. Infologic modeling** The concept of the subject area. Infologic modeling and its purpose. Conceptual , logical and physical levels of modeling. Entity-Relationship Diagrams (ERD) model. Constructive elements of infological models. Fundamental types of connections. Data normalization. Overview of ER notations. The general method of constructing ER-diagrams. Case tools for the automation of info modeling. . Lit.: [3], [6]	2
7	**Lecture 7. Unified modeling language UML. Use case diagrams** General concepts and essence of the object-oriented approach (OOP) to software development. Principles of OOP. RUP (Rational Unified Process) methodology. Characteristics and conceptual foundations of UML (Unified Modeling Language). Types of UML diagrams. Case tools for automating object-oriented modeling. Diagrams and options for use. Lit.: [4 – 6]	2
8	**Lecture 8. Unified modeling language UML. Details of use cases** Ways of describing use cases. Event streams (main, alternate). Textual description of event streams. State diagrams and activity diagrams, their structural elements. Features of construction of status and activity diagrams. Lit.: [4 – 6]	2
9	**Lecture 9. Unified modeling language UML. Documenting use cases with interaction diagrams. Class diagrams** The concept of a usage scenario. Main and alternative scenarios. Forms of presentation of interaction diagrams. Sequence diagram and cooperation diagram, their elements. Peculiarities of constructing interaction diagrams. Class diagram, its elements. Features of class diagram construction. Lit.: [4 – 6]	2
10	**Lecture 10. The process of developing software requirements** The concept of software requirements. Classification of requirements. Methodologies and standards regulating work with software requirements (SWEBOK, DSTU ISO/IEC/IEEE 12207:2018, ISO/IEC/IEEE 29148-2018, DSTU ISO/IEC TR 24766:2016) . Properties of software requirements. Participants in the development of software requirements. The general process of developing software requirements. Year : [1], [2], [5]	2
11	**Lecture 11. Sources and strategies for identifying software requirements** Sources of software requirements. Basic strategies for identifying requirements. Peculiarities of identifying requirements for a software product "to order" and for the open market. Feasibility analysis. Prototyping. Modeling. Communications about the project team and the customer in the process of developing software requirements. Letters: [1], [2], [5]	2
12	**Lecture 12. Analysis, systematization and specification of software requirements** Purpose of software requirements analysis. Basic procedures for verification and validation of software requirements. Specification of software requirements. Specification of functional requirements for software using use cases.	2

	Formalization of requirements. Technical task (TOR) for the software product. Rules for drawing up technical specifications. Letters: [1], [2], [5]	
1	*2*	*3*
13	**Lecture 13. Fundamentals of software design** Basic concepts of software design. Software design according to SWEBOK, according to standards DSTU ISO / IEC / IEEE 12207 :2018, DSTU ISO/IEC/IEEE 42010:2018 . Architectural and detailed design. Concept of software structure and architecture. The main types of software architectures. Architectural styles and templates (patterns). Physical presentation of architectural models of systems in the UML language (implementation diagrams). Methodical, technological, instrumental and organizational support for the software design process. Structural and functional software design. Software design using an object-oriented approach. User interface design . Letters: [1], [2], [5], [6]	2
14	**Lecture 14. Fundamentals of Software Construction** Software design according to SWEBOK. Software design management. Construction models. Measurement in construction. Overview of programming paradigms . Classification of programming languages. Applied and theoretical methods of programming. Tools and integrated programming environments . Translators . Coding. Module design rules. Integration. Program optimization. Ways to save memory. Ways to reduce execution time. Software design standards. Letters: [1], [2], [5]	2
15	**Lecture 15. Fundamentals of software testing and quality** The concept and purpose of software testing. The main types of testing. Levels of testing. Testing strategies. Localization and correction of software errors. Peculiarities of testing object-oriented modules and object -oriented integration. User interface testing. Verification and validation (attestation) of software. Development of test plans. Strategy for development of test data sets. Basic concepts in the field of software quality. Metrics and quality models of software systems. Software quality standards. Letters: [1], [2] , [5]	2
16	**Lecture 16. Software documentation** General principles of developing documentation related to the software life cycle. Pre-project and project documentation. Documentation intended for the user of the software. Electronic documentation. Interactive electronic guides. Standards regulating the development of software documentation. Technical writers. Automation of development of software documentation. Lit.: [1]	2
17	**Lecture 17. Configuration management in the software life cycle. Software implementation, support and monitoring** Software configuration management processes. Stages and procedures for software configuration management. Technological support in software configuration management. The main stages of software product readiness - (alpha version, beta version, release). Organization, methods and procedures of - software development. Escort techniques. Reengineering, reverse reengineering. Refactoring. Reverse engineering. Letters: [1], [5]	2
18	**Lecture 18. Ethical and cultural aspects of software engineering** The concept of ethics and morality. Categories of ethics. Professional and corporate ethics. Computer ethics . Principles _ _ and the rules of behavior . _ _ _ The role of professional communities in the development of ethical responsibilities of software developers. IEEE - CS / ACM Code of Ethics and Professional Practice and its principles. Corporate culture of software development (using the example of the Microsoft company).	2

	Lit.: [2] , [5]	
	Total for the 2nd semester:	**36**

5.2 Content of laboratory classes

No s/p	**The topic of the laboratory session**	**Number hours**
	Second semester	
1	Modeling based on electronic spreadsheets. Analysis of models Lit.: [7]	4
2	Structural and functional modeling according to the SA D T methodology (IDEF 0) Lit.: [7]	4
3	Analysis and modeling of data flows. Decomposition of processes in IDEF 3 notation Lit.: [7]	4
4	Infologic modeling. Construction of entity- relationship diagrams Lit.: [7]	4
5	Analysis of software requirements. Construction of UML - diagrams of use cases Lit.: [7]	4
6	Analysis of software requirements. Construction of UML - state and activity diagrams Lit.: [7]	4
7	Analysis of software requirements. Construction of UML diagrams of interaction (sequence and cooperation) Lit.: [7]	4
8	Class design. Development of UML - implementation diagrams. Lit.: [7]	8
	Total for the 2nd semester:	**36**

5.3 Content of independent (including individual) work

Independent work of students consists in systematic processing of program material from relevant sources of information, preparation for performance and defense of laboratory work, preparation for testing from theoretical and practical material.

The guidance of students' independent work is carried out by the teacher according to the schedule of consultations outside of class time.

Number of the week	**Type of independent work**	**Number of hours**
	Second semester	
1	*2*	*3*
1	Development of theoretical material. Preparation for the implementation of LR1.	4
2	Development of theoretical material. Preparation for the defense of LR1. Preparation for the implementation of LR2.	5
3	Development of theoretical material. Preparation for the defense of LR2. Preparation for the implementation of LR3.	4
4	Development of theoretical material. Preparation for the defense of LR2. Preparation for the implementation of LR3.	5
5	Development of theoretical material. Preparation for the defense of LR3.	4

	Preparation for the implementation of LR4.	
6	Development of theoretical material. Preparation for the defense of LR3. Preparation for the implementation of LR4.	4
7	Development of theoretical material. Preparation for the defense of LR4. Preparation for execution of LR5. Preparation for test control on topics 1-2.	5

1	*2*	*3*
8	Development of theoretical material. Preparation for the defense of LR4. In - preparation for the implementation of LR5. Preparation for test control on topics 1-2.	5
9	Development of theoretical material. Preparation for the defense of LR5. Preparation for the implementation of LR6.	4
10	Development of theoretical material. Preparation for the defense of LR5. Preparation for the implementation of LR6.	4
11	Development of theoretical material. Preparation for the defense of LR6. Preparation for the implementation of LR7.	4
12	Development of theoretical material. Preparation for the defense of LR6. Preparation for the implementation of LR7.	4
13	Development of theoretical material. Preparation for the defense of LR7. Preparation for the implementation of LR8.	4
14	Development of theoretical material. Preparation for the defense of LR7. Preparation for the implementation of LR8.	4
15	Development of theoretical material. Preparation for test control on topics 3-5.	5
16	Development of theoretical material. Preparation for the defense of LR8. Preparation for test control on topics 3-5.	5
17	Development of theoretical material. Preparation for the defense of LR8.	4
18	Development of theoretical material.	4
Total for the 2nd semester:		**78**

6 TEACHING METHODS

The teaching process in the discipline is based on the use of traditional and modern technologies, in particular: lectures (using methods of problem-based learning and visualization); laboratory classes (with the use of information technology methods and modern Case modeling and design tools, master classes, workshops), independent work, and are aimed at students' mastery of special terminology; study by students of those aspects of activity that make up the essence of the profession of a software developer; their acquisition of practical skills in software modeling and design; formation of students' basic competencies necessary for further study of the cycle of professionally oriented disciplines.

7 ASSESSMENT FORMS AND METHODS

Current control is carried out during laboratory classes, as well as on the days of control activities established by the work program and schedule of the educational process . At the same time, the following methods of current control are used: an oral survey before admission to a laboratory session;

protection of laboratory works; test control of theoretical material on the topic ; presentation of completed tasks.

The semester control is conducted in the form of credit.

A student who does not have a positive weighted average for the current work in the semester is considered to be absent.

The assessment of the academic achievements of a higher education applicant is carried out in accordance with the "Regulations on the control and evaluation of the results of studies of higher education applicants at KhNU". Each type of work in the discipline is evaluated on an institutional ***four-point*** scale. The semester final grade is defined as a weighted average of all types of academic work completed and passed ***positively*** , taking into account the weighting factor. The weighting factors change depending on the structure of the discipline and the importance of certain types of its work.

The grade given for the laboratory work consists of the following elements: an oral interview of students before admission to the laboratory work; knowledge of theoretical material on the topic; the quality of the report on laboratory work ; the student's fluency in special terminology and the ability to professionally justify the adopted constructive decisions; timely protection of laboratory work.

The deadline for the defense of the laboratory work is considered timely if the student defended it in the next class after the completion of the work.

is obliged to complete the missed laboratory class no later than two weeks before the end of theoretical classes in the semester.

The student's assimilation of the theoretical material of the discipline is assessed by testing.

Structuring of the discipline by types of work and evaluation of the learning results of full-time students in the semester by weighting coefficients

Auditory work			**Semester control**
Laboratory work	Test control		
No. 1 - 8	TC1 (T1-2)	TC2 (T3-5)	test
WC: 0.5	**0.5**		

Conventional designations : *WC – weighting factor, TK – test control, T – topic of discipline.*

Evaluation of test tasks

The thematic test for each student consists of 20 test tasks, each of which is evaluated by one point. The maximum number of points a student can score is 20. Test tasks for each student are randomly generated from a common bank of questions in the Moodle learning environment. Evaluation of the student's answers is carried out automatically. Evaluation is carried out on a four-point scale. The sum of points is proportional to the number of correct answers. Correspondence of the scored points for the test task to the grade assigned to the student is presented in the table below.

The sum of points for test tasks	1 – 11	12 - 14	15 - 18	19 - 20
Evaluation on a 4-point scale	2	3	4	5

25 minutes are allotted for testing. If a negative evaluation is received, the test should be retaken before the next control period.

The final semester grade according to the institutional scale and the ECTS scale is set in an automated mode after the teacher enters all the grades into the electronic journal.

Credit is given if the weighted average score received by the student in the discipline is between 3.00 and 5.00 points. At the same time, according to the institutional scale, the grade "credited" is given , and according to the ECTS scale, the letter designation of the grade corresponding to the number of points scored by the student is given. Correlation of the institutional rating scale and the ECTS rating scale given in the table.

The ratio of institutional assessment scales and ECTS assessment scales

Evaluation of ECTS	Institutional interval scoring scale	Domestic assessment, criteria	
A	4.75-5.00	Enrolled	EXCELLENT – deep and complete mastery of the educational material and identification of relevant skills and abilities
B	4.25-4.74		GOOD - complete knowledge of the educational material with a few minor errors
C	3.75 -4.24 _		GOOD - a generally correct answer with two or three significant errors
D	3, 25 -3, 74		SATISFACTORY - incomplete mastery of the software material, but sufficient for practical activities in the profession
E	3 , 00 -3, 24		SATISFACTORY – incomplete mastery of the program material that meets the minimum evaluation criteria
FX	2.00 -2.99	Not counted	UNSATISFACTORY – unsystematic knowledge acquired and the impossibility of continuing education without additional knowledge of the discipline
F	0.00-1.99		UNSATISFACTORY - serious further work and re-study of the discipline is required

8 QUESTIONS FOR SELF-CONTROL OF LEARNING RESULTS

1. Concept of program, software tool, software. Types of software by types of licenses .
2. Reasons and prerequisites for the emergence of the "Software Engineering" discipline. Definition of software engineering.
3. Software engineering as an engineering and scientific discipline .
4. Standardization in software engineering.
5. Main areas of SWEBOK knowledge .
6. Software life cycle.
7. General characteristics of the cascade model of the software life cycle; its advantages and disadvantages.
8. General characteristics of the spiral model of the software life cycle and its advantages and disadvantages.
9. General characteristics of the iterative model of the software life cycle and its advantages and disadvantages.
10. General characteristics of the incremental (step-by-step) software life cycle model and its advantages and disadvantages.
11. General characteristics of the rapid application development methodology RAD (Rapid Application Development) and its advantages and disadvantages.
12. RUP methodology (Rational Unified Process). General characteristics of the software development process.
13. MSF (Microsoft Solutions Framework) project team model .
14. MSF (Microsoft Solutions Framework) process model .
15. Loud software development methodologies. General characteristics.
16. Extreme Programming (XP).
17. Scrum methodology .
18. International and national standards of the software life cycle.
19. The essence and methods of structural analysis and modeling.
20. General characteristics of structural analysis and modeling methodologies.
21. Basic principles of structural analysis and modeling.
22. But that's me IDEF0 for the graphical representation of the SADT model.
23. Syntax and semantics of DFD diagrams.
24. Basic concepts of "entity-relationship" information modeling (ERD) .
25. Overview of notations used in the construction of ER diagrams.
26. Methodology for building ER-diagrams.
27. The essence of object-oriented analysis and modeling; its advantages and disadvantages.
28. Basic principles of object-oriented analysis and modeling.
29. General characteristics of the UML language. UML diagrams.
30. The general process of developing software requirements.
31. Properties of software requirements.
32. Sources and strategies for identifying software requirements.
33. Analysis, systematization and verification of software requirements.
34. Specification and formalization of software requirements.
35. Software specification.
36. The concept of software architecture. Main classes of architectures.
37. Basic principles of software architecture design.
38. The modular structure of the software product. The main characteristics of the modules.
39. Types of software modules and their structure.
40. Top-down and bottom-up software design.
41. Development of the software structure using an object-oriented approach.
42. Principles, rules and stages of software interface design.
43. Programming paradigms.

44. Concept of programming language. Development stages of programming languages.
45. Classification of programming languages.
46. Tools and integrated programming environments. Translators.
47. Basics of software design. Standards in construction. Design management.
48. Classification of software errors. Software debugging methods. General software debugging technique.
49. The main types of testing.
50. Levels of testing (module, integration and system testing).
51. Tests. Classification of tests. Validation tests. Measurement of test results.
52. Features of object-oriented testing.
53. Peculiarities of testing object-oriented modules.
54. Software verification and attestation.
55. General principles of drafting and design of software documentation.
56. The main types of software documentation.
57. Characteristics of the documentation intended for the user of the software .
58. Electronic reference system.
59. Software configuration management processes.
60. Stages and procedures in software configuration management.
61. Technological support for software configuration management.
62. The main stages of software product readiness. Alpha version. Beta version. Release
63. Organization and methods of software maintenance. Stages and procedures for software maintenance.
64. Escort techniques. Reengineering, reverse reengineering.
65. Code of ethics for software engineering professionals (IEEE - CS / ACM) .
66. Corporate ethics and culture.

9 METHODOLOGICAL SECURITY

The educational process in the discipline is fully and in sufficient quantity provided with the necessary educational and methodological developments in the Modular educational environment. Access to the resource: https :// msn . khmnu _ edu . ua / course / view . php ? id = 3403).

1 0 RECOMMENDED LITERATURE

Main

1. Бородкіна І., Бородкін Г. Інженерія програмного забезпечення : посіб. для студ. вищих навч. закладів. К : Вид-тво "Центр навчальної літератури", 2018. 204 с.
2. Левус Є., Мельник Н.. Вступ до інженерії програмного забезпечення. Л. : Львівська політехніка, 2018. 248 с.
3. Моделювання бізнес-процесів та управління ІТ-проектами : навч. посіб. / Крижановський Є. М., Ящолт А. Р., Жуков С. О., Козачко О. М. Вінниця: ВНТУ, 2018. 91 с.
4. Постіл С. Д. UML. Уніфікована мова моделювання інформаційних систем : навч. посіб. Ірпінь : УДФСУ, 2019. 322 с.
5. Rod Stephens. Beginning Software Engineering: Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. Published simultaneously in Canada and the United Kingdom. 2022. 685 P.
6. Elvis C. Foster. Software Engineering. A Methodical Approach: Second Edition. Published 2022 by CRC Press, Taylor & Francis Group. Boca Raton-London-New York. 579 P.

Additional

7. Електронні версії методичних вказівок до лабораторних робіт.

8. Ronald J. Leach. Introduction to Software Engineering: Second Edition. Howard University, Washington, DC, USA. CRC Press, Taylor & Francis Group. 2018. 420 P.

9. Автоматизоване проектування інформаційних систем. URL: https://posibniki.com.ua/catalog-avtomatizovane-proektuvannya-informaciynih-sistem---denisova-o-o

10. Левус Є. В., Марусенкова Т. А. Вступ до інженерії програмного забезпечення : 1024 завдання для підготовки до контрольних заходів. Львів : Львівська політехніка, 2021. 188 с.

11. Мартін Роберт. Чистий Agile: назад до основ; пер. з англ. В. Луненко. Харків : Вид-во «Ранок»: Фабула, 2021. 224 с.

12. Software Engineering Body of Knowledge (SWEBOK). URL: https://www.computer.org/education/bodies-of-knowledge/software-engineering

13. Introduction to the Microsoft Solutions Framework. URL: https://learn.microsoft.com/en-us/previous-versions/tn-archive/bb497060(v=technet.10)?redirectedfrom=MSDN

14. Manifesto for Agile Software Development. URL: http://agilemanifesto.org/

15. The Software Engineering Code of Ethics and Professional Practice. URL: https://ethics.acm.org/code-of-ethics/software-engineering-code/

16. ACM Code of Ethics and Professional Conduct. URL: https://ethics.acm.org/code-of-ethics/

17. ДСТУ ISO/IEC/IEEE 24765:2018 (ISO/IEC/IEEE 24765:2017, IDT). Інженерія систем і програмних засобів. Словник термінів.

18. ДСТУ ISO/IEC/IEEE 12207:2018. Інженерія систем і програмних засобів. Процеси життєвого циклу програмних засобів (ISO/IEC/IEEE 12207:2017, IDT).

19. ДСТУ ISO/IEC TR 24774:2016 (ISO/IEC TR 24774:2010, IDT). Інженерія систем і програмних засобів. Керування життєвим циклом. Настанови щодо опису процесу.

20. ДСТУ ISO/IEC TS 24748-1:2018 (ISO/IEC TS 24748-1:2016, IDT). Інженерія систем і програмних засобів. Керування життєвим циклом. Частина 1. Настанови щодо керування життєвим циклом.

21. ДСТУ ISO/IEC TR 24748-3:2016 (ISO/IEC TR 24748-3:2011, IDT). Інженерія систем і програмного забезпечення. Керування життєвим циклом. Частина 3. Настанова щодо застосування ISO/IEC 12207 (Процеси життєвого циклу програмного забезпечення).

22. ISO/IEC/IEEE 29148-2018. Systems and software engineering – Life cycle processes – Requirements engineering.

23. ДСТУ ISO/IEC TR 24766:2016 (ISO/IEC 24766:2009, IDT). Інформаційні технології. Інженерія систем і програмних засобів. Настанови щодо розроблення технічних вимог до програмних засобів.

24. ДСТУ ISO/IEC/IEEE 42010:2018 (ISO/IEC/IEEE 42010:2011, IDT). Інженерія систем і програмних засобів. Опис архітектури.

25. ДСТУ ISO/IEC/IEEE 15289:2019 (ISO/IEC/IEEE 15289:2017, IDT). Інженерія систем і програмних засобів. Уміст інформаційних об'єктів життєвого циклу (документації).

26. ДСТУ ISO/IEC/IEEE 26515:2018 (ISO/IEC/IEEE 26515:2011, IDT). Інженерія систем і програмних засобів. Розроблення документації користувача в гнучкому середовищі.

11 . INFORMATION RESOURCES

1. MOODLE Learning Platform. Access to the resource https://msn.khmnu.edu.ua.
2. University Electronic Library. Access to the resource: http://library.khmnu.edu.ua.
3. University Repository. Access to the resource https://elar.khmnu.edu.ua/home.