КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

Кафедра інформаційних систем

«ЗАТВЕРДЖУЮ»

Заступник декана
_____ Кашпур О.Ф.

«____»_____2018 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

Програмна інженерія

для студентів

галузь знань	12 – «Інформаційні технології»
спеціальність	121 – «Інженерія програмного забезпечення»
освітній рівень	бакалавр
освітня програма	«Програмна інженерія»
вид дисципліни	вільного вибору (блоками)

Форма навчання	**денна**
Навчальний рік	**2018/2019**
Семестр	**7**
Кількість кредитів ECTS	**4**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**екзамен**

Викладач: **к.ф.-м.н., доцент Ченцов О.І.** (лекції, лабораторні заняття)
к.ф.-м.н., асистент Терлецький Д.О. (лабораторні заняття)

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

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

КИЇВ – 2018

Розробник: Ченцов Олексій Ілліч, к.ф.-м.н., доцент кафедри інформаційних систем.

ЗАТВЕРДЖЕНО
В.о. зав. кафедри інформаційних систем
_____ (Іванов Є.О.)

Протокол № 8 від «22» травня 2018 р.

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики
Протокол № ___ від «18» червня 2018 року
Голова науково-методичної комісії _____ (Хусаінов Д.Я.)

«_____» _____ 20___ року

1. Мета дисципліни – розвинути у студента розуміння концепцій, принципів та методів розробки якісних програмних систем різного ступеня складності, на практичному прикладі підготувати студента до виконання програмних проектів на виробництві.

2. Попередні вимоги до опанування або вибору навчальної дисципліни:

1. Мати уявлення: про розробку програм

2. Знати: основи ООП, деякі розділи дискретної математики.

3. Вміти: програмувати на мові C++ та/або Java, послуговуватися різними стилями програмування.

4. Володіти: сучасними технологіями, зокрема web-програмуванням, та методами розробки інформаційних систем, навичками з розробки невеликих програмних систем.

3. Анотація навчальної дисципліни:

Навчальна дисципліна «Програмна інженерія» є складовою освітньо-професійної програми підготовки фахівців за освітньо-кваліфікаційним рівнем «бакалавр» галузі знань 12 – «Інформаційні технології» зі спеціальності 121 «Інженерія програмного забезпечення», за освітньою програмою «Програмна інженерія».

Дана дисципліна є дисципліною вільного вибору студента блоками за ***програмою*** «Програмна інженерія». Викладається у 7 семестрі 4 курсу бакалаврату в **обсязі – 120 год.** ***(4 кредити ECTS)*** зокрема: *лекції – 26 год., лабораторні – 28 год., самостійна робота – 64 год., консультації – 2 год.* У курсі передбачено 2 *змістових модулі* та 1 *модульна контрольні роботи*. Завершується дисципліна – **екзаменом.**

В результаті вивчення навчальної дисципліни студент повинен

знати: принципи розробки програмного забезпечення, моделі та складові процесу розробки, підходи до моделювання програмних систем, способи оцінки та підвищення якості програмного забезпечення.

вміти: виконувати програмні проекти різного рівня складності у складі команди, організовувати та управляти проектами середнього розміру, оформлювати технічну супроводжуючу документацію, підсумкові та проміжні звіти про виконання проекту

4. Завдання (навчальні цілі):

набуття знань, умінь та навичок (компетентностей) на рівні новітніх досягнень у програмуванні, відповідно до кваліфікації фахівець з інформаційних технологій. Зокрема, розвивати:

- здатність застосовувати знання у практичних ситуаціях;
- здатність оцінювати та забезпечувати якість виконуваних робіт;
- здатність ефективно використовувати та модифікувати сучасне системне програмне забезпечення;
- здатність до алгоритмічного мислення.

5. Результати навчання за дисципліною:

Результат навчання (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		**Форми (та/або методи і технології) викладання і навчання**	**Методи оцінювання та пороговий критерій оцінювання (за необхідності)**	**Відсоток у підсумковій оцінці з дисципліни**
Код	**Результат навчання**			
РН1.1	*Знати принципи розробки програмного забезпечення, моделі та складові процесу розробки, підходи до моделювання програмних систем, способи оцінки та підвищення якості програмного забезпечення.*	*Лекція*	*Контрольна робота, 60% балів, екзамен*	40%

РН2.1	*Вміти виконувати програмні проекти різного рівня складності у складі команди, організовувати та управляти проектами середнього розміру, оформлювати технічну супроводжуючу документацію, підсумкові та проміжні звіти про виконання проекту*	*Лабораторне заняття, самостійна робота*	*Захист лабораторної роботи, екзамен*	40%
РН3.1	*Обґрунтовувати власний погляд на задачу, спілкуватися з колегами з питань проектування та розробки програм*	*Лабораторне заняття*	*Захист лабораторної роботи*	10%
РН4.1	*Організовувати свою самостійну роботу для досягнення результату*	*Самостійна робота*	*Захист лабораторної роботи*	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни **Програмні результати навчання**	**РН 1.1**	**РН 2.1**	**РН 3.1**	**РН 4.1**
(з опису освітньої програми)				
ПР-3. Знати основні процеси, фази та ітерації життєвого циклу програмного забезпечення	+			
ПР-16. Мати навички командної розробки, погодження, оформлення і випуску всіх видів програмної документації.		+	+	+
ПР-22. Знати та вміти застосовувати методи та засоби управління проектами.		+		+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

1. Контрольна робота: РН 1.1 — 20 балів/12 балів.

2. Лабораторні роботи 1-6: РН2.1, РН 3.1, РН4.1 – 50 балів/30 балів.

- підсумкове оцінювання

максимальна кількість балів які можуть бути отримані студентом: 30 балів;
результати навчання які будуть оцінюватись: РН1.1, РН2.1;
форма проведення і види завдань: письмова.
Види завдань: 3 письмових завдання.

7.2 Організація оцінювання:

Терміни проведення форм оцінювання:

1. Контрольна робота: до 14 тижня семестру.
2. Лабораторна робота 1: до 3 тижня семестру.
3. Лабораторна робота 2: до 6 тижня семестру.
4. Лабораторна робота 3: до 6 тижня семестру.
5. Лабораторна робота 4: до 11 тижня семестру.
6. Лабораторна робота 5: до 14 тижня семестру.
7. Лабораторна робота 6: до 14 тижня семестру.

Студент має право на одне перескладання кожної контрольної роботи із можливістю отримання максимально 80% початково визначених за цю контрольну роботу балів. Термін перескладання визначається викладачем.

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

У разі неякісного виконання лабораторної роботи, викладач має право не зарахувати лабораторну роботу, або знизити за неї бали.

Студент має право здавати лабораторні роботи після закінчення визначеного для них терміну, але з втратою одного балу за кожен тиждень, який пройшов з моменту закінчення терміну її здачі.

7.3 Шкала відповідності оцінок

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

8. Структура навчальної дисципліни. Тематичний план лекцій і лабораторних занять

№ лекції	Назва теми/лекції	Кількість годин		
		Лекції	Лабораторні роботи	Самост. робота
Частина 1. ***Основні процеси розробки програмного забезпечення та управління ними***				
1	**Тема 1.** *Вступ до курсу*.	2	2	6
2	**Тема 2.** *Процес розробки програмного забезпечення*.	2	2	4
3	**Тема 3.** *Управління програмними проектами*.	2	2	4
4	**Тема 4**. *Гнучка методологія розробки*.	2	2	4
5	**Тема 5**. *Інженерія вимог до програмного забезпечення*. **Лекція 5**. *Вимоги до програмного забезпечення*.	2	2	4
6	**Тема 5**. *Інженерія вимог до програмного забезпечення*. **Лекція 6.** *Інженерія вимог до ПЗ*.	2	2	4
7	**Тема 6**. *Моделювання систем*.	2	2	4
8	**Тема 7**. *Архітектурне проектування*.	2	2	6
9	**Тема 8**. *Проектування та реалізація*.	2		4
Модульна контрольна робота			2	
Частина 2. ***Тестування та еволюція програмного забезпечення***				
10	**Тема 8**. *Тестування програмного забезпечення*. **Лекція 10.** *Процес тестування програмного забезпечення*.	2	2	4
11	**Тема 8**. *Тестування програмного забезпечення*. **Лекція 11.** *Методи тестування програмних компонентів*.	2	2	4
12	**Тема 8**. *Тестування програмного забезпечення*. **Лекція 12.** *Методи тестування програмних систем*.	2	2	4
13	**Тема 9**. *Еволюція програмного забезпечення у ході його експлуатації*.	2	2	8
14	**Тема 10**. *Реверсна інженерія програмного забезпечення*		2	4
Екзамен				
ВСЬОГО		26	28	64

Загальний обсяг ***120*** *год.,* в тому числі:
Лекцій **– *26*** *год.*
Лабораторні роботи – ***28*** *год.*
Самостійна робота – ***64*** *год.*
Консультації – ***2*** *год.*

Теми, винесені на самостійне вивчення:
Кодекс етики програміста
Програмний інструментарій підтримки виконання проектів
Рекомендації щодо оформлення специфікації вимог до програмного забезпечення

Виявлення вимог на основі етнографічного дослідження
Специфікація вимог в екстремальному програмуванні
Архітектурне проектування багаторівневих застосунків. Проектування рівня доступу до даних. Проектування бізнес-рівня та рівня представлення.
Реверсна інженерія програмного забезпечення

Умови лабораторних робіт:
Лабораторна робота 1: Вибір програмного проекту, формування команди, підготовка статуту.
Лабораторна робота 2: План проекту.
Лабораторна робота 3: Специфікація вимог до програмної системи.
Лабораторна робота 4: Звіт з проектування програмної системи.
Лабораторна робота 5: Звіт з тестування програмної системи.
Лабораторна робота 6: Індивідуальний звіт з проекту.
Деталізовані умови лабораторних робіт розміщено за посиланнями:
https://docs.google.com/document/d/1ak-sI-QetalHsbkWzl3Yoba3AvEGkc5W7aBLEJZy_54

9. Рекомендовані джерела:

Основні

1. *Sommerville I.* Software Engineering, 10th ed. — Addison-Wesley / Pearson Education Limited, 2015. — 816 p.
2. *Соммервилл И.* Инженерия программного обеспечения, 6 изд. — М.: "Вильямс", 2002. — 624 с.
3. *Pressman R.* Software Engineering: A Practitioner's Approach, 7th ed. — McGraw-Hill, 2010. — 928p.

Додаткові

4. *McConnell S.*, Code Complete: A Practical Handbook of Software Construction, Second Edition. — Microsoft Press, 2004. — 960 p.
5. *Hunt A., Thomas D.* The Pragmatic Programmer: From Journeyman to Master. — Addison-Wesley Longman Publishing, 1999. — 320p.
6. *Ghezzi C., Jazayeri M., Mandriol D.* Fundamentals of Software Engineering, 2nd ed. — Prentice Hall, 2003. — 604p.
7. *Лавріщева К.М.* Програмна інженерія. — Київ, 2008. — 319с.
8. *Липаев В.* Программная инженерия. Методологические основы. — М.: Теис, 2006. — 608с.
9. *Brooks F.* The Mythical Man-Month, Anniversary Edition (2nd ed.) — Addison-Wesley Professional, 1995. — 336p.
10. Guide to Software engineering body of knowledge – 2004 Version / Exec. eds: Alain Abran, James W. Moore. — IEEE Computer Society, 2004. — 200p. — http://www.computer.org/portal /web/swebok/htmlformat
11. Software engineering: Report of a conference sponsored by the NATO Science Committee / Eds.: Peter Naur, Brian Randell. — Scientific Affairs Division, NATO, 1969. — 231p.
12. *Wirth Niklaus*. A Brief History of Software Engineering // IEEE Annals of the History of Computing. — v. 30, No. 3, 2008. — pp. 32–39.
13. *Brooks F*. No silver bullet // IEEE Computer. — vol. 20, no. 4, April 1987. — pp. 10–19.
14. ISO/IEC 12207:2008, Systems and software engineering – Software life cycle processes. — ISO, 2008. — 123p.
15. Software Engineering Code of Ethics and Professional Practice (Version 5.2). — http://www.acm.org/about/se-code
16. *Boehm B.* A Spiral Model of Software Development and Enhancement. — ACM SIGSOFT Software Engineering Notes", "ACM". — v. 11(4), August 1986. — pp. 14–24.
17. 830-1998, IEEE Recommended Practice for Software Requirements Specifications. — IEEE, 1998. — 31p.
18. 1233-1998, IEEE Guide for Developing System Requirements Specifications. — IEEE, 1998.

19. *Viller S., Sommerville I.* Ethnographically informed analysis for software engineers. // Int. J. Human-Computer Studies. — 53 (1), 2000. — pp.169–196.
20. *Mellor S., Balcer M.* Executable UML: A Foundation for Model-Driven Architecture. — Addison-Wesley, 2002. — 416p.
21. Object Management Group: Semantics of a foundational subset for executable UML models (fUML), v1.0. — OMG, 2011. — 404p. — http://www.omg.org/spec/FUML/.
22. Object Management Group: Concrete Syntax For UML Action Language (Action Language For Foundational UML – ALF). — OMG, 2010. — 425p. — http://www.omg.org/spec/ALF/.
23. *Schwaber K., Sutherland J.* — The Scrum Guide: The Definitive Guide to Scrum – The Rules of the game. — 2011. — 17p. — http://www.scrum.org/Scrum-Guides.
24. *Kruchten P.* The 4+1 View Model of Architecture // IEEE Software. — vol. 12, no. 6, November 1995. — pp. 42–50.
25. Microsoft Application Architecture Guide, 2nd ed. / Microsoft Patterns & Practices Team. — Microsoft Press, 2009. — 496p. — http://msdn.microsoft.com/en-us/library/dd673617.aspx
26. Руководство Microsoft по проектированию архитектуры приложений, 2-е изд. / Microsoft Patterns & Practices Team. — Microsoft Press, 2010. — 529p. — http://apparchguide.ms/
27. *Mills H., Dyer M., Linger R.* Cleanroom Software Engineering // IEEE Software. — v. 4, no. 5, September 1987. — pp.19–25.
28. *Chikofsky E., Cross J.* Reverse engineering and design recovery: a taxonomy // IEEE Software. — vol. 7, no. 1, January 1990. — pp.13–17.
29. Rugaber S. White paper on reverse engineering. — 1994. — 11p. — http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.199.9825
30. *Leffingwell D.* Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise. — Addison-Wesley, 2011. — 518p.