Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

АПЕПС

Кафедра автоматизації проектування енергетичних процесів і систем

ТЕХНОЛОГІЇ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	Перший (бакалаврський)
Галузь знань	12 Інформаційні технології
Спеціальність	122 Комп'ютерні науки
Освітня програма	Комп'ютерний моніторинг та геометричне моделювання процесів та систем
Статус дисципліни	Нормативна
Форма навчання	очна(денна)
Рік підготовки, семестр	3 курс весняний семестр
Обсяг дисципліни	135 год
Семестровий контроль/ контрольні заходи	Екзамен
Розклад занять	Науково-педагогічний працівник
Мова викладання	Українська
Інформація про керівника курсу / викладачів	Лектор: к.т.н., доцент каф. АПЕПС, Тихоход Володимир Олександрович Комп'ютерні практикуми: к.т.н., доцент каф. АПЕПС, Тихоход Володимир Олександрович Беспала Ольга Миколаївна
Розміщення курсу	Кампус

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

В цій дисципліні детально вивчається архітектура системного програмного забезпечення операційно системи Linux. На сьогоднішній день затребувані фахівці, які впроваджують новітні технології, в основі яких лежить саме Linux. Також розглядається і архітектура операційної системи Window 10, ця система є найбільш вживаною серед користувачів. Отримані знання дозволяють виконувати роботи з адміністрування операційної системи, застосування технології контейнерів для розгортання, поширення та функціонування програмного забезпечення, створення мікросервісів, а також як результат будуть сприяти кар'єрному зростанню, самореалізації, застосуванню отриманих знань для вирішення практичних завдань.

Метою дисципліни є опанування студентами теоретичних знань архітектури системного програмного забезпечення, побудови, функціонування, використання засобів операційних систем, технології контейнерів для реалізації упаковки, розгортання та функціонування програмного забезпечення.

Предмет дисципліни - вивчення принципів побудови, архітектури, основних функцій, режимів роботи, засобів операційних систем (ОС), розроблення мікросервісів на основі технології Docker.

Завдання. В результаті вивчення дисципліни у студентів повинні сформуватися наступні компетентності:

інструментальні:

- здатність вирішувати проблеми в професійній діяльності на основі аналізу й синтезу (ІК-2).

професійні:

- здатність до проектування інформаційних систем (ІС) та інформаційних технологій (ІТ), включаючи формальний опис їх структури та проведення моделювання бізнес-процесів (ПК-2)
- здатність до проектування архітектури комп'ютерної системи, вибору і інтегруванню компонентів технічного і стандартного програмного забезпечення при реалізації ІС та ІТ (ПК-3)
- здатність використовувати сучасні комп'ютерні технології для системного, функціонального, конструкторського та технологічного проектування складних об'єктів і систем (ПК-7)
- здатність розробляти методичну, нормативну та технічну документацію, проводити заходи щодо реалізації розроблених проектів і програм (ПК-10)
- здатність застосовувати сучасні парадигми програмування під час програмної реалізації професійних задач (ПК-11)
- здатність розв'язувати проблеми масштабованості (ПК-12)
- здатність організовувати в підрозділі роботи із стандартизації та сертифікації ПЗ та інформаційних технологій (ПК-16)
- здатність використовувати знання стандартів, методів і засобів управління процесами життєвого циклу інформаційних систем, продуктів і сервісів інформаційних технологій (ПК-21)

Після засвоєння навчальної дисципліни студенти мають продемонструвати такі знання та вміння:

***знання*:**

- інформаційних технологій, мов програмування, інструментарію програміста;
- адміністративних, правових та економічних основ професійної діяльності;
- сучасних технологій та інструментальних засобів розробки програмних систем;
- CASE-технологій проектування інформаційних та програмних систем;
- мов програмування, сучасних теорій організації баз даних та знань, методів і технологій їх розробки.
- методів та стандартів оформлення документації;
- міжнародних стандартів з оцінки якості програмного забезпечення (ISO 9126:2001, тощо), правил та методів забезпечення якості ІТ-систем;

вміння:

- застосовувати мови програмування, мови опису інформаційних технологій, мови специфікацій;
- застосовувати інструментальні засоби при проектуванні та створенні інформаційних систем, продуктів і сервісів інформаційних технологій;
- володіти методами і засобами підтримки командної роботи.
- використовувати державні та міжнародні стандарти в галузі інформаційних технологій;
- застосовувати сучасні технології та інструментальні засоби розробки на всіх етапах життєвого циклу ІС;
- застосовувати CASE засоби;
- моделювати системи та процеси, стани та поведінки складних об'єктів інформатизації в процесі проектування інформаційних систем і технологій;
- володіти сучасними технологіями автоматизації проектування складних об'єктів і систем, продуктів і сервісів інформаційних технологій, сучасними парадигмами та мовами програмування;
- складати технічну документацію;
- організовувати в підрозділі роботи із стандартизації та сертифікації ПЗ та інформаційних технологій;
- застосовувати стандарти, методи та засоби управління процесами життєвого циклу розробки програмних систем;
- застосовувати у роботі міжнародні стандарти з оцінки якості програмного забезпечення.

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

Пререквізити дисципліни. Вивчення дисципліни спирається на знання, отримані в дисциплінах з попередніх семестрів, зокрема: “Основи програмування та алгоритмічні мови”, «Об’єктно-орієнтоване програмування», «Системи баз даних», «Проектування та використання баз даних».

Постреквізити дисципліни. Матеріал дисципліни може бути використаний при вивченні дисциплін «Технології комп’ютерного проектування», «Програмування комп'ютерних мереж», «Геометричне моделювання та комп'ютерна графіка», “Технології розподілених систем та паралельних обчислень”, «Теорія прийняття рішень», «Управління ІТ-проектами» та інших, що подаються в наступних семестрах.

3. Зміст навчальної дисципліни

1. Життєвий цикл і процеси розробки програмного забезпечення
2. Аналіз вимог замовника до ПЗ
3. Проектування ПЗ.
4. Конфігурація ПЗ.
5. Архітектура ПЗ, стандарти опису архітектур ПЗ.
6. Патерни проектування ПЗ.

4. Навчальні матеріали та ресурси

Основна література

1. Фаулер М. UML. Основы // Пер. с англ.. — 3-издание. – СПб: СимволлПлюс, 2004. – 192 с.
2. Буч Г. Язык UML. Руководство пользователя / Буч Г., Рамбо Д., Якобсон И.//Пер. с англ. Мухин Н. — 2-е изд. – М.: ДМК Пресс. – 496 с.
3. Фаулер Мартин. Шаблоны корпоративных приложений.: Пер. с англ. — М.: ООО «И.Д. Вильямс», 2016. — 544 с.
4. Лавріщева К.М. Програмна інженерія. — К. — 2008. — 319 с.
5. Гамма Э. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Гамма Э., Хелм Р., Джонсон Р., Влиссидес Д. — СПб.: Питер, 2017. — 368 с.

Додаткова література

6. Коберн Алистер. Современные методы описания функциональных требований к системам. — М.: Издательство "Лори", 2002. — 288 с.
7. Вигерс К. Разработка требований к программному обеспечению/ Карл Вигерс. Пер, с англ. — 3-е издание.— М.: Издательство «Русская Редакция»; СПб.: БХВ-Петербург, 2004. —736 с.
8. Паттон Д. Пользовательские истории. Искусство гибкой разработки ПО/ // . — Питер-Пресс, 2017. — 288 с.
9. Кон М. Пользовательские истории: гибкая разработка программного обеспечения / Майкл Кон. — М.: Диалектика-Вильямс, 2020. — 256 с.
10. Орлов С.А. Программная инженерия / С.А. Орлов // Учебник для вузов. — 5-е издание — СПб.: Питер, 2016. — 640 с.
11. Чакон С. Git для профессионального программиста / Чакон С., Штрауб Б. — СПб.: Питер, 2016. — 496 с.
12. Pro Git. Режим доступу: https://git-scm.com/book/uk/v2 .
13. Керівництва GitHub. Режим доступу: https://guides.github.com/
14. Книберг Х. Kanban и Scrum: выжимаем максимум / Хенрик Книберг, Маттиас Скарин. — C4Media, Издательство InfoQ.com, 2010. — 78 с.
15. Книберг Х. Scrum и XP: заметки с передовой. Как мы делаем Scrum. — C4Media, Издательство InfoQ.com, 2010. — 94 с.
16. Рубин К. Основы Scrum. Практическое руководство по гибкой разработке ПО / Рубин Кеннет С. — М.: Вильямс, 2016. — 544 с.

17. Кон М. Scrum: гибкая разработка ПО / Майкл Кон. — М. Вильямс, 2016. — 576 с.
18. Макконнелл С. Совершенный код. Мастер-класс / Пер. с англ. — М. : Издательско- торговый дом «Русская Редакция» ; СПб.: Питер, 2005. — 896 стр.
19. Босуєлл Д. Читаемый код, или Программирование как искусство / Босуэлл Д., Фаучер Т. — СПб.: Питер, 2012. — 208 с.
20. Фаулер М. Рефакторинг: улучшение существующего кода. – Пер. с англ. – СПб: Символл Плюс, 2003. – 432 с.
21. Мартін Р. Чистий код: створення і рефакторинг за допомогою Agile / пер. з англ. І. Бондар-Терещенко. — Харків : Вид-во «Ранок» : Фабула, 2019. — 448 с.
22. Симан М. Внедрение зависимостей в .NET. — СПб.: Питер, 2014. — 464 с.
23. Эспозито Д. Microsoft .NET: архитектура корпоративных приложений / Дино Эспозито, Андреа Сальтарелло. — 2-е издание. — М.: ООО «И.Д. Вильямс», 2016. — 432 с.
24. Арлоу Д.. UML 2 и Унифицированный процесс. Практический объектно-ориентированный анализ и проектирование/ Арлоу Д., Нейштадт И. // Пер. с англ. — 2-е издание. — СПб: Символ-Плюс, 2007. – 624 с.
25. Брауде Э. Технология разработки программного обеспечения. — СПб.: Питер, 2004. — 655 с.
26. Руководство Microsoft по проектированию архитектуры приложений. — 2-е издание. — Microsoft, 2009. — 529 с.
27. .NET Microservices: Architecture for Containerized .NET Applications/ —Edition v2.1.02. — Microsoft.
28. Мартін Роберт. Чиста архітектура: Мистецтво розроблення програмного забезпечення / пер. з англ. І. Бондар-Терещенко. — Харків : Вид-во «Ранок» : Фабула, 2019. — 368 с.
29. Месарош Дж. Шаблоны тестирования xUnit: рефакторинг кода тестов / Джерард Месарош : Пер. с англ. — М. : ООО ‘‘И.Д. Вильямс’’, 2009. — 832 с.
30. Ошероув Р. Искусство автономного тестирования с примерами на C# / Пер. с англ. — 2-е издание. – М.: ДМК Пресс, 2014. – 360 с.
31. Кент Бек. Экстремальное программирование. Разработка через тестирование. — СПб.: Питер, 2017. — 224 с.
32. Эберхард Вольф. Continuous delivery. Практика непрерывных апдейтов. — СПб.: Питер, 2018. — 320 с.
33. Хамбл Д. Непрерывное развертывание ПО: автоматизация процессов сборки, тестирования и внедрения новых версий программ/ Хамбл Джез, Фарли Дейвид // Пер. с англ. — М. : ООО “И.Д. Вильямс”, 2011. — 432 с.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

Тема 1. Життєвий цикл і процеси розробки програмного забезпечення

Лекція 1. Моделі життєвого циклу ПЗ.

Технології розроблення програмного забезпечення. Програмний продукт. Життєвий цикл програмного забезпечення. Моделі життєвого циклу. Стратегії розробки ПЗ. Водоспадна модель (Waterfall model) ЖЦ. Інкрементна (incremental) модель. Макетування. Спіральна модель. Компонентно-орієнтована модель. RAD-модель. V- модель (V-Model)

Лекція 2. Гнучкі методології. Методологія Scrum.

Важкі та полегшені процеси. Agile-маніфест. Методологія Kanban. Методологія XP (Extreme Programming). Методологія Scrum. Порівняння Agile-методологій

Тема 2. Аналіз вимог замовника до ПЗ

Лекція 3. Інженерія вимог до ПЗ. Методики формування вимог.

Вимоги до програмного забезпечення. Категорії вимог. Етапи інженерії вимог. Типи вимог. Формування вимог. Діаграма прецедентів. Аналіз вимог (класичний підхід). Характеристики

якісних вимог. Опис вимог у вигляді текстових сценаріїв. Формування та аналіз вимог в процесі Scrum. Історія користувача (user story). Критерії якісної історії користувача. Ієрархія історій. Критерії якісної задачі. Карта історій.

Лекція 4. Аналіз вимог до ПЗ. Опис детальних вимог за допомогою діаграм діяльності та станів UML.
Мова моделювання UML. Використання UML для аналізу вимог. Режими використання. Режим ескізу. Режим проектування. Режим мови програмування. Типи діаграм UML 2.0. Діаграма діяльності. Діаграма станів.

Тема 3. Проектування ПЗ.

Лекція 5. Проектування програмного забезпечення. Діаграма класів та об'єктів UML.
Проектування програмного забезпечення. Використання нотації UML для проектування ПЗ. Діаграма класів: клас, ім'я, видимість полів та атрибутів, атрибути, операції, відношення між класами, генерування коду. Діаграма об'єктів.

Лекція 6. Проектування ПЗ з допомогою структурних та поведінкових діаграм UML.
Діаграма послідовності (Sequence diagram). Діаграма пакетів (Package Diagram). Діаграма компонентів (Component Diagram). Діаграма розгортання (Deployment Diagram).

Лекція 7. Якість ПЗ.
Якість ПЗ. Види якості. Модель якості програмних систем. Модель Мак-Кола. Модель якості ISO/IEC 9126. Стандартні показники якості. Метрики якості.

Лекція 8. Проектування UI.
Місце прототипу в життєвому циклі ПЗ. Стандарти проектування людино-машинного інтерфейсу. Засоби створення прототипів.

Лекція 9. Об’єктні моделі та реляційні бази даних.
Архітектурні рішення. Функціональні проблеми. Зчитування даних. Взаємне відображення об’єктів та реляційних структур. Реалізація відображення. Наслідування з одною таблицею (Single Table Inheritance). Наслідування з таблицями для кожного класу (Class Table Inheritance). Наслідування з таблицями для кожного конкретного класу (Concrette Table Inheritance).

Лекція 10. Чистий код.
Поняття чистого коду. Стандарти створення чистого коду: змістовні імена, стандарти написання функцій, коментарів, класів. Оброблення помилок.

Лекція 11. Модульне тестування.
Призначення. Переваги. Характеристики якісного теста. Тестові двійники: об'єкт-заглушка (Dummy Object), тестовая заглушка (Test Stub), тестовий агент (Test Spy), імітація (Fake), підставний об'єкт (Mock). TDD. Фреймворки тестування. Підтримка модульного тестування на рівні мови. Рекомендації.

Тема 4. Керування конфігурацією

Лекція 12. Системи контролю версій. Основи Git.
Про контроль версій. Типи СКВ. Основні поняття. Типовий порядок роботи з СКВ. Особливості GIT: принципи збереження даних, стани файлу в робочому каталозі, процеси роботи з гілками, робочий процес одного розробника, робочий процес взаємодії декількох розробників, хостинг репозиторіїв, GitHub.

Лекція 13. Безперервна інтеграція та розгортання.

Концепція безперервної інтеграції. Автоматизація складання. Гаки (hooks) в системах контролю версій. Інструменти складання. Засоби безперервної інтеграції.Концепція безперервного розгортання. Засоби безперервного розгортання.

Тема 5. Архітектура ПЗ, стандарти опису архітектур ПЗ

Лекція 14. Принципи проектування. Шаблони DI.
Зв’язність та зв’язаність. Інверсія залежності (Dependency Inversion). Шаблони DI: впровадження в конструктор, впровадження у властивість, впровадження в метод, навколишній контекст (Ambient Context). Антишаблони Dependency Injection: Диктатор (Control Freak), «Гібридне впровадження» (Bastard Injection), Обмежене конструювання» (Constrained Construction), Локатор сервісів (Service Locator). Принципи SOLID: принцип єдиної відповідальності (SRP), принцип відкритості / закритості (Open / Closed Principle), принцип підстановки Лісков (Liskov Substitution Principle), принцип поділу інтерфейсів (Interface Segregation Principle), принцип інверсії залежностей (Dependency Inversion Principle, DIP). Принцип KISS. Принцип YAGNI. Принцип DRY. Принцип «Говори, а не питай» (Tell-Don't-Ask).

Лекція 15. Архітектурні стилі.
Огляд. Поєднання архітектурних стилів. Архітектура клієнт/сервер. Компонентна архітектура. Проектування на основі предметної області. Багатошарова архітектура (N-layer architecture). Лукова архітектура (Onion Architecture). Гексагональна архітектура (Hexagonal Architecture). Архітектура, основана на шині повідомлень. N-рівнева / 3-рівнева архітектура (N-tier architecture). Об’єктно-орієнтована архітектура. Сервісно-орієнтована архітектура (Service-Oriented Architecture). Мікро-сервісна архітектура.

Лекція 16. Багатошарова архітектура.
Концепція розшарування. Архітектура базових типів застосунків. Мобільні застосунки. Насичені клієнтські застосунки. Насичені Інтернет-застосунки. Служба. Веб-застосунок. Особливості типових шарів. Рівень представлення. Шаблони реалізації бізнес логіки. Шаблони реалізації шару доступу до даних. Шар служб. Наскрізна функціональність. Чиста архітектура.

Тема 6. Патерни проектування ПЗ

Лекція 17. Породжуючі шаблони проектування.
Factory Method. Абстрактна фабрика (Abstract Factory). Одинак (Singleton). Прототип (Prototype). Будівельник (Builder).

Лекція 18. Структурні шаблони проектування.
Декоратор (Decorator). Адаптер (Adapter). Фасад (Facade). Компоновщик (Composite). Заступник (Proxy). Міст (Bridge). Пристосуванець (Flyweight).

Лекція 19. Поведінкові шаблони проектування.
Стратегія (Strategy). Спостерігач (Observer). Команда (Command). Шаблонний метод (Template Method). Ітератор (Iterator).

Лекція 20. Поведінкові шаблони проектування. Частина 2.
Стан (State). Ланцюжок Обов'язків (Chain of responsibility). Інтерпретатор (Interpreter). Посередник (Mediator). Хранитель (Memento). Відвідувач (Visitor).

6. Самостійна робота студента

Тема 1 Життєвий цикл і процеси розробки програмного забезпечення
Порівняння Agile-методологій.

Тема 2 Аналіз вимог замовника до ПЗ
Рівні деталізації сценаріїв за Алістером Коберном.

Тема 3 Проектування ПЗ
Діаграма потоків даних (data flow diagrams).

Тема 4 Керування конфігурацію
Ознайомлення з основними командами командного рядка Git.

Тема 5 Архітектура ПЗ, стандарти опису архітектур ПЗ
Архітектура DDD.

Тема 6 Патерни проектування ПЗ
Принципи застосування патернів проектування. Підходи до вибору патернів.

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Відвідування лекційних та лабораторних занять є обов'язковим за винятком поважних причин (хвороби, форс-мажорних обставин).

В разі пропущення занять з поважних причин викладач надає можливість студенту виконати усі або деякі лабораторні завдання (винятком є виконання деяких завдань у зв'язку із закінченням навчального процесу).

В разі пропущення занять без поважних причин, а також через порушення граничного терміну виконання завдання (deadline) студент може отримати 80% від максимальної оцінки відповідного завдання.

Протягом семестру студенти:

- виконують та захищають комп'ютерні практикуми у відповідні терміни (на кожну лабораторну роботу відводиться один тиждень для здачі),
- пишуть модульну контрольну роботу,
- повинні позитивно закрити дві атестації (в кінці березня та в середині травня),
- по закінченні навчального процесу складають екзамен.

8. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Система рейтингових (вагових) балів та критерії оцінювання

1) Лабораторні практикуми

Кожний практикум оцінюється максимально в 10 балів.

Для оцінки практикуму використовується наступні критерії оцінювання:

- виконаний своєчасно (протягом двох тижнів з моменту видачі), у повному обсязі – відповідний бал згідно номеру комп'ютерного практикуму;
- виконаний із запізненням – знімається 10 – 30% від максимальної кількості балів в залежності від терміну запізнення;
- виконаний не самостійно, із запізненням – знімається 50% від максимальної кількості балів;
- невиконаний протягом відведеного часу – 0 балів.

Максимальна кількість балів за усі виконані комп'ютерні практикуми дорівнює $S_{max} = 50$ балів. Для підрахунку кількості набраних студентом за практикуми балів протягом семестру використовується наступна формула:

$$r_{\text{практ}} = \frac{\sum_{i=1}^{13} K\phi_i * v_i}{\sum_{i=1}^{13} Kmax_i * v_i} * S_{max}$$

де $К\text{ф}_i$ — це отриманий (фактичний) бал за практикум під номером *i*, $К\text{м}_i$ — це максимальний бал за практикум *i*, v_i — ваговий коефіцієнт практикуму *i*.

Вагові коефіцієнти за комп'ютерні практикуми представлено в наступній таблиці.

№ з/п	Назва комп'ютерного практикуму	Ваговий коефіцієнт
1	Формування первинних вимог до ПЗ.	2
2	Аналіз вимог до ПЗ. Діаграма активностей та станів.	2
3	Проектування ПЗ. Діаграми класів та об'єктів.	2
4	Проектування ПЗ. Діаграми послідовностей та компонентів.	2
5	Заділ продукту.	1
6	Система контролю версій Git. GitHub.	1
7	Реалізація шару доступу до даних.	3
8	Реалізація шару бізнес-логіки.	3
9	Модульне тестування	3
10	Породжуючі шаблони проектування.	3
11	Структурні шаблони проектування.	3
12	Поведінкові шаблони проектування.	3
13	Поведінкові шаблони проектування 2	3

2) Модульна контрольна робота

Максимальна кількість балів за модульну контрольну роботу дорівнює 10 балів.

Якість виконання роботи:

- усі відповіді вірні та повні –10 балів,
- у відповідях допущені несуттєві неточності – 8 балів,
- половина відповідей вірна – 5 балів,
- відповіді з суттєвими неточностями, але без критичних помилок – 2 бали,
- менше половини відповідей вірна – 0 балів.

Штрафні та заохочувальні бали за:	
- активність на комп'ютерних практикумах	+ 2 бали
- виконання комп'ютерного практикуму з використанням власного оптимального алгоритму	+ 1 бали
- відсутність на занятті без поважної причини	– 2 бали
- несвоєчасна здача комп'ютерного практикуму (пізніше ніж за тиждень)	– 0,5 балів;

3) Складання іспиту

Максимальний ваговий бал $r_{\text{ісп}}$=40

На іспиті студент виконує письмову контрольну роботу, яка містить два теоретичних питання і одне практичне питання. Теоретичні питання оцінюються максимально по 10 балів, практичне – 20 балів.

Умови позитивної проміжної атестації

Для отримання „зараховано" з першої проміжної атестації студент повинен захистити 4 практикуми.

Для отримання „зараховано" з другої проміжної атестації студент повинен набрати не менше ніж 22 балів.

Умови допуску до *іспиту*

Необхідною умовою допуску до іспиту є зарахування усіх комп'ютерних практикумів та виконання модульної контрольної роботи, а також стартовий рейтинг (R_c) не менше 40 балів.

Розрахунок шкали (R) рейтингу:

Сума вагових балів контрольних заходів протягом семестру (шкала рейтингу) складає:

$$R = r_{практ} + r_{мод} + r_{icп} = 50 + 10 + 40 = 100 \text{ балів.}$$

Максимальний стартовий рейтинг становить $R_c = r_{практ} + r_{мод} = 60$ балів.

Рейтинг іспиту дорівнює 40 балів. Мінімальний рейтинг допуску до іспиту становить 40 балів.

Таким чином, рейтингова шкала з кредитного модуля складає

$$R = 60+40=100 \text{ балів.}$$

Для отримання студентом відповідних оцінок рейтингова оцінка студента переводиться згідно таблиці:

Бали	**Оцінка**
95 - 100	Відмінно
85 - 94	Дуже добре
75 - 84	Добре
65 - 74	Задовільно
60 - 64	Достатньо
Менше 60	Незадовільно
R < 40 є незараховані роботи комп'ютерного практикуму або не виконані інші умови допуску до екзамену	Не допущено

Робочу програму навчальної дисципліни (силабус):

Складено доцент, к.т.н., доцент каф. АПЕПС,Тихоход Володимир Олександрович

Ухвалено кафедрою _____ (протокол № ___ від _____)

Погоджено Методичною комісією факультету[1] (протокол № __ від _____)

[1] Методичною радою університету – для загальноуніверситетських дисциплін.