Міністерство освіти і науки України
Тернопільський національний технічний університет
імені Івана ПУЛЮЯ

кафедра програмної інженерії

М Е Т О Д И Ч Н І В К А З І В К И

щодо самостійної роботи студентів
та модульного контролю знань
з дисципліни

МОДЕЛЮВАННЯ ТА АНАЛІЗ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

для студентів першого рівня вищої освіти
за спеціальністю No 121 Інженерія програмного забезпечення

Тернопіль 2018

Методичні вказівки щодо самостійної роботи студентів та модульного контролю знань з дисципліни “Моделювання та аналіз програмного забезпечення” для студентів першого рівня вищої освіти за спеціальністю No 121 Інженерія програмного забезпечення / / Уклад.: М.Р. Петрик, О.Ю.Петрик - Тернопіль: ТНТУ 2018 - 28 с.

Призначені для полегшення засвоєння дисципліни “Моделювання та аналіз програмного забезпечення” і контролю знань студентів. Складені з урахуванням модульної системи навчання, рекомендацій до самостійної роботи і індивідуальних завдань, тем практичних та лабораторних занять, тестів, екзаменаційних питань, типової форми та вимог для комплексної перевірки знань з дисципліни

Укладачі: М.Р. Петрик, О.Ю.Петрик
Відповідальний за випуск: М.Р.Петрик
Рецензент: к.т.н., доцент Н.Б. Гащин

Розглянуто на засіданні кафедри програмної інженерії, протокол №2 від 3.09.2018р.

Схвалено на засіданні методичної ради факультету комп’ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя, протокол №2 від 24 жовтня 2018р.

ВСТУП

Метою викладання дисципліни “Моделювання та аналіз програмного забезпечення” є розвинення у студентів навичок моделювання програмного забезпечення шляхом аналізу предметної області, застосування методів декомпозиції складної системи та аналізу взаємозв’язків між її елементами, абстрагування та класифікація сутностей предметної області для встановлення ієрархії класів, що реалізовують функціональності системи для конкретних представників (екземплярів) користувача, використовуючи сучасні методи та інструментальні засоби моделювання.

Завданням вивчення дисципліни є набуття студентами практичних навичок та теоретичних знань з моделювання та аналізу програмного забезпечення з використанням засобів UML.

За результатами вивчення дисципліни студент повинен продемонструвати такі результати навчання:

1) володіння методами аналізу вимог до розроблюваної програмної системи з урахуванням специфіки предметної області;
2) розуміння принципів побудови, сутності та структури складних систем;
3) знати об’єктно-орієнтовані методи побудови моделей складних систем, методи системного аналізу та декомпозиції складної системи на елементи, моделювання зв’язків між ними; методи побудови основних абстракцій предметної області, їх системної класифікації для встановлення ієрархії класів, що реалізовують функціональності системи для конкретних представників користувача;
4) основи застосування універсальної мови моделювання UML, засоби для моделювання вимог і побудови відповідним їм моделей варіантів використання програмної системи;
5) засоби побудови моделей ієрархії класів, включаючи моделювання відношень між ними, моделей поведінки системи (станів, переходів, взаємодій);
6) основні найбільш використовувані сучасні зразки (патерни) і шаблони проектування для створення архітектури програмного забезпечення.
7) Вміння створювати та застосовувати об’єктні моделі предметної області; використовувати об'єктно-орієнтовані технології; будувати з використанням UML діаграми використання, діаграми ієрархії класів і

об’єктів; застосовувати вивчені зразки і шаблони проектування для створення архітектури створюваного програмного забезпечення.

Вивчення навчальної дисципліни передбачає формування та розвиток у студентів компетентностей:

загальних:

- дотримання професійної етики програмної інженерії.
- Здатність використовувати можливості апаратного забезпечення.
- Здатність використовувати можливості операційних систем, офісних.
- Здатність виявляти, ставити та вирішувати проблеми, приймати обґрунтовані рішення.
- Здатність діяти на основі етичних міркувань (мотивів).
- Здатність до пошуку, оброблення та аналізу інформації з різних джерел, проведення досліджень на відповідному рівні.
- Здатність оцінювати та забезпечувати якість виконуваних робіт.
- Здатність працювати в як індивідуально, так і в команді, мотивувати людей та рухатися до спільної мети.
- Здатність аргументовано переконувати колег у правильності пропонованого рішення, вміти донести до інших свою позицію.

фахових:

- базові уявлення про основи моделювання програмного забезпечення, типи моделей, основні концепції уніфікованої мови моделювання UML.
- Володіння основами методів та технологій об'єктно-орієнтованого програмування.
- Здатність здійснювати аналіз вимог, розробляти специфікацію програмних вимог.
- Здатність моделювати різні аспекти системи, для якої створюється програмне забезпечення.
- Здатність проектувати компоненти архітектури програмного продукту.
- Сучасні уявлення про основи інженерії вимог до програмного забезпечення.

– Сучасні уявлення про структуру та архітектуру програмного забезпечення, методи проектування програмного забезпечення.

– Уміння застосовувати знання у практичних ситуаціях.

– Уміння адаптуватись до нових ситуацій.

– Уміння ефективно працювати як автономно, так і у складі команди.

– Уміння відповідально ставитись до виконуваної роботи та досягти поставленої мети.

– Уміння спілкуватись включаючи усну та письмову комунікацію українською мовою та однією з іноземних мов (англійською, французькою, німецькою).

– Уміння використовувати інформаційні і комунікаційні технології для вирішення різних дослідницьких і професійних завдань.

– Уміння здійснювати пошук інформації в різних джерелах для розв’язання задач спеціальності.

– Уміння приймати обґрунтовані рішення та оцінювати їх наслідки.

– Уміння використовувати базові знання основ філософії, психології, педагогіки в професійній і соціальній діяльності.

– Уміння здійснювати моделювання процесів і об'єктів з використанням стандартних програмних технологій.

– Володіння технологією розроблення програмного забезпечення відповідно до вимог і обмежень замовника інформаційних систем і технологій.

– Сучасні уявлення про структуру та архітектуру програмного забезпечення, методи проектування програмного забезпечення.

– Здатність до проектної діяльності в професійній сфері, уміння будувати і використовувати моделі для опису об’єктів і процесів, здійснювати їх якісний аналіз.

Знання, вміння та навички, отримані студентами під час вивчення даної дисципліни будуть необхідними для їхньої професійної діяльності, а також використовуватимуться при написанні курсових та дипломних проектів з програмної інженерії.

Завданням лекційних занять є ознайомлення студентів з принципами та підходами до моделювання програмного забезпечення, їх класифікацією та властивостями, вивчення підходів до опису взаємодій між складовими частинками програмних систем та принципів управління та планування.

Завдання лабораторних занять полягає у практичному засвоєнні студентами навичок з побудови моделей програмних систем, які включають побудову діаграм класів, діаграм станів і переходів, діаграм взаємодії, діаграм об’єктів, діаграм модулів і процесів.

1. МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ І ТЕМИ ІНДИВІДУАЛЬНИХ ЗАВДАНЬ

1.1 СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ ЗА МОДУЛЬНОЮ СИСТЕМОЮ

Погодинний розподіл тем курсу наведено у таблиці 1

Таблиця 1 - Погодинний розподіл тем курсу

Назви змістових модулів і тем	Кількість годин											
	денна форма						Заочна форма					
	усього	у тому числі					усього	у тому числі				
		л	п	лаб	інд	с.р.		л	п	лаб	інд	с.р
1	2	3	4	5	6	7	8	9	10	11	12	13
Змістовий модуль 1. Основи структурного моделювання ПЗ												
Тема1. Основні підходи та концепції моделювання ПЗ	6	2	–	-	–	4	8	–		–	–	8
Тема2. Загальний підхід до моделювання системної архітектури	8	2		2	–	4	7	1	-	–	–	6
Тема3. Життєвий цикл ПЗ. Раціональний уніфікований процес проектування ПЗ	8	2	–	4	–	2	9	1	-	2	–	6
Тема4. Формалізовані мови моделювання ПЗ та сучасні інструментальні засоби проектування ПЗ. Основні завдання та характеристики UML.	6	2	–	-	–	4	8	-	-	-	–	8
Тема5. Моделювання варіантів використання	8	2		4	–	2	9	1	–	2	–	6
Тема6.Моделювання класів	10	4	–	4	–	2	9	1	–	2	–	6

Продовження таблиці 1

Тема7.Моделювання відношень між класифікаторами (класами, варіантами використання, підсистемами, компонентами та ін.)	8	2	-	4	-	2	6	-	-	-	-	6
Тема8. Моделювання інтерфейсів	6	2	-	-	-	4	6	-	-	-	-	6
Тема9. Моделювання об’єктів	6	2	-	-	-	4	6	-	-	-	-	6
Разом за змістовим модулем 1	66	20	–	18	–	28	68	4	-	6	–	58
Змістовий модуль2. Основи моделювання поведінкових характеристик ПЗ												
Тема10. Варіанти використання для визначення вимог	8	2	–	-	–	6	12	–	–	–	–	12
Тема11. Sequence Diagram. Моделювання повідомлень як взаємодії об'єктів. Comunication Diagram	13	4	–	4	–	5	18	2	–	2	–	14
Тема12. Моделювання станів-переходів, діяльностей та потоків керування	14	2	–	4	–	8	17	2	-	2	–	13
Тема13. Моделювання кооперацій, компонентів та вузлів обладнання	16	4	–	4	–	8	10	-		-	–	10
Тема14. Моделі та підходи проектування ПЗ, що ґрунтуються на використанні узагальнених проектних рішень	18	4		6	–	8	10	–	–	–	–	10
Разом за змістовим модулем 2	69	16		18	–	35	67	4	-	4	–	59
Усього годин	135	36	–	36	–	63	135	8	-	10	–	117

1.2 КОНТРОЛЬНІ ПИТАННЯ ДЛЯ САМОПЕРЕВІРКИ

Питання для самоперевірки складені за матеріалами всієї дисципліни “Моделювання та аналіз програмного забезпечення ” і є для студентів допоміжним засобом вивчення пропонованого курсу. Нижче приводяться контрольні питання щодо дисципліни.

Змістовий модуль 1. Основи структурного моделювання ПЗ

Тема 1. Основні підходи та концепції моделювання ПЗ

Актуальність моделювання ПЗ. Батоваріантність підходів до розробки ПЗ. Моделювання ПЗ як усталена і загальноприйнята інженерна методика. Суть моделі. Опис системи з різних точок зору. Мета і завдання моделювання ПЗ. Неформальне і формальне моделювання. Принципи моделювання на основні системного та архітектурного розуміння ПЗ.

Тема 2. Загальний підхід до моделювання системної архітектури ПЗ

Архітектура як набір суттєвих рішень. Взаємозв’язок архітектури ПЗ, вимог, функціональності, продуктивності, зручності, повторного застосування коду, економічних і технологічних обмежень, естетики. Взаємозалежні архітектурні вигляди системи: Use case view, Design view, Process view, Implementation view), Deployment view та основні діаграми.

Тема 3. Життєвий цикл ПЗ. Раціональний уніфікований процес проектування ПЗ

Rational Unified Software Development Process RUP. Ітеративний процес (iterative), що включає керування потоком виконавчих версій системи. Інкрементний або покроковий процес (incremental) як процес безперервної інтеґрації системної архітектури з метою випуску версій, кожна наступна з яких удосконалена в порівнянні з попередньою. Виявлення чинників, що складають найбільший ризик для проекту в цілому, та зведення їх до мінімуму.

Фази RUP: аналіз і планування вимог Requirements, проектування Design, побудова Construction, впровадження Transition. Контрольні точки (milestones). Діаграма потоку робіт для різних його складових частин в різних фазах.

Основні робочі процеси. Ітерація як - чітка послідовність дій з ясно сформульованим планом і критеріями оцінки, що приводить до появи нової версії системи, яка може бути виконана, протестована і оцінена.

ЖЦ ПЗ як безперервний потік версій, що виконуються, реалізують архітектуру системи із проміжною корекцією для зниження потенційного ризику.

Тема 4. Формалізовані мови моделювання ПЗ та сучасні інструментальні засоби проектування ПЗ. Основні завдання та характеристики UML.

Концептуальна модель UML як інструментальний засіб моделювання ПЗ, візуалізації, специфікації, конструювання і документування ПЗ. Основні елементи UML. Базові об'єктно-орієнтовані будівельні блоки моделей UML. Структурні сутності - «іменники» у моделях UML. Класи. Інтерфейси. Активні класи. Варіанти використання. Компоненти. Артефакти. Вузли.

Поведінкові базові сутності моделей (Behavioral things): взаємодія (interaction), повідомлення, діяльності (actions) і коннектори (з'єднання між об'єктами), стани, переходи, події, а також дії (реакції на переходи).

Основні завдання, можливості та характеристики інструментального засобу проектування ПЗ IBM Rational SoftWare Architect.

Тема 5. Моделювання варіантів використання

Принцип проектування прецедентів. Діаграми ВВ (Use Case) для моделювання вимог. Приклад Use Case діаграми (короткий аналіз). Візуалізація ВВ. Актори. Зміст діаграми ВВ. Контекст і суб'єкт. Організація ВВ. Відношення Актори-ВВ. Відношення між ВВ. Залежність, включення (В) і розширення (Р). Повторне використання з допомогою В і Р. Декомпозиція з допомогою В і Р. Їх зміст. Відношення узагальнення між ВВ. Відношення узагальнення між акторами. Ідентифікація акторів (як головних користувачів ПЗ). Актори і ролі. Стереотип «secondairy». Екземпляр актора. Формування оптимального списку ВВ. Опис ВВ як потоку подій. Текстові описи ВВ. Ідентифікація ВВ. Описи послідовностей реалізації ВВ.

Тема 6. Моделювання класів

Діаграми ієрархії класів. Основні завдання діаграм класів. Приклад. Загальна UML-нотація класу. Концепція абстракції (класи, асоціації) і екземпляра реалізації сутності (об'єкти, зв'язки). Класи і об'єкти. Властивості: атрибути і операції. Розміщення в нотації і синтаксис атрибутів. Поле операцій, синтаксис. Операції (сигнатура) і методи (реалізація). Операція - як реалізація послуги об'єкта класу, на яку може бути запит зі сторони іншого об'єкта чи актора (екземпляра актанта), щоб викликати певну його поведінку об'єкта класу. Вибір імені операції.

Тема 7. Моделювання відношень між класифікаторами (класами, варіантами використання, підсистемами, компонентами та ін.)

Відношення між класами: успадкування, залежність, асоціація, агрегація. Класифікатори. Асоціація: нотація, множинність, навігація. Одно- і двонапрямна асоціація. Приклад імплементації в С++-коді. Рефлективна асоціація. Асоціація типу агрегація і композиція. Ролі в

асоціації. Роль, яку відіграє клас, що перебуває на кінці асоціації. Множинність ролі (role multiplicity) асоціації.

Узагальнення (generalization) як зв'язок типу «є»: «клас-підклас», або «батько-нащадок»). Зображення узагальнення. Приклад імплементації в Java-коді. Інкапсуляція і модифікатори коду доступу до атрибутів. Пакетування. Приклад діаграми з інкапсуляцією. Доступ до властивостей при успадкуванні. Абстрактні класи і операції. Перевантаження і поліморфізм операцій при успадкуванні. Множинне успадкування. Узагальнення між іншими елементами (актантами, ВВ, вузлами).

Залежність. Зображення залежності. Види залежності. Зв'язки використання. Залежності між іншими елементами (примітками і пакетами). Асиметричність зв'язків залежності і узагальнення.

Ключові відмінності між зв’язками асоціації, залежності, узагальнення.

Тема 8. Моделювання інтерфейсів

Інтерфейси. Наданий та необхідний інтерфейси. Клас-клієнт інтерфейсу. Аналогія абстрактних класів та інтерфейсів. Приклади інтерфейсів. Суть з'єднань в складних системах. Моделювання зв’язків «інтерфейс реалізація». Стандартизовані підходи використання реалізації: інтерфейсів та кооперацій. Формалізовані форми інтерфейсів (стереотип interface), абстрактні класи, класи зі стереотипами, скорочена форма (lollipop notation). Стандартизовані підходи до специфікації наданого та необхідного інтерфейсу. Зв'язок між ВВ і кооперацією, яка його реалізує через інтерфейси.

Обов'язки і контракт, визначений інтерфейсом. Тип (type) як стереотип класу.

Тема 9. Моделювання об’єктів

Основні завдання. Абстракція та екземпляр. Зображення екземплярів. Ім’я екземпляра. анонімний об'єкт. Екземпляри операції. Стан об'єкту. Зміна стану об'єкту.

Діаграми класів і об’єктів. Їх взаємозв’язок. Приклад. Зв’язки між об’єктами. Реалізовані екземпляри залежностей між об’єктами. Приклад.

Автомат. Активні і пасивні елементи. Посилання і асоціації. Статичний атрибут. Об’єкти із стереотипами. Зв'язки залежності між об'єктами і класами: instanceof, instantiate. Характеристика добре структурованого екземпляру.

Моделі тестування на основі моделей взаємодії екземплярів. Діаграми об'єктів в тестуванні ПЗ. Необхідність розробки динамічних сценаріїв тестування на основі реальних значень їх атрибутів. Моделі розкадрування

сценаріїв. Статичний кадр в динамічному сценарії. Використання екземплярів абстракцій. Прямі і непрямі екземпляри. Анонімні об’єкти. Екземпляри операції. Стан об'єкту. Зміна стану об'єкту. Суть діаграм об'єктів (object diagrams). Загальні властивості діаграм об'єктів. Моделювання структур об'єктів для тестування сценарії реалізації.

Змістовний модуль2. Основи моделювання поведінкових характеристик та підсистем ПЗ

Тема 10. Варіанти використання для визначення вимог

Оцінка та класифікація ВВ по порядку важливості. Приклад класифікації. Модель предметної області. Її реалізація через ідентифікацію класів, що реалізовують усі необхідні для користувача функціональні операції. Приклад моделі предметної області. Її структуризація через пакетування. Визначення зв’язків між структурними елементами.

Взаємодія системи з користувачем через використання діаграм послідовності системи (Sequance Diagrame of System SDS). Приклад використання SDS. Операції системи.

Класи учасників: класи діалогу (інтерфейсні або граничні класи), класи керування, класи сутностей. Діаграми класів учасників (Diagrams of Participants Class - DPC). Приклади DPC. Загальна характеристика діаграм взаємодій (поведінкового вигляду ПЗ):послідовності, комунікації, станів, діяльності. Вимоги до створення діаграм при моделюванні виглядів системи.

Тема 11. Діаграми послідовностей (Sequence Diagram SD). Моделювання повідомлень як взаємодії об'єктів. Діаграми комунікацій (Comunication Diagram)

Основні завдання SD. Приклади елементів взаємодій. SD як екземпляри реалізації сценаріїв ВВ. Контекст взаємодій. Об’єкти і ролі у взаємодії. Описи термінології SD. Лінія життя. Повідомлення (message) як засіб взаємодії і обміну інформації між об’єктами та очікування наступних дій. Основні типи та екземпляри повідомлень (синхронні, асинхронні, сигнали). Відповідність повідомлень операціям. Імплантація синхронного повідомлення в С++-код. Відповідність асинхронних повідомлень сигналам. Створення і ліквідація об’єктів на лінії життя. Повідомлення завершені, загублені, знайдені. Синтаксис повідомлення. Повідомлення повернення. Синтаксис. Випадок ініціації події (occurrence). Типи повідомлень, моделювання яких забезпечує UML: call (викликати) return (повернути); send (послати); create (створити);destroy (знищити).

Посилання (link) (екземпляр асоціації) як семантичне з’єднання об’єктів один з одним і як шлях надсилання повідомлення між об’єктами.

Опис посилання. Обмеження. Прототипні об’єкти (ролі) та прототипні зв’язки асоціації (коннектори).

Діаграми послідовності і комунікації. Репрезентація повідомлень. Імплементація повідомлень в Java-код. Нумерація послідовності повідомлень. Еквівалентність з діаграмою послідовностей. Одночасні повідомлення. Оператор вибору взаємодії. Синтаксис. Використання Діаграм послідовності для моделювання сценаріїв варіантів використання та методів класів.

Тема 12. Моделювання станів-переходів, діяльностей та потоків керування

Діаграми станів-переходів. Приклади. Початкові і кінцеві стани. Події. Складні стани і початкові та кінцеві стани. Складні стани і внутрішні переходи. Стани історії (архів). Інтерфейси складних станів. Конкурентні стани. Конкурентні переходи.

Моделювання потоків керування. Діаграми діяльності. Приклад. Дії. Передача. Структура умовного керування. Приклад. Структуровані діяльності. Розпаралелювання активностей на елементи. Багатократне розпаралелювання. Параметри, що повертаються. Точки з’єднання активностей. Конкурентні активності. Синхронізація. Вузли завершення потоків контролю. Діяльності комунікації. Асинхронні виклики. Виключення. Приклади. Блоки переривання. Типи зон розширення.

Порядок передачі повідомлення об’єктом. Посилання сигналів. Потоки та послідовності повідомлень. Процедурний потік керування. Плоский (послідовний) потік. Ідентифікація процесів і потоків керування. Створення, модифікація та знищення об’єктів і посилань. Моделювання потоку керування.

Види керування, застосовні при моделюванні динамічних характеристик систем: необов’язкове виконання (тег opt); умовне виконання (тег alt); паралельне виконання (тег par); циклічне (ітераційне) виконання (тег loop). Ітерація (iteration). Розгалуження (branching).

Поняття про діаграми Перта і Гента.

Використання Діаграм активності для моделювання сценаріїв варіантів використання та методів класів.

Тема 13. Моделювання кооперацій, компонентів та розміщення вузлів обладнання

Кооперація. Приклад. Діаграма кооперацій. Кооперації і взаємодії.

Засоби пакетування. Суть і необхідність компонентної розробки ПЗ.

Компоненти та інтерфейси. Зв'язок між компонентою (що надає сервіс) і інтерфейсом. Інтерфейс, реалізований компонентою. Зв'язок між

компонентою та її інтерфейсами. Заміннність. Заміщувана компонента. Загальні принципи організації компонентів.

Діаграми компонентів і розміщення. Компонента. Діаграма компонентів. Приклад моделювання компонентів. Інтерес діаграм компонентів. Архітектура обладнання системи. Діаграма розміщення. Вузол. Артефакт. Конкретизація вузлів і артефактів. Реалізація компонентів.

Тема 14. Моделі та підходи проектування ПЗ, що ґрунтуються на використанні узагальнених проектних рішень

Узагальнене проектування та вимоги у вигляді класів проектування, кооперацій, підсистем. Декомпозиція системи на підсистеми. Вимоги до проектування класів підсистем. Зв’язки використання між класами різних підсистем. Інструментарій пакетів в моделюванні підсистем і компонентів.

Приклад проектної реалізації підсистем проектування і реалізації на основі раціонального уніфікованого підходу.

1.3 ТЕМИ ЛАБОРАТОРНИХ ЗАНЯТЬ

Ціль проведення лабораторних занять - закріплення теоретичних положень, викладених на лекціях. Студентам пропонуються лабораторні завдання, проблеми їх вирішення, довідкова література і методичні вказівки, розроблені для самостійної роботи. Викладач контролює хід рішення практичних задач і оцінює уміння студентів застосовувати знання, одержані на лекціях.

Зміст лабораторних занять наведено в табл. 2.

Таблиця 2 - Тематика лабораторних занять курсу

№	Тема заняття	Кількість годин	
		ДФН	ЗНФ
1	Вибір концептуальної моделі розробки ПЗ на основі аналізу вимог предметної області (згідно варіантів)	4	-
2	Аналіз сутностей предметної області	4	-
3	Діаграми варіантів використання	4	1
4	Побудова діаграм взаємодії (interaction diagrams)	4	1
5	Побудова діаграм класів	6	
6	Діаграми станів та переходів	4	2
7	Побудова діаграм діяльності	4	2
8	Цілісний проект	6	
Усього годин		36	6

ТЕМИ ДЛЯ САМОСТІЙНОГО ВИВЧЕННЯ

Самостійна робота студента є основним видом засвоєння навчального матеріалу у вільний від аудиторних занять час.

Метою самостійної роботи є вироблення студентами навичок і вміння працювати з літературою, віднаходити головні аспекти проблем, що потребують стійкого засвоєння.

Предметом самостійної роботи студентів є опрацювання ними як окремих тем програми курсу в цілому, так і деяких розділів тем, самостійне розв'язування конкретних задач з метою практичного поглиблення здобутих знань.

Перевірка рівня засвоєння матеріалу самостійно опрацьованих тем здійснюється при проведенні поточного тестового контролю з окремих тем згідно програми та під час написання домашніх і лабораторних робіт. Розподіл годин, винесених на самостійне опрацювання, наведено в табл. 3

Таблиця 3 - Розподіл годин на самостійну роботу

№	Найменування робіт	Кількість Годин	
		ДФН	ЗФН
1	Опрацювання лекційного матеріалу	10	4
2	Розв'язування індивідуальних завдань лабораторних робіт.	10	10
3	Оформлення звітів, підготовка лабораторних робіт до захисту.	6	6
4	Підготовка до складання екзамену, тестування.	6	30
5	Опрацювання окремих розділів програми, які не виносяться на лекції:		
5.1	Практичні приклади моделей предметної області (МПО). Реалізація МПО через ідентифікацію класів, що реалізовують усі необхідні для користувача функціональні операції. Приклад моделі глосарію предметної області. Визначення зв'язків між структурними сутностями.	3	8
5.2	Методи визначення кандидатів в класи, методи і атрибути. Розширене моделювання відношень між класами (способи взаємодії і комбінування абстракцій): на базі використання засобів	4	9

	декомпозиції та ОО-аналізу - успадкування , залежності, асоціаціє (агрегації).		
5.3	Поглиблене розуміння суті об'єктів та їх станів в моделюванні поведінки ПЗ. Створення і ліквідація об'єктів на ЛЖ. Суть повідомлень різних типів (завершені, загублені, знайдені, повідомлення повернення) як способів взаємодії об'єктів і ролей.	4	9
5.4	Розширене розуміння діаграм послідовностей щодо практичного використання. Різні типи взаємодій об'єктів і прототипних ролей на. Фокус керування. Основний зміст діаграми послідовності і комунікації. Їх принципові відмінності. Впорядковані (за часом та об'єктами) взаємодії на ЛЖ. Види керування на діаграмах послідовності.	4	9
5.5	Практичні аспекти проектування інтерфейсів (I). I, реалізований компонентою або класом (надаваний). Необхідний інтерфейс компоненти або класу. Розширені зв'язки між компонентою та її інтерфейсами. Заміннність. Приклади компонентів і розміщення для різних предметних областей.	4	10
5.6	Механізми UML-пакетування як засоби вирішення проблеми складності системи. Пакети як підсистеми, що містять компоненти ПЗ. Суть і необхідність компонентної розробки ПЗ. Внутрішня структура пакетів, підсистем компонент. Приклади.	4	8
5.7	Відповідальність класів. Вимоги проектування класів як елементів архітектури та реалізаторів варіантів використання системи. Розширене розуміння кооперації класів. Класи аналізу і проектування.	4	8
5.8	Засоби забезпечення прийняття стандартизованих рішень при проектуванні ПЗ. Застосування зразків і шаблонів проектування різних категорій в створенні моделей предметних областей. Переваги та недоліки патернів різних категорій (патерни створення та поведінки. Приклади їх структури. Імплементація в Java- або C++-код.	4	10
Разом		63	121

2. КОНТРОЛЬ ЗНАНЬ СТУДЕНТІВ

Контроль знань студентів включає оцінку знань студента включає в себе – виконання індивідуальних завдань, оформлення звіту з лабораторних робіт, захист звітів із лабораторних робіт, підсумковий контроль з кожного модуля дисципліни. Підсумковий модульний контроль передбачає комп'ютерного тестування знань теоретичних питань з дисципліни і рішення практичного завдання згідно варіанту.

Типові тести для контролю знань

1. ВВ потрібно створювати як:

   одне ціле, яке можна змінювати, рецензувати, тестувати і управляти ним;
   декілька частин, які окремо можна змінювати, рецензувати, тестувати і управляти ним;
   окремі не зв'язні частини.

2. Буква «U» в абревіатурі «UML» означає:

   United
   Unified
   Universal

3. Життєвий цикл програмної інженерії це:

   контекст проектування: від вимог до тестів
   контекст програмування: від вимог до тестів
   контекст моделювання: від варіантів використання до тестів

4. Яка з перелічених діаграм не належить до стандартних діаграм UML

   діаграма програмного коду
   діаграма пакетів
   діаграма класів

5. Щоб відобразити пакети і взаємодію між пакетами в системі використовують

   діаграму пакетів
   діаграму варіантів використання
   діаграму кооперації

6. Модель UML складається з (вкажіть зайве):

   сутностей
   відносин
   множин

7. Сутності UML поділяються на (вкажіть зайве):

структурні
поведінкові
графічні
групуючі
анотаційні

8. Відносини UML поділяються на (вкажіть зайве):

залежності
асоціації
уточнення
узагальнення
реалізації

9. Структурні сутності UML включають в себе (вкажіть зайве)

класи
вузли
пакети
варіант використання
інтерфейси

10. Поведінкові сутності UML включають в себе (вкажіть зайве)

стани
діяльності
варіанти використання
інтерфейси

11. Сутностями UML є (вкажіть зайве)

класи
вузли
залежності
примітки
варіанти використання

12. Групуючі сутності UML включають в себе

класи
вузли
пакети
примітки

13. Анотаційні сутності UML включають в себе

класи
вузли
пакети
примітки

14. Відношення залежності в UML є

симетричними
асиметричними
транзитивними

15.Відношення узагальнення в UML є

симетричними
асиметричними
транзитивними

16.Відносшення асоціації (без доповнень) до UML є

симетричними
асиметричними
транзитивними

17.Відношення реалізації в UML є

симетричними
асиметричними
транзитивними

18.Множина канонічних діаграм UML

визначається стандартом мови
є угодою користувачів мови
визначається виробниками інструментів, що підтримують UML

19.Множина канонічних структурних діаграм UML містить у собі (вкажіть зайве)

Діаграми класів
Діаграми використання
Діаграми компонентів
Діаграми об'єктів

20.Множина канонічних структурних діаграм UML включає в себе

Діаграми послідовності
Діаграми (кооперації) комунікації
Діаграми використання
Діаграми розміщення

21.Множина канонічних поведінкових діаграм UML містять у собі (вкажіть зайве)

Діаграми станів
Діаграми діяльності
Діаграми послідовності
Діаграми потоків даних

22.До множини канонічних поведінкових діаграм UML входять

Діаграми класів
Діаграми компонентів

Діаграми послідовності
Діаграми розміщення (розгортання)

23.Множина канонічних діаграм UML містить у собі (вкажіть зайве)

Діаграми класів
Діаграми станів
Діаграми послідовності
Діаграми потоків даних

24.Канонічні діаграми реалізації призначені для опису

поведінки
використання
структури

25.Канонічні діаграми класів призначені для опису

поведінки
використання
структури

26.Канонічні діаграми об'єктів призначені для опису

поведінки
використання
структури
композиції

27.Канонічні діаграми станів призначені для опису

поведінки
використання
структури
залежності від часу

28.Канонічні діаграми послідовності призначені для опису

поведінки
використання
структури

29.Канонічні діаграми кооперації призначені для опису

поведінки
використання
структури

30.Канонічні діаграми розміщення призначені для опису

поведінки
використання
структури

31.Діаграми діяльності призначені для опису

поведінки

використання
структури
застосування

32.Канонічні діаграми компонентів призначені для опису

поведінки
використання
структури

33.Діаграма взаємодії (Interaction diagram) показує взаємодію, що складається з набору об'єктів і виконуваних ними ролей, включаючи:

повідомлення, які можуть передаватися між ними.
атрибути, які можуть передаватися між ними.
пакети, які можуть передаватися між ними.

34.Діаграми взаємодії поділяються на діаграми:

станів і звязків
послідовностей і діаграми комунікацій.
процесів та апаратів

35.Канонічні діаграми взаємодії призначені для опису

поведінки
використання
структури

36.Які діаграми використовуються для опису моделі взаємодії?

діаграма послідовності
діаграма станів
діаграма діяльності

37.На діаграмі класів UML застосовують такі основні типи сутностей

Класи
Варіанти використання
Вузли
Інтерфейси
Компоненти

38.На діаграмі класів UML застосовують такі основні типи відносин між класами

Залежність
Узагальнення
Асоціація
Реалізація
Розширення

39.На діаграмі класів UML застосовують такі основні типи відносин між класом і інтерфейсом

Залежність
Узагальнення
Асоціація
Реалізація

40.На діаграмі класів UML застосовують такі основні типи відносин між інтерфейсами і класами

Залежність
Узагальнення
Асоціація
Реалізація

41.На діаграмі компонентів UML застосовують такі основні типи сутностей

Компоненти
Стани
Вузли
Інтерфейси

42.На діаграмі розміщення (розгортання) UML застосовують такі основні типи сутностей

Вузли
Стани
Об'єкти
Компоненти

43.Щоб показати, що клас є абстрактним, в UML застосовується

Підкреслення імені класу
Курсив імені класу
Жирний текст імені класу
Стереотип «abstract»

44.Ім'я стереотипу в UML виділяється

Підкресленням
Курсивом
Напівжирним зображенням
Лапками «»

45.Додаткові елементи нотації (прикраси) користувач UML може

включати чи не включати в модель
показувати чи не показувати на діаграмі
встановлювати або не встановлювати в інструменті

46.Класифікаторами в UML є (вкажіть зайве)

клас
інтерфейс
варіант використання
стан

47.Якщо класифікатор А є узагальненням класифікатора В, то

Всякий примірник класифікатора А є екземпляром класифікатора В
Всякий примірник класифікатора В є екземпляром класифікатора А
Жодна відповідь не вірна

48.Якщо ім'я атрибута класифікатора підкреслено, то

цей атрибут не змінює свого значення
цей атрибут є атрибутом об'єкта
усі примірники даного класифікатора мають одне значення цього атрибута
цей атрибут є ключовим

49.Кратність в UML є властивістю (вкажіть зайве)

класифікатора
полюси асоціації
операції
атрибута

50.Видимість в UML не є властивістю

класифікатора
асоціації
операції
атрибута
примітки

3. ЕКЗАМЕНАЦІЙНІ ПИТАННЯ

Екзаменаційні білети з дисципліни “ Моделювання та аналіз програмного забезпечення ” складені за всім курсом із внесенням питань, що представлені в розділі 1.2 “Контрольні питання для самоперевірки”.

Кількість білетів - 40 шт. Білети включають всі розділи курсу та є практично рівноцінними за своє складністю.

Формуляр екзаменаційного білету:

Форма № Н-5.05

Тернопільський національний технічний університет імені Івана Пулюя

Освітньо-кваліфікаційний рівень бакалавр
Спеціальність 121 «Інженерія програмного забезпечення» Семестр I
Навчальна дисципліна «Об’єктно-орієнтоване програмування»

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 1

1. Класифікація систем.
2. UML-диаграми варіантів використання.
3. Проект.

Затверджено на засіданні кафедри програмної інженерії
Протокол № 1 від „___” __ 201_ року

Завідувач кафедри _____
Екзаменатор _____

4. КОНТРОЛЬНІ ПИТАННЯ З ДИСЦИПЛІНИ

1) Складові частини об’єктно-орієнтованого підходу
2) Класи і відношення між ними
3) Категорії класів
4) Ознаки складної системи
5) Структура та проектування складних систем.
6) Декомпозиція системи
7) Структура системи
8) Структура системи
9) Специфікації
10) Об’єктна модель
11) Діаграми станів і переходів
12) Діаграми об’єктів
13) Системна архітектура ПЗ
14) Об’єктно-орієнтований аналіз, проектування та програмування
15) Діаграми взаємодій
16) Класифікація систем
17) Управління проектами
18) Ідентифікація компонентів системи
19) Механізми абстракції
20) Системна архітектура програмної системи
21) Варіанти використання
22) UML як мова моделювання ПЗ
23) Якість ПЗ
24) UML-диаграми класів
25) Планування ПЗ
26) Універсальний підхід моделювання ПЗ
27) Декомпозиція системи
28) UML-діаграми об’єктів
29) UML-диаграми дій
30) Складові частини об’єктно-орієнтованого підходу
31) UML- діаграми сценаріїв реалізації
32) Об’єктна модель
33) Складові частини об’єктно-орієнтованого підходу
34) Об’єктно-орієнтований аналіз, проектування та програмування
35) Об’єктна модель
36) Класифікація систем
37) Об’єктно-орієнтований аналіз, проектування та програмування

38) Механізми абстракції
39) Класифікація систем
40) UML-діаграми варіантів використання
41) Складність ПЗ.
42) Структура та проектування складних систем.
43) Об’єктна модель системи.
44) Складові частини об’єктного підходу.
45) Застосування об’єктних моделей.
46) Класифікація моделей їх ідентифікація.
47) Основні абстракції та механізми.
48) Діаграма класів.
49) Діаграма станів та переходів.
50) Діаграма об’єктів.
51) Діаграма взаємодій.
52) Діаграма модулів і процесів.
53) Управління і планування.
54) Якість і вимірювання.

ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. «Моделювання та аналіз програмного забезпечення» для студентів 121 - Інженерія програмного забезпечення, рівень вищої освіти перший (бакалаврський) / Уклад. Петрик М.Р., Петрик О.Ю. –Тернопіль: ТНТУ імені Івана Пулюя, 2014. [Електронний ресурс]. - Режим доступу: URL: *https://dl.tntu.edu.ua/content.php? course=1351*
2. Петрик О.Ю. Методичні вказівки щодо виконання лабораторних робіт з дисципліни "Моделювання та аналіз програмного забезпечення " для студентів напряму 6.050103 - "Програмна інженерія " / М.Р. Петрик, О.Ю. Петрик - Тернопіль: ФОП Паляниця В.А.,- 2014. – 75 с.
3. Петрик М. Моделювання та аналіз програмного забезпечення. Науково-методичний посібник. / Петрик М., Петрик О. - Тернопіль: Вид-во ТНТУ ім. І. Пулюя.-2014.-180с.
4. Петрик М.Р. Моделювання програмного забезпечення: науково-методичний посібник / М.Р. Петрик, О.Ю. Петрик – Тернопіль : Вид-во ТНТУ імені Івана Пулюя, 2015. – 200 с. URL: http://elartu.tntu.edu.ua/handle/123456789/17796
5. Буч, Г. Введение в UML от создателей языка; пер. с англ. / Г. Буч, Дж. Рамбо, И. Якобсон – ДМК Пресс, 2015. – 496 с.
6. Booch G. Object-Oriented Analysis and Design with Applications 3 Ed, Addison-Wesley, 2007.
7. Якобсон А., Буч Г., Рамбо Д. Унифицированный язык моделированя UML. Руководсто пользователя. - М.: Изд. дом «Вильямс», 2004.–460с.
8. Якобсон А., Буч Г., Рамбо Д. Рациональный унифицированный подход. - М.: Изд. дом «Вильямс», 2003.–600с.
9. Якобсон А., Буч Г., Рамбо Дж. Унифицированный процесс разработки программного обеспечения The Unified Software Development Process, - СПб: Питер, 2002.- 496с.
10. Фаулер, М. UML основы. Второе издание. Краткое руководство по унифицированному языку моделирования./ М. Фаулер, К. Скотт – СПб.: Символ-плюс, 2002.– 192с.

НАВЧАЛЬНЕ ВИДАННЯ МЕТОДИЧНІ ВКАЗІВКИ ЩОДО САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ ТА МОДУЛЬНОГО КОНТРОЛЮ ЗНАНЬ

з дисципліни “Моделювання та аналіз програмного забезпечення”

для студентів першого рівня вищої освіти

за спеціальністю No 121 Інженерія програмного забезпечення

Укладачі: Петрик Михайло Романович,

Петрик Оксана Юліанівна