**МОДУЛЬ 1**

***Лабораторна робота 1.3***

**ДОСЛІДЖЕННЯ ВЗАЄМОЗВ'ЯЗКУ ШАБЛОНІВ ПРОЕКТУВАННЯ ТА ОСНОВНИХ ПРИНЦИПІВ ПРОЕКТУВАННЯ**

**Завдання**

1. Спроектувати, представити у вигляді UML діаграми(-ам) функціонал згідно з варіантом. На діаграмі виділити використаний шаблон проектування.
2. Розробити програму на мові C#, яка відповідає вимогам у варіанті. Для демонстрації роботи використати текстові повідомлення на консолі через операції вводу-виводу.
3. Діаграма(-и) та вихідний код повинні відповідати основним принципам проектування: OOP, SOLID, Law of Demeter (LoD), DRY, YAGNI, KISS, cohesion – coupling, inheritance with caution.
4. Для отримання балів, що відповідають «задовільно» необхідно реалізувати завдання, відповідно варіанту (табл. 1) з одним шаблоном проектування на власний вибір.
5. Для отримання балів, що відповідають «добре» необхідно у реалізацію завдання з табл. 1 додати імплементацію ще мінімум одного чи більше шаблонів проектування. Пояснити вибір, а також існуючі альтернативи. Які переваги дають шаблони в порівнянні з реалізацією такого ж завдання в попередній лабораторній роботі? Необхідною умовою є використання репозиторію Git. Початковий проект – попередня лабораторна робота. Показати та пояснити зроблені коміти (мінімум 3). Локальний репозиторій повинен бути обов’язково зв’язаний з віддаленим приватним репозиторієм.
6. Для отримання балів, що відповідають «відмінно» необхідно реалізувати реалізувати завдання у варіанті з табл. 1 з використанням 2-х і більше шаблонів проектування. Один із шаблонів повинен бути Observer, реалізований за допомогою механізму подій з виокремленням класів аргументів подій, генератора подій, спостерігачів та підписок (матеріал дисципліни ООП – делегати та події). Допускається в певних випадках використання інших шаблонів при достатній аргументації. Перед вибором шаблону варто проконсультуватися у викладача.

**Методичні рекомендації**

При виконанні роботи слід пам’ятати:

* вибір шаблону проектування повинен бути обгрунтованим. При цьому введення шаблону (-ів) не повинно порушувати відповідність коду принципам, вказаним у завданні в п.3. Якщо по-іншому неможливо ввести шаблон, варто або відрефакторити код так, щоб відновити відповідність принципам, або обрати інший шаблон;
* деякі шаблони проектування можуть мати кілька варіантів реалізації. Тому варто пояснити вибір саме цієї конкретної реалізації, та пояснити відмінності з іншими способами;
* при реалізації наблону Observer необхідно відокремити аргументи подій, генератор подій, спостерігачі та підписки, використавши механізм подій;
* різні характеристики/ сутності та специфічні операції не повинні бути в одному класі/ модулі;
* функціонал повинен бути легко розширюваним без необхідності внесення змін в існуючий код (чи при мінімальних змінах) – якщо в завданні не сказано протилежне;
* при необхідності реалізації однотипних операцій вварто використати абстракцію;
* код не повинен повторюватися і бути читабельним;
* коміти повинні бути атомарними та містити робочий код.

**Вимоги по реалізації застосування**

Використовувати багаторівневу архітектуру в даній лабораторній не є необхідним (але і не забороняється).

Головним критерієм якості є реалізація функціоналу з використанням шаблонів проектування та відповідність базовим принципам проектування. Використання в завданні шаблону проектування ні в якому разі не повинно призводити до порушення базові принципи проектування і створювати Code Smells. В цьому випадку варто провести рефакторинг коду. Якщо при цьому проблеми залишаються, варто ще раз задуматися про доцільність використання саме цього шаблону саме в цьому місці застосування.

Для сповіщення про зміни в стані об’єкту варто використовувати події, а також підписуватися чи відписуватися від них. Наприклад, у випадку зміни сума коштів на рахунку клієнт може отримувати смс-повідомлення; при цьому він може відмовитися від цих повідомлень. Чи коли хазяїну тварини бажано знати, коли його тварина зголодніла. Реалізація механізму подій в шаблоні Observer також ні в якому разі не повинна порушувати принципи проектування. Генерація події, підписки на події та інші компоненти цього шаблону повинні бути відокремлені. Також відокремленою повинна бути і підписка на події.

За необхідності можна додавати нові властивості, сутності та розширювати функціонал. Але ця необхідність повинна бути обґрунтована.

Управління та відповідь застосування здійснюється через консоль. Використання консолі, наприклад, для виведення повідомлень, потрібно винести за межі функціоналу предметної області. Тобто, вивід повідомлень на консоль повинен бути за межами методів, що виконують певні дії. Наприклад, вивід повідомлення про те, що тварина голодна повинна бути окремо від методу, який визначає стан тварини голодний. Методи класів предметної області мають змінювати певним чином стан об’єкту, а не просто виводити повідомлення на консоль (наприклад, на банківському рахунку змінюється сума коштів, чи тварина стає не голодна і т.д.).

Це ж саме стосується і використання шаблонів. Вони повинні бути використані саме для предметної області. Відповідно, в методах класів шаблону не повинно бути операцій вводу-виводу.

Сутності у завданнях (тварини, пристрої, ігри та ін.) створюються за схожими механізмами. Для цього можна використати деякі Creational шаблони;. Поведінка та умови цієї поведінки тварин, використання пристроїв, ігри теж мають деякі спільні риси. Тут можна використати як Behavioral, так і Structural шаблони.

Виконання операцій повинно здійснюватися тільки при настанні певних умов. Наприклад, багаторазовий виклик методу годування тварини не повинен приводити до її смерті від переїдання, або зняття коштів з рахунку не повинно призводити до негативного баланса. Для вирішення цієї задачі також можна використати шаблони проектування.

Для демонстрації роботи програми необхідно підготувати набір вхідних даних.

**Варіанти**

**Табл. 1**

| Номер  варіанту | Завдання |
| --- | --- |
| 1 | Розробити застосування, яке симулює роботу з комп’ютерною технікою: комп’ютер, ноутбук, смартфон, процесор, пам’ять (різні види), акумулятор (за необхідності). На пристроях можна працювати, грати, спілкуватися, слухати музику, дивитися відео, використовувати джерело безперебійного живлення при відсутності електроенергії. Якщо електроенергія відсутня, то пристрій може працювати: 1) від акумулятора (при наявності), залежно від ємності 2000-3000 мАг – до 48 год неінтенсивного використання (без ігор та відео) чи 16 год при інтенсивному використанні (усі функції), 2) від акумулятора 5000-7000 мАг (35000-5000 мВтг) – до 12 год неінтенсивного використання (без ігор та відео) чи 4 год при інтенсивному використанні (усі функції), 3) або джерела безперебійного живлення (не більше 30 хв). Врахувати, що для всіх можливих операцій повинні бути виконані передумови: встановлене програмне забезпечення, є підключення до мережі, наявні зовнішні пристрої чи інші. При відсутності ПЗ, підключення до мережі чи зовнішніх пристроїв пристроєм користуватися повноцінно неможливо (наприклад, без ПЗ не можна зовсім запустити програму, без мережі – тільки, ті, які не потребують мережі, без динаміків/ навушників чи гарнітури неможливо нічого почути з комп’ютера). Передбачити можливе розширення функціоналу пристроїв. Нові пристрої додаватися не будуть. |
| 2 | Розробити застосування, яке симулює поведінку тварин: кіт, папуга, змія. Вони мають лапи та крила і здатність їсти, ходити, бігати, говорити, літати, повзати. При цьому тварина повинна їсти мінімум один раз на день, максимум – 5, через приблизно однакові проміжки часу. В інших випадках тварина помирає. Якщо у тварини попередній прийом їжі був більше 8 год тому, то вона не може бігати чи літати, але може ходити, повзати чи говорити. Тварини можуть жити у хазяїна або на волі. Якщо тварина живе у хазяїна, то він за нею доглядає. Догляд за твариною передбачає в першу чергу годування. Передбачити урізноманітнення способів догляду за твариною, додавши опцію прибирання. Якщо у тварини прибирається 1 раз на день чи частіше, вона є щасливою. Прибирання та щастя тварини не є необхідною умовою виживання тварини. У хазяїна може бути тільки одна тварина. |
| 3 | Розробити застосування, яке симулює комп’ютерні ігри різних жанрів: стратегія, симулятор, online casino. Ігри сумісні тільки з Windows PC, за винятком online casino (для неї потрібний тільки браузер). Мають вимоги по продуктивності: процесор, оперативна пам’ять, відеокарта та розмір на HDD; online casino вимагає тільки наявність браузера. Ігри перед використання потрібно інсталювати, за винятком браузерної гри (вона запускається відразу в браузері). Якщо гра не інстальована, не можна проводити будь-які маніпуляції з нею. Гру не можна інсталювати, якщо недостатньо вільного місця на HDD. В online гру не можна грати без підключення до мережі. Гру не можна запустити, якщо апаратне забезпечення не відповідає мінімальним вимогам гри. В симуляторі можна використовувати кермо (але не можна в іграх інших жанрів). Воно також доступне в усіх інших симуляторах. Кермо не обов’язкове, але при наявності дозволяє покращити якість гри. В кожні грі можна зберегти поточний стан і завантажити збережене. Якщо збережених станів немає, то завантажити немає чого. Доки гра не запущена, не можна використовувати її функціонал. У гравця може бути тільки в одна гра кожного жанру, окрім симуляторів – їх може бути кілька. Грати в певний момент часу можна тільки в одну гру (тобто, іншу не можна запустити до тих пір, доки запущена гра). |
| 4 | Розробити застосування, що симулює процес вивчення різних університетських дисциплін: ООП, архітектура ПЗ, англійська. Дисципліни передбачають різні види активностей: лекції (крім англійської), практичні (тільки англійська), лабораторні роботи (крім англійської), курсова робота (крім англійської), модульні контрольні роботи, екзамен, залік (для ООП та англійської). Дисципліни вивчаються групами студентів, їх ведуть визначені викладачі. Лабораторні роботи розбиваються на підгрупи. Не може існувати підгрупи, меншої за 10 чол. Для вивчення дисциплін обов’язкова наявність комп’ютера чи ноутбука (при відсутності – вивчення дисципліни неможливе). Для англійської – наявність також звуковідтворюючого обладнання та додаткових активностей: читання, написання тестів, розповіді на задану тему. Без звуковідтворюючого обладнання проводити заняття з англійської неможливо. Передбачити можливість додавання нових дисциплін, для яких необхідне додаткове обладнання (наприклад, фізика чи хімія потребує наявність лабораторії та проведення дослідів) – врахувати в тому числі курс. В одній дисципліні різні види активностей може проводити один чи кілька викладачів. Дисципліни можуть вивчатися тільки один раз протягом усього періоду навчання на певному курсі. ООП – 1 курс, архітектура – 2 курс, англійська – 1-2 курс. Якщо група на 3 курсі і далі, то вона вивчати ці дисципліни не може. Викладач не може вести одночасно заняття з кількох дисциплін. Без зданих робіт протягом семестру, студент не допускається до МКР чи екзамену. Залік виставляється автоматом. |
| 5 | Розробити застосування, яке симулює роботу з комп’ютерною технікою: ноутбук, смартфон, планшет, процесор, пам’ять (різні види), сенсорний екран. На пристроях можна працювати, грати, чатитися, слухати музику, дивитися відео, друкувати на принтері. Якщо відсутня електроенергія, то пристрій може працювати: 1) від акумулятора (при наявності), залежно від ємності 2000-3000 мАг – до 48 год неінтенсивного використання (без ігор та відео) чи 16 год при інтенсивному використанні (усі функції), 2) від акумулятора 5000-7000 мАг (35000-5000 мВтг) – до 12 год неінтенсивного використання (без ігор та відео) чи 4 год при інтенсивному використанні (усі функції). Врахувати, що для всіх можливих операцій повинні бути виконані передумови: встановлене програмне забезпечення, є підключення до мережі, наявні зовнішні пристрої. При відсутності ПЗ, підключення до мережі чи зовнішніх пристроїв пристроєм користуватися повноцінно неможливо (наприклад, без ПЗ не можна зовсім запустити програму, без мережі – тільки, ті, які не потребують мережі, без динаміків/ навушників чи гарнітури неможливо нічого почути з комп’ютера). Передбачити можливе розширення переліку пристроїв. Новий функціонал додаватися не буде. |
| 6 | Розробити застосування, яке симулює поведінку тварин: собака, канарка, ящірка. Вони мають очі, лапи та крила і здатність їсти, бігати, ходити, співати, літати. При цьому тварина повинна їсти мінімум один раз на день, максимум – 5, через приблизно однакові проміжки часу. В інших випадках тварина помирає. Якщо у тварини попередній прийом їжі був більше 8 год тому, то вона не може бігати, співати чи літати, але може ходити, повзати. Тварини можуть жити у хазяїна або в зоомагазині. Незалежно від місця проживання за твариною передбачається догляд (прибирання). Якщо у тварини прибирається 1 раз на день чи частіше або вона живе на волі, вона є щасливою. Прибирання та щастя тварини не є необхідною умовою виживання тварини. Передбачити можливість випустити тварину на волі (в цьому разі за нею не доглядають). У хазяїна може бути кілька різних тварин. |
| 7 | Розробити застосування, яке симулює комп’ютерні ігри різних жанрів: стратегія, пригоди, RPG. Ігри сумісні з різними платформами (десктоп/ мобайл – незалежно від типу пристрою), крім стратегії – її можна використовувати тільки на Windows PC. Мають вимоги по продуктивності: процесор, оперативна пам’ять, відеокарта та розмір на HDD. Ігри перед використання потрібно інсталювати. Якщо гра не інстальована, не можна проводити будь-які маніпуляції з нею. Гру не можна інсталювати, якщо недостатньо вільного місця на HDD. Гру не можна запустити, якщо апаратне забезпечення не відповідає мінімальним вимогам гри. При запуску гри необхідно увійти обліковий запис гри (без входу грати неможливо). RPG підтримує режим мультиплеєра за допомогою кількох підключених маніпуляторів (при відсутності кількох маніпуляторів мультеплеєр недоступний). В кожні грі можна зберегти поточний стан і завантажити збережене. Якщо збережених станів немає, то завантажити немає чого. Доки гра не запущена, не можна використовувати її функціонал. З мобільного пристрою доступна можливість трансляції на інший пристрій. У гравця можуть бути різні ігри різних жанрів. Грати в певний момент часу можна тільки в одну гру (тобто, іншу не можна запустити до тих пір, доки запущена гра). |
| 8 | Розробити застосування, що симулює процес вивчення різних університетських дисциплін: основи програмування, ООП, алгоритми та структури даних. Дисципліни передбачають різні види активностей: лекції, лабораторні роботи, курсова робота, модульні контрольні роботи, екзамен, залік (залік тільки для основ програмування). Дисципліни вивчаються групами студентів, їх ведуть визначені викладачі. Лабораторні роботи розбиваються на підгрупи. Не може існувати підгрупи, меншої за 10 чол. Для вивчення дисциплін обов’язкова наявність комп’ютера чи ноутбука (при відсутності – вивчення дисципліни неможливе). Передбачити можливість зміни видів активностей в межах дисципліни, а також викладача (ів), що їх ведуть. В загальному на дисципліну повинно відводитися не менше 64 аудиторних годин. Дисципліну може вести не менше одного викладача. І не більше, ніж кількість підгруп + лектор. В одній дисципліні різні види активностей може проводити один чи кілька викладачів. Дисципліни можуть вивчатися тільки один раз протягом усього періоду навчання на певному курсі. Основи програмування – 1 курс, алгоритми та структури даних – 2 курс, ООП – 1-2 курс. Якщо група на 3 курсі і далі, то вона вивчати ці дисципліни не може. Викладач не може вести одночасно заняття з кількох дисциплін. Без зданих робіт протягом семестру, студент не допускається до МКР чи екзамену. Залік виставляється автоматом. |
| 9 | Розробити застосування, яке симулює процес гри в настільну гру: монополія, мафія, alias. Для гри може бути потрібно: ігрове поле, кубики, фішки, картки. Без одного із необхідних компонентів у гру грати неможливо. При цьому «зайвих» компонентів теж не повинно бути. Гру неможливо почати доки всі гравці не познайомляться з правилами. За хід гравець може посунути фішку, щось купити чи продати, виконати певну дію по завданню (але тільки одну дію, якщо протилежне не вказане в інструкції). Грати можна тільки по черзі (тобто, хід робить саме той гравець, хто є наступним). Передбачити, що грати можна вдвох чи з більшою кількістю гравців. В іншому випадку гру почати неможливо. Одночасно зробити хід кілька гравців не можуть. Передбачити можливість, що будуть додаватися ігри в, які можуть грати тільки 2 гравці (наприклад, шашки), а також вводитися додаткові компоненти гри (наприклад, додаткові картки). Гра триває доти, доки хтось із гравців не виграє (якщо інше не передбачається правилами). Додати можливість задавати параметри гри правилами (ігрові компоненти, гравці, дії під час ходу). |
| 10 | Розробити застосування, яке симулює роботу з комп’ютерною технікою: комп’ютер, смартфон, пам’ять (різні види), відеокарта. На пристроях можна працювати, грати, дивитися відео, використовувати динаміки чи гарнітуру. Якщо електроенергія відсутня, то пристрій може працювати: 1) від акумулятора (при наявності), залежно від ємності 2000-3000 мАг – до 48 год неінтенсивного використання (без ігор та відео) чи 16 год при інтенсивному використанні (усі функції), 2) від акумулятора 5000-7000 мАг (35000-5000 мВтг) – до 12 год неінтенсивного використання (без ігор та відео) чи 4 год при інтенсивному використанні (усі функції). У комп’ютера акумулятора немає. іВрахувати, що для всіх можливих операцій повинні бути виконані передумови: встановлене програмне забезпечення, є підключення до мережі, наявні зовнішні пристрої. При відсутності ПЗ, підключення до мережі чи зовнішніх пристроїв пристроєм користуватися повноцінно неможливо (наприклад, без ПЗ не можна зовсім запустити програму, без мережі – тільки, ті, які не потребують мережі, без динаміків/ навушників чи гарнітури неможливо нічого почути з комп’ютера). Передбачити можливе розширення переліку та функціоналу пристроїв. |
| 11 | Розробити застосування, яке симулює поведінку тварин: собака, сова, ящірка. Вони мають очі, лапи та крила і здатність їсти, бігати, літати. При цьому тварина повинна їсти мінімум один раз на день, максимум – 5, через приблизно однакові проміжки часу. В інших випадках тварина помирає. Якщо у тварини попередній прийом їжі був більше 8 год тому, то вона не може бігати, літати, але може ходити, повзати. Тварини можуть жити у хазяїна, в зоомагазині або на волі. В зоомагазині та у хазяїна за твариною передбачається догляд (прибирання). Якщо у тварини прибирається 1 раз на день чи частіше або вона живе на волі, вона є щасливою. Прибирання та щастя тварини не є необхідною умовою виживання тварини. Передбачити можливість хазяїну завести будь-яку іншу тварину, яка відповідає заданим вище критеріям. У хазяїна може бути кілька різних тварин. |
| 12 | Розробити застосування, яке симулює комп’ютерні ігри різних жанрів: симулятор, шутер, online пригоди. Ігри сумісні з різними платформами (десктоп, мобайл, консоль). Мають вимоги по продуктивності: процесор, оперативна пам’ять, відеокарта та розмір на HDD. Ігри перед використання потрібно інсталювати, за винятком браузерної гри (вона запускається відразу в браузері). Якщо гра не інстальована, не можна проводити будь-які маніпуляції з нею. Гру не можна інсталювати, якщо недостатньо вільного місця на HDD. В online гру не можна грати без підключення до мережі. Гру не можна запустити, якщо апаратне забезпечення не відповідає мінімальним вимогам гри. До консолі обов’язково повинні бути підключені маніпулятори (без маніпуляторів, на консолі грати неможливо). При запуску гри необхідно увійти обліковий запис гри (без входу грати неможливо). В кожні грі можна зберегти поточний стан і завантажити збережене. Якщо збережених станів немає, то завантажити немає чого. Доки гра не запущена, не можна використовувати її функціонал. Також можна поставити чи подивитися рейтинг гри. З мобільного пристрою та консолі доступна можливість трансляції. З доступних ігор є тільки по одній грі кожного жанру. Але при цьому є можливість додати гру іншого жанру. Грати в певний момент часу можна тільки в одну гру (тобто, іншу не можна запустити до тих пір, доки запущена гра). |
| 13 | Розробити застосування, що симулює процес вивчення різних університетських дисциплін: вища математика, англійська, фізика. Дисципліни передбачають різні види активностей: лекції (крім англійської), практичні (англійська, математика), лабораторні роботи (для фізики), курсова робота (крім англійської), модульні контрольні роботи, екзамен, залік (для англійської). Дисципліни вивчаються групами студентів, їх ведуть визначені викладачі. Лабораторні роботи розбиваються на підгрупи. Не може існувати підгрупи, меншої за 10 чол, і групи, меншої за 13 чол. Дисципліну може вести не менше одного викладача. І не більше, ніж кількість підгруп + лектор. В одній дисципліні різні види активностей може проводити один чи кілька викладачів. Для вивчення англійської обов’язкова наявність звуковідтворюючого обладнання та додаткових активностей: читання, написання тестів, розповіді на задану тему. Без звуковідтворюючого обладнання проводити заняття з англійської неможливо. Фізика потребує наявність лабораторії та проведення дослідів. Без лабораторії лабораторні роботи з фізики проводити неможливо. Передбачити можливість додавання нового обладнання: наприклад, комп’ютера, та проведення симуляцій. Комп’ютер не є необхідним, але дозволяє краще засвоювати матеріал і успішно здавати дисципліни. Дисципліни можуть вивчатися тільки один раз протягом усього періоду навчання на певному курсі. Фізика – 1 курс, вища математика – 1-2 курс, англійська – 1-2 курс. Якщо група на 3 курсі і далі, то вона вивчати ці дисципліни не може. Викладач не може вести одночасно заняття з кількох дисциплін. Без зданих робіт протягом семестру, студент не допускається до МКР чи екзамену. Залік виставляється автоматом. |
| 14 | Розробити застосування, яке симулює процес гри в настільну гру: монополія, шашки, нарди, шахи. Для гри може бути потрібно: ігрове поле, кубики, фішки, фігури, шашки. Без одного із необхідних компонентів у гру грати неможливо. При цьому «зайвих» компонентів теж не повинно бути. За хід гравець може посунути фішку, фігуру чи шашку, щось купити чи продати, «взяти» фігуру чи шашку (але тільки одну дію, якщо протилежне не вказане в інструкції). Грати можна тільки по черзі (тобто, хід робить саме той гравець, хто є наступним). Передбачити, що грати можна вдвох чи з більшою кількістю гравців. В іншому випадку гру почати неможливо. В деякі ігри можна грати тільки вдвох. Доступні ігри та гравці змінюватися не будуть. Гра триває доти, доки хтось із гравців не виграє (якщо інше не передбачається правилами). Додати можливість задавати параметри гри правилами (ігрові компоненти, гравці, дії під час ходу). |
| 15 | Розробити застосування, яке симулює поведінку різних тварин. Вони мають очі, лапи та крила і здатність їсти, бігати, співати, літати, повзати. При цьому тварина повинна їсти мінімум один раз на день, максимум – 5, через приблизно однакові проміжки часу. В інших випадках тварина помирає. Якщо у тварини попередній прийом їжі був більше 8 год тому, то вона не може бігати, літати, співати, але може ходити, повзати. Тварини можуть жити у хазяїна, в зоомагазині або на волі. В зоомагазині та у хазяїна за твариною передбачається догляд (прибирання). Якщо у тварини прибирається 1 раз на день чи частіше або вона живе на волі, вона є щасливою. Прибирання та щастя тварини не є необхідною умовою виживання тварини. Передбачити можливість хазяїну чи в зоомагазині завести будь-яку тварину, яка відповідає заданим вище критеріям. У хазяїна може бути тільки одна тварина, в зоомагазині та на волі – багато різних. |

**Контрольні запитання**

1. Що таке Code Smells? Які ви знаєте види? Наведіть приклади. Чому це погано?
2. Що таке spaghetti code? Чому це вважається поганою практикою?
3. Які Code Smells належать до недоліків рівня застосування? Чому?
4. Які Code Smells належать до недоліків рівня класів? Чому?
5. Які Code Smells належать до недоліків рівня методів? Чому?
6. Для чого потрібні шаблони проектування? Які задачі/ проблеми вони допомагають вирішити?
7. Чи завжди потрібно використовувати багато шаблонів проектування? Чому?
8. Які існують групи шаблонів проектування? Чому вони розділяються саме так?
9. Що таке Creational шаблони? Для чого вони потрібні і як можуть бути використані?
10. Що таке Structural шаблони? Для чого вони потрібні і як можуть бути використані?
11. Що таке Behavioral шаблони? Для чого вони потрібні і як можуть бути використані?
12. Про що говорить GRASP?
13. Чим відрізняються централізовані репозиторії (Subversion) від розподілених (Git)?
14. Яка структура Git репозиторію?
15. Які види вказівників існують в Git? В чому різниця між ними?
16. Що таке коміт? Яка його структура? Як зробити коміт?
17. В яких статусах може перебувати файл репозиторія?
18. Які стани репозиторію існують і як вони пов’язані між собою?
19. Як можна переміщувати різні вказівники Git? Для яких цілей?
20. Як зв’язати локальний репозиторій з віддаленим і здійснювати синхронізацію?
21. Які архітектурні шаблони та стилі ви знаєте? Як вони розділені по категоріях?
22. В чому полягає поєднання архітектурних стилів?
23. Архітектура клієнт-сервер.
24. Компонентна архітектура.
25. Багаторівнева архітектура та N-рівнева архітектура.
26. Шина повідомлень.
27. Об’єктно-орієнтована архітектура.
28. Сервісно-орієнтована архітектура.
29. Що варто реалізовувати, як наскрізну функціональність? Чому?

**Примітка:** Також можуть бути задані питання про шаблони проектування з лабораторної роботи № 1-2.