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

Кафедра обчислювальної техніки

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*F Інформаційні технології*
Спеціальність	*F2 Iінженерія програмного забезпечення*
Освітня програма	*Iінженерія програмного забезпечення комп'ютерних систем*
Статус дисципліни	*Нормативна*
Форма навчання	*Очна (денна)*
Рік підготовки, семестр	*2 курс, осінній семестр*
Обсяг дисципліни	*5 кредитів,150 годин: з них лекційних 30 годин, лабораторні роботи 30 годин, самостійна робота 90 годин*
Семестровий контроль/ контрольні заходи	*Екзамен, Модульна контрольна робота, Розрахунково-графічна робота*
Розклад занять	*Два заняття на тиждень* *Розклад за адресою https://schedule.kpi.ua/*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	*Лекції: доцент кафедри ОТ, кандидат техн. наук, Порєв Віктор Миколайович* *v_porev@ukr.net* *Лабораторні роботи: асистент кафедри ОТ, Рекечинський Дмитро Олександрович*
Розміщення курсу	На платформі дистанційного навчання «Сікорський»: https://classroom.google.com/c/NDIyMzA5OTc5Nzgy?cjc=wgho4qv

Програма навчальної дисципліни

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

Метою вивчення дисципліни є формування у студентів здатностей:

- опанувати об’єктно-орієнтований підхід для розробки та розвитку програмних систем;
- розуміти методологію об’єктно-орієнтованого проектування, оволодіти нею і використовувати її впродовж життєвого циклу програмного забезпечення;
- розробляти програмне забезпечення за допомогою сучасних інструментальних засобів створення програмного забезпечення

Основні завдання при вивченні дисципліни

Дисципліна «Об'єктно-орієнтоване програмування» забезпечує програмні компетентності і програмні результати освітньо-професійної програми (ОПП). Згідно з вимогами ОПП здобувачі після засвоєння дисципліни (як освітнього компонента ОПП) «Об'єктно-орієнтоване програмування» мають продемонструвати компетентності:

- ЗК01. Здатність до абстрактного мислення, аналізу та синтезу.
- ФК02. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування.
- ФК03. Здатність розробляти архітектури, модулі та компоненти програмних систем.

Відповідно до ОПП програмними результатами мають бути:

- ПРН12. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення
- ПРН15. Мотивовано обирати мови програмування та технології розробки для розв’язання завдань створення і супроводження програмного забезпечення.
- ПРН17. Вміти застосовувати методи компонентної розробки програмного забезпечення.

Успішне виконання завдань курсу «Об'єктно-орієнтоване програмування» забезпечує

знання:

- об'єктно-орієнтованої мови програмування;
- патернів об'єктно-орієнтованого проектування;
- інструментальних засобів та інтегрованих середовищ для створення програмного забезпечення

уміння:

- аналізувати вимоги до програмного забезпечення;
- проектувати архітектуру програмного забезпечення;
- використовувати патерни об'єктно-орієнтованого проектування;
- виконувати рефакторинг програмного коду;
- розробляти та налагоджувати програмне забезпечення;
- використовувати потрібні інструментальні засоби та інтегровані середовища для вирішення завдань;

набуття практичного досвіду:

- розроблення програмного забезпечення;
- роботи з інформаційними комп’ютерними технологіями

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

Для успішного оволодіння дисципліною "Об'єктно-орієнтоване програмування" відповідно до освітньої програми необхідно попередньо вивчити дисципліни: "Основи програмування", "Алгоритми та структури даних".

Компетентності, знання та вміння, отримані в рамках вивчення дисципліни "Об'єктно-орієнтоване програмування", можуть бути застосовані для вивчення дисциплін "Компоненти інженерії програмного забезпечення", "Технології розгортання програмного забезпечення", “Математичні основи і програмування комп'ютерної графіки та віртуальної реальності”.

3 Зміст навчальної дисципліни

В навчальній дисципліні «Об'єктно-орієнтоване програмування» заплановано вивчення наступних тем

Розділ 1. Вступ до курсу ООП
Тема 1.1. Огляд основних понять розробки програмного забезпечення

Розділ 2. Знайомство з мовою C++ об'єктно-орієнтованого програмування
Тема 2.1. Базові елементи мови C++
Тема 2.2. Препроцесор C++. Модульність програм
Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм
Тема 2.4. Класи C++. Множинне спадкування
Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи
Тема 2.6. Особливості програмування для операційної платформи Android. Особливості Android API

Розділ 3. Відношення класів та операції з об'єктами
Тема 3.1. Діаграми UML
Тема 3.2. Вкладені та локальні класи
Тема 3.3. Операції з об'єктами

Розділ 4. Елементи узагальненого програмування та стандартні бібліотеки C++
Тема 4.1. Шаблони C++
Тема 4.2. Стандартні бібліотеки C++. Контейнери
Тема 4.3. Стандартні бібліотеки C++. Алгоритми, функціональні об'єкти

Розділ 5. Організація взаємодії класів та об'єктів
Тема 5.1. Callback-функції
Тема 5.2. Інтерфейси
Тема 5.3. Статичні члени

Розділ 6. Патерни об'єктно-орієнтованого проектування

Тема 6.1. Поняття патерну проектування. Патерн Singleton
Тема 6.2. Різновиди патернів Factory
Тема 6.3. Патерни Facade, Adapter, Dependency Injection, Bridge
Тема 6.4. Патерни Decorator, Observer, Visitor

Розділ 7. Особливості об'єктно-орієнтованого проектування
Тема 7.1. Об'єктно-орієнтований підхід vs функціонально-процедурний
Тема 7.2. Принципи SOLID
Тема 7.3. Рефакторинг

Методичні рекомендації

Для успішного вивчення об'єктно-орієнтованого програмування необхідно набуття вмінь та навичок роботи створення проектів на основі певних мов програмування у сучасних середовищах розробки програмного забезпечення.

У якості базової обрана мова програмування C++. Вона підтримує найбільш повний набір елементів об'єктно-орієнтованого програмування. Разом з цим, у курсі даної дисципліни розглядаються та порівнюються можливості інших мов програмування: Java, C#, Python тощо.

Для виконання робіт комп'ютерного практикуму рекомендується використання сучасних версій середовищ розробки, зокрема, Microsoft Visual Studio, Qt, Android Studio. Студентам надається можливість самостійного вибору мови програмування та середовища розробки як для виконання завдань комп'ютерного практикуму так ы для розрахунково-графічної роботи.

4 Навчальні матеріали та ресурси

Базова література:

1. Об'єктно-орієнтоване програмування: конспект лекцій [Електронний ресурс]: навч. посіб. для студ. освітньої програми «Інженерія програмного забезпечення комп'ютерних систем» спеціальності 121 «Інженерія програмного забезпечення» / Порєв В.М.; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 4,8 МБайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 271 с. URL: https://ela.kpi.ua/handle/123456789/51571 Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 1 від 02.09.2022 р.)
2. Об'єктно-орієнтоване програмування: комп'ютерний практикум [Електронний ресурс]: навч. посіб. для студ. освітньої програми «Інженерія програмного забезпечення комп'ютерних систем» спеціальності 121 «Інженерія програмного забезпечення» / КПІ ім. Ігоря Сікорського; уклад. Порєв В.М. – Електронні текстові дані (1 файл: 2,5 МБайт). – Київ : КПІ ім. Ігоря Сікорського, 2022. – 105 с. URL: https://ela.kpi.ua/handle/123456789/51572 Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 1 від 02.09.2022 р.)
3. Ілюстративні матеріали курсу, презентації лекцій знаходяться за посиланням https://drive.google.com/drive/folders/18Tc1v15pZTlpwDN3_wN6_0vo6F__Mac6?usp=sharing
4. Актуальні завдання робіт комп'ютерного практикуму знаходяться за посиланням https://drive.google.com/drive/folders/1B3UfyuFSyM-ODTKeBeVStYAVLzZ9NpSv?usp=sharing

Додаткова:

5. Патерни проектування. REFACTORING GURU [Online]. Available from: https://refactoring.guru/uk/design-patterns
6. C and C++ in Visual Studio. [Online] Available from: https://learn.microsoft.com/en-us/cpp/overview/visual-cpp-in-visual-studio?view=msvc-170
7. C# documentation. [Online] Available from: https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/
8. Develop for Android. [Online] Available from: https://developer.android.com/develop
9. Gerbert Schildt. C++: A Beginner's Guide. McGrawHil, 2012, 542 p.
10. Gerbert Schildt. Java: A Beginner's Guide. McGrawHil (Eighth Edition), 2018, 814 p.
11. Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts. Refactoring: Improving the Design of Existing Code. Addison Wesley. Professional, Second Edition, 2018, 448 p.
12. Overview of Windows Programming in C++. [Online] Available from: https://docs.microsoft.com/en-us/cpp/windows
13. The Unified Modeling Language. [Online] Available from: https://www.uml-diagrams.org/
14. Working Draft, Standard for Programming. Language C++. ISO/IEC N4582, 1514 pp. https://www.open-std.org/jtcl/sc22/wg21/docs/papers/2016/n4582.pdf

Обладнання, що необхідне для проведення занять

Лекційні заняття проводяться в аудиторії, яка обладнана проектором, заняття компютерного практикуму – в комп'ютерному класі. Для дистанційних занять потрібен доступ до інтернет та програмне й апаратне забезпечення для проведення відеоконференцій.

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента)

5.1. Лекційні заняття

Передбачено проведення 18 лекцій обсягом 2 аудиторні години кожна.

№ лекції з/п	Назва теми (тем) лекцій та перелік основних питань
1	**Тема 1.1. Огляд основних понять розробки програмного забезпечення** Різновиди парадигм програмування. Середовища виконання програм. Побудова програми, керованої повідомленнями **Тема 2.1. Базові елементи мови С++** Абетка, операції, оператори. Умовні оператори, оператори циклу. Прості типи. Покажчикі. Масиви, структури. Функції: оголошення, визначення, виклик. Головна функція програми та її аргументи
2	**Тема 2.2. Препроцесор С++. Модульність програм** Роль препроцесора. Макроси. Особливості програмного коду модулів на С++. Заголовок модуля. Приховування даних та функцій у модулях. Зв'язки між модулями, ієрархія модулів. Реалізація модульності в інших мовах програмування.

3	**Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм** Оголошення класу, визначення його членів, створення екземплярів об’єктів. Обмеження доступу до членів класів. Спадкування. Ієрархії класів. Віртуальні функції та абстрактні класи. Поняття та реалізація поліморфізму. Реалізація поліморфізму в інших мовах програмуваання.
4	**Тема 2.4. Класи C++. Множинне спадкування** Поняття множинного спадкування. Приклади множинного спадкування. Ромбічне спадкування. Віртуальні базові класи. Проблеми реалізації множинного спадкування. Наявність множинного спадкування в інших мовах програмування. **Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи** Поняття API та можливості розроблення програмних додатків. Повідомлення Windows та іхня роль у функціонуванні системи. Вікна. Вікна діалогу – модальні та немодальні. Функція вікна.
5	**Тема 2.6. Особливості програмування для операційної платформи Android. Особливості Android API** Основи побудови додатків для Android. Використання мов Java, Kotlin. Програмування MainActivity та вікон діалогу.
6	**Тема 3.1. Діаграми UML** Можливості опису систем діаграмами UML. Основні види, різновиди діаграм. Диаграми прецедентів та класів. Типи відношень класів та об’єктів. Залежність, асоціація, агрегація, композиція, узагальнення.
7	**Тема 3.2. Вкладені та локальні класи** Поняття вкладених класів. Обмеження доступу до членів вкладеного класу і охоплюючого класу. Локальні класи. Особливості таких класів в C++ та в інших об'єктно-орієнтованих мовах програмування.
8	**Тема 3.3. Операції з об'єктами** Види операцій над об'єктами, як єдиним цілим. Копіювання, присвоювання об’єктів, додавання об’єктів. Конструктор копії. Перевантаження операторів
9	**Тема 3.3. Операції з об'єктами** Передавання об’єкта у якості параметра функції. Об'єкт як результат виклику функції. Огляд можливостей виконання над об’єктами в C++
10	**Тема 4.1. Шаблони C++** Шаблонні функції та шаблонні класи. Спеціалізація шаблонів. Доцільність шаблонів. **Тема 4.2. Стандартні бібліотеки C++. Контейнери** Еволюція стандартних бібліотек C та C++. Склад стандартної бібліотеки C++. Стандартна бібліотека шаблонів (STL). Види бібліотечних класів. Клас complex. Клас string.
11	**Тема 4.2. Стандартні бібліотеки C++. Контейнери** Поняття класу-контейнеру . Вимоги для користувацьких класів у якості елементів контейнерів. Класи vector, list, map **Тема 4.3. Стандартні бібліотеки C++. Алгоритми, функціональні об'єкти** Поняття біблотеки шаблонів алгоритмів. Шаблонні класи алгоритмів та їхні члени. Ітератори. Перелік стандартних алгоритмів. Алгоритми count_if, find_if, for_each, remove. Функціональні об'єкти. Класи стандартних функціональних об'єктів. Приклади використання функціональних об’єктів.
12	**Тема 5.1. Callback-функції** Необхідність організації функції зворотного виклику. Приклади callback-функцій. Можливості програмування callback в C++. Callback-функції і використання Windows

	API. Організація зв’язків між модуліми з використанням callback. Можливості реалізації техніки callback класами C++. **Тема 5.2. Інтерфейси** Оголошення класу-інтерфейсу. Особливості членів класу-інтерфейсу. Використання класів-інтерфейсів. Особливості реалізації класів-інтерфейсів в C++. Використання ключового слова interface в різних реалізаціях C++. Доцільність класів інтерфейсів в архітектурі програм.
13	**Тема 5.3. Статичні члени** Особливості статичних локальних перемінних. Статичні члени класів. Доцільність статичних даних-членів класів. Статичні функції-члени класів. **Тема 6.1. Поняття патерну проектування. Патерн Singleton** Поняття патерну проектування та програмування. Доцільність патернів. Книга Design Patterns. Класифікація патернів. Патерн Singleton та його реалізації – класична та Singleton Меєрса.
14	**Тема 6.2. Різновиди патернів Factory** Патерн Simple Factory. Доцільність такого патерну. Патерн Factory Method. Патерн Abstract Factory.
15	**Тема 6.3. Патерни Facade, Adapter, Dependency Injection, Bridge** Доцільність створення фасаду програмної системи. Призначення та реалізація патерну Adapter. Патерн Bridge.
16	**Тема 6.4. Патерни Decorator, Observer, Visitor** Основне призначення патерну Decorator. Патерн Observer та його складові. Реалізація спостереження стану декількох об’єктів. Патерн Visitor та можливості розширення функціоналу системи класів. **Тема 7.1. Об'єктно-орієнтований підхід vs функціонально-процедурний** Приклад для ілюстрації співвідношення об'єктно-орієнтованого та функціонально-процедурного підходів. Дуальність. Expression Problem
17	**Тема 7.2. Принципи SOLID** Ознаки поганих проектів. Огляд принципів, що входять до складу SOLID: Single responsibility, Open/Closed, Liskov, Interface segregation, Dependency inversion. Приклади програмного коду.
18	**Тема 7.3. Рефакторинг** Поняття рефакторингу. Ітеративність розробки та вдосконалення коду. Класифікація методів та прийомів рефакторингу. Приклади рефакторингу.

5.2. Лабораторні роботи

Основна мета виконання лабораторних робіт – закріплення теоретичних знань та отримання практичних навичок по роботі з інструментальними засобами розробки програм.

Варто відзначити, що за даним курсом практичну складову можна також організувати у вигляді комп'ютерного практикуму.

Для успішного вирішення завдань треба виконати самостійну роботу по ознайомленню з теоретичними відомостіми і набуттям вмінь та навичок роботи з комп'ютерними засобами створення та налагодження програмного забезпечення.

Заплановано виконання 6 лабораторних робіт. Терміни виконання цих робіт завчасно оголошуються викладачем, який веде заняття.

№ з/п	Назва лабораторної роботи	Кількість ауд. годин
1	**Робота №1. Знайомство із середовищем розробки програм Microsoft Visual Studio та складання модульних проектів програм на C++** Мета: отримати перші навички створення програм для Windows на основі проектів API Win32 для Visual C++ і навчитися модульному програмуванню на C++. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: – Тема 1.1. Огляд основних понять розробки програмного забезпечення. Зокрема питання, що стосуються побудови програм, керованих повідомленнями забезпечення графічного інтерфейсу користувача – Тема 2.2. Препроцесор C++. Модульність програм. Зокрема, потрібно навчитися складати багатомодульні проекти з декількох файлів у середовищі розробки програм на C++. – Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи Також потрібно навчитися розробляти програмні слабкозв'язані модулі-компоненти з мінімальним винесенням назовні подробиць реалізації кожного модуля. Необхідно розробити найпростіший інтерфейс взаємодії між модулями.	**6**
2	**Робота №2. Розробка графічного редактора об'єктів на C++** Мета роботи – отримати вміння та навички використовувати інкапсуляцію, абстракцію типів, успадкування та поліморфізм на основі класів C++, запрограмувавши простий графічний редактор в об'єктно-орієнтованому стилі. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: – Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм – Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи Потрібно опанувати питання щодо створення ієрархії класів для опису графічних об'єктів та передбачити оброблення повідомлень для запису масиву об'єктів та поліморфного відображення	**6**
3	**Робота №3. Розробка інтерфейсу користувача на C++** Мета роботи – отримати вміння та навички використовувати інкапсуляцію, абстракцію типів, успадкування та поліморфізм на основі класів C++, запрограмувавши графічний інтерфейс користувача. Окрім вивчення особливостей програмування додатків з графічним інтерфейсом, керованих повідомленнями, студенти вчаться документовати програми діаграмами UML, зокрема, опановують діаграми класів. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: – Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм – Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи – Тема 3.1. Діаграми UML	**6**
4	**Робота №4. Вдосконалення структури коду графічного редактора об'єктів на C++**	**6**

	Мета роботи – отримати вміння та навички проектування класів, виконавши модернізацію коду графічного редактора в об’єктно-орієнтованому стилі для забезпечення зручного додавання нових типів об'єктів. Фактично, для виконання роботи виконується рефакторинг. Студенти отримують певні навички розробки складних проектів шляхом ітераційного вдосконалення. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: − Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм − Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи − Тема 3.1. Діаграми UML − Тема 7.3. Рефакторинг	
5	**Робота №5. Розробка багатовіконного інтерфейсу користувача для графічного редактора об’єктів** Мета роботи – отримати вміння та навички програмувати багатовіконний інтерфейс програми на C++ в об’єктно-орієнтованому стилі. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: − Тема 2.3. Класи C++. Інкапсуляція. Спадкування. Поліморфізм − Тема 2.4. Класи C++. Множинне спадкування − Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи − Тема 6.4. Патерни Decorator, Observer, Visitor	**6**
6	**Робота №6. Побудування програмної системи з множини об’єктів, керованих повідомленнями** Мета роботи: отримати вміння та навички використовувати засоби обміну інформацією та запрограмувати взаємодію незалежно працюючих програмних компонентів. Основним завданням буде побудувати інтерфейс повідомленнями між об’єктами-компонентами програмних систем. Для цього потрібно обрати певний засіб обміну повідомленнями, визначити формат повідомлення, засоби його передавання-отримання. Для виконання роботи потрібно вивчення окремих відомостей за наступними темами: − Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи − Тема 3.1. Діаграми UML − Тема 6.4. Патерни Decorator. Observer, Visitor − Тема 7.1. Об'єктно-орієнтований підхід vs функціонально-процедурний	**6**

Примітка. При розробці програмних застосунків в ході виконання завдань Робіт №1-6 замість мови C++ можуть бути використані інші мови та середовища розробки, наприклад, C#, Java, Kotlin, Javascript тощо. Цільова платформа ОС також може бути іншою – Android, iOS тощо. Головне – виконання функціональних вимог завдань робіт, а також архітектури рішення (для Робіт №2-6). Усе це має бути узгоджено з викладачем, який веде лабораторні роботи (комп'ютерний практикум)

5.3. Розрахунково-графічна робота

Розрахунково-графічна робота (РГР) планується у вигляді проєкту створення конкретного програмного застосунка і виконується в ході самостійної роботи здобувачів. Для кожного здобувача індивідуальне завдання. Завдання пов’язані з реалізацією парадигми об'єктно-орієнтованого програмування шляхом виконання певного проєкту створення програмного забезпечення. Орієнтовний перелік тем РГР нижче

1. Запрограмувати модуль, якмй містить клас універсального вікна ToolWin, і створити програму, яка використовує об'єкт відповідного похідного класу.
2. Створити програму показу та обробки табличних даних.
3. Розробити програму – векторний графічний редактор.
4. Розробити програму для обробки растрових зображень – растровий графічний редактор.
5. Створити програму для автоматичної побудови діаграм залежностей модулів проектів C++.
6. Створити програму, яка послідовно показує панелі вводу інформації для реєстрації
7. Запрограмувати програму-калькулятор з можливістю парсінгу виражень
8. Створити власну бібліотеку класів вікон та запрограмувати програму, яка використовує немодальні вікна діалогу
9. Створити власну бібліотеку класів для створення модальних діалогових вікон та запрограмувати програму, яка використовує такі вікна діалогу
10. Створити програму, яка зберігає список студентів групи і виконує розрахунки успішності навчання
11. Створити комп'ютерну гру з графічними можливостями
12. Створити програму для показу тривимірної графіки
13. Створити програму для відображення діаграм класів
14. Розробити програму для побудови графіків довільних функцій y = f(x)
15. Розробити програму для побудови поверхонь функцій z = f(x,y)
16. Створити програму для вводу та обробки текстів
17. Створити програму для показу кордонів країн на поверхні Землі з можливістю вводу атрибутів для кожної країни
18. Створити програму, яка здатна взаємодіяти з іншими через засоби комунікації смартфона
19. Створити програмну систему у вигляді множини сумісно працюючих незалежних компонентів
20. Розробити програмну систему для моделювання процесів у часі

Примітка. Наведений перелік тем не є фіксованим і може змінюватися викладачем при взаємодії зі здобувачами – уточнюватися, розширюватися відповідно до інтересів здобувачів, їхнього рівня підготовки, знань та умінь, набутих в інших курсах.

Здобувачі можуть запропонувати власні теми для РГР. Також здобувачі можуть обирати мову програмування та середовище розробки програмного забезпечення для реалізації проєктів. Кожний здобувач узгоджує з викладачем тему РГР та основні положення завдання. Тема та завдання на РГР узгоджується не пізніше як за два місяця до кінця семестру.

5.4 Модульна контрольна робота

Модульна контрольна робота виконується студентами на останній лекції. На МКР виносяться питання з Розділів 1-7 Орієнтовний перелік питань надано нижче.

1. Розробка програмного забезпечення. Інтерпретатори та компілятори
2. Програми типу Message Driven Application. Як вони програмуються?
3. Парадигми програмування

4. Модульність програм C++. Стратегія модульності
5. Структура вихідного тексту модуля C++
6. Класи C++. Члени. Оголошення. Визначення
7. Класи C++. Конструктор. Конструктор за замовчуванням. Деструктор
8. Класи C++. Способи створення екземплярів об'єктів
9. Класи C++. Динамічні об'єкти. Масив об'єктів
10. Класи та модульність. Дві ролі класів
11. Успадкування класів. Поняття. Приклади
12. Можливості обмеження доступу до членів базового класу у похідних класах
13. Віртуальні функції. Абстрактні класи
14. Поліморфізм на основі класів C++. Приклад для масиву об’єктів
15. Поняття множинного успадкування класів C++
16. Ромбічне успадкування
17. Віртуальні базові класи
18. Поліморфізм при множинному успадкуванні
19. Статичні члени класів та як вони використовуються
20. Вкладені та локальні класи
21. Ідентифікація типів в ході виконання програми. RTTI
22. Перевантаження операторів. Основні поняття. Обмеження
23. Перевантаження операторів. Оператор як функція-член класу. Приклади
24. Перевантаження оператора “=“
25. Перевантаження оператора "+"
26. Перевантаження оператора "+" для випадку: число + об’єкт
27. Дружні функції. Яка від них користь? Приклад
28. Перевантаження оператора [ ]
29. Присвоювання об’єктів: B = A
30. Ініціалізація одного об’єкта іншим об’єктом: classname B = A
31. Передача об’єкта через параметри функції: Func(obj)
32. Об’єкт у якості результату роботи функції: obj = Func()
33. Конструктор копії. Приклад
34. Огляд операцій над об’єктами: B = A, classname B = A, Func(obj), obj = Func(), classname *p = Func() та способів вирішення проблем
35. Поняття шаблонів C++. Приклад
36. Шаблонні функції C++. Спеціалізація шаблонів функції. Приклад
37. Шаблонні класи C++. Приклад шаблонного класу
38. Стандартна бібліотека C++
39. Клас complex. Приклад використання
40. Класс string. Приклад використання
41. Клас vector. Приклад використання
42. Клас list. Приклад використання
43. Клас map. Приклад використання
44. Стандартні алгоритми бібліотеки шаблонів C++
45. Алгоритм count_if. Приклад використання
46. Алгоритм find_if. Приклад використання
47. Алгоритм for_each. Приклад використання
48. Алгоритм remove. Приклад використання
49. Поліморфізм. Обгрунтування. Тлумачення
50. Класи-інтерфейси і навіщо вони потрібні
51. Особливості класів-інтерфейсів C++

52. Поняття Callback - функції. Обгрунтування необхідності таких функцій
53. Приклади Callback – функцій
54. Реалізація Callback - функцій з використанням вказівників
55. Реалізація CallBack-функцій віртуальними функціями
56. Діаграми UML
57. Діаграма прецедентів
58. Діаграма класів. Побудова засобами Microsoft Visual Studio
59. Множинне спадкування та структура програми
60. Заміна множинного спадкування – чим та як?
61. Основні види відношень між класами, об’єктами на діаграмах класів
62. Відношення Залежність, Асоціація на діаграмах UML
63. Відношення Узагальнення, Агрегація, Композиція на діаграмах класів
64. Композиція vs Успадкування
65. Поняття шаблону проектування. Огляд патернів
66. Патерн Одинак (Singleton). Відомі реалізації цього патерну
67. Патерн Simple Factory
68. Патерн Фабричний метод (Factory Method)
69. Патерн Абстрактна Фабрика (Abstract Factory)
70. Патерн Builder
71. Патерн Dependency Injection
72. Патерн Façade
73. Патерн Adapter
74. Патерн Bridge
75. Патерн Decorator
76. Патерн Observer
77. Патерн Visitor
78. Процедурний підхід vs Об’єктно-орієнтований. Expression Problem
79. Принципи об'єктно-орієнтованого проектування та програмування. Принципи SOLID
80. Закон Деметри
81. Рефакторинг в ООП

Примітка. Цей перелік питань може змінюватися та уточнюватися у відповідності до викладання відповідних тем впродовж семестру. При наявності змін уточнений перелік питань повідомляється студентам заздалегідь не пізніше трьох діб до проведення МКР.

6. Самостійна робота студента

Впродовж вивчення навчальної дисципліни “Об'єктно-орієнтоване програмування” передбачається виконання самостійної роботи студента (СРС):

- для опрацювання матеріалів, викладених під час лекцій та поглиблення знань згідно тематиці завдань на СРС
- для підготовки до виконання лабораторних робіт – інсталяції програмного забезпечення, для створення та налаштування проєктів у середовищі розробки програмного забезпечення та при опрацюванні завдань
- при виконанні розрахунково-графічної роботи
- при підготовці до модульної контрольної роботи
- при підготовці до екзамену – 30 годин

Загальний обсяг СРС – 90 годин.

Завдання на самостійну роботу відповідно темам

Теми	Завдання на СРС
Тема 1.1. Огляд основних понять розробки програмного забезпечення	1. Різновиди та приклади сучасних інтерпретаторів та компіляторів. Інтегровані середовища розробки програм 2. Програмна реалізація обробників повідомлень 3. Особливості програмування додатків для Windows та Android 4. Приклади API
Тема 2.1. Базові елементи мови С++	1. Оператор switch. Динамічні масиви 2. Головна функція для програм Windows
Тема 2.2. Препроцесор С++. Модульність програм	1. Відображення ієрархії залежностей модулів 2. Складання проекту з декількох модулів
Тема 2.3. Класи С++. Інкапсуляція. Спадкування. Поліморфізм	1. Динамічні об'єкти. Масиви об'єктів 2. Класи та модульність. Дві ролі класів
Тема 2.4. Класи С++. Множинне спадкування	1. Поліморфізм при множинному спадкуванні 2. Необхідність використання множинного спадкування, можливості відмови від нього. 3. Множинне спадкування у інших мовах програмування
Тема 2.5. Особливості програмування для операційної платформи Windows. Об'єктна орієнтованість системи	1. Програмування модальних вікон діалогу 2. Програмування немодальних вікон діалогу. Проблеми організації роботи множини немодальних вікон. 3. Вікна Windows-програм та ООП 4. Можливості використання классів С++ для програмування вікон
Тема 2.6. Особливості програмування для операційної платформи Android. Особливості Android API	1. Основи побудови додатків для Android у середовищі Android Studio. Використання мов Java, Kotlin. 2. Огляд класів Android API 2. Програмування MainActivity та вікон діалогу. 3. Створення шаблону для вікон Alert Dialog
Тема 3.1. Діаграми UML	1. Засоби створення діаграм UML 2. Формування діаграм класів у інтегрованому середовищі розробки програм 3. Використання діаграм UML для документування програмних систем
Тема 3.2. Вкладені та локальні класи	1. Відмінності програмування вкладених та локальних класів С++ та Java
Тема 3.3. Операції з об'єктами	1. Дружні функції 2. Глобальні оператори перевантаження 3. Особливосты реалізації перевантаження операторів С++ та інших мовах
Тема 3.3. Операції з об'єктами	1. Особливості використання локальних та динамічних об'єктів в операціях над об'єктами 2. Відмінності посилань від покажчиків на об'єкти
Тема 4.1. Шаблони С++	1. Поняття узагальненого програмування в С++ 2. Узагальнене програмування в Java, C# та інших

	мовах програмування
Тема 4.2. Стандартні бібліотеки C++. Контейнери	1. Використання класу string для різних типів кодування символів 2. Перетворення даних string у char, wchar і навпаки 3. Використання стандартних контейнерів для зберігання графічних об'єктів класів Shape з робіт комп'ютерного практикуму 4. Класи array, multimap та їхнє можливе використання
Тема 4.3. Стандартні бібліотеки C++. Алгоритми, функціональні об'єкти	1. Еволюція шаблонів алгоритмів в різних стандартах C++ (C++ 11, C++ 14 та інших) 2. Еволюція функціональних об'єктів в різних стандартах C++ (C++ 11, C++ 14 та інших)
Тема 5.1. Callback-функції	1. Особливості CALLBACK функцій вікон програм для Windows. 2. Спорідненість callback і функціональних об'єктів.
Тема 5.2. Інтерфейси	1. Співвідношення інтерфейсів і абстрактних класів 2. Особливості інтерфейсів в мовах Java, C# та інших мовах програмування
Тема 5.3. Статичні члени	1. Декілька контекстів використання слова static в C++, Java та інших мовах програмування
Тема 6.1. Поняття патерну проектування. Патерн Singleton	1. Можливості програмування патерну Singleton для редактора об'єктів у роботі комп'ютерного практикуму 2. Проблеми реалізації патерну Singleton на C++ для багатопотокових застосувань
Тема 6.2. Різновиди патернів Factory	1. Доцільність використання патернів Factory Method та Abstract Factory. Приклади реалізації.
Тема 6.3. Патерни Facade, Adapter, Dependency Injection, Bridge	1. Навіщо відокремлювати користувача програмної системи від внутрішніх класів (патерн Facade)? 2. На чому грунтується техніка Injection? 3. Відмінності патерну Bridge від патерну Strategy
Тема 6.4. Патерни Decorator, Observer, Visitor	1. Особливості програмування класів та об'єктів згідно патерну Decorator. 2. Роль патерну Visitor у подоланні Expression Problem
Тема 7.1. Об'єктно-орієнтований підхід vs функціонально-процедурний	1. Проаналізувати можливості та засобів об'єктно-орієнтованого програмування повідомлень в додатках для Windows та інших операційних платформ.
Тема 7.2. Принципи SOLID	1. Антипатерн God object 2. Техніка Dependensy inversion для керування залежностіми модулів програмної системи
Тема 7.3. Рефакторинг	1. Заміна switch поліморфізмом 2. Проанаізувати можливості рефакторингу коду редактора об’єктів у роботах комп'ютерного практикуму

Політика та контроль

7. Політика навчальної дисципліни (освітнього компонента)

Всі студенти повинні відвідувати лекційні та лабораторні заняття – як при звичайному навчанні у аудиторіях (фізичне відвідування), так і при дистанційному навчанні (віртуальне відвідування).

Результати виконання завдань лаб. робіт та індивідуальних завдань РГР оформлюються у електронному форматі у вигляді файлів звітів, виконуваних файлів та інших файлів. Такі файли повинні містити результати відповідно завданням, вимогам та методичним вказівкам для кожної роботи. Робота програм перевіряється викладачем в ході прийому робіт.

За роботи, які виконані оригінально, творчо, ініціативно можуть бути нараховані додаткові бали при умові вчасної здачі (відповідно встановленим термінам виконання) робіт.

Усі письмові роботи перевіряються на наявність плагіату, фальсифікації, списування. Несанкціоноване значне запозичення чужого тексту рішення може призвести до незадовільної оцінки роботи. Списування під час контрольних робіт заборонені (в т. ч. із використанням мобільних пристроїв).

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

Робота студента щодо вивчення дисципліни "Об'єктно-орієнтоване програмування" оцінюється балами, які він отримує:

- за виконання 6 лабораторних робіт ($R_{ЛАБ}$);
- за виконання розрахунково-графічної роботи ($R_{РГР}$);
- за виконання модульної контрольної роботи ($R_{МКР}$);
- за екзамен (R_E).

Відповідно до "Положення про систему оцінювання результатів навчання в КПІ ім. Ігоря Сікорського" (https://osvita.kpi.ua/sites/default/files/downloads/Pologennia_RSO_2025.pdf) застосована система оцінювання типу РСО-2, для якої підсумкова рейтингова оцінка ($R_Д$) складається з:

- стартової оцінки (R_C) – оцінювання заходів впродовж семестру: $R_C = R_{ЛАБ} + R_{РГР} + R_{МКР,}$
- екзаменаційної оцінки R_E

Таким чином, $R_Д = R_C + R_E$

Розрахунок шкал оцінювання

Шкала оцінювання (максимально можлива оцінка)

Вид навчальної роботи	Всього за видом роботи (max)
Виконання та захист лабораторної роботи №1	6
Виконання та захист лабораторної роботи №2	6
Виконання та захист лабораторної роботи №3	6
Виконання та захист лабораторної роботи №4	6
Виконання та захист лабораторної роботи №5	6
Виконання та захист лабораторної роботи №6	6
Максимальна оцінка за усі роботи ***RЛАБ***	36
Максимальна оцінка за розрахунково-графічну роботу ***RРГР***	18
Максимальна оцінка за модульну контрольної роботи ***RМКР***	6
Максимальний стартовий рейтинг за семестр ***RС = RЛАБ + RРГР + RМКР***	60

Максимальна екзаменаційна оцінка ***RE***	40
Усього за семестр ***R*** = ***RC*** + ***RE***	100

Мінімально можлива загальна позитивна оцінка: ***RMIH*** = 0.6 × 100 = 60 балів

Критерії оцінювання виконання завдань лабораторнтх робіт

Оцінка за кожну роботу виставляється в результаті її захисту. На захисті перевіряється відповідність вимогам, коректність рішення, програмний код, стан проєкту у середовищі розробки програмного забезпечення, відсутність помилок роботи програмного застосунку, правильність оформлення звіту, своєчасність надсилання рішення викладачу на перевірку, правильність усних відповідей на контрольні запитання.

За кожну роботу максимальна оцінка 6 балів. Вона складається з

- 2 балів за правильність відповідей щодо теоретичної складової рішення;
- 2 балів за правильність відповідей щодо практичної складової (реалізації);
- 2 бали за демонстрацію на занятті роботи програми та проєкту у середовищі розробки;

Заохочувальні бали

За виконання творчих робіт з навчальної дисципліни (наприклад, участь у олімпіадах, конференціях, конкурсах наукових робіт, підготовка оглядів наукових праць чи наукових публікацій тощо) а також за оригінальність рішення, творчий підхід до виконання завдань комп'ютерного практикуму можуть додаватися заохочувальні бали, які не входять до загальної шкали оцінювання. Відповідно до "Положення про систему оцінювання результатів навчання в КПІ ім. Ігоря Сікорського" сума заохочувальних балів не може перевищувати 10 балів, при цьому загальний рейтинговий бал здобувача не може перевищувати 100 балів.

Визначення умов допуску до екзамену

Умовою допуску до екзамену є виконання з певною успішністю завдань впродовж семестру, тобто, студент має впродовж семестру заробити деякий ненульовий стартовий рейтинг (***RC*** > 0).

Мінімальний стартовий рейтинг для допуску до екзамену ***RC*** має бути не менше 25 балів.

Таким чином, при умові допуску до екзамену, студент в результаті екзамену буде мати підсумкову рейтингову оцінку (***RД***), яка буде сумою усіх отриманих оцінок у балах. Підсумкова рейтингова оцінка також переводиться в оцінку за університетською шкалою.

Таблиця переведення підсумкової рейтингової оцінки в університетську шкалу оцінок

Підсумкова рейтингова оцінка ***RД***	Університетська шкала оцінок
95...100	Відмінно
85...94	Дуже добре
75...84	Добре
65...74	Задовільно
60...64	Достатньо
Менше 60	Незадовільно
Стартовий рейтинг (***Rc***) менше 25	Не допущено

9. Визнання результатів неформальної освіти

Визнання результатів неформальної освіти здійснюється у відповідності до Положення про визнання в КПІ ім. Ігоря Сікорського результатів навчання, набутих у неформальній / інформальній освіті: https://osvita.kpi.ua/node/179

Зарахування окремої частини данного курсу може бути здійснено при індивідуальному розгляді сертифікатів, які надані відповідними провайдерами неформальної освіти.

Робочу програму навчальної дисципліни (силабус):

Склав доцент кафедри обчислювальної техніки, к.т.н. Порєв В.М.

Ухвалено кафедрою обчислювальної техніки (протокол № 18 від 24.06.2026 р.)

Погоджено Методичною комісією факультету інформатики та обчислювальної техніки (протокол № 12 від 26.06.2026 р.)