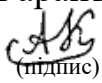
Міністерство освіти і науки України
Національний аерокосмічний університет
«Харківський авіаційний інститут»

**Кафедра Інформаційних технологій проєктування (№ 105)**

**ЗАТВЕРДЖУЮ**

Гарант освітньої програми

(підпис) Олександр КАРАТАНОВ
(ім'я та ПРІЗВИЩЕ)

«_29_» _08_ 2025 р.

# СИЛАБУС ОБОВ'ЯЗКОВОЇ НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

**Інструментальні засоби візуального програмування**
(шифр і назва навчальної дисципліни)

**Галузь знань:** 12 «Інформаційні технології»
(шифр і найменування галузі знань)

**Спеціальність:** 122 «Комп'ютерні науки»
(код і найменування спеціальності)

**Освітня програма:** «Інформаційні технології проектування»
(найменування освітньої програми)

**Рівень вищої освіти**: ***перший (бакалаврський)***

**Силабус введено в дію з 01.09.2025**

**Харків – 2025 р.**

Розробник: доцент кафедри 105 Інформаційних технологій проєктування,
(посада, науковий ступінь і вчене звання, ім'я та прізвище)
к.т.н., доцент Віктор ОВСЯННІК _____
(підпис)

Силабус навчальної дисципліни розглянуто на засіданні кафедри № 105 Інформаційних технологій проєктування
(назва кафедри)

Протокол № 1 від 28.08.2025 року

В.о. завідувача кафедри 105 _____ Аліна АРТЬОМОВА
(підпис) (ім'я та прізвище)

## 1. Загальна інформація про викладача

| |
|---|
| ПІБ: Овсяннік Віктор Миколайович |
| Посада: доцент кафедри кафедри 105 |
| Науковий ступінь: кандидат технічних наук |
| Вчене звання: доцент |
| Перелік дисциплін, які викладає:<br>*Об'єктно-орієнтоване програмування,*<br>*Інструментальні засоби візуального програмування* |
| Напрями наукових досліджень:<br>*Основи методології побудови єдиного інформаційного простору* |
| Контактна інформація: *Електронна пошта:*<br>*v.ovsiannik@khai.edu*<br>*Телеграм: @victor2166* |

## 2. Опис навчальної дисципліни

| | |
|---|---|
| Форма здобуття освіти | *Денна* |
| Семестр | 4, 5 (КП) |
| Мова викладання | Українська |
| Тип дисципліни | *Обов'язкова* |
| Обсяг дисципліни: кредити ЄКТС/ кількість годин | *денна*: 6,5 кредитів ЄКТС / 195 годин (96 аудиторних, з яких: лекції – 48, лабораторні роботи – 48, СРЗ – 99); 2 кредити ЄКТС / 60 годин (32 аудиторних, з яких: практичні заняття – 32, СРЗ – 28) |
| Види навчальної діяльності | Лекції, практичні та лабораторні заняття, самостійна робота |
| Види контролю | Поточний контроль, модульний контроль, семестровий контроль – іспит, диференційний залік |
| Пререквізити | *Об'єктно-орієнтоване програмування* |

.

## 3. Мета та завдання навчальної дисципліни, переліки компетентностей та очікуваних результатів навчання

**Мета:** Надати здобувачам освіти знання та практичні навички з розробки програмного забезпечення для роботи під керівництвом ОС Windows з широким використанням можливостей концепції об'єктно-орієнтованого програмування (ООП).

**Завдання дисципліни:** Засвоїти професійне застосування основних принципів технологій об'єктно-орієнтованого та візуального програмування; навчити методам проектування, розробки та відлагодження застосунків з використанням середовищ Microsoft Visual Studio та QT**;** сформувати навички використання мови UML; розкрити засади забезпечення ефективності, наочності та надійності застосунків Windows**;** ознайомити з сучасними тенденціями розвитку технологій та концепцій візуального програмування.

**Компетентності, які набуваються:**

***Інтегральна компетентність***: Здатність розв'язувати складні спеціалізовані задачі та практичні проблеми у галузі комп'ютерних наук або у процесі навчання, що передбачає застосування теорій та методів інформаційних технологій і характеризується комплексністю та невизначеністю умов

***Загальні компетентності (ЗК)***

***Після вивчення цієї дисципліни здобувач освіти опанує:***

ЗК2. Здатність застосовувати знання у практичних ситуаціях.

ЗК3. Знання та розуміння предметної області та розуміння професійної діяльності.

ЗК9. Здатність працювати в команді.

ЗК11. Здатність приймати обґрунтовані рішення.

***Спеціальні компетентності (СК)***

***Після засвоєння цієї дисципліни здобувач освіти опанує:***

СК8. Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно - орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.

***Програмні результати навчання (ПР):***

ПР5. Проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій.

ПР15. Застосовувати знання методології та CASE -засобів проектування складних систем, методів структурного аналізу систем, об'єктно-орієнтованої методології проектування при розробці і дослідженні функціональних моделей організаційно-економічних і виробничо-технічних систем.

## 4. Зміст навчальної дисципліни

# МОДУЛЬ 1

### Змістовний модуль 1. Об’єктно-орієнтований аналіз та технологія ООП

**Тема 1. Поняття об’єктно-орієнтованого аналізу та проектування застосунків Windows з використанням мов C++ C++/CLI C#**

***Анотація:*** *Структура курсу. Поняття структур, сутності, атрибутів відношення, класу предметної області, простору проблем та простору рішень. Визначення цих понять та їх використання. Об’єкт як абстракція сутності реального світу і його структура, Поведінка об’єкта, його властивості , зв’язки між об’єктами одної ієрархії та об’єктами різних типів*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій.*

**Тема 2. Об’єктна модель предметного середовища і принципи її побудови**

***Анотація:*** *Опис предметного середовища та постановка задачі Принципи побудови моделі, їх використання та приклади застосування*

***Теми лекцій:*** *Структури, об’єднання формати опису та визначення*

***Теми лабораторних/практичних занять:*** *«Формати описів та визначень структур, класів, об’єднань».*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

***Проектування та розроблення класу реалізації зв'язкового списку за індивідуальними завданнями:***

1) Вузол списку Трактор з член-даними марка (char *), колір (char *), об'єм (float) та потужність двигуна. Реалізувати член-функцію «додавання на початок списку нового вузла»

2) Вузол списку Людина з член-даними ПІБ (char*), ідентифікаційний код, вік. Реалізувати член-функцію «видалення першого вузла списку»

3) Вузол списку Книга з член-даними назва, число сторінок, мова видання (char*), наявність ілюстрацій (bool). Реалізувати член-функцію «видалення останнього елемента списку»

4) Вузол списку Кімната з член-даними ширина (float), довжина (double) та колір стін (char*). Реалізувати член-функцію «пошук вузла списку із заданим порядковим номером»

5) Вузол списку Птах з член-даними вага (float), максимальні висота та швидкість польоту або бігу, порода (char*). Реалізувати член-функцію «видалення елемента із заданим порядковим номером»

6) Вузол списку Двигун з член-даними тип двигуна ((char*)бензиновий, дизельний, електричний, комбінований), фірма-виробник (char*) та потужність двигуна (float). Реалізувати член-функцію «заміна елемента із заданим порядковим номером на новий»

7) Вузол списку вектор з член-даним покажчик на цілочисельний тип даних, його розмір і фактичне число елементів, тобто. масив. Реалізувати член-функцію "пошук у списку масиву із заданим вмістом". Потрібно знайти у списку всі вузли, які містять масив із заданим вмістом. Різні вузли списку повинні(можуть) містити різну кількість елементів масиву. у програмі повинен бути описаний і доведений до користувача формат файлу з даними.

8) Вузол списку Муха з член-даним вигляд (char *), число крил та їх розмах (float). Реалізувати член-функцію "вставка нового вузла перед вузлом із вказаним порядковим номером"

9) Вузол списку Сік з член-даними марка (char*), виробник (char*) та об’єм упаковки. Реалізувати член-функцію «видалення всіх вузлів, що містять опис соку із заданою маркою

10) Вузол списку Дитина з член-даними вага, зріст і раса (char *). Реалізувати член-функцію «вставка елемента після вказаного порядкового номера вузла списку

11) Вузол списку Комп'ютер з член-даними марка (char*), число процесорів, обсяг оперативної пам'яті. Реалізувати член-функцію «додавання до кінця списку нового вузла»

12) Вузол списку Вікно з член-даними меню (char *) та число тем меню, наявність інструментальної панелі (bool) та рядка статусу (bool). Реалізувати член-функцію «заміна вузла із заданим порядковим номером на новий

13) Вузол списку з член-даними ПІБ (char*), стать (char), спеціальність (char*), курс навчання. Реалізувати член-функцію заміна вузла списку із заданим порядковим номером на новий»

14) Вузол списку викладач з член-даними ПІБ (char *), стаж роботи, посада (char *), курс (char *), що читається викладачем. Реалізувати член-функцію пошук вузлів списку із заданим прізвищем викладача

**Тема 3. Поняття об'єктів, класів та їх взаємовідносин**

***Анотація:*** *Поняття класів і об'єктів, формат їх опису та складові частини*

***Теми лекцій:*** *«Формати опису класів, приклади створення та визначення, використання заголовкових файлів та файлів реалізації».*

***Самостійна робота здобувача освіти:*** *опрацювати матеріали лекції.*

**Тема 4. Принцип інкапсуляції та регулювання доступу до компонентів класів**

***Анотація:*** *Тема присвячена специфікаторам доступу в класах, сферам видимості та правилам їх використання. Розглядаються опис і визначення атрибутів класів із застосуванням покажчиків і посилань, використання псевдо-змінної this, а також визначення методів класу у тілі класу та поза ним у файлі реалізації.*

***Теми лекцій:*** *«Специфікатори доступу до компонентів класів, сфери видимості та правила їх використання».*

***Теми лабораторних/практичних занять:*** *«опис та визначення атрибутів класів з використанням покажчиків та посилань структур, об'єднань та об'єктів і покажчиків на екземпляри класів». Псевдо-змінна this: призначення та правила і приклади використання. Визначення методів класу у тілі класу та поза ним у файлі реалізації .*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуальних завдання, підготовка до захисту лабораторних робіт.*

***Проектування та розроблення застосунку(лабораторної роботи) згідно з індивідуальними завданнями:***

Завданням лабораторної роботи є розробка такої програми, на вхід якої подається заголовний файл (наприклад, Foo.h, а на виході з'являються також два файли(краще з новими іменами наприклад foo2.h і foo2.cpp), але в них міститься максимально доопрацьований опис класу, причому ці файли повинні безпомилково компілюватися.

На основі опису класу, наданого у заголовному файлі Foo.h застосунок повинен згенерувати:

1) конструктор без параметрів;
2) конструктор з параметрами, що ініціалізує значення атрибутів класу;
3) деструктор;
4) конструктор «глибокого» копіювання;
5) так звані аксесори (accessories) для доступу до атрибутів класу;

У всіх варіантах завдання передбачається неухильне дотримання наступних вимог і умов:

– обов'язковим інтерфейсом програми має бути меню з командами "Відкрити", "Зберегти", "Зберегти як" та "Вихід";звичайно зберігатись повинні як заголовні файли, так і відповідні файли реалізації типу cpp.

– програма повинна забезпечити запитувати імена вхідних файлів, наприклад Foo.h і Foo.cpp, а також ім'я класу, наприклад, CFoo. Рекомендовано використовувати для цього клас CfileDialog, або інші подібні функції Windows. Вимога нежорстка і її можна обійти, запросивши, наприклад, тільки ім'я класу і по ньому визначити імена файлів, а також запитавши ім'я каталогу, в якому вони мають бути розміщені; ім'я каталогу повинно бути коротким і без вживання символів кирилиці та пробілів, наприклад c:/temp.

– вміст вхідних та вихідних файлів повинен бути приблизно таким:

– можна припускати, що ім'я класу знаходиться у файлі слідом за словом class, яке має бути першим у рядку, але не обов'язково у файлі;

– рядки коментарів, що починаються з символів //, можуть розташовуватися в будь-якому місці тіла класу і повинні ігноруватися, але не видалятися із вхідних файлів. Не забороняється, а навіть заохочується використання коментарів вигляду /* оце коментар так коментар!*/;

– можна припускати, що опис член-даних класу починається з рядка, наступного за рядком, що містить слово private, і закінчується рядком, в якому знаходиться слово public або кінець опису класу - фігурна дужка, що супроводжується крапкою з комою;

– ідентифікатори член-даних та член-функцій можуть складатися з довільної кількості символів, причому різних регістрів. Ці імена повинні підкорятися вимогам для ідентифікаторів умовиС++;

– подібним чином можна також припускати, що опис член-функцій класу, у тому числі конструктора та деструктора, починається з рядка, наступного за рядком, що містить слово public, і закінчується рядком, в якому знаходиться слово private або кінець опису класу;

– прототипи опису аксесорів як і будь-яких інших методів класу повинні бути у секції public тіла класу(файл .h ), а реалізації – у файлі .cpp. Аксесори повинні бути реалізовані у всіх варіантах завдань без винятку. Імена аксесорів повинні починатися з префіксів set_і get_;

– член-дані масиви ініціалізувати (у конструкторах) не потрібно, а їх аксесорами має бути перевантажена операція [] (див. клас TVector;

– для член-даних покажчиків у деструкторі класу передбачити звільнення пам'яті з перевіркою значень покажчиків. Деструктор повинен бути реалізований у будь-якому варіанті завдання, де є атрибути- покажчики. Кількість елементів масивів потрібно передавати у конструктор через параметр.

– важливо: якщо у вашому варіанті завдання передбачені, наприклад, атрибути типів char і int, то це означає, що їх може бути довільна кількість;

– вочевидь, що після обробки файлів вашою програмою вони мають компілюватися без помилок.

## Змістовний модуль 2. Технологія візуального проектування та програмування застосунків Windows

### Тема 5. Основи об’єктно-орієнтованого проектування програм з використанням мови UML.

***Анотація:*** *Тема присвячена вивченню UML як засобу аналізу, візуального моделювання та проєктування програмного забезпечення. Розглядаються типи UML-діаграм, нотація Г. Буча та практичні підходи до проєктування й реалізації класів мовою С++ на основі діаграм класів.*

***Теми лекцій:*** *Поняття та призначення UML як мови, що призначена для аналізу, візуального моделювання та проектування програмного забезпечення. Реалізація в UML таких загальних функцій як візуалізація, специфікація, конструювання та документування елементів програмних систем. Типи моделей UML та їх використання, приклади реалізації. Структурні діаграми, діаграми поведінки та взаємодії. Позначення в термінах нотації Г.Буча. проектування і розробка класів мовою С++ шляхом використання діаграм класів на мові UML*

### Тема 6. Основи об’єктно-орієнтованої мови програмування С++/CLI

***Анотація:*** *У темі розглядаються історичні передумови розвитку ООП, особливості платформи .NET Framework та мови С++/CLI. Значна увага приділяється механізмам обробки виключних ситуацій, синтаксису їх ініціювання та створенню власних класів винятків.*

***Теми лекцій:***

*Коротка історична довідка стосовно мов Сімула та Smalltalk. Технологія .NET Framework як найсучасніша концепція розробки програм. Мови об'єктно-орієнтованого програмування платформи .NET Framework. Обробка виняткових ситуацій. Вступ до опрацювання виключних ситуацій. Синтаксис обробки виключних сітуацій. Синтаксис ініціювання виключних сітуацій. Приклади генерації і обробки виключних ситуацій. Власні класи виключних сітуацій*

**Тема 7. Розмежування доступу до компонентів класів і їх ініціалізація**

***Анотація:*** *Тема охоплює принципи розмежування доступу до компонентів класів, призначення та формат опису конструкторів і деструкторів. Розглядаються особливості ініціалізації об'єктів, робота з динамічною пам'яттю та використання покажчика this.*

***Теми лекцій:*** *Призначення і формат опису конструкторів і деструкторів. Особливості виклику конструкторів і деструкторів класів. Розміщення екземплярів класів у «купі(heap)». Огляд CRect як прикладу класу, що має декілька конструкторів, а також інші компоненти. Використання покажчика this.*

**Тема 8. Варіанти розміщення екземплярів класу у пам'яті**

***Анотація:*** *У темі аналізуються способи розміщення об'єктів у глобальній, стековій та динамічній пам'яті. Розглядаються особливості виклику конструкторів і деструкторів залежно від області пам'яті та їх вплив на життєвий цикл об'єктів.*

***Теми лекцій:*** *«Розміщення екземплярів класу у глобальній пам'яті, стеку програми та у динамічній пам'яті». особливості викликів конструкторів та деструкторів в залежності від місця розташування об'єктів класу*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуальних завдань, підготовка до захисту лабораторних робіт.*

**Модульний контроль 1.**

## МОДУЛЬ 2. Успадкування класів і поліморфізм
## Змістовний модуль 3 Успадкування класів

**Тема 9. Визначення похідного класу**

***Анотація:*** *Тема присвячена сутності успадкування, правилам опису та визначення похідних класів, а також впливу специфікаторів доступу на взаємодію з компонентами базового класу. Аналізується роль ключового слова class у механізмах успадкування.*

***Теми лекцій:*** *Сутність успадкування класів формат опису та визначення похідного класу специфікатори доступу до компонентів батьківського класу Особливості доступу похідного класу до компонентів батьківського класу Вплив ключа класу на успадкування*

**Тема 10. Сумісність об'єктів ієрархії класів**

***Анотація****: У темі розглядається поняття сумісності об'єктів у межах ієрархії класів та особливості передачі об'єктів як параметрів глобальних функцій. Аналізуються типи параметрів функцій і їх вплив на коректність взаємодії об'єктів.*

***Теми лекцій:*** *Поняття сумісності об'єктів Передача об`єктів як параметрів глобальних функцій. Тип параметрів глобальних функцій*

**Тема 11. Видимість компонентів класів ієрархії**

***Анотація:*** *Тема присвячена впливу специфікаторів доступу на доступність компонентів базового класу в похідних класах. Розглядаються принципи керування видимістю даних і методів у складних ієрархіях класів.*

***Теми лекцій:*** *Вплив специфікаторів доступу на доступність компонентів батьківського класу у похідному*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

**Тема 12. Віртуальні функції**

***Анотація:*** *У темі розкривається призначення віртуальних функцій на прикладі ієрархії графічних фігур та механізми раннього і пізнього зв'язування. Окрема увага приділяється таблиці віртуальних функцій як основі реалізації поліморфізму.*

***Теми лекцій:*** *Приклад ієрархії класів графічних фігур Поняття та призначення віртуальних функцій формат опису віртуальних функцій, раннє (early binding) та пізнє (late binding) зв'язування функцій Таблиця віртуальних функцій*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

**Тема 13. Віртуальні функції конструктори і деструктори**

***Анотація:*** *Тема охоплює особливості роботи конструкторів і деструкторів у ієрархії класів з віртуальними функціями. Розглядається специфіка їх опису та порядку виклику під час створення і знищення об'єктів.*

***Теми лекцій:*** *Конструктори і деструктори в ієрархії класів*
*специфіка опису та виклику конструкторів в ієрархії класів з віртуальними функціями*

***Самостійна робота здобувача освіти:*** *опрацювати матеріали лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

**Тема 14. Абстрактні класи**

***Анотація:*** *У темі розглядаються абстрактні віртуальні функції, принципи опису абстрактних класів та їх роль як базових елементів ієрархій з підтримкою поліморфізму.*

***Теми лекцій:*** *абстрактні віртуальні функції поняття та формат опису абстрактного класу роль абстрактного класу як базового класу ієрархіїз віртуальними функціями*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

**Тема 15. Дружні функції й класи**

***Анотація:*** *Тема присвячена призначенню дружніх функцій і класів, а також форматам їх опису та визначення. Аналізуються можливості керованого доступу до закритих компонентів класів.*

***Теми лекцій:*** *Призначення дружніх функцій і дружніх класів; формати опису та визначення дружніх функцій і дружніх класів.*

***Самостійна робота здобувача освіти:*** *опрацювання матеріалів лекцій, виконання індивідуального завдання, підготовка до захисту лабораторних робіт.*

**Модульний контроль 2.**

**Практичні заняття** використовуються для групових консультацій по курсовому проекту в семестрі 5.

## 5. Індивідуальні завдання

**Курсовий проект полягає у проєктуванні та розробленні застосунка Windows за індивідуальною темою і написанні пояснювальної записки до нього**

**Мета проекту**: отримання досвіду створення ієрархії класів на основі аналізу об'єктів реального світу і моделювання їх поведінки шляхом розроблення відповідних методів спроектованих класів

**Обсяг** пояснювальної записки – 20-25 сторінок формату А4.

**Таблиця 5.1** Основні етапи та графік виконання курсового проекту

| № | Етап | Тиждень семестру | Форма звіту | % обсягу |
|---|---|---|---|---|
| 1 | Вибір та затвердження теми завдання | 3 | Найменування теми та стислий опис ієрархії класів | 5 |
| 2 | Розробка технічного завдання | 4 | Розділ записки (електронна форма) | 10 |

| | | | | |
|---|---|---|---|---|
| 3 | Проектування програми (ієрархія класів, перелік атрибутів та методів класів) | 5 | Розділ записки (електронна форма) | 20 |
| 4 | Проектування програми (вибір каркасу програми; проектування вікон, меню, панелей інструментів) | 7 | Розділ записки (електронна форма), проект програми | 40 |
| 5 | Кодування програми | 10 | Програма | 60 |
| 6 | Налагодження програми | 11 | Програма | 80 |
| 7 | Оформлення пояснювальної записки та завершення розробки програми | 12 | Пояснювальна записка | 90 |
| 8 | Здача пояснювальної записки та захист проекту (здача програми) | 14 | Записка та програма | 100 |

**Приклади тем курсових робіт, орієнтованих в першу чергу на розробку розгалуженої ієрархії класів:**

1) пілотовані та безпілотні літальні апарати, які не мають двигунів

2) планери парашути повітряні кулі, аеростати, параплани (без двигуна) крила, що закріплюються на людині і т.п.

3) пілотовані літальні апарати з двигунами: літаки гвинтокрили дирижаблі параплани з двигуном парамотори, паральоти, літаючі авто (з крилами),ранцеві двигуни ракетного або гвинтокрильного типу тощо

4) безпілотні ЛА: БПЛА з крилами, коптери тощо

5) ракети: балістичні, сигнальні, крилаті тощо

6) надводні плавзасоби без двигунів: човни, круги дошки баржі тощо

7) надводні плавзасоби з двигунами

8) підводні плавзасоби з двигунами

9) Велосипеди

10) Мотоцикли та ровери з двигунами

11) Самокати

12) Одноколісні транспортні засоби з двигунами, гіроскутери, сегвеї тощо

13) Літальні апарати, що можуть злітати вертикально, тобто без використання підйомної сили крил

14) Автотранспортні засоби

15) Рейковий транспорт: трамваї, паровози, тепловози тощо

16) Гармати.

**Перелік тем курсових робіт, орієнтованих в першу чергу на розробку алгоритмів:**

1) Створення нового проекту на базі існуючого у Visual Studio MVS-2019

2) Інтелектуальний світлофор

3) Перемикає світло на перехресті з урахуванням кількості авто на різних напрямах перехрестя може бути досить складним для руху

4) Суперінтелектуальний світлофор, запам’ятовує обстановку на перехресті і використовує її для оптимізації переключень сигналів світлофорів шляхом прогнозування руху транспорту на основі використання попередньо накопичених даних, тобто такого собі свого досвіду.

5) Розрахунки польоту ракети-перехоплювача рухливої наземної цілі

6) Розрахунки польоту ракети-перехоплювача повітряної цілі та візуалізація польоту ракети та її цілі

7) Моделювання роботи диспетчерської служби залізничного вокзалу

8) Моделювання роботи диспетчерської служби автовокзалу

9) Моделювання роботи диспетчерської служби аеропорту

10) Пошук заданого рельєфу місцевості крилатою ракетою з урахуванням вірогідності розпізнавання рельєфу

11) Розробка алгоритму управління польотом крилатої ракети

12) Розробка алгоритму управління польотом балістичної ракети

13) Індивідуальна тема
14) Планування польотів БПЛА

**Вимоги до функціональності застосунка та оформлення текстів програм**

На початку файлу, що містить опис головної функції main, повинні бути вказані в коментарях:

- ПІБ розробника, номер групи, навчальний рік;
- призначення програми.

Програма не повинна мати ім'я на кшталт Project, 1, ttt, Kurs і т.п. Іншими словами, ім'я програми має бути оригінальним і нести смислове навантаження.

Програма має бути працездатною і не повинна перериватися за будь-яких помилок користувача. Воно, зокрема, не повинно втрачати свою працездатність при простому перенесенні файлу, що виконується, в інший каталог.

Файли з текстами програм повинні мати коментарі. Для кожної функції, у тому числі методів класів, має бути вказано її призначення та наведено опис параметрів. Для атрибутів функцій класів також має бути зазначено їх призначення.

Оператори програми повинні бути виділені відступами та порожніми рядками для пояснення алгоритму та забезпечення наочності коду.

Імена файлів, з якими працює програма, повинні запитуватись у користувача, а не бути «зашитими» в ній як константи. Слід також уникати використання у програмі «магічних чисел», наприклад:

```
int Vec[100];
for(int i=0;i<100;i++) Vec[i]=0;
```

У цьому прикладі слід замінити 100 іменованою константою, наприклад, const int VecSize=100; і далі використовувати лише її.

Усі дії користувача – команди або відповіді на запити програми – повинні контролюватись так, щоб під час введення будь-яких символів або натискання будь-яких клавіш програма не завершувалася аварійно. Для обробки хибних станів потрібно використовувати механізм обробки виняткових ситуацій (ВС).

Програма повинна, в обов'язковому порядку, мати і використовувати свою ієрархію класів, що включає принаймні три родинні класи. Класи ієрархії повинні мати конструктори, деструктори, компонентні дані та компонентні функції, у тому числі віртуальні, делегати (delegate C++/CLI). Зовнішня програма не може мати безпосереднього доступу до даних класу (їх треба оголосити особистими – private або захищеними – protected), а використовуватиме цієї мети властивості (property).

Програма повинна дозволяти завантажувати, створювати, видаляти, редагувати та зберігати довільну кількість об'єктів класів ієрархії. Під час читання даних із текстового файлу (файлів) передбачити контроль їхньої коректності за допомогою механізму виключених ситуацій (ВС). Виведення повідомлень про помилки у файлі має бути інформативним з точною вказівкою місця та причини помилки. Збереження членів-даних об'єктів, так само як і їхнє завантаження, повинно виконуватися в текстовому або двійковому файлах. Природно, програма має дозволяти користувачеві переглядати дані об'єктів у зручному вигляді, легко та невимушено.

Також програма має бути забезпечена файлом довідки, який найпростіше заповнити вмістом розділу записки «Посібник користувача». Формат файлу довідки повинен бути одним з наступних: .chm, .html або, можливо, іншим за погодженням з викладачем. У будь-якому випадку файл довідки обов’язково повинен мати зміст.

**Вимоги до оформлення пояснювальної записки**

Склад записки має бути таким:

- Титульна сторінка
- Зміст
- Введення

- Технічне завдання
- Проектування програми
- Керівництво користувача
- Керівництво програміста
- Висновки
- Перелік посилань

**Проєктування програми.**

На цьому етапі належить:

- опрацювати ієрархію класів, тобто. уточнити перелік компонентів – даних та функцій – для кожного класу ієрархії;
- крім ієрархії класів, доцільно використовувати контейнерні класи
- визначити, які функції мають бути віртуальними, перевантаженими, компонентними чи зовнішніми(глобальними);
- спроектувати застосунок, тобто. визначитися остаточно з каркасом програми, командами меню, панелями інструментів тощо;
- для всіх член-функцій класів треба навести їх призначення та описати параметри

На цьому етапі необхідно також вирішити задачу подання даних користувачеві в максимально зручному для нього вигляді.

**Кодування програми.**

На цьому етапі треба:

- створити проект програми відповідно до обраного каркасу застосунку Windows;
- створити файли із розробленими вами класами;
- написати програмний код, що «оживить» застосунок, тобто. створюватиме спроектовані об'єкти та відображатиме їхні дані на моніторі.

**Оформлення пояснювальної записки та завершення** розробки програми

На цьому етапі необхідно завершити оформлення пояснювальної записки відповідно до вимог, викладених у розділі «Вимоги до оформлення пояснювальної записки».

**Здача пояснювальної записки та програми**

На цьому заключному етапі необхідно здати пояснювальну записку викладачеві на перевірку, при необхідності виправити зауваження.

## 6. Методи навчання

Лекції проводяться з використанням демонстрації окремих прийомів роботи в середовищі обговорюваних програмних середовищ.

Лабораторні роботи виконуються з використанням ліцензійних зразків програмного забезпечення.

Самостійна робота включає підготовку до лабораторних робіт, модульного контролю та іспиту, виконання поза аудиторної частини індивідуального завдання з використанням навчально-методичної літератури та документації до програмного забезпечення.

## 7. Методи контролю

Контроль здійснюється згідно з «Положення про рейтингове оцінювання досягнень студентів».

Поточний контроль – відповідно до повноти, якості та своєчасності виконання лабораторних робіт та розділів курсового проекту; проміжний (модульний) контроль – письмові контрольні роботи на 8-му та 16-му тижнях; підсумковий контроль - у вигляді письмового іспиту та диференційного заліку на основі захисту курсового проекту .

## 8. Критерії оцінювання та розподіл балів, які отримують студенти

Таблиця 8.1 – Розподіл балів, які отримують здобувачі освіти

| Складові навчальної роботи | Бали за одне заняття (завдання) | Кількість занять (завдань) | Сумарна кількість балів |
|---|---|---|---|
| Модульні контрольні роботи | 0–25 | 2 | 0–50 |
| Лабораторні/практичні роботи | 0–5 | 10 | 0–50 |
| | | **Усього за семестр** | 0–100 |

Семестровий контроль (іспит/залік) проводиться у разі відмови студента від балів поточного тестування та за наявності допуску до іспиту/заліку. При складанні семестрового іспиту/заліку студент має можливість отримати максимум 100 балів.

Білет для іспиту складається з 2 теоретичних запитань , по 50 балів за кожне.

Приклади запитань

1. У чому полягає специфічність конструкторів і деструктора класу як його методів
2. Що таке глибоке копіювання і коли його треба реалізовувати
3. Назвіть і поясніть види поліморфізму
4. Яка різниця між абстрактним класом і інтерфейсом, якщо вона є.

Таблиця 8.2 – Розподіл балів, які отримують здобувачі освіти за виконання курсової роботи (проекту).

| Пояснювальна записка | Застосунок | Захист роботи | Сума |
|---|---|---|---|
| 0…25 | 0…25 | 0…50 | 100 |

Таблиця 8.3 – Шкали оцінювання: бальна і традиційна

| Сума балів | Оцінка за традиційною шкалою | |
|---|---|---|
| | Іспит, диференційний залік | Залік |
| 90 – 100 | Відмінно | Зараховано |
| 75 – 89 | Добре | |
| 60 – 74 | Задовільно | |
| 0 – 59 | Незадовільно | Не зараховано |

**Критерії оцінювання роботи здобувача протягом семестру**

**Задовільно (60-74).** Знати основні положення теоретичного матеріалу. Вміти користуватися засобами інформаційної підтримки середовищ розробки програм Microsoft Visual Studio QT та інших. Вміти вирішувати задачі в з використанням засобів візуального програмування пов'язаних з розробкою застосунків та динамічних бібліотек DLL. Вміти розробляти програмні додатки з використанням необхідних засобів SDK

**Добре (75-89).** Знати основний теоретичний матеріал в повному обсязі. Володіти технологією пошуку довідкової літератури. Вміти вирішувати задачі. Вміти розробляти програмні додатки в середовищах на основі сучасних інструментальних засобів програмування. Вміти встановлювати і налаштовувати середовища IDE та . підтримувати їх працездатність

**Відмінно (90-100).** Знати основний і додатковий теоретичний матеріал в повному обсязі. Орієнтуватися в довідковій літературі.. Вміти встановлювати IDE і адаптувати її до власних потреб . Вміти розширювати можливості IDE за рахунок власних програмних додатків.

## 9. Політика навчального курсу

**Відвідування занять.** Регуляція пропусків. Інтерактивний характер курсу передбачає обов’язкове відвідування практичних занять. Здобувачі освіти, які за певних обставин не можуть відвідувати практичні заняття регулярно, повинні протягом тижня узгодити із викладачем графік

індивідуального відпрацювання пропущених занять. Окремі пропущені заняття мають бути відпрацьовані на найближчій консультації протягом тижня після їх пропуску. Відпрацювання занять здійснюється усно у формі співбесіди за питаннями, визначеними планом заняття. В окремих випадках дозволяється письмове відпрацювання пропущених занять шляхом виконання індивідуального письмового завдання.

**Дотримання вимог академічної доброчесності** здобувачами освіти під час вивчення навчальної дисципліни. Під час вивчення навчальної дисципліни здобувачі освіти мають дотримуватися загальноприйнятих морально-етичних норм і правил поведінки, вимог академічної доброчесності, передбачених Положенням про академічну доброчесність Національного аерокосмічного університету «Харківський авіаційний інститут» (https://khai.edu/assets/files/polozhennya/polozhennya-pro-akademichnu-dobrochesnist.pdf).
Очікується, що роботи здобувачів освіти будуть їх оригінальними дослідженнями або міркуваннями. Відсутність посилань на використані джерела, фабрикування джерел, списування, втручання в роботу інших здобувачів освіти становлять, але не обмежують, приклади можливої академічної недоброчесності. Виявлення ознак академічної недоброчесності в письмовій роботі здобувача освіти є підставою для її незарахування викладачем незалежно від масштабів плагіату чи обману.

**Вирішення конфліктів.** Порядок і процедури врегулювання конфліктів, пов'язаних із корупційними діями, зіткненням інтересів, різними формами дискримінації, сексуальними домаганнями, міжособистісними стосунками та іншими ситуаціями, що можуть виникнути під час навчання, а також правила етичної поведінки регламентуються Кодексом етичної поведінки в Національному аерокосмічному університеті «Харківський авіаційний інститут» (https://khai.edu/ua/university/normativna-baza/ustanovchi-dokumenti/kodeks-etichnoi-povedinki/).

## 10. Методичне забезпечення

Навчально-методичний комплекс дисципліни у електронному вигляді знаходиться на сайті дистанційної освіти ХАІ «Ментор»:

- силабус дисципліни;
- конспект лекцій, підручники (навчальні посібники), в тому числі в електронному вигляді, які за змістом повністю відповідають робочій програмі дисципліни;
- методичні вказівки та рекомендації для виконання курсових робіт та проектів, розрахункових та розрахунково-графічних робіт, лабораторних та практичних робіт, а також рекомендації для самостійної підготовки;
- тематики індивідуальних завдань;
- приклади розв'язування типових задач чи виконання типових завдань;
- питання, тести для контрольних заходів;
- каталоги інформаційних ресурсів.

## 11. Рекомендована література

### 11.1 Базова

1. Овсяннік, В. М. Мова С++ не для чайників [Електронний ресурс] : навч. посіб./ В. М. Овсяннік**,** О. К. Погудіна. – Харків : Нац. аерокосм. ун-т ім. М. Є. Жуковського «Харків. авіац. ін-т», 2020. – 130 с.

2. Основи програмування [Текст] : навч. посіб. / О. К. Погудіна, В. М. Овсяннік, В. І. Калашнікова, А. В. Погудін. – Харків: Нац. аерокосм. ун-т ім. М. Є. Жуковського «Харків. авіац. Ін-т», 2021. - 116 с.

3. Основи програмування [Електронний ресурс] : Методичні рекомендації до виконання лабораторних робіт О. К. Погудіна, В. М. Овсяннік, М. О. Бичок, А. В. Погудін – Харків : Нац. аерокосм. ун-т ім. М. Є. Жуковського «Харків. авіац. ін-т», 2021. – 73 с.

4. В.В. Бублик Об’єктно-орієнтоване програмування: [Підручник] / В.В. Бублик. – К.: ІТ-книга, 2015. – 624 с.: іл.

**11.2Допоміжна**

1. Bjarne StroustrupC++Programming Language FourthEdition 2013. 1366c.

2. Веклич Р. А. Вступ до програмування мовою С++. Структури даних: навч. посіб. / Р. А. Веклич, Т. О. Карнаух, А. Б. Ставровський – К. : ВПЦ*"Київський університет", 2018. – 99 с*

*3.* ДСТУ 2938-94. Системи оброблення інформації. Основні поняття. Терміни та визначення. – К. : Держстандарт України, 1995. – 32 с.

4. Learning C++cplusplus. Pdf844.pdf

5. Кравець П.О. Об’єктно-орієнтоване програмування: навч. посібник/ П.О. Кравець. – Львів: Видавництво Львівської політехніки, 2012. – 624с.

## 12. Інформаційні ресурси

1. C++/CLI Programing [Електронний ресурс]. — Режим доступу: http://www.functionx.com/cppcli