Вступ

Однією із важливих задач підготовки спеціалістів за напрямом "Комп'ютерні науки" та "Системний аналіз" є забезпечення високого рівня теоретичних знань і практичних навичок проектування та експлуатації великих інформаційних і програмних систем. Сучасні програмні системи є складними для розроблення, супроводження та використання. Для забезпечення чіткої структурної організації, здатності до розширення та семантичної визначеності вони будуються на основі парадигми об'єктно-орієнтованого програмування.

Фахівці у галузі комп'ютерних інформаційних технологій повинні володіти сучасними методами та засобами об'єктно-орієнтованого аналізу та проектування програмних систем різного призначення.

"Об'єктно-орієнтоване програмування" належить до нормативних дисциплін циклу професійної та практичної підготовки студентів базових напрямів підготовки 6.050101 "Комп'ютерні науки", 6.040303 "Системний аналіз" та ін.

Об'єктно-орієнтоване програмування (ООП) – це технологія програмування, яка ґрунтується на понятті класів, об'єктів та успадкуванні елементів базових класів похідними класами. Клас визначає абстрактний тип даних, семантика якого обумовлена проблематикою розв'язуваної задачі, а об'єкт є екземпляром або значенням з типом класу.

Необхідність становлення ООП обумовлена кризою розвитку або бар'єром можливостей технології процедурного програмування, що виникли у середині 60-х років XX століття. Причиною кризи стало значне зростання розміру та складності програмних систем. При використанні процедурної технології ускладнюється структура програми та можливість її модифікації у зв'язку із зростанням числа глобальних змінних, функцій та зв'язків між ними. Крім цього, процедурний підхід розділяє дані та код для їх опрацювання, що не відповідає картині реального світу, який складається з різноманітних об'єктів, які одночасно характеризуються властивостями (даними) та поведінкою (діями). Логічне об'єднання даних та операцій над ними є головною ідеєю об'єктно-орієнтованої методології, яка забезпечила подолання цих труднощів.

Концепція об'єктного підходу не нова. Дослідження у галузі психології довели, що мислення людини в процесі діяльності оперує поняттями на рівні об'єктів та зміни їх властивостей.

Об'єктно-орієнтована методологія орієнтована на колективне розроблення великих програмних систем групою програмістів і складається з об'єктно-орієнтованого аналізу, проектування та програмування.

Об'єктно-орієнтований аналіз на основі декомпозиції системи виявляє концептуальні сутності проблемної області для розуміння і пояснення того, як вони взаємодіють між собою.

Об'єктно-орієнтоване проектування полягає у формуванні класів, об'єктів та відношень між ними на основі атрибутів (характеристик) та операцій (дій) виділених концептуальних сутностей.

Наприклад, для розв'язання задач освітньої галузі створюються об'єкти Навчальний план, Викладач, Студент, Розклад занять, Аудиторний фонд тощо, встановлюються відношення між ними, розробляються моделі та алгоритми функціонування згідно із заданими критеріями.

Об'єктно-орієнтоване проектування ґрунтується на трьох принципах:

- інкапсуляція даних та функцій (методів) їх опрацювання в єдиній інформаційній структурі, яка називається класом;
- успадкування класів – імпорт властивостей базових класів у похідні класи;
- поліморфізм, який полягає у використанні однакових інтерфейсів для роботи з різними за функціональністю об'єктами.

ООП перетворює розроблений проект на програмний продукт, використовуючи засоби об'єктно-орієнтованої мови програмування. Для цього здійснюється оголошення необхідних класів, їх успадкувань, визначення об'єктів та методів роботи з ними. Так, програмна реалізація динамічного поліморфізму використовує механізм пізнього зв'язування, який основується на перевизначенні віртуальних функцій в ієрархії успадкування класів та пов'язуванні конкретної реалізації цих функцій з об'єктами під час виконання програми.

ООП ґрунтується на пріоритеті даних над кодом, на передаванні повідомлень об'єктам за допомогою викликів відповідних функцій. На відміну від традиційного процедурного програмування, в ООП не код керує даними, а дані керують кодом програми.

Для роботи з об'єктами використовуються об'єктно-орієнтовані мови (ООМ) програмування. Історично першою мовою ООП є Simula, розроблена у 1967 р. Закладена у Simula об'єктна концепція була розвинута у мовах ООП: Smalltalk, Objective C, Eiffel, C++, CLOS, Ada, Oberon, Object Pascal, Visual Basic, Java, PHP, Perl, Python, C# (C Sharp), Ruby, Action Script 3 та ін.

ООМ підтримують абстракцію даних та усі наведені вище принципи ООП. Їх поділяють на дві групи: чисті та гібридні.

Чисті ООМ, до яких належать Smalltalk, Objective C, Eiffel, Java, C#, Ruby, Action Script 3, у повній мірі підтримують концепцію ООП. У цих мовах усе розглядається як елементи певних класів і у програмі не можна оперувати з глобальними даними або функціями, що не належать класам.

Гібридні ООМ, такі як Object Pascal, C++, Perl, є більш універсальними, оскільки реалізують засоби процедурного та об'єктно-орієнтованого програмування. Процедурне програмування використовується для економного використання ресурсів комп'ютера та побудови швидкодіючих програм, однак

вимагає більших затрат часу для створення проекту програми. ООП забезпечує зменшення часу розроблення та кращу керованість програмного проекту, особливо при його колективній розробці, необхідному супроводі та модифікації при експлуатації. Необґрунтноване поєднання обох засобів в одній програмі може призвести до зменшення надійності її роботи.

Застосування ООП призначено для повторного використання та для зменшення довжини коду програми і часу її проектування за рахунок розбиття задачі на складові частини – об'єкти, побудови схеми їх підпорядкування, взаємодії та повторного використання. Крім того, засоби ООП дають можливість будувати відкриті програмні системи, здатні до розширення коду та вдосконалення функцій.

Метою дисципліни "Об'єктно-орієнтоване програмування" є комплексне засвоєння студентами сучасних об'єктно-орієнтованих мов та технологій програмування. Для вивчення об'єктно-орієнтованого програмування вибрано мову C++, яка започаткувала ряд споріднених мов, таких як Java, C#, PHP та інші, які ґрунтуються на подібних синтаксичних конструкціях та засобах програмування. Сьогодні більшу частину розробленого у світі програмного забезпечення написано цими мовами. Освоєння об'єктно-орієнтованого програмування базовою мовою C++ забезпечить розуміння споріднених мов та за потреби зменшить час освоєння практичного програмування цими мовами.

До складу дисципліни "Об'єктно-орієнтоване програмування" входять: курс лекцій, практичних, лабораторних занять та виконання розрахункової роботи з побудови та експлуатації об'єктно-орієнтованих програмних систем.

Отримані теоретичні відомості допоможуть розробляти та використовувати на практиці об'єктно-орієнтовані програмні продукти та засоби автоматизації програмування. Навички застосування інструментальних засобів об'єктно-орієнтованого програмування дають можливість за короткий час будувати оптимізовані, швидкодіючі, відкриті програмні системи, придатні до вдосконалення та нарощування функціональності.

У результаті вивчення дисципліни "Об'єктно-орієнтоване програмування" фахівець повинен *знати*:

- основи об'єктно-орієнтованого аналізу та проектування програмних систем;
- засоби мови C++ для розроблення об'єктно-орієнтованих програм.

Підготовлений фахівець повинен *вміти*:

- розробляти програмні системи засобами об'єктно-орієнтованого програмування мовою C++;
- відлагоджувати та реалізовувати програми в одному із середовищ програмування "Visual C++", "Borland C++", "Borland Builder C++" або в іншому.

Для вивчення об'єктно-орієнтованого програмування мовою C++ необхідні базові знання програмування мовою C.

Цей навчальний посібник сформовано на основі авторських матеріалів лекцій та джерел, вказаних у списку літератури. У посібнику розглядаються приклади тільки консольних програм Windows. Це спрощує виклад предмета, дає змогу зосередитися на суті розглянутих питань, позбавляючись від великої кількості коду порівняно з використанням графічних інтерфейсів користувача. У всіх прикладах програм необхідно вважати, що підключено файл `<iostream>` для організації потокового введення–виведення та використано простір імен `std`. Результати виконання програм можуть залежати від виду та версії компілятора C++. Усі наведені у посібнику повні програми перевірено компілятором Visual C++.

Виконуючи тестові завдання самоконтролю, насамперед необхідно уважно проаналізувати наведені приклади програм та визначити результат їх роботи або причину помилки. У кожному завданні правильним є тільки один варіант відповіді. Якщо вибраний варіант не збігається з правильним, наведеним у кінці посібника, то рекомендується виконати програму на комп'ютері і повернутися до її аналізу.

Розділи посібника можна читати у різній послідовності залежно від початкового рівня підготовки. Для початкового вивчення можна скористатися наведеною нижче схемою залежностей, де числами позначено номери розділів посібника. На схемі не зображено прямих зв'язків між розділами за наявності опосередкованих.

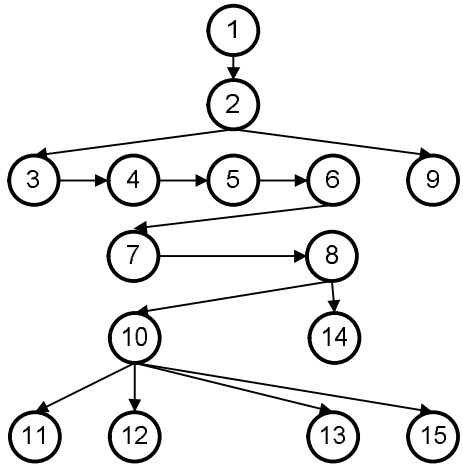

Схема залежностей розділів посібника