Передмова

Цей підручник виник як підсумок викладання автором об'єктно-орієнтованого програмування студентам-інформатикам Київського національного університету імені Тараса Шевченка і Національного університету «Києво-Могилянська академія» протягом останніх десяти років. Курс розрахований на 4–6 аудиторних годин на тиждень протягом року та значний обсяг самостійної роботи зі створення, реалізації та випробувань програмних проектів. Практика багаторічного спілкування зі студентами дала можливість сконцентруватися на складніших для розуміння конструкціях, підібрати прозоріші приклади.

Слухачі мали певний власний досвід програмування, здебільшого на Паскалі, а останнім часом мовою Java. Тому ми не зупинятимемося на традиційних проблемах початкового курсу програмування, а виходитимемо з того, що традиційні структури керування й структури даних високого рівня добре відомі, так само як техніка програмування рекурентних співвідношень, індуктивних функцій, дослідження інваріантів тощо. Читачеві мають бути відомі також засоби алгоритмічної декомпозиції, наприклад, на рівні розробки алгоритмів сортування й пошуку, обробки дерев, списків та інших структур даних. Для розуміння курсу цілком достатньо підготовки в обсязі книги Н. Вірта [1], на жаль, незаслужено забутої.

Головна мета курсу – оволодіння технікою об'єктно-орієнтованого програмування на основі мови програмування C++. Ця мова багатопарадигменна, тому її вивчення служить доброю базою для порівняльного аналізу парадигм імперативного програмування. Крім цього, мова C++ базується на мові програмування C. Залишаючись мовою програмування високого рівня, мова C дотримується концепції, вираженої лозунгом повернення до апаратури (*back to hardware*). Остання обставина дозволяє зосередити увагу на важливих питаннях нижчого рівня: роботою з машинними структурами даних (бітами, байтами, машинними словами), указниками, машинною арифметикою, програмованим розподілом пам'яті тощо. Отже, обрана нами мова органічно поєднує підходи нижчого й вищого рівнів і при цьому відповідає тенденції до поглиблення типізації. Саме програмування на C++ найповніше відповідає влучній характеристиці умінь програміста, даній академіком А.П.Єршовим: «Програміст повинен володіти здібністю першокласного математика до абстракції і логічного мислення у поєднанні з талантом Едісона споруджувати все, що

завгодно, з нуля і одиниці. Він повинен поєднувати акуратність бухгалтера з прозорливістю розвідника, фантазію автора детективних романів з тверезою практичністю економіста».

В центрі об'єктно-орієнтованої парадигми перебуває поняття типу, або класу, яке поєднує в собі чотири найважливіші властивості об'єктної моделі: абстракцію, інкапсуляцію, модульність та ієрархію. Кожна окрема властивість притаманна й іншим парадигмам. Так ми маємо абстракцію процедур, інкапсуляцію даних у структурі, ієрархію підструктур, модулі з процедур і даних в Object Pascal або Delphi. Для повноцінної реалізації об'єктно-орієнтованої парадигми залишається тільки доповнити ієрархію успадкуванням, відділити успадкування реалізацій від успадкувань типів, запровадити можливість налаштування поведінки за допомогою динамічного поліморфізму й ввести узагальнене поняття абстрактного класу або інтерфейсу, керуючись тезою Е.Дейкстри про те, що абстрактність не зводиться до невизначеності, а полягає в створенні нового абсолютно прозорого семантичного рівня.

Увага в курсі об'єктно-орієнтованого програмування концентрується не на складних алгоритмах – на них просто не вистачає часу – а на побудові обґрунтованих ієрархій, складанні абстрактних інтерфейсів та універсальних сигнатур методів. Ідеал об'єктно-орієнтованого програмування – «програми-пристосуванці» (*programming for reuse*), придатні до налагодження та функціонування за нових умов лише завдяки дописуванню, але не переписуванню коду. В літературі з програмування часом наводять приклад фрази, яку вважають повною протилежністю алгоритму: «Піди туди, не знаю куди, за тим, не знаю за чим». Але ж ця фраза – прекрасний приклад абстрактного інтерфейсу, не закованого в пута реалізації. Цей інтерфейс має справу з тими, які ходять і приносять, а куди ходять і що приносять – на цьому етапі неважливо. Це деталі, для з'ясування яких пізніше залишається достатньо простору. Наскільки було б гірше, якби довелося виправляти наслідки походу не туди, куди треба, з доставкою непотрібного або навіть шкідливого чи небезпечного. Сьогодні необхідно вчитися програмувати, заглядаючи у майбутнє, розробляти гнучкі програми, придатні до нових, можливо, навіть несподіваних застосувань. Образно кажучи, парадигму *«reuse»* можна сформулювати, як спробу відкласти на завтра все, чого можна не уточнювати сьогодні.

Підручник з об'єктно-орієнтованого програмування не замінить посібників з мови програмування C++. Він допоможе вивчати матеріал у правильній послідовності, виділити найважливіші з погляду розробки програмного забезпечення концепції, опрацювати змістовні приклади. Пе-

редбачено, що деталі синтаксису та семантики мовних конструкцій читачі вивчатимуть самостійно за широко доступною літературою, наприклад, за книгами авторів С [2] та С++ [3]. Мова програмування С++ знаходиться у постійному розвитку, ретроспективу якого можна прослідкувати за книгою Бьярне Страуструпа [4], а з її новітнім станом можна ознайомитися, вивчаючи [5]. Однак автор глибоко переконаний у перспективності навчання методам, а не мовам програмування. Кожна конструкція вивчається й аналізується з погляду доцільності та обґрунтованості її застосування в розробленні програм. Протягом усього курсу явна перевага віддається принципам структурованого програмування [6], а при написанні коду рекомендуємо керуватися міркуваннями, викладеними Стівом Макконнеллом [7].

Для подальшого вивчення можна рекомендувати поглиблений курс методів програмування з науковим семінаром, де увага зосереджуватиметься на рекомендаціях з ефективного програмування Скотта Мейерса [8, 9] , задачах і стандартах програмування Герба Саттера й Андрея Александреску [10, 11, 12], стратегіях проектування Александреску [13], взірцях проектування (*design pattern*) «банди чотирьох» [14] та особливостях нового стандарту С++ [5]. Дуже корисно організувати практикум групової розробки програм для практичного засвоєння набутих знань (*capstone project* в термінології відомих рекомендацій ACM до змісту навчальних програм з інформатики [15]).

Серйозною виявилася проблема української термінології. У програмуванні вона, на жаль, ще не склалася остаточно. Доступні книги російськомовні, термінологія російською мовою загалом склалася, хоча теж не вільна від критики. Деякі терміни, як наприклад, *framework, toolkit, design pattern* ще чекають на влучні переклади, інші перекладають дослівно навіть у тих випадках, коли можна вживати більш звичні слова, наприклад, *довизначення* або *розширення* замість «перегрузки функций». Значну допомогу в перекладі термінів надають англо-українські та російсько-українські словники з інформатики та обчислювальної техніки, зокрема підготовлені в Національному університеті ім. Тараса Шевченка [28], [29]. Але самі вони ще не встигли остаточно визначити мовну норму, а радше розпочали експеримент, відтак проблема термінології ще чекає на своє вирішення. Підбираючи власний варіант перекладу терміну, я керувався здоровим глуздом, власним досвідом та академічними виданнями словників української мови.

Робота над курсами програмування і підручником, зокрема, завдячує постійній підтримці колег і, головне, інтересу і активності студентів, те-

перішніх і колищніх, частина яких вже самі стали колегами автора. Потрібно згадати професора М.М.Глибовця, завдяки чиїй наполегливості цей підручник побачить світ, Ігоря Завадського і Олександра Левченка за їх активну співпрацю над рукописом, Богдана Троценка, Антона Шабінського, Володимира Ляшка, Анну Дьоміну, Назима Сітманбетова, Данила Фітеля, Андрія Давиденка, Андрія Чайку, Василя Куцика, Дмитра Зважія та багатьох інших за допомогу, стимулюючі питання, активність на форумах курсу, і, головне, за підтримку в автора впевненості у потрібності розпочатої справи.

Особливої вдячності заслуговує компанія Інфопульс за фінансування видання підручника, а також постійну допомогу факультету інформатики Києво-Могилянської академії.

Автор буде глибоко вдячний кожному, хто висловить свої зауваження до підручника, їх можна надіслати на адресу **boublik@ukma.kiev.ua**.

Веб-підтримка курсу

http://distedu.ukma.kiev.ua – портал електронного навчання, що супроводжує викладання курсу в Києво-Могилянській академії. Крім навчальних матеріалів, тут містяться вправи та задачі для самостійного розв'язування, завдання для програмних проектів, а також форуми для обговорення проблемних питань. Портал надає гостьовий доступ всім зацікавленим в ознайомленні з практичною частиною електронного курсу. Курс розміщено в розділі «другий рік навчання».

http://oop.in.ua – портал з об'єктно-орієнтованого програмування, створений силами студентів-ентузіастів факультету інформатики Києво-Могилянської академії. Портал супроводжує роботу гуртка любителів програмування, містить презентації студентських доповідей, добірки цікавих матеріалів, розв'язання задач та багато іншого.

http://itknyga.com.ua – сайт видавництва. Тут ви можете замовити книжки, завантажити презентації для лекторів, інтерактивні навчальні модулі до окремих тем курсу, а також отримати доступ до онлайнового навчального середовища курсу для вашого закладу.