УДК 331; 004.42

АКТУАЛЬНІ ПИТАННЯ ТРАНСФОРМАЦІЇ ВИКЛАДАННЯ ПРОГРАМУВАННЯ ДЛЯ ЕКОНОМІЧНИХ СПЕЦІАЛЬНОСТЕЙ

КОЛДОВСЬКИЙ В. В.

кандидат економічних наук

Суми

Програмування – дисципліна, яка займає важливе місце у підготовці фахівців не лише технічних спеціальностей, а й деяких економічних, таких як, наприклад, «Економічна кібернетика». Із поширенням інформаційних технологій (ІТ) роль дисципліни тільки зростає, сама вона змінюється в основному у напрямку ускладнення, що не сприяє її опануванню студентами. Водночас відбулися трансформації у сфері застосування програмного забезпечення (ПЗ), які недостатньо добре відображаються у традиційних підходах до викладання програмування. Від автоматизації наукових розрахунків і роботи підприємств ПЗ поступово перемістилося практично у всі сфери людського життя і зайняло місце як на робочому просторі будь-якого інтелектуального працівника, так і в особистому багатьох людей, навіть не пов'язаних з ІТ професійними навиками, за рахунок значного збільшення доступності традиційних ПК, а також «розумних» пристроїв таких як смартфони, планшети, нетбуки та ін. Відповідно змінилося і саме ПЗ: його внутрішня будова у більшості випадків стала простішою, ніж наукоємні розробки минулих років, а на перший план виходить дизайн, зручність використання, інтуїтивність інтерфейсу, простота та інші якості, яким у минулому приділялося достатньо мало уваги при викладанні відповідних дисциплін.

Таким чином, виникає актуальна проблема привести процес викладання програмування у відповідність із сучасними вимогами економіки і стану науки і техніки у даній галузі.

Питанням вдосконалення викладання програмування у вищій школі присвячена значна кількість досліджень, проведених як вітчизняними, так і зарубіжними вченими. Зокрема, вітчизняні дослідники З. С. Сейдаметова і Ф. В. Шкарбан стверджують, що останнім часом спостерігається зменшення зацікавленості серед вступників вузів до спеціальностей, які передбачають вивчення програмування [1], незважаючи на те, що в цілому потреба у подібних фахівцях останнім часом лише зростає. Західні фахівці Moskal B., Lurie D. та Cooper S. наводять статистичні дані, згідно з якими зацікавленість серед студентів, які мають можливість обирати дисципліни до подальшого вивчення, у продовженні вивчення програмування за умови, що їх перед цим навчали на основі традиційних академічних підходів, є достатньо низкою і становить близько 15% [2].

Виходячи із вищезазначеного, вважаємо за доцільне розглянути існуючі проблеми, які виникають у процесі викладання програмування традиційними методами, і запропонувати деякі підходи стосовно їх вирішення.

На наш погляд, у процесі викладання програмування слід приділити увагу вирішенню таких завдань:

- привернути увагу до дисципліни, зробити її цікавою;
- спростити процес викладання, зробити дисципліну більш зрозумілою;
- вчитися на реальних задачах, зробити дисципліну більш практичною;
- вчити студентів добрим практикам програмування, зробити дисципліну більш корисну для подальшого працевлаштування випускників.

Як показує досвід, вирішення першої задачі є найважливішим і найскладнішим одночасно. Цікавість до дисципліни знижується одночасно із тим, як студенти знайомляться із тією колосальною відмінністю, що являє собою ПЗ «ззовні» і яким чином воно побудоване «зсередини». Бажання програмувати часто виникає внаслідок того, що студенти отримують натхнення від

ПЗ, з яким вони стикаються особисто, однак на практиці своєрідним «холодним душем» для них є знайомство з основами алгоритмів, структур даних, синтаксису і семантики мов програмування. На нашу думку, значною мірою проблема полягає в «нашаруванні» процесу викладання дисципліни аналогічно тому, як аналогічній процес відбувався у процесі розвитку програмування взагалі: низькорівневе програмування замінялося високорівненим, далі отримав розповсюдження структурний підхід, який пізніше змінився на об'єктно-орієнтований. У результаті сама по собі дисципліна виявляється достатньо складною для опанування, якщо вона представлена у подібному вигляді. При цьому той факт, що на практиці кожна наступна парадигма програмування призводила до суттєвого полегшення у роботі програмістів, студентам полегшення не приносить, оскільки вони знаходяться у ситуації перенавантаження великим обсягом інформації, оцінити корисність якої як для практичного використання, так і як теоретичний фундамент для подальшого опанування професії, самостійно не в змозі. Перехід від класичного академічного підходу у викладанні дисципліни до спрощених і більш наглядних методів, особливо на початкових етапах навчання, має приводити до зростання зацікавленості студентів у дисципліні. Як свідчать результати досліджень, використання систем навчання програмування на основі сценарного підходу, зокрема відкритого проекту «Аліса» (alice.org), дозволяє збільшити інтерес до дисципліни з 15% до 88% [2]. Крім згаданого, існують інші проекти зі спрощеного викладання програмування, доступні для використання у вищій школі [3].

Суттєве спрощення дисципліни є, на наш погляд, особливо важливою задачею. Із розвитком ІТ і зростання можливостей обчислювальних машин і, відповідно, мов і середовищ програмування, програмування в цілому значно спростилося, наприклад, мови програмування з неявним керуванням пам'яттю, такі як Java і C# суттєво полегшили ООП у порівнянні з мовами програмування попередньої генерації. Водночас, як зазначалося раніше, у вузах викладання програмування традиційно відбувалося не у напрямку спрощення і відмови від застарілих підходів, а у напрямку «нашарування» нових рішень на старі, що лише ускладнює опанування дисципліни студентами. При цьому слід зазначити, що окремі застарілі технології не мають жодної практичної цінності, потреба у їх вивченні може диктуватися лише з точки зору академічної цікавості, що, звичайно, може мати право на існування, однак не за рахунок більш глибокого вивчення тем, які особливо актуальні у сучасній практиці програмування.

Так звана «академічність», на наш погляд, є істотною перепоною на шляху до формування стійких практичних навичок у студентів. Зокрема, задачі, які підкріплюють вивчення теоретичних розділів дисципліни, традиційно підбираються без урахування їхньої кінцевої практичної цінності. Крім того, розробка навчальних програм з дисциплін, як правило, здійснюється таким чином, щоб охопити максимальну кількість тем, причому до уваги не сприймається той факт, що окремі теми є особливо важливими, і вивчення їх на високому рівні має бути обов'язковою умовою для опанування майбутньої професії, в той час як деякі інші не мають такого важливого практичного значення. Слід зазначити, що студенти не в змозі самостійно розставити пріоритети, і більш з них просто не отримають на належному рівні той необхідний мінімум практичних навичок, потрібних для подальшого працевлаштування.

Таким чином, збільшення практичної цінності дисципліни, на наш погляд, слід здійснювати як у напрямку заміни «академічних» задач на ті, які мають більшу практичну цінність, зокрема, вирішення комплексних прикладних завдань, які більше схожі на те, з чим зустрічається у своїй роботі програміст, так і у напрямку розстановки пріоритетів на темах, які є особливо важливими.

Добрі практики програмування, на нашу думку,– особливо важливе питання, яке традиційно незаслужено обходять увагою у вузах. Наприклад, один із найбільш авторитетних фахівців з теорії і практики програмування, С. Макконелл стверджує, що написання якісного коду є особливо важливою рисою сучасного програміста [4]. Як показує вивчення досвіду працевлаштування випускників, при прийомі на роботу особливо перевіряються не базові фундаментальні знання, які, звісно, мають бути наявними у випускника вузу, а його вміння писати якісний документований і зручний для подальшої підтримки код, навики володіння поширеними програмними бібліотеками, гарні знання паттернів проектування і стандартів кодування – більшість з того, що традиційно не входить до навчальних програм вищих навчальних закладів, які вивчають програмування на економічних спеціальностях.

Крім того, особливого значення слід приділяти таким питанням, як навики командної роботи й управління програмними проектами. Сучасні програмні проекти дуже рідко реалізуються програмістами-одинаками. Звичайно вони вимагають скоординованої командної роботи, використання спеціальних інструментальних засобів, таких як системи контролю версій, баг-трекери, системи управління вимогами, прожект-менеджери та ін. Однак викладачі у вузах не зацікавлені у тому, щоб займатися формуванням навиків саме командної роботи у студентів, оскільки це достатньо складна задача, яка вимагає спеціальних, часто інноваційних, підходів, що погано узгоджуються з тими методами, за якими працює вища школа. У результаті навіть найкращі студенти-випускники, які продемонстрували високий рівень опанування дисципліни у вузі, однак не мали можливості паралельно з навчанням працювати за напрямком майбутньої професії, мають значні проблеми із подальшим працевлаштуванням, оскільки роботодавці ставлять перед ними такі вимоги, яким не відповідає середньостатистичний випускник вищого навчального закладу.

ВИСНОВКИ

Таким чином, нами були підняті деякі важливі питання стосовно трансформації викладання програмування дисциплін у вищій школі.

Використання практичних завдань і вивчення добрих практик програмування дозволяє підготувати студентів для того, щоб отримані ними знання і навики не потребували подальшого «перенавчання» для того, щоб могли бути використаними у реальних проектах.

Таким чином, нами пропонуються деякі нестандартні підходи, які, на наш погляд, мають суттєво покращити відношення і, найважливіше, – рівень опанування студентами дисципліни «програмування». Сподіваємося, що запропоновані ідеї знайдуть своє впровадження у досвіді роботи працівників галузі освіти і науки, для яких подібна задача є актуальною. ■

ЛІТЕРАТУРА

1. Сейдаметова З. С., Шкарбан Ф. В. Сценарний підхід в навчанні інформатики: об'єктність, наочність, креативність [Електронний ресурс].– Режим доступу: www.ii.npu.edu.ua/files/Zbirnik_KOSN/16/7.pdf

2. Moskal B., Lurie D., Cooper S. Evaluating the Effectiveness of a New Instructional Approach [Електронний ресурс].– Режим доступу: http://portal.acm.org/citation.cfm?id=971328

3. Колдовский В. В. Требуются революционеры // Компьютерное обозрение.– 2008.– № 7.– С. 50 – 56.

4. Макконелл С. Совершенный код. / М.: Питер, Русская редакция, 2007.– 896 с.