«БАГТРЕКІНГ» ЛЕКЦІЙНОГО МАТЕРІАЛУ З ВИЩОЇ МАТЕМАТИКИ ЯК СКЛАДОВА ТЕХНОЛОГІЇ НАВЧАННЯ МАЙБУТНІХ ІНЖЕНЕРІВ З ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Оксана ДУБІНІНА

У статті обґрунтовано необхідність створення та викладено результати розробки і особливості практичної реалізації частини акме-технології формування математичної культури майбутніх фахівців з програмної інженерії, яка стосується розвитку математичної мови.

The article substantiates the need to create and presents the results of development and implementation features of the part of acme-technology for future experts in software engineering mathematical culture formation, that touches on the development of mathematical language.

Постановка проблеми. Розмірковуючи над тим, якою має бути педагогічна технологія формування і, що не менш важливо, неперервного розвитку математичної культури майбутніх інженерів з програмного забезпечення, головне, що стало предметом наукового пошуку – це її методологічна основа, тобто такі специфічні особливості галузі програмної інженерії, які слугували б постійним поштовхом для всебічного і ґрунтовного вивчення усіх математичних дисциплін, передбачених навчальними планами. При цьому постало першочергове завдання обов'язкового збереження десятиліттями напрацьованого надбання математичної підготовки інженерних кадрів національною вищою школою, зважаючи на великий об'єм математичного матеріалу, потрібного для підготовки кваліфікованого фахівця означеної галузі виробництва.

Аналіз останніх досліджень та наукових публікацій з проблеми підтверджує, що математична мова як складова математичної культури завжди привертала увагу науковців, серед яких К. Б. Бєктаєв, В. С. Герасимова, Р. Г. Піотровський, Н. Г. Салміна, В. Н. Худяков та інші.

Мета написання статті – представлення результатів розробки професійно спрямованої педагогічної технології формування математичної культури студентів за напрямом підготовки «Програмна інженерія».

Виклад основного матеріалу. Загальноприйнятою провідною формою навчання студентства є академічна лекція, що має багатовікову історію і є такою формою організації процесу навчання, що уявляє собою усний монологічний систематизований послідовний виклад навчального матеріалу лектором. Основним призначенням лекцій циклу математичних дисциплін є формування основ математичного знання, як фундаменту інженерної професії, а також визначення напряму, загального змісту, характеру усіх видів навчальних занять і самостійної роботи студентів з формування їх математичної культури. Тобто лекція передує всім іншим формам організації навчального процесу, що дозволяє оперативно актуалізувати навчальний матеріал з визначених математичних дисциплін. Проте лекція з математики для програмних інженерів згідно акме-технології формування математичної культури зазначених фахівців підпадає під певне професійно-спрямоване навантаження. На жаль, не рідко до тепер у вітчизняних технічних навчальних закладах зустрічається формалізоване читання лекцій, обмежене надиктовуванням визначень і теорем.

Чимало менеджерів ІТ компаній стверджують, що їм потрібні не просто співробітники, а вмілі розробники з інженерними здібностями. А таких дійсно не вистачає. Вимогою сьогодення підготовки відповідних фахівців стає те, що навчальний заклад має готувати фахівців, здатних приступати до виробничої діяльності одразу після одержання диплома [8]. Аналіз публікацій з програмної інженерії підтверджує [6], що різниця в якості роботи кваліфікованого і некваліфікованого розробника у проекті може відрізнятися в десятки разів. Тому мета лекцій з математики технології «відстеження помилок» - готувати фахівців, які задовольняли б найвищим вимогам, надаючи їм одночасно фундаментальну академічну освіту, формувати засобами математики необхідні професійні якості, навички та вміння.

До найважливіших речей, які повинен знати кваліфікований інженер з програмного забезпечення, належить набір концепцій, які знову і знову використовуються у його роботі. Методики та інструментарій можуть змінюватися, а потужні інтелектуальні схеми, набуті в ході формування математичної культури залишаються на роки і складають фахову скарбницю програмного інженера. Більша їх частина напрацьовується роками і вимагає поступового вдосконалення, крок за кроком, під керівництвом викладачів.

На лекціях згідно пропонованої технології вважаємо доцільним спиратися на базові концепції, які є основою всієї області програмної інженерії [7]: системний аналіз, налагодження, перемагання складнощів. Щоб вирішити завдання за допомогою програмного забезпечення, спочатку необхідно зрозуміти і описати її, тому системний аналіз - невід'ємна частина програмної інженерії та, мабуть, одна з найскладніших її частин. Помилки - звична складова повсякденної роботи програмного інженера. Для того щоб впоратися з ними, необхідно систематично й ефективно використовувати методи налагодження. Хоча в підручниках, як правило, на цьому не акцентують увагу, більшу частину повсякденної роботи виробникам програмного забезпечення доводиться витрачати на з'ясування того, чому програма не функціонує належним чином, незалежно від того, чи то власна помилка безпосередньо створювача програми або когось ще. Фахівці з програмного забезпечення - найкращі в світі створювачі хаосу. Саме цей факт і пов'язане з ним вміння відстежувати помилки враховуємо в технології формування математичної культури студентів за напрямом професійної підготовки «програмна інженерія». Оскільки програмна інженерія - грандіозна інтелектуальна робота, то професіонал повинен знати, як розпізнати суть за наявним безладом. Навчитися не пасувати перед інтелектуальними складнощами – пряме завдання циклу математичних дисциплін.

У методичній літературі [3, с. 213] зазначається, що формування культури ведення лекційних записів є важливим педагогічним завданням. Але не існує єдиних правил конспектування лекції. Вважається, що це залежить від індивідуальних особливостей вимог викладачів і індивідуальних якостей особистості студентів. У цьому відношенні слухачів можна розділити на чотири групи: перші – уважно слухають лектора, аналізують інформацію і фіксують лекційний матеріал; інші – майже дослівно намагаються записати текст лекції, іноді навіть не усвідомлюючи її зміст; треті – уважно слухають, аналізують, але не роблять ні яких записів; студенти четвертої групи – нічого не слухають, часто займаються сторонніми справами, порушують ділову обстановку і дисципліну. Конспект є корисним лише тоді, коли з самого початку зорієнтований на одночасне прослуховування лекції та мисленнєву переробку матеріалу, на виділення та фіксацію у тезисно - аргументованій формі головного змісту лекції. Важливо враховувати, що у студентів, як правило, стихійно

складається прагнення найбільш повно записати увесь лекційний матеріал, що не сприяє його глибокому розумінню і засвоєнню.

Для процесу відстеження помилок, дефектів - «багів» - у галузі програмної інженерії використовується англійська назва - *bug tracking* - «багтрекінг». Як правило, це прикладна програма, розроблена з метою допомогти розробникам програмного забезпечення, а саме програмістам, тестувальникам тощо, враховувати і контролювати помилки і недоліки, знайдені в програмах, побажання користувачів, а також стежити за процесом усунення цих помилок і виконання або невиконання побажань.

Розглянемо застосування багтрекінгу на прикладі лекції з математичного аналізу. Лекція (лат. *lectio* - читання) у вищій школі в класичному її розумінні є методом навчання і виховання, який полягає у послідовному монологічному викладі системи ідей у певній галузі [1, с. 47], що передбачає наукове і послідовне викладення навчального матеріалу, будь-якого питання, теми, розділу, предмету, методів науки тощо. ЇЇ типові ознаки – системність, логічна послідовність, строга структурність, наукова обґрунтованість [5]. Якщо говорити про лекції з математичних дисциплін, то вони відносяться здебільшого до проблемного типу.

По суті *багтрекінг лекції – це спеціально розроблена з навчальною метою педагогічна технологія, яка в рамках викладання фундаментальної дисципліни імітує модель технології відстеження помилок у професійній діяльності інженера галузі виробництва програмної продукції*. Студенти, залучаючись до виробничої проблемної ситуації, вчаться находити правильне її вирішення використовуючи математичне мислення, знання математичної мови, вже засвоєні математичні знання. Поряд з цим студенти опановують інструмент, який їм стане у пригоді у професійній діяльності, набувають вміння користуватися ним. Аналогом *програмного коду* будемо вважати конспект лекції у зошиті. Лектор у створеній виробничій ситуації виступає в ролі *програміста*, а студенти – *тестувальниками*. У якості прикладної програми по відстеженню помилок виступає мисленнєва діяльність студентів на основі власної математичної культури. Проаналізувавши види помилок, які частіше трапляються в конспекті лекцій з математичних дисциплін, виділимо найпоширеніші: *арифметичні, семантичні і синтаксичні.*

Арифметика (давньогрецькою *ἀριθμητική*, від *ἀριθμός* – число) - розділ математики, що вивчає числа, в першу чергу невід'ємні раціональні числа: цілі та дробові, дії над ними [5, с. 77], їх найпростіші відношення і властивості. *Арифметичними* помилками вважаємо всі, що пов'язані з основними операціями над числами та їх властивостями. На більшості лекцій з вищої математики використовуються арифметичні дії. Навмисну помилку в розрахунках можна зробити, наприклад, ілюструючи якийсь теоретичний матеріал. Проте бажано зробити її наприкінці розв'язуваного завдання, щоб не нашкодити основній навчальній меті.

У програмуванні поширеними є *помилки пріоритету операцій*, пов'язані з порядком виконання математичних дій. Знаходити їх важко, оскільки виконання програми при цьому не зупиняється, а результат отримуємо не вірний. Такими помилками в арифметиці є не дотримання пріоритетності операцій стосовно дії у дужках, множення і ділення, додавання та віднімання.

Відносно синтаксичних помилок, то вони є найпоширенішими у програмуванні, тому основну увагу при реалізації технології багтрекінгу лекцій з математичних дисциплін зосередимо на них. Під *синтаксисом* (від лат. *σύνταξις* - побудова, порядок, складання) формалізованої мови розуміємо систему правил побудови виразів цієї мови та перевірки того, чи є ці вирази правильно побудованими

формулами, аксіомами, теоремами, висновками або доказами, включаючи набір граматичних правил. Має значення і правильне написання знаків та символів. Треба брати до уваги, що комп'ютери більш вимогливі до грамотної мови, ніж люди у спілкуванні. Тому від створювача програмного забезпечення вимагається надзвичайна уважність. Це означає, наприклад, якщо рядок згідно визначеної мови програмування треба брати в лапки, то не можна сподіватися, що хоча б одну з лапок вдасться пропустити – комп'ютер цього не дозволить. Вбачаючи аналогію в тому, що і математична мова і будь-яка мова програмування є формалізованими мовами – спроектуємо характер помилок, що допускаються в ході написання коду програми.

Формалізовані мови, які використовуються для формалізації математичних теорій, зазвичай називають логіко-математичними мовами, оскільки в них поєднується використання математичної і логічної символіки. Побудова будь-якої формалізованої мови починається з абетки (алфавіту), тобто переліку символів, з яких будуються усі вирази. Потім описується синтаксис – правила побудови осмислених виразів. В логіко-математичних мовах серед таких виразів розрізняють *терми* і *формули*. За визначеними правилами із предметних констант і змінних будуються більш складні терми. Із термів за допомогою предикатних символів і символів логічних операцій будуються формули – вирази, які відповідають висловлюванням і висловлюваним формам звичайної мови. Із елементарних формул будуються більш складні, при цьому символи логіки грають ту ж саму роль, що і союзи та інші службові слова при побудові складних речень звичайної мови [5, с. 609]. Найбільш вживаними на лекціях з математики є наступні логічні символи: $\supset$ - знак імплікації, $\&$ - знак кон'юнкції, $\vee$ - знак диз'юнкції, $\neg$ - знак заперечення, $\forall$ - квантор всезагальності, $\exists$ - квантор існування та інші. Розуміння осмислених виразів математичної мови складає її *семантику*. Опис синтаксису і семантики зазвичай йде на природній мові, яка таким чином виступає як метамова по відношенню до математичної мови.

Семантичні (від грецької - *σημαντικος* - який означає) *помилки* - це помилки, пов'язані з неправильним змістом дій, порушенням логіки і використанням неприпустимих значень величин. Такі помилки досить важко знайти, доводиться переглядати практично весь конспект лекції з аналізом того, в якому місці насправді порушена логіка.

Для реалізації запропонованої технології ми пропонуємо анонсувати на початку лекції одну синтаксичну, можливо арифметичну помилку, якщо всі розрахунки наведені на дошці, але до семантичних помилок підходити з особливою обережністю. Оскільки при цьому може порушуватися принцип науковості, що вважаємо неприпустимим, оскільки застосування педагогічної технології не повинно нашкодити смисловому наповненню лекції. Проте, треба зауважити, що добре продумана логічна помилка є цікавою. Якщо студенти виявили помилку в ході лекції, що часто трапляється при напрацюванні цього навику, то на цей випадок необхідно мати в запасі ще кілька заздалегідь заготовлених продуманих помилок. Викладач кожен раз дякує за допомогу у точності викладення на дошці лекційного матеріалу, підкреслюючи тим важливість навчально-виробничої ситуації. При цьому помилки, відстежені протягом викладення лекційного матеріалу, не вважаються анонсованою помилкою, тобто до відома студентів не доводиться, що помилка знайдена. Отже наприкінці лекції повинна обов'язково залишитися одна, і не більше, помилка. Оскільки лекція з математичного аналізу має свої чіткі навчальні та наукові цілі, і ні в якому разі не повинна звестися суто до технології програмної інженерії. Одну навмисну помилку залишаємо для того, щоб за період часу порядку 3 - 5 хвилин,

який викладач навмисно залишає до закінчення аудиторного заняття зосередити увагу студентів, надати їм можливість сконцентруватися на навчальному матеріалі, сформулювати запитання.

Тому, хто перший знайшов помилку, обов'язково виставляється бал, який потім зараховується до загального рейтингу студента. Якщо помилку знайшли відразу кілька студентів, то бали виставляються всім, згідно концептуально значимого для програмного інженера командного виконання роботи. У випадку не винайдення помилки за короткий проміжок часу, вона залишається в якості домашнього завдання і виправлення може бути надіслане лекторові електронною поштою упродовж будь-якого часу аж до наступної лекції. Лектор оцінює тільки перше за часом надіслане виправлення, але не повідомляє про це інших студентів, фіксуючи для себе їх активність і надаючи їм можливість повідомити про свій успіх.

Оцінювання відстеженої помилки є обов'язковим. На наш погляд, один чи два «бонусні» бали, отримані таким чином, істотно не вплинуть на підсумкові бали, які отримають студенти за тематичний модуль з певної дисципліни математичного циклу, що зазвичай становить 100 – 200 балів. Оскільки оцінювання фактично відбувається за накопичувальною системою, тобто студент набирає якусь кількість балів на кожному практичному занятті, проходячи коротке тестування з попереднього матеріалу, отримує також певні бали за підсумкову контрольну роботу по вирішенню практичних завдань, складання колоквіуму з теоретичної частини навчального матеріалу, а також за обов'язкове розрахунково-графічне домашнє завдання. Проте усвідомлення студентами того, що на лекціях з математичного аналізу вони підвищуючи свій рівень математичної культури одночасно продуктивно і наочно залучаються до технологій отримуваної професії є потужним мотиваційним чинником. В результаті ми отримуємо добре опрацьовану, вдумливо законспектовану лекцію, яка є провідником у царину математичного знання.

Хоча в жодному звичайному програмному проекті вища математика зовсім не застосовується повсякденно, певне розуміння того, що програмування та мови програмування - це математичні об'єкти, що підлягають формальному опису, є ключовою вимогою необхідності високого рівню математичної культури інженера з програмного забезпечення. Студенти, що навчилися застосовувати математичне мислення, яке є невід'ємною складовою математичної культури, до розробки програмного забезпечення, мають переваги в порівнянні з тими, хто подібними навичками не володіє. Цикл математичних дисциплін передбачений університетським навчальним курсом, але дуже важко опанувати його потім, після того, як випускники тривалий термін пропрацювали на виробництві, розуміючи вже на своєму власному досвіді, що математична культура куди важливіша для учасників проекту, ніж знання швидко змінюваних в програмній інженерії інструментарію та методик. Отже, технологічне коригування навчальних програм з циклу математичних дисциплін, який підпадає під найбільше навантаження з формування професійно спрямованої математичної культури майбутніх програмних інженерів, може серйозно змінити якість підготовки студентів.

Висновки та перспективи подальших досліджень. *Переваги застосування багтрекінгу лекційного матеріалу з математичних дисциплін:*

– це не просто *встановлення міжпредметного зв'язку*, а майже злиття професійної і фундаментальної математичної підготовки за допомогою педагогічної технології без втрати якості, з набуттям потужного мотиваційного чиннику; – застосування

проектної методології, на якій ґрунтується методологія програмної інженерії, є потужним засобом формування і розвитку професійної культури за напрямом підготовки студентів «Програмна інженерія»; – в ході використання налагодженої системи багтрекінгу теоретичного лекційного матеріалу відбувається більш глибоке усвідомлення студентом мети певної лекції, покращення сприйняття за рахунок посилення уваги, налаштування на *розуміння, а не прослуховування*, з першої хвилини, прискорена мисленнєва переробка лекційного матеріалу, обміркування і повторення або впродовж заняття, або відразу після закінчення доповіді викладача; – надаючи викладачеві право на помилку, студенти вчаться брати *інтелектуальну відповідальність* на себе; – відбувається поглиблене вивчення математичної мови, яка є вагомою складовою математичної культури майбутніх фахівців з програмної інженерії через семантичний та синтаксичний підходи [4]; – систематичне застосування технології відстеження помилок до певної міри позитивно впливає на проблему *комунікації і взаємодії* між студентами і лектором упродовж всього семестру, є стимулятором виникнення *зворотного зв'язку* після прочитання теоретичного матеріалу з фундаментальної дисципліни; – отримуємо активізацію навчально-пізнавальної діяльності студентів з одночасною її діагностикою; – практичне застосування педагогічної акме-технології в ході багтрекінгу привчає студентство до *створення ситуації успіху*, а викладач отримує додатковий контроль точності викладення навчального матеріалу; – наочне використання багтрекінгу лекцій з математичних дисциплін стимулює до вивчення і застосування професійних систем відстеження помилок *Anthill, Atlassian JIRA, Bugzilla, Codestriker, Tracker, Trello* та інших. **Перспективами подальших досліджень** є вивчення і створення педагогічних умов необхідних для реалізації запропонованої акме-технології.

БІБЛІОГРАФІЯ

1. Воронин А. С. Словарь терминов по общей и социальной педагогике / А. С. Воронин. – Екатеринбург: ГОУ ВПО УГТУ–УПИ, 2006. – 135 с.
2. Математический энциклопедический словарь / гл. ред. Ю. В. Прохоров. – М.: Советская энциклопедия, 1988. – 847 с.
3. Подольская Е. А. Педагогика и психология высшей школы: учеб. пособие / Е. А. Подольская. – Харьков: НУА, 2010. – 316 с.
4. Худяков В. Н. Семантический и синтаксический подходы к изучению математического языка / В. Н. Худяков, Е. Ф. Габрик // Сб. науч. труд. преп. и сотр. РИПОДО ЧГПУ. - Челябинск, 1999. - С. 25-30.
5. Ягупов В. В. Педагогіка: навч. посібник / В. В. Ягупов. – К.: Либідь, 2002. – 560 с.
6. Boehm B. W. Software Engineering Economics / B. W. Boehm. – Prentice-Hall, 1981. – 320 p.
7. Meyer B. Software Engineering in the Academy / Bertrand Meyer / Computer Society (IEEE), May 2001. – vol. 34, no. 5, p. 28-35.
8. Vliet Hans van. Reflections on Software Engineering Education / H. van Vliet. – IEEE Software, May/June, 2006. – P. 55 – 61.

ВІДОМОСТІ ПРО АВТОРА

Дубініна Оксана Миколаївна – кандидат технічних наук, доцент кафедри комп'ютерної математики та математичного моделювання Національного технічного університету «Харківський політехнічний інститут».

Коло наукових інтересів: математична культура особистості.