Від редакції. Ми із задоволенням друкуємо цю статтю майже без редакційних змін, в якій розглядається актуальні питання змісту навчання у середній і вищій школі. У статті наведено цікаві й оригінальні ідеї і думки досвідчених викладачів і авторів підручників з інформатики. Можливо, слід погодитися з основними висновками авторів. Однак, на нашу думку, деякі з них не є незаперечними і потребують ретельного обговорення. Тому закликаємо наших читачів до широкого обговорення порушених проблем.

ЧОМУ МИ ВИБИРАЄМО VISUAL BASIC

Глинський Я.М., Ряжська В.А.

У курсі інформатики в середній і вищій школах можна виокремити два важливі розділи. Це «Інформаційні технології» й «Основи алгоритмізації та програмування». У 2011/2012 навчальному році для вивчення розділу «Основи алгоритмізації та програмування» у середній школі на рівні стандарту надається відповідно до чинної програми 5% від загальної кількості годин трирічного курсу, а на академічному рівні — 22% [1]. Правда, у схваленій науково-методичною комісією з інформатики 03.06.2011 додатковій програмі з інформатики на вивчення цього розділу на рівні стандарту надається 10% годин, що кількісно становить 11 годин.

У вищих навчальних закладах на інженерно-технічних некомп'ютерних напрямах підготовки на вивчення відповідного розділу програми у рамках дисциплін, які, зазвичай, називаються «Інформатика» або «Інформатика та комп'ютерна техніка», надається від 30 до 50% від загальної кількості аудиторних годин. Не можна не погодитися з тезою, що в умовах неодноразової зміни освітніх парадигм і технологій навчання, апаратних платформ і технологій програмування актуальним стає перехід до нової моделі навчання, що формує в майбутнього студента і фахівця здатність до розв'язування нестереотипних задач, до творчого мислення на основі фундаментальних знань [2].

Питання фундаменталізації курсів базової інформатики є актуальним і може вирішуватися головно зростанням ролі інформаційного моделювання, розвитком алгоритмічного компонента діяльності і в дещо меншій мірі математизацією змісту навчання. Актуальним знову постає питання наступності змісту навчання в середній і вищій школі, яке було зняте з порядку денного декілька років тому, коли школи забезпечили певний рівень знань своїх випускників, який більш-менш корелювався із задекларованими у навчальних програмах рівнями учнівських компетентностей.

Як показали результати інтернет-опитування кількість школярів, які вчаться за програмою рівня стандарту і фактично не будуть ознайомлені з основами алгоритмізації і програмування, становить орієнтовно 60–70%. Це означає, що програми з інформатики для вищих технічних начальних закладів треба буде знову переглядати з метою передбачення вивчення розділу «Основи алгоритмізації та програмування» від самих початків.

Доцільність вивчення цього розділу в курсі базової інформатики нині не викликає сумніву [3]. Однак, на жаль, під час розробки нових галузевих стандартів вищої освіти цей розділ необґрунтовано випав із програм декількох напрямків підготовки. Так, на напрямку «Менеджмент» не тільки він, а й ціла дисципліна «Інформатика та комп'ютерна техніка» на першому курсі підмінена зовсім іншою дисципліною «Інформаційні системи та технології», яка відповідно до тематичного наповнення мала би вивчатися на третьому курсі.

Відкритими й актуальними залишаються питання вибору правильної методики навчання інформатики, яка має передбачити високий рівень мотивації навчання, врахування специфіки різних напрямів підготовки, націленість на розвивальне навчання, на гармонійний розвиток особистості майбутнього студента чи фахівця, на формування загальної інформаційної культури і здобуття відповідних компетентностей. Роль розділу «Основи алгоритмізації та програмування» як провідника концепції фундаменталізації курсів інформатики є значною.

Головними завданнями цього розділу насамперед є:

- навчити моделювати реальні об'єкти і явища;
- навчити аналізувати і знаходити розв'язки задач, які є моделями життєвих (розвиток асоціативного мислення);
- розвивати алгоритмічне мислення;
- розвивати творчі здібності особистості;
- роз'яснити принципи функціювання комп'ютера і роботи з даними;
- навчити добирати засоби для розв'язування типових задач;
- навчити аналізувати і приймати правильні рішення, що є неодмінною вимогою до сучасного спеціаліста будь-якого напрямку підготовки: будівничого, економіста, менеджера, інженера-механіка тощо.

Важливо, щоб навчальний процес був сучасним, умотивованим, максимально зрозумілим і цікавим. Не можна однаково подавати матеріал для майбутніх економістів, хіміків, математиків чи програмістів. Разом з тим варто виокремити набір задач, які становлять інваріанту частину в навчанні учнів і студентів різних напрямів підготовки. Власне такі задачі й мають розглядатися в школі й на першому курсі навчання в рамках базового предмета інформатика.

У цій статті ми робимо спробу проаналізувати деякі підходи до навчання розділу алгоритмізації й програмування, що актуально для різних профілів навчання у школі й напрямків підготовки у вищих навчаль-

них закладах і пропонуємо шляхи вирішення посталих проблем.

Нині під час вивчення розділу «Алгоритмізація та програмування» звертається увага на такі технології програмування:

- традиційне процедурне програмування;
- розробка програм і проектів у візуальних середовищах;
- об'єктно-орієнтоване програмування (ООП).

Розглянемо ці три технології. Освоєння процедурного програмування, зазвичай, зводиться до вивчення алгоритмічної мови Паскаль, що нині є слабо мотивованим, оскільки сфери застосування мови постійно звужуються, хоча ще залишаються міцними. Мова відігравала важливу роль у 80–90-х роках минулого століття, увійшла в навчальний процес в Україні з десятирічним запізненням і, здається, покидати вона його збирається з тим самим десятирічним запізненням. Тепер роботою в консольному режимі ні учнів, ні студентів зацікавити не можливо. Для сьогоднішнього рівня загальної підготовки учнів і студентів мова є заскладною, часто з нездоланними синтаксисом і семантикою. Вивчення основ програмування у відповідних візуальних середовищах (Delphi чи Lazarus) може ще на деякий час відтермінувати її відхід, але не надовго і лише за умови застосування вдалих методик навчання.

Другу з перелічених технологій хотілось би назвати візуальним програмуванням, розширюючи традиційне трактування цього поняття. Традиційно візуальним програмуванням називають розробку програм засобами візуальних елементів, коли кодів «вручну» не пишуть взагалі. Таке візуальне програмування використовується лише для розв'язування вузького кола специфічних задач специфічними програмними засобами. У цій статті пропонуємо візуальне програмування трактувати дещо ширше, а саме: як технологію програмування, де домінує конструювання інтерфейсу програми з візуальних елементів керування (компонент), а написання кодів «вручну» зводиться до створення кодів реакцій на події в стилі технології процедурного програмування. Це дає змогу процедурне програмування розглядати як частковий випадок візуального, а візуальне програмування (ВП) трактувати як розробку візуального інтерфейсу (ВІ) із застосуванням процедурного програмування (ПП), тобто маємо, що ВП=ВІ+ПП.

Підхід, у якому візуальне програмування можна розглядати як візуальну оболонку над процедурним є дуже продуктивним у методичному плані. Візуальне програмування — це той прорив у програмуванні, який можна порівняти з переходом від MS DOS до Windows, що відбувся в операційних системах двадцять років тому.

Фактично реалізація процедур у ВП здійснюється мовами, які належать до класу об'єктно-орієнтованих, але про об'єктно-орієнтоване програмування тут мова не йтиме.

Під час візуального програмування учні і студенти із задоволенням створюють форми, вставляють різні елементи керування, змінюють їхні властивості, програмують кнопки тощо, оперуючи поняттями об'єкт, метод об'єкта, властивість об'єкта, значення властивості, що дає змогу побудувати нову модель навчання, яка базується на принципах фундаменталізації предмету. Під час роботи над дизайном форми з'являється додатковий простір і поштовх для творчості й самоутвердження, що часто потребує креативних підходів, самостійного освоєння окремих тем чи питань, які, як з'ясувалося, можна легко подолати, знайомлячись з довідковими системами чи он-лайн допомогою, чи звертаючись до спільнот в інтернеті. Достатньо вчителеві чи лектору вимовити фразу «Ви можете вставити елемент керування класу «медіаплеєр» на форму і переглядати відео», як далі все відбувається автоматично і без участі викладача — на учнівських чи студентських формах з'являться не тільки медіаплеєри, але й інші екзотичні елементи керування: власні браузери, таймери тощо. Усі основні традиційні задачі курсу «Основи алгоритмізації та програмування» легко реалізуються зі значно більшим зовнішнім ефектом і внутрішнім розумінням. Рівень знань і вмінь за такого підходу вищий і вмотивованість й зацікавленість у навчанні набагато більша.

Традиційні задачі, які рекомендується вивчати в розділі процедурного програмування, можна оформляти як візуальні проекти і використання консольного режиму роботи зводити до мінімуму. Візуальне програмування з розробкою змістовних проектів замість закостенілого процедурного програмування — це суттєвий аргумент на користь збереження теми «Основи алгоритмізації та програмування» як важливого розділу базової інформатики на рівні стандарту і використання на її викладання варіативної години навчального плану.

Чи можна програмування з налаштуванням стандартних візуальних об'єктів, якими є елементи керування, назвати об'єктно-орієнтованим. Однозначно, ні. Об'єктно-орієнтоване програмування починається зі створення власних класів і дотримання відповідної парадигми ООП, яка є досить строгою.

Програмування з побудовою власних класів, що базується на принципах моделювання предметних областей, зокрема, мовою UML (Unified Modeling Language), на нашу думку, слід використовувати виключно для студентів, зокрема, комп'ютерних і математичних спеціальностей, хоча в школі це передбачено програмою інформаційно-технологічного профілю. Безумовно, це найсучасніша технологія програмування, оволодівши якою, можна стати досвідченим фахівцем у даній галузі. Але для більшості інженерно-технічних напрямів підготовки такий підхід є заскладним і надлишковим.

Можна продемонструвати принципи ООП на прикладі інтерфейсної частини проекту, яку візуальне середовище створює автоматично під час вставляння на форму елементів керування. Можна продемонструвати формальне застосування принципу інкапсуляції для побудови власного класу, але лише з ознайомчою метою, тому що такий підхід без застосування наслідування, поліморфізму, конструкторів і деструкторів є непродуктивним.

Щодо вибору середовища програмування, то слід зауважити, що завжди підкреслювалось, що не важливо, якою мовою (Бейсик, Паскаль чи С) програмувати і в якому середовищі (Visual Basic .NET, VBA, Delphi, C# у Visual Studio) навчати учнів і студентів. Реалізації концепцій навчання у відповідних середовищах програмування з точністю до синтаксису мов дуже схожі. Опанувавши одну з мов програмування, не складно розібратись самостійно і з іншою.

Однак ми вважаємо, що нині зваженої альтернативи мові Visual Basic немає. Більше того, на нашу думку, відмова в 90-х роках від Бейсика і перехід у школах до вивчення мови Паскаль, був шкідливим для освітнього

процесу. Адже на початку 90-х сформувалися версії мови Бейсик і середовища програмування, які не поступалися середовищу Turbo Pascal, що залишилося незаміченим чиновниками від освіти і їхніми радниками. На сьогоднішній день захоплює перш за все високоінтелектуальний рівень середовищ програмування мовою Visual Basic. Під час введення тексту кодів постійні підказки просто не дають змоги помилитися, а деякі помилки середовище виправляє автоматично. Послаблення щодо строгості декларування змінних, на нашу думку, корисне для початківців, які не планують стати програмістами. Синтаксис мови цілком і повністю відповідає принципам структурного стилю програмування. Невідповідність принципам структурного програмування була притаманна лише раннім версіям мови Бейсик, що вважалося головним недоліком ранніх версій мови. Зауважимо, що працюючи з мовою Паскаль, ми добивались ефекту якісного засвоєння знань орієнтовно на 20–30%, а працюючи з мовою Visual Basic, якість засвоєння матеріалу учнями і студентами зросла до 80%. Питання, пов'язані з проведенням олімпіад, легко вирішуються. На часі вирішення питання про організацію ЗНО з інформатики, яке без розділу «Основи алгоритмізації і програмування» немає сенсу. Звертаємо увагу на необхідність формулювання завдань цього розділу в термінах мов Паскаль і Visual Basic.

Застосування мови Visual Basic, окрім навчально-методичних, має низку технічних переваг, що пов'язано, зокрема, з легкістю інсталяції студійних експрес-середовищ чи гарантованою наявністю візуального середовища VBA на типових шкільних комп'ютерах.

Треба також врахувати питання легальної доступності того чи іншого середовища програмування. Це ще один аргумент на користь доцільності зупинити вибір на мові Visual Basic у середовищах класу Visual Basic .NET чи VBA.

Такий вибір обумовлений декількома причинами:

1) навчальні заклади, учасники програми MSDN AA, започаткованої корпорацією Microsoft, можуть отримати пакет програм Visual Studio 2008 і деякі програми з пакета MS Office 2007, що дає можливість легально використовувати Visual Basic для навчальних цілей;

2) середовище Visual Basic 2010 Express можна безкоштовно завантажити на локальний комп'ютер із сервера компанії Microsoft (http://www.microsoft.com/express/downloads);

3) на багатьох комп'ютерах встановлено програми з пакета Microsoft Office різних років випусків. Усі ці програми мають вбудований засіб для створення повноцінних проектів у VBA-середовищі;

4) пакет вільнопоширюваних офісних програм OpenOffice.org також містить середовище розробки, подібне до VBA.

Отже, для вивчення основ програмування не потрібно встановлювати додаткові, зазвичай платні, програми на кшталт Delphi.

Ми не просто за Visual Basic, ми за активне використання VBA в навчальному процесі у середній і вищій школах. Адже всі найважливіші поняття і принципи програмування легко засвоїти власне у найпростішому середовищі — VBA. Отримані знання і навички мобільні, тобто переносяться на інші середовища програмування з незначними адаптаціями. Уявіть собі, ви пропонуєте учням чи студентам запустити програму MS Word і прямо в документі оформити розв'язування задачі додавання двох таблиць (масивів). Цю задачу можна розв'язати за 10 хвилин, написавши нескладний VBA-код додавання двох двовимірних масивів для кнопки «Пуск» (рис. 1, [4]), фактично змоделювавши роботу з грідами (сітками) даних.

Будь-яку традиційну задачу з програмування можна розв'язати засобами VBA у середовищі програм MS Word, MS Excel чи MS Visio (власне остання доступна безкоштовно відповідно до ліцензії MSDN AA, чого достатньо для застосування VBA).

Для цього є два шляхи. Перший шлях – це перехід у середовище VBA і робота за таким алгоритмом:

1) відкрити офісну програму-додаток;

2) перейти в середовище VBA за допомогою комбінації клавіш **Alt+F11**;

3) вставити у робоче поле форму командою **Insert UserForm**;

4) заповнити форму елементами керування і запрограмувати їх.

Другий шлях – це програмування безпосередньо на сторінці документа MS Word. Для цього треба виконати такий алгоритм:

1) відкрити MS Word;

2) відкрити панель інструментів Visual Basic;

3) відкрити панель елементів керування;

4) вставити з неї на сторінку потрібні елементи керування з класів «кнопки», «написи», «поля» чи «зображення» тощо;

5) запрограмувати кнопку, двічі клацнувши на ній у режимі конструктора (це перша кнопка на панелі елементів);

6) повернутися на сторінку, згорнувши вікно коду. Отримаємо готовий проект, приклад якого наведено на рис. 2.

Починаючи роботу у VBA, треба пам'ятати про необхідність понизити рівень безпеки комп'ютера до середнього чи низького (кнопкою **Безпека...** на панелі Visual Basic), інакше робота з VBA може бути заблокована з повідомленням **Marcos enabled**. Усі потрібні для роботи й навчання відомості можна почерпнути з багатьох джерел, зокрема [4], де паралельно описано два середовища програмування: Visual Basic 2010 Express і VBA.

Повернемося до питання наступності у навчанні інформатики в середній і вищій школах. Нині у вищих начальних закладах починають навчати розділу «Основи алгоритмізації та програмування» на досить високому рівні, орієнтуючись на старі шкільні програми, де вивчення цього розділу декларувалось всіма учнями на рівні, достатньому для подальшого навчання студентів у темпі «мозкового штурму». Тепер стандарти і відповідні програми вищої школи стануть нереальними для виконання. Адже навіть на комп'ютерні факультети вступають випускники зі шкіл різного профілю підготовки і рівня вивчення інформатики. Наприклад, випускники

Рис. 1. Моделювання грідів у документі MS Word

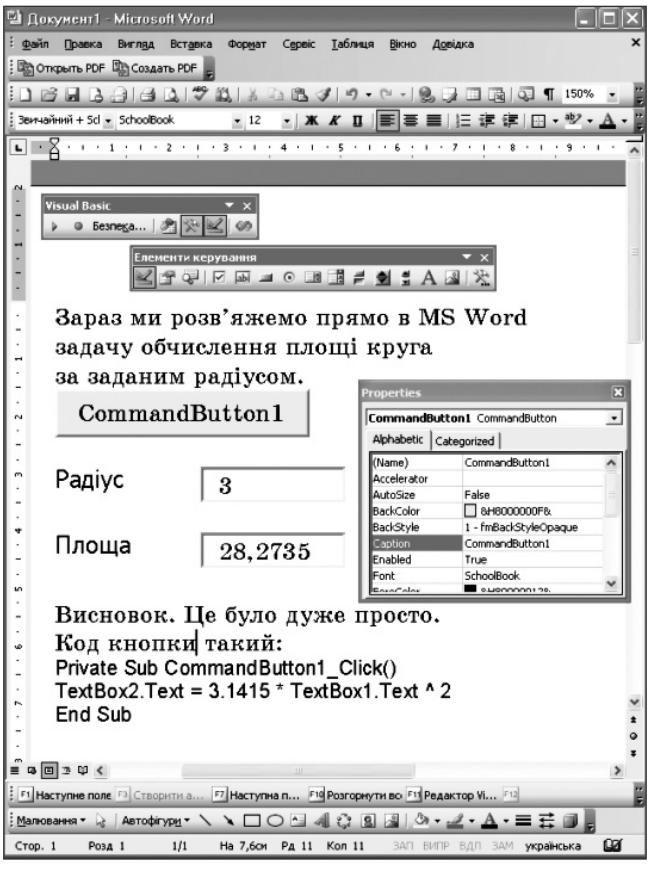

Рис. 2. Розв'язування задач засобами VBA в документі MS Word

англійської спецшколи можуть поступати на комп'ютерний факультет, що цілком реально. Зараз у цих школах не те, що розділ «Основи алгоритмізації та програмування» не вивчають, там сам предмет «Інформатика» є другорядним. Лише впровадження у близькому майбутньому ЗНО з інформатики може врятувати цей предмет від повної деградації.

А тепер ми можемо потурбуватись щодо підняття престижу розділу «Основи алгоритмізації і програмування». Як добитися успіхів у навчанні цього розділу? Перш за все потрібно застосувати Visual Basic. Варто внести зміни до курсу математики або хоча би апелювати до колег-математиків більше звертати увагу на питання, дотичні до алгоритмізації й моделювання. На жаль, сучасні учні часто не вміють проаналізувати поставлені перед ними задачі, змоделювати відповідний процес чи явище. Асоціативне мислення в них розвинуте слабо. Задача обчислити площу клумби (навіть з уточненням, що вона має форму круга) у студентів-першокурсників викликає легкий шок, а формулу для розв'язування $s=\pi r^2$ знає лише кожний п'ятий. В ідеалі інформатика мала б займатися дослідженням готових формальних моделей, а побудова моделей — це завдання, яке треба розв'язувати в курсах математики, фізики та інших дисциплін.

Поняття табулювання функції, як задачі створення таблиці значень аргумента і функції з деяким кроком зміни аргумента, учні часто не знають. Формулу для периметра трикутника вони записують так: $a+b+c=p$, а потім так її і програмують. Про те, що $0.03=3*10^{-2}$, а отже $3.0e^{-2}=0.03$ знає кожний десятий.

За останні роки не було ні конференції, ні нарад фахівців, ні публікацій, які могли би дати оцінку програмам з математики в контексті потреб, які виникають під час вивчення інформатики у школі. У вищій школі навчальні програми з інформатики прийнято погоджувати з викладачами вищої математики й навпаки. Такі погодження варто робити і на шкільному рівні.

Саме навчання треба зробити доступним і багаторівневим. Не варто ставити занадто складних задач. Кожний учень і студент під час навчання мають зрозуміти, що вони можуть чогось досягти і що є до чого прагнути. У школі треба відстежувати чотири рівні навчальних досягнень: початковий, середній, достатній і високий. На початковому рівні варто пропонувати розібратись з готовими проектами, алгоритмами й кодами, відтворити їх у середовищі програмування. На середньому рівні необхідно виконати нескладні завдання на модифікацію, наприклад, додати на форму певні елементи керування, змінити властивості деяких об'єктів, внести незначні зміни в коди тощо. На достатньому рівні потрібно вміти робити складніші завдання на модифікацію: перенаправляти потоки введення-виведення даних, додавати нові елементи до проекту й самостійно складати до них коди. На високому рівні слід створювати свої власні проекти з моделюванням предметних галузей, творчо використовувати отримані під час вивчення знання, а також відомості з мережі, онлайн-спільнот і систем допомоги.

Навчання треба будувати «від задач» і «від зразків проектів». Вивчати конкретну тему з програмування потрібно не заради тієї чи іншої алгоритмічної або синтаксичної конструкції, а заради розв'язування конкретних задач, бажано близьких до життєвих. Навчальна література з програмування часто має такий недолік: вона є лише детальним довідником з мови програмування. Наприклад, у мові Visual Basic є цикли For і While, а також чотири цикли Do Loop, які є абсолютно надлишкові у базовому курсі, проте їх детально описують у багатьох підручниках. Для успішної роботи у візуальному середовищі достатньо лише декількох елементів керування й окремих їхніх властивостей. У підручниках часто наводять повні списки і елементів керування і їх властивостей, що є надлишковим.

Правильно організований навчальний процес, добре підібрані дидактичні й методичні матеріали, узгодженість з напрямом і профілем навчання під час вивчення розділу «Алгоритмізація і програмування» є необхідними складовими для досягнення педагогічної мети і побудови нової моделі навчання. Сьогоднішні учні й студенти мають вміти планувати, прогнозувати, відшуковувати оптимальні розв'язки поставлених перед ними життєвих задач, а наш обов'язок — їм у цьому допомогти.

Література

1. Інтернет-ресурс, http://www.mon.gov.ua/education/average/prog12/inf_st.doc.
2. *Семеріков С.О.* Теоретичні та методичні основи фундаменталізації навчання інформатичних дисциплін у вищій школі / Семеріков С.О., Теплицький І.О. // Теорія та методика навчання математики, фізики, інформатики : збірник наукових праць. Вип. VIII. — Вид. відділ НметАУ, 2010. — Т. 3. — С. 223–239.
3. *Глинський Я.М.* Наступність вивчення базових розділів інформатики у середній і вищій школі / Глинський Я.М., Глинський Ю.Я., Ряжська В.А. // Теорія та методика навчання математики, фізики, інформатики: зб. наук. праць. Вип. VIII. — Вид. відділ НметАУ, 2010. — Т. 3. — С. 50–55.
4. *Глинський Я.М.* Основи алгоритмізації і програмування мовою Visual Basic. — Львів: СПД Глинський, 2011. — 272 с.