- розширити можливості для підвищення кваліфікації вчителів інформатики області, для вдосконалення їх педагогічної майстерності;
- об'єднати зусилля вчителів інформатики, методичних формувань єдиної мережі з метою використання сучасних освітніх технологій.

Досягнення вчителів інформатики Запорізької області, їх високий професійний рівень був відмічений на Міжрегіональній конференції «Шкільна інформатика: сучасний стан, проблеми й перспективи», яка була присвячена 25-річчю шкільного курсу інформатики. Уся спільна праця методистів ЦІТО і РМК, керівників РМО і вчителів інформатики Запорізької області за 2007–2010 рр. відображена на порталі http://ciit.zp.ua.

Ресурсно-мережева модель системи підвищення кваліфікації педагогічних працівників робить діяльність учасників відкритою і доступною для вивчення в будь-який час. У цьому перевага ресурсно-мережевої моделі. Але є й істотний недолік — така модель залежна від технічних умов. Для її функціонування необхідним є підключення до Інтернету всіх учасників ресурсно-мережевої взаємодії.

Ресурсно-мережева модель системи підвищення кваліфікації вчителів у стадії становлення. Ще багато проблем, завдань, які необхідно розв'язувати. Методисти ЦІТО ЗОІППО знають проблеми і працюють далі для того, щоб знайти шляхи їх розв'язання, щоб ресурсно-мережева модель була досконалішою і кориснішою для всіх її учасників.

Література

1. *Гагарина О.Ф.* Условия повышения эффективности функционирования методической службы в системе повышения квалификации работников образования. — Ставрополь, 2005.
2. *Горбунова Л.М., Семибратов А.М.* Построение системы повышения квалификации педагогов в области информационно-коммуникационных технологий на основе принципа распределенности. Конференция ИТО-2004 [Електронний ресурс]. — Режим доступу: http://ito.edu.ru/2004/Moscow/ Late/Late-0-4937.html.
3. Ключевые компетенции и образовательные стандарты : доклад А.В. Хуторского на Отделении философии образования и теоретической педагогики РАО 23 апреля 2002 г. — Центр «Эйдос» [Електронний ресурс]. — Режим доступу: http://www.eidos.ru/.
4. *Кочелаева Е.Р.* Интернет как двигатель профессионального развития учителя [Електронний ресурс]. — Режим доступу: http://ekochelaeva.narod.ru/theory.html.
5. *Круцило О.І., Захарчук В.М.* Атестація педагогічних працівників: метод. посібник. — К.: Шкільний світ, 2006. — 176 с.
6. *Максимов В.В.* Повышение квалификации учителей информатики и организаторов информатизации образования // Материалы науч.-практ. конф. «Информационные технологии в образовании: опыт, проблемы, перспективы», — Якутск : ЯГУ, 2003. [Електронний ресурс]. — Режим доступу: http://www.nerungri.edu.ru/muuo/pweb/resurs/nitobr/plen/mvv.htm.
7. *Ремаренко Е.В.* Сборник научных трудов Международной научно-практической конференции // Информатизация образования. Школа XXI века : сб. науч. трудов междунар. науч.-практ. конф. — С. 168 [Електронний ресурс]. — Режим доступу: http://conference.school.informika.ru/2007/materials.html.

* * *

ОСОБЛИВОСТІ ВИКОРИСТАННЯ ЗАДАЧНОГО ПІДХОДУ У ПРОЦЕСІ ВИВЧЕННЯ ОСНОВ АЛГОРИТМІЗАЦІЇ І ПРОГРАМУВАННЯ

Циммерман Г. А., Циммерман О. В.

У даний час у курсі шкільної інформатики досить складною для розуміння й подальшого опанування учнями є лінія алгоритмізації та програмування. На наш погляд, є декілька об'єктивних пояснень цьому явищу. По-перше, кількість уроків інформатики на тиждень, очевидно, недостатня для освоєння цієї непростої навчальної дисципліни. По-друге, учні не можуть одразу зрозуміти практичної значущості навчання програмування, оскільки методика викладання програмування фактично зводиться до швидкого вивчення особливостей основних конструкцій конкретної мови програмування. По-третє, спеціальна література з програмування, до якої зацікавлені учні намагаються звертатися самостійно, розкриває в основному конструкції мови, виходячи з логіки і правил побудови самої мови.

Завдання навчання полягає в тому, щоб забезпечити позитивні умови для навчання і досягнення мети навчання. Зрозуміло, що максимальний обсяг інформації можна засвоїти тільки за умови, якщо строго викласти формальні конструкції мови і розв'язати мінімум завдань, необхідних для первинного закріплення навчального матеріалу.

Традиційно схема пояснення нового матеріалу зводиться до введення нових операторів програмування або вбудованих функцій. Учням пропонується алгоритмічна конструкція у вигляді блок-схеми, потім уводиться оператор, за допомогою якого можна реалізувати дану конструкцію, конкретизуються його ключові слова, параметри, правила й особливості застосування. Далі пропонується низка завдань, під час розв'язування яких можна використовувати зазначений оператор, причому під розв'язуванням задачі найчастіше розуміється складання програми та її відлагодження. Ця схема логічна і методично грамотна з погляду принципів науковості, зрозумілості, наочності. Але в ній відсутній дуже важливий аспект — початкова зацікавленість учнів у вивченні теми, мотивація.

Ми вважаємо, що метою навчання основ алгоритмізації і програмування повинні бути не формальне вивчення конструкцій мови, а вміння застосовувати одержані відомості для розв'язання практичних життєвих

завдань, тому під час навчання учнів програмування завжди прагнемо використовувати задачний підхід, основними компонентами якого є питання, пов'язані з моделюванням і формалізацією. Уведення кожного нового оператора або функції має відбуватися на фоні розв'язування спеціально підібраних задач.

Якщо перед учнем цікава задача, він отримує додаткову мотивацію для її розв'язування, навіть коли не має достатньої підготовки, і саме відсутність цієї підготовки в подальшому спонукають його отримати цю підготовку. Феномен цікавої задачі врахований відомими майстрами задачі і слова М. Гарднером, Я. Перельманом, Л. Керролом, Г. Остером та ін. На цей позитивний фактор успішного досягнення навчальної мети звернули увагу останнім часом укладачі задач для олімпіад з програмування (ситуативні задачі формулюються так, що, не зважаючи на їх складність, виникає велике бажання їх розв'язати). До того ж природне суперництво у класі, прагнення не відставати від лідерів, які розв'язують задачі для самоствердження в колективі, робить свою справу — сильні учні підтягують за собою інших.

Звідси — основна мета, яку ми ставимо перед собою: побудувати виклад матеріалу так, щоб учні усвідомили, що написання правильної програми не є самоціллю. Під час розв'язування задачі найважливішим є висунення й обґрунтування ідеї її розв'язання (правильна постановка завдання), уміння сконструювати цю ідею, а потім уже формалізувати її засобами мови програмування.

Тому у відборі навчальних задач ми керуємося такими правилами:

- задача, на прикладі якої проводиться пояснення або закріплення матеріалу, вибирається змістовна, практично значуща, цікава;
- для розв'язування задачі вводиться мінімум нових операторів, типів даних, стандартних функцій мови програмування;
- спочатку задача формулюється не чітко, її умова уточнюється в процесі обговорення з учнями (формування початкових даних, результатів);
- задача, на прикладі якої проводиться закріплення вивченого матеріалу, повинна передбачати використання вивченого оператора (або типу даних) у стандартному і модифікованому вигляді, що відображає інші аспекти його застосування;
- у задачі має бути можливість варіювання складністю для забезпечення адекватного навантаження на диференційований за можливостями учнів клас.

У табл. 1 наведемо основні етапи розв'язування задач при використанні задачного підходу в навчанні програмування.

Найважливішим на початковому етапі навчання основ алгоритмізації та програмування, з нашого погляду, є розуміння базових алгоритмічних конструкцій, зокрема розгалуження і повторення, і мовних засобів їх реалізації. Аналіз друкованих Інтернет-джерел із питань використання цікавих задач для вивчення алгоритмізації і програмування показав, що вказана інформаційна база критично мала й існує нагальна необхідність у створенні друкованих збірок і ВЕБ-ресурсів, які будуть допомагати учням і вчителям з питань ефективного засвоєння основ алгоритмізації і програмування. Тож ця стаття є певним кроком у розв'язанні цієї проблеми.

Розглянемо декілька навчальних прикладів, які ми результативно використовували у своїй педагогічній діяльності.

Приклад 1

Метою пояснення нового матеріалу є введення нового оператора мови, який описує алгоритмічну конструкцію «Розгалуження». Пропонуємо таке завдання: «Послуги оператора телефонного зв'язку оплачуються за таким правилом: розмови до 1000 хв. за місяць оплачуються в розмірі B гривень, а розмови понад встановлену норму оплачуються з розрахунку C гривень за хвилину. Скласти алгоритм обчислення платні за користування телефоном для відомого часу розмов за місяць».

Перший етап роботи — створення описової інформаційної моделі. Під час створення такої моделі виділяються істотні параметри для розв'язання задачі, і так ми довизначаємо умову задачі. Уточнення відбувається в процесі діалогу, у результаті якого учні відповідають на такі запитання:

Таблиця 1

Етапи розв'язування задачі	Очікуваний результат	Дидактична мета етапу
Знайомство з текстом задачі і його обговорення	Виявлення основної проблеми навчального завдання	Виділення аспектів, істотних для відповіді на головні питання задачі
Висунення підходів до розв'язування задачі	Вибір методу розв'язування задачі	Формулювання ідеї розв'язування мовою «дані — операції над даними»
Інформаційне моделювання	Побудова інформаційної або математичної моделі задачі	Формування вмінь і закріплення навичок інформаційного моделювання
Виділення основних етапів розв'язування задачі	Побудова формалізованого алгоритму	Формування вмінь і закріплення навичок структуризації алгоритму
Уточнення, які дані (змінні, константи) будуть потрібними для реалізації кожного з етапів	Введення й опис для кожного з етапів основних і допоміжних змінних	Формування умінь і закріплення навичок структуризації даних
Визначення, за допомогою яких засобів мови програмування можна реалізувати незнайомий етап розв'язування	Формалізація найбільш незрозумілого етапу, запис його у вигляді фрагменту програми	Знайомство з новим оператором або функцією. Формалізація етапу розв'язування за правилами оператора, що вводиться
Складання програми з окремих фрагментів	Отримання всіх фрагментів, з яких складатиметься програма	Встановлення і перевірка правильності взаємозв'язків окремих частин цілого
Введення і відлагодження програми	Отримання правильно працюючої програми	Закріплення навичок роботи із середовищем програмування
Обговорення можливості модифікації програми під час розв'язування подібних задач	Висновки про адекватність програми змісту і меті розв'язування початкової задачі	Формування вміння аналізувати й оцінювати результати діяльності

1. Що відомо з умови (тексту) задачі? Назвіть аргументи.

а) 1000 хв. — постійна величина, що позначає межу переходу від одного варіанту оплати до іншого.

б) Вартість оплати цієї норми — B. Це змінна, її потрібно буде визначити на початку алгоритму.

в) Вартість оплати хвилини розмови понад норму — C. Також змінна величина, тому потребує визначення на початку алгоритму.

г) Ще має бути відомим час розмов за місяць для оплати. Позначимо його через V.

2. Що має бути результатом у цій задачі?

а) Сума оплати V хв. розмови. Позначимо її через S.

Наступний етап роботи — це створення математичної моделі, тобто описова інформаційна модель за допомогою математичних формул, рівнянь або нерівностей перетворюється на математичну модель. Ми пропонуємо учням самостійно вибрати метод розв'язування задачі, тобто пригадати відомі їм математичні формули і з їх допомогою скласти модель розв'язування задачі.

Необхідно знайти S. Складаємо вирази для знаходження цієї величини:

якщо $V<=1000$, тоді $S=B$;

якщо $V>1000$, тоді $S=V+C*(V-1000)$.

Потім, увагу учнів концентруємо на тому, що на відміну від лінійних алгоритмів, де дії виконуються строго послідовно одна за одною у порядку їх запису, у нас з'явилася умова, залежно від виконання або невиконання якої формується результат розв'язування задачі.

На наступному етапі роботи математичну модель необхідно перетворити на комп'ютерну — побудувати алгоритм розв'язування задачі і закодувати алгоритм за правилами певної мови програмування. Під час побудови алгоритму вводимо поняття «розгалуження» і новий оператор умовного переходу.

Приклад 2

Для закріплення вивченого матеріалу використовуємо задачу, у якій задіяні два види алгоритмічної структури розгалуження — структура повного розгалуження і структура неповного розгалуження: «До фіналу конкурсу кращого за професією «Фахівець комп'ютерної обробки даних» були допущені троє — Шевченко, Козаченко, Іванчук. Змагання проходили в три тури. У першому турі Шевченко набрав $A1$ балів, в другому — $A2$, в третьому — $A3$, Козаченко — відповідно $B1$, $B2$, $B3$, Іванчук — $C1$, $C2$, $C3$. Скільки балів набрав переможець?».

Розв'язування:

1. Що є відомим з тексту задачі? Назвіть аргументи.

а) Кількість балів, набраних Шевченком: $A1$, $A2$, $A3$.

б) Кількість балів, набраних Козаченком: $B1$, $B2$, $B3$.

в) Кількість балів, набраних Іванчуком: $C1$, $C2$, $C3$.

Якого типу будуть значення цих аргументів?

2. Що є результатом у цій задачі? Кількість балів, набраних переможцем. Позначимо його через MAX.

3. Чи потрібні проміжні величини? Якщо так, тоді позначимо їх відповідно A, B, C — це суми балів, набрані кожним учасником. Запишемо вирази для знаходження цих величин:

$A=A1+A2+A3$,

$B=B1+B2+B3$,

$C=C1+C2+C3$.

Визначимо спочатку більше серед перших двох значень — A, B — і запишемо його до змінної MAX. Потім знайдемо більше серед знайденого значення і третьої величини C. Для того щоб це зробити, в алгоритм необхідно ввести два послідовні розгалуження — повне, з яким ми вже знайомі і неповне, у якому дії виконуються тільки у тому випадку, коли переможцем є третій учасник.

Наступним прикладом цікавих задач для учнів у нашій практиці стали так звані криптарифметичні задачі. Криптарифметична задача — це арифметична рівність, у якій усі або деякі цифри замінені буквами або іншими символами. У таких завданнях потрібно розшифрувати значення кожного символу і відновити числовий запис.

В Індії і Китаї цей вид математичних розваг з'явився більше 1000 років тому. У Європі подібні завдання стали конструювати лише з початку XX століття. У європейських і американських журналах назва криптарифметичні стала загальноприйнятою. У літературі радянського і пострадянського простору їх, зазвичай, називають числовими головоломками або ребусами. Установилися і деякі правила шифрування — різні цифри замінюють різними буквами, іноді застосовують символ *, що може замінювати будь-яку цифру. Бажано, щоб завдання мало єдиний розв'язок. Першим до створення таких завдань причетний відомий англійський укладач математичних головоломок Г. Е. Дьюдені.

Приклад 3

Необхідно розв'язати криптарифметичну задачу two+two=four.

Перший етап процесу розв'язування задачі — створення описової інформаційної моделі. Під час створення такої моделі виділяються істотні параметри задачі. Разом з учнями відповідаємо на вже відомі запитання.

1. Що відомо з умови (тексту) задачі? Назвіть аргументи.

а) two — тризначне число;

б) four — чотиризначне число;

в) t, w, o, f, u, r—цифри цього числа.

2. Що має бути результатом у цій задачі?

а) значення перелічених цифр, а у підсумку числа, що відповідають словам two, four.

Наступний етап роботи — створення математичної моделі. Разом з учнями ми пригадуємо відомі математичні формули, які можна раціонально використати під час розв'язування задачі.

Зазначимо, що w, o, u, r—цифри від 0 до 9, а t та f—цифри від 1 до 9, відповідно до умови задачі.

Число two можна отримати з відповідних цифр так $two=t*100+w*10+o$.

Аналогічно, $four=f*1000+o*100+u*10+r$.

Подальшу увагу учнів фіксуємо на тому, що перебравши всі можливі варіанти цифрових значень літер можна знайти потрібний розв'язок задачі. Комп'ютер виконає перебирання набагато швидше за людину. Перебирання значень однієї цифри організується звичайним *оператором циклу з параметром*. Якщо потрібно перебирати можливі значення для декількох літер раціонально використати конструкцію *вкладених циклів*.

На наступному етапі роботи математичну модель перетворимо на комп'ютерну — побудувавши алгоритм і закодувавши його за правилами мови програмування, наприклад Turbo Pascal. При побудові алгоритму вводимо поняття циклу (багаторазового повторення групи команд), вкладеного циклу, акцентуємо увагу на

особливостях графічного зображення і запису за правилами мови програмування відповідних операторів циклу з параметром. Після попередньої роботи у цьому напрямку отримаємо такий програмний код.

```
var t,w,o,f,u,r,two,four:integer;
begin
 for t:=1 to 9 do
  for w:=0 to 9 do
   for o:=0 to 9 do
    for f:=1 to 9 do
     for u:=0 to 9 do
      for r:=0 to 9 do
      begin
        two:=t*100+w*10+o;
        four:=f*1000+o*100+u*10+r;
        if two+two=four then
        begin
          writeln(' ',two);
          writeln('+');
          writeln(' ',two);
          writeln('————');
          writeln(four);
          writeln;
        end;
end.
```

Але, перевіривши цю програму на працездатність експериментально, легко побачити, що серед отриманих розв'язків багато таких, що не відповідають правилу «різним літерам відповідають різні цифри». Тому програма потребує доопрацювання (оптимізації). Знайомимо учнів з поняттям оптимізації, шукаємо механізми (алгоритмічні й мовні) для її реалізації. Сформулювавши умову й додаючи її перевірку у програму, отримаємо новий варіант програмного коду.

```
var t,w,o,f,u,r,two,four:integer;
begin
 for t:=1 to 9 do
  for w:=0 to 9 do
   for o:=0 to 9 do
    for f:=1 to 9 do
     for u:=0 to 9 do
      for r:=0 to 9 do
       if (t<>w) and (t<>o) and (t<>f) and (t<>u) and (t<>r)
       and (w<>o) and (w<>f) and (w<>u)and (w<>r) and
       (o<>f) and (w<>r) and (o<>f) and (o<>u) and (o<>r)
       and (f<>u) and (f<>r) and (u<>r) then
          begin
            two:=t*100+w*10+o;
            four:=f*1000+o*100+u*10+r;
            if two+two=four then
            begin
              writeln(' ',two);
              writeln('+');
              writeln(' ',two);
              writeln('————');
              writeln(four);
              writeln;
            end;
end.
```

Така програма буде отримувати правильні розв'язки. Але з причини виконання досить великої кількості перевірок (пояснюємо чому це так) складних умов (показуємо у чому ця складність полягає) програма буде працювати повільно. Тому, розбивши умову на частини й усунувши з циклів інваріанти (пояснюємо, що це один з методів так званої оптимізації циклів), отримаємо останню версію найбільш оптимізованої програми, яка дозволить нам побачити розв'язки початково сформульованої задачі.

```
var t,w,o,f,u,r,two,four:integer;
begin
 for t:=1 to 9 do
 for w:=0 to 9 do
 if t<>w then
  for o:=0 to 9 do
  if (t<>o) and (w<>o) then
   for f:=1 to 9 do
   if (t<>f) and (w<>f) and (o<>f) then
    for u:=0 to 9 do
    if (t<>u) and (w<>u) and (o<>u) and (f<>u) then
     for r:=0 to 9 do
     if (t<>r) and (w<>r) and (o<>r) and (f<>r)
     and (u<>r) then
      begin
          two:=t*100+w*10+o;
          four:=f*1000+o*100+u*10+r;
          if two+two=four then
          begin
            writeln(' ',two);
            writeln('+');
            writeln(' ',two);
            writeln('————');
            writeln(four);
            writeln;
          end;
end.
```

Для закріплення матеріалу, що виклав і пояснив учитель учням, пропонується самостійно розв'язати наступні задачі (учням дозволяється користуватися програмою учителя як зразком):

Send+more=money
One+two+five=eight
Zero+one+two=three
Two+three+seven=twelve
Линия+линия=фигура
Задача+анализ=решение
Три+три+один=семь
Корова+трава+доярка=молоко
М*а=т-е=м:а=т:и=к-а
Икс^2=базис
У-р=а:в=н*е=н+и=е
По= $\sqrt{\text{ду}}$ =ма-й
Боря+иди+будь=добр
One+two=three
Три+два=пять

Перефразуючи, інколи ускладнюючи або спрощуючи умови задач, ми маємо змогу результативно реалізовувати принцип диференційованого навчання. А найбільш підготовленим учням рекомендовано виконати проектне завдання, яке стосується не лише конкретної теми, а розширює знання із суміжних тем або навіть спонукає застосувати їх до вирішення проблем інших навчальних предметів. У такий спосіб ми маємо можливість застосувати метод проектів і реалізувати міждисциплінарні зв'язки у навчанні. Наприклад, зараз декілька наших учнів зацікавлено працюють над проектом «Генератор криптарифметичних задач з обмеженим словником».

Наприкінці навчального року або на тижні інформатики ми обов'язково знов звертаємо увагу на розв'язані задачі й виконані проекти з інформатики і програмування, організуємо конкурси індивідуальних програмних розробок, публічно нагороджуємо переможців книжками з популярно викладеними матеріалами з практичного застосування інформаційних технологій і програмування.

Отже, у нас складається послідовна політика свідомого, цікавого і мотивованого вивчення алгоритмізації і програмування як потрібного в сучасному інформаційному суспільстві напряму в інформатиці.