Твердохліб І.А.
НПУ імені М.П. Драгоманова

Використання знань з логічних основ інформатики при вивченні інформатичних дисциплін

Реформування шкільної та вищої освіти значною мірою пов'язується з підвищенням теоретичного рівня змісту навчальних предметів, а фундаментальна підготовка є однією з головних умов професійної освіти.

Вивчення фундаментальних основ теоретичної інформатики, що складають загальноосвітнє ядро цієї галузі знань, повинне займати важливе місце в профільній підготовці майбутніх вчителів інформатики, в формуванні важливих компонентів інформаційної культури та професійних компетентностей майбутніх фахівців.

Варто зауважити, що важлива складова такої фундаментальної підготовки здійснюється значною мірою при вивченні логічних основ інформатики. При цьому формуються відповідні знання, вміння та навички майбутніх вчителів, їхня інформаційна культура та відповідні компетентності, необхідні для проведення базових, профільних та факультативних курсів з інформатики. Знання логічних основ інформатики сприяє розвитку логічного мислення студентів, набуттю вмінь оперувати поняттями та математичним апаратом логіки, широко використовується в програмуванні, в процесі формування запитів на пошук та вибірку в базах даних, пошуку повідомлень в глобальній мережі [1].

Знання з логічних основ інформатики необхідні студентам зокрема, при вивченні програмування, баз даних та електронних таблиць, мережі Інтернет, елементів схемотехніки тощо. В процедурно-алгоритмічному програмуванні використовуються логічні величини та логічні вирази. Використання цих засобів дає змогу розв'язувати складні математичні та логічні задачі, моделювати логіку людського мислення в програмних системах штучного інтелекту. Логічні вирази часто використовують при реалізації алгоритмічних структур розгалуження та циклічних алгоритмічних структур.

В мовах програмування є логічний тип даних. Для прикладу, в мові програмування Pascal є логічний тип даних *Boolean* (змінна може набувати значення *true* або *false* – відповідно в логіці $|A|=1$ або $|A|=0$), при використанні якого студентам потрібні знання з логіки висловлень. При написанні програм використовують прості логічні вирази – два арифметичні вирази, з'єднані знаком порівняння ($x<y$, $x^2+y^2>3$); та складені логічні вирази – прості логічні вирази, до яких застосовані логічні операції *not* (не), *and* (і) *or* (або) (в логічних основах інформатики: $\overline{A}$, $A\wedge B$, $A\vee B$ відповідно). При використанні логічних операцій варто пам'ятати про їх пріоритетність (черговість виконання). Серед зазначених логічних операцій найвищий пріоритет має операція *not*, а найнижчий – операція *or*.

Наприклад, розглянемо задачу, в якій необхідно за довжинами сторін трикутника визначити – трикутник рівнобедрений, рівносторонній чи довільний.

```
program Trukytnuk;
var a,b,c: real;
begin
  writeln('Введіть довжини сторін трикутника');
  readln(a,b,c);
  if (a=b) and (a=c) and (b=c) then write('Трикутник рівносторонній')
  else
  begin
    if (a=b) or (a=c) or (b=c) then write('Трикутник рівнобедрений')
    else write('Трикутник довільний')
  end;
  readln
end.
```

Для написання програми студенти повинні знати структуру опису оператора розгалуження *IF*, вміти застосовувати логічні операції кон'юнкції та диз'юнкції при описуванні умов. Варто зауважити, що логічні операції в мовах програмування процедурного типу використовуються при написанні майже кожної програми.

Як відомо, парадигма логічного програмування є альтернативою до парадигми процедурного програмування. Відмінність логічного програмування від процедурного полягає в тому, що тут потрібно описувати логічну структуру задач, а не вказувати, як розв'язувати ту чи іншу задачу. Логічне програмування базується на концепції логічного виведення, а логіка предикатів першого порядку використовується як мова програмування високого рівня [2]. Таким чином, для використання мов логічного програмування необхідні ґрунтовні знання з логічних основ інформатики, а саме з логіки предикатів першого порядку.

В процесі навчання інформатики студенти використовують знання з логічних основ інформатики при вивченні баз даних, зокрема в реляційних базах даних логічними величинами є поля логічного типу, логічні вирази використовуються в умовах пошуку даних при використанні фільтрів та формуванні запитів.

При вивченні баз даних студенти повинні вміти використовувати логічні типи полів, що призначені для зберігання логічних значень *Істинно* чи *Хибно*. Прикладом використання поля логічного типу є реляційна база даних *УСПІШНІСТЬ,* в якій містяться відомості про середній бал учнів класу і отримані нагороди на олімпіадах, структура якої:

УСПІШНІСТЬ (*УЧЕНЬ, СЕРЕДНІЙ БАЛ, НАГОРОДИ*)

В даному випадку логічний тип даних має поле *НАГОРОДИ*, яке може набувати двох значень *Істинно* чи *Хибно*, яким відповідають наявність чи відсутність нагород в учня відповідно.

В реляційних базах даних також використовуються логічні вирази для задання умов фільтру та запитів. Логічний вираз в базах даних – це деяке твердження, що стосується полів бази даних і може бути істинне або хибне для конкретного поля. Умови відшукання даних поділяють на прості та складені. При створенні простих умов використовують числа, текст і математичні вирази та співвідношення, за допомогою яких описують критерії пошуку даних в даному полі. До складених умов належать умови, утворені з простих умов з використанням логічних операцій заперечення, кон'юнкції та диз'юнкції.

Розглянемо як приклад реляційну базу даних *БІБЛІОТЕКА*, в якій містяться відомості про автора, назву книги, рік видання та номер полиці на якій знаходиться дана книга, реляційна структура якої:

БІБЛІОТЕКА (*АВТОРИ, НАЗВА КНИГИ, РІК ВИДАННЯ, ПОЛИЦЯ*)

Розглянемо приклади простих умов:

а) для числових полів: *>5, =2000, >=8, <>2004*:

б) для текстових полів: *Like"Д*"* або *Д** – усі записи, що починаються з літери *Д, Like[П-Т]* – усі слова, що починаються з *П, Р, С* або *Т*.

Прикладами складних умов можуть бути: *Between 1990 And 2008, not 9, 6 or 9, Like "Д*" And not 2003* тощо. Наприклад при використанні умови відбору *Like "Д*" Or Like "Г*" And <>6* при поданні запиту на вибірку виводиться на екран список книг, прізвища авторів яких починаються з літер *Г* або *Д*, і які не лежать на *6*-й полиці.

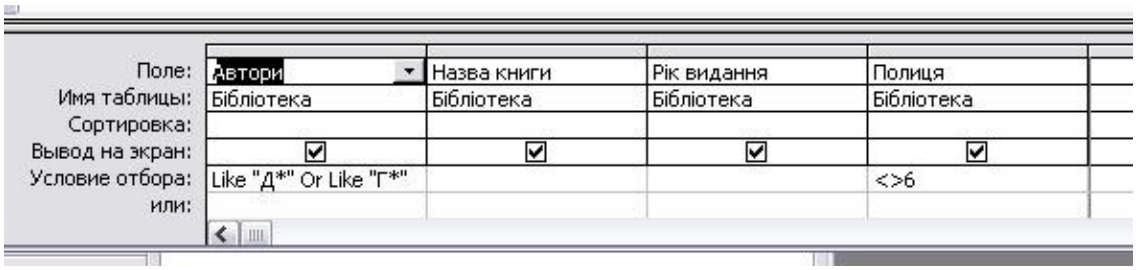

Рис. 1. Приклад складеного запиту

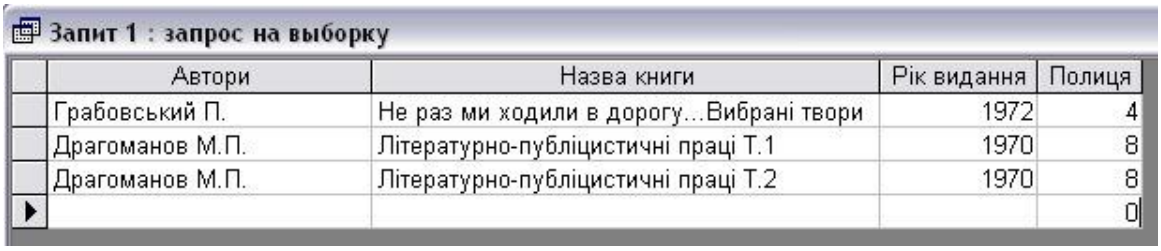

Рис. 2. Результат виконання складеного запиту

Також варто пам'ятати, що умови відбору вводяться у рядок умов. Умови, що стосуються різних полів і пов'язані логічною операцією „*І*" ($F(A,B) = A \wedge B$), записуються в один рядок умов, а умови що пов'язані логічною операцією „Або" ($F(A,B) = A \vee B$) і стосуються одного поля, вводяться одна під одною. Важливою умовою правильної вибірки є правильно розставлені дужки та врахування пріоритету арифметичних та логічних операцій.

Отже, для ефективної роботи з базами даних необхідні знання з логічних основ інформатики, що дає змогу ефективно відшукати потрібні дані при роботі з фільтрами та запитами.

Мову електронних таблиць можна інтерпретувати як своєрідну табличну мову програмування для розв'язування розрахункових задач. Для розв'язування таких задач дуже часто доводиться користуватися знаннями з логічних основ інформатики для реалізації алгоритмів розгалуження, циклічних структур, використання логічних функцій.

Необхідність використовувати елементи логіки зустрічається при роботі з електронними таблицями при використанні *умовного форматування*. При умовному форматуванні умовою може бути значення або формула. Найпростіше форматувати клітинки за значенням. Якщо значення в клітинці порівнюється (>, <, <= тощо) з деяким значенням, тоді застосовується певне форматування цієї клітинки. Наприклад:

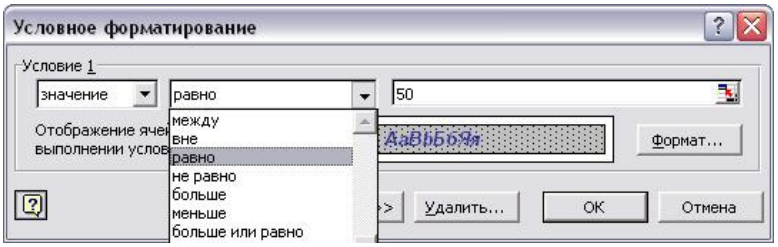

Рис. 3. Умовне форматування за значенням

У вікні форматування клітинок за значенням можна задавати умови, при яких виконується відповідне форматування клітинок, що задається при використанні послуги *ФОРМАТ*. На одну клітинку можна накладати кілька умов форматування, використавши послугу *А ТАКОЖ*.

Складнішим є умовне форматування за формулою. В цьому випадку можна перевіряти вміст одних клітинок, а форматувати інші. Прикладом може бути виділення червоним кольором клітинок, значення в яких більше за середнє.

В електронних таблицях також часто доводиться користуватися логічними функціями, які є аналогами логічних операцій заперечення (*НЕ*), кон'юнкції (*І*) та диз'юнкції (*АБО*), і операцією розгалуження *ЯКЩО*.

Для реалізації алгоритму розгалуження в електронних таблицях використовується логічна функція *ЯКЩО*, яка має три складові: логічне твердження в умові, значення, яке виводиться в разі істинності твердження в умові, та значення, яке виводиться, коли твердження в умові хибне. За допомогою функції *ЯКЩО* перевіряється лише одна умова, тому, в разі необхідності перевірити кілька умов, для організації складних алгоритмів розгалуження, використовується вкладена умова *ЯКЩО*, яку можна вкладати не більше семи разів.

Наприклад, розглянемо розв'язування квадратного рівняння засобами електронних таблиць. Для коректного розв'язування задачі необхідно врахувати 3 випадки в залежності від значення дискримінанта. Для цього використано вкладання функцій *ЯКЩО*.

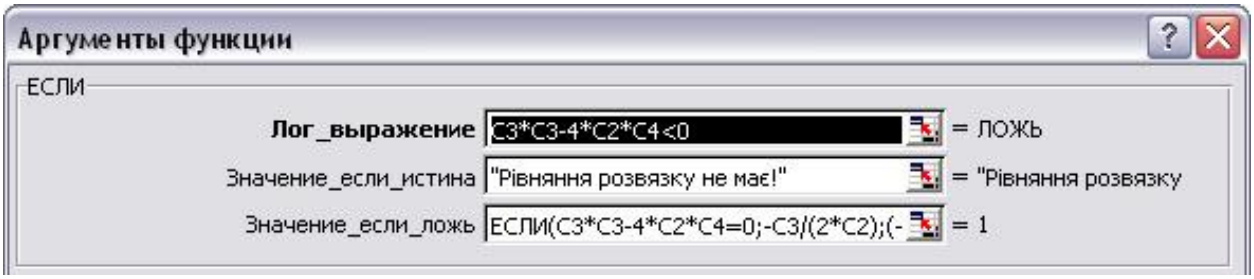

Рис. 4. Вкладання функції *ЯКЩО* в функцію *ЯКЩО*

Для коректного відображення результатів обчислення використано функцію *ЯКЩО* та умовне форматування клітинок. В електронних таблицях використовуються також логічні функції *НЕ, І, АБО*.

Таким чином, використання логічних функцій та умовного форматування дає можливість розв'язувати досить складні розрахункові задачі засобами електронних таблиць.

Сьогодні у веб-просторі зберігаються мільйони документів з неструктурованими даними (текстом, графікою, музикою тощо). Для відшукання потрібних даних користувачеві Інтернет нерідко доводиться витрачати багато часу для перегляду веб-сторінок і пошуку потрібних даних.

Для спрощення і пришвидшення пошуку створені пошукові системи – комплекс програм та надпотужних комп'ютерів, до функцій яких належать: постійний перегляд веб-сторінок, відбір ключових слів та адрес веб-сторінок, у відповідь на запит користувача відбір у своїх базах даних посилань на потрібні сторінки, надсилання знайдених сторінок на комп'ютер користувача.

Для того, щоб сформувати запит, користувачеві необхідно спочатку виділити ключові слова, за якими він буде шукати документ. Ключові слова – це слово чи комбінація слів, які тим чи іншим чином відображають зміст документа. Потім ці ключові слова необхідно ввести в пошукову систему за певними правилами, в основі яких лежить алгебра логіки. Однак варто зауважити, що пошук в Інтернет менше формалізований ніж, в базах даних.

Розглянемо деякі правила виконання запитів на пошук та вибірку в пошукових системах мережі Інтернет. Як приклад розглянемо найпростіший запит: знайти в Інтернет документи, які містять слово „логіка".

Якщо запит буде складатися з кількох слів, записаних через пропуск, то він відповідатиме запиту з використання логічної операції диз'юнкції (*АБО)*. Наприклад, на запит *математична логіка* будуть вибрані всі сторінки, які містять слова „математична", або „логіка", або обидва слова одночасно. Логічна структура запиту має вигляд: $A =$"ìàòåìàòè÷íà", $B =$"ëîã³êà", $F(A,B) = A \vee B$.

Кілька слів, записаних в лапках, сприймаються пошуковою системою як одне ціле. Якщо ж слова з'єднані знаком „+", то запит буде аналогічний використанню логічної операції кон'юнкції (*І*). Наприклад, на запит *математична + логіка* будуть відібрані документи, в яких наявні обидва ці слова. Логічна структура даного запису така: $A =$"ìàòåìàòè÷íà", $B =$"ëîã³êà", $F(A,B) = A \wedge B$. Легко бачити, що кількість таких документів буде не меншою, ніж кількість документів, вибраних за запитом „математична логіка".

Велике значення має застосування логіки висловлень в теорії комбінаційних схем та скінченних автоматів. Використання логіки висловлень в комп'ютерній схемотехніці засноване на тому, що цифрові елементи характеризуються двома станами і через це можуть бути описані за допомогою булевих функцій. Комбінаційні схеми найчастіше будують з елементарних логічних елементів, за допомогою яких реалізують найпростіші логічні операції. Одним із способів фізичної реалізації комбінаційних схем служать релейно-контактні схеми. Ототожнюючи ці операції над контактами з операціями (інверсія, кон'юнкція, диз'юнкція) логіки висловлень, можна утворювати різноманітні логічні функції для контактів так само, як це робилося для висловлень [2]. Отже, знання основних логічних операцій, їх пріоритетності, вміння будувати, аналітично подавати та мінімізувати булеві функції є необхідними для ефективного конструювання та аналізу комбінаційних схем та скінченних автоматів.

Таким чином можна зробити висновок про те, що для вивчення та розуміння багатьох тем та розділів курсу інформатики у вищому навчальному закладі студенти повинні мати міцні знання з логіки висловлень, знати основні логічні операції та вміти будувати логічні формули, а для оволодіння логічним програмуванням – володіти знаннями теорії логіки предикатів першого порядку. Іншими словами, для ефективного засвоєння навчального матеріалу з інформатики, набуття вмінь та навичок роботи з програмними засобами та комп'ютерною технікою, розуміння принципів її функціонування, студенти мають володіти знаннями логічних основ інформатики.

Література

1. Рамський Ю.С. Логічні основи інформатики: навч. посіб. / Юрій Савіянович Рамський. – К.: НПУ імені М.П. Драгоманова, 2003. – 286 с.: іл.

2. Теория и методика обучения информатике: учебник / [М.П. Лапчик, И.Г. Семакин, Е.К. Хеннер, М.И. Рагулина и др.] ; под ред. М.П. Лапчика. – М.: Издательский центр «Академия», 2008. – 592 с.