УДК 628.321

В.М. Заяць
Національний університет “Львівська політехніка”,
кафедра “Інформаційно-комп’ютерних технологій та систем”
zvm@rambler.ru

ОСНОВНІ ПІДХОДИ ДО ОПТИМІЗАЦІЇ ФУНКЦІОНАЛЬНИХ ПРОГРАМ

У статті розглядаються основні підходи та особливості побудови функціональних програм та їх оптимального подання, що дозволяє суттєво підвищити ефективність та читабельність створюваних програмних засобів. Розглянуто метод рекурсій, параметрів нагромадження, додаткових підфункцій та композиції функцій. Показано особливості та переваги кожного з них.

Доцільність використання описаних підходів ілюструється конкретними прикладними розробками, які реалізовано в середовищі стандартного Ліспу, який відноситься до декларативних мов програмування.

Функціональні мови за ефективністю на сучасних комп’ютерах не можуть конкурувати з традиційними мовами, якщо ефективність є абсолютною вимогою. Однак програми, написані з елементним присвоєнням структурованих значень, є програмами низького рівня. Вимагають значних зусиль для їх побудови, а тому є важкими для правильної інтерпретації.

Підвищення швидкості комп’ютерів та місткості їх пам’яті на сучасному етапі робить доступними ряд важливих функційних програм прикладного характеру. Але якісного стрибка у використанні функційних мов можна очікувати лише з появою комп’ютерів нової архітектури, яка орієнтована не на використання технічних засобів, а на особливості самої мови, яка будується, виходячи з природних потреб користувача.

Спискова форма подання даних і програм, яка характерна для функціональних мов програмування, дозволяє успішно оперувати зі значною кількістю арифметичних, логічних та синтаксичних умов і додаткових обмежень при застосуванні кожного із наведених підходів.

Відсутність поняття змінної, операторів присвоєння, циклів, умовних та безумовних переходів в явному вигляді, які характерні для алгоритмічних мов програмування, дає можливість елегантного і легкозрозумілого запису функціональної програми та її доступного тестування.

Застосування розглянутих підходів дозволяє оптимізувати складно структуровану функціональну програму по швидкодії та формі запису, уникнути зациклювань і повторюваності логічних вислідів, що актуально при роботі з великими обсягами даних, обробці символьної інформації та створенні інтерпретаторів інших мов програмування.