Міністерство освіти і науки України
Чернівецький національний університет
імені Юрія Федьковича

Тетяна КАРАВАНОВА

Теорія алгоритмів

Частина 1. Необчислювальні алгоритми

Навчальний посібник

Чернівці
Чернівецький національний університет
імені Юрія Федьковича
2022

УДК 519.16(072)
К 21

Друкується за ухвалою Вченої ради
Чернівецького національного університету імені Юрія Федьковича
(протокол №11 від 31.10.2022 р.).

Рецензенти:

Міца О.В., доктор технічних наук, доцент, зав. кафедри інформаційних управляючих систем та технологій ДВНЗ «Ужгородський національний університет»;

Чернікова Л.А., кандидат педагогічних наук, доцент, проректор з навчально-методичної роботи комунального закладу «Запорізький обласний інститут післядипломної педагогічної освіти» Запорізької обласної ради.

Караванова Т.П.

К 21 Теорія алгоритмів. Частина 1. Необчислювальні алгоритми : навч. посіб. / Т.П. Краванова. Чернівці : Чернівец. нац. ун-т. ім. Ю. Федьковича. 2022. 268 с.

У навчальному посібнику розглядаються питання методики загальної побудови алгоритмів, структур даних, пошукових алгоритмів та сортування. Усі теми супроводжуються питаннями для самоконтролю та завданнями, виконання яких дозволить закріпити новий матеріал.

Для студентів вищих навчальних закладів спеціальностей «Комп'ютерні науки», «Системний аналіз», «Прикладна математика» та інших технічних спеціальнсотей.

УДК 519.16(072)

Від автора

Навчальний посібник «Теорія алгоритмів» є базовим при викладанні відповідного курсу студентам спеціальностей «Комп'ютерні науки», «Системний аналіз», «Прикладна математика» та інших технічних спеціальнсотей. Доповнююча назва посібника «Необчислювальні алгоритми» обумовлена тим, що перш ніж знайомитися з методами розв'язування прикладних задач, необхідно бути обізнаними з основами теорії алгоритмів, як науки, зі структурами даних, уміти визначати, які саме пошукові алгоритми та алгоритми сортування необхідно застосувати у тій чи іншій конкретній задачі. Такі алгоритми можна назвати необчислювальними, оскільки вони є базовими для побудови алгоритмів, що обчислюють результат сформульованої задачі. Необчислювальні алгоритми є платформою для розв'язування задач на графи, лінійного та динамічного програмування тощо.

У посібнику наводиться велика кількість готових фрагментів програм, однак немає жодної готової програми, яка відповідає алгоритму, що вивчається. Саме таку готову до виконання на комп'ютері програму повинен розробити сам читач.

Велика увага також приділена виробленню навичок тестування розроблених алгоритмів. Поради щодо тестування алгоритмів містяться або в теоретичній частині матеріалу, або у завданнях, які обов'язково супроводжують його.

Посібник може бути успішно використаний на спецкурсах, факультативах, гурткових заняттях, а також і самостійно при підготовці студентів до олімпіад з програмування.

Дуже хочеться, аби та позитивна енергетика, яку автор отримував від роботи над посібником, передалася і його читачам.

Тетяна Караванова

Теорія алгоритмів

”...програміст зобов’язаний володіти здібністю першокласного математика до абстракції та логічного мислення у сукупності з едісонівським талантом споруджувати все що завгодно з нуля та одиниці, він повинен поєднувати в собі акуратність банківського клерка з проникливістю розвідника, фантазією автора детективних романів з тверезою практичністю бізнесмена, а, окрім того, мати смак до колективної праці...”

Академік А.П.Єршов

Основні поняття алгоритмів

Історія розвитку алгоритмів

Будь-яка теорія вивчає питання дослідження сукупності ідей, поглядів, концепцій, вчень, уявлень про об’єктивну реальність. Теорія виростає з практики, узагальнює її та обґрунтовується нею. Будь яка теорія є узагальненням об’єктивних фактів, способом опису та поясненням закономірностей реальної дійсності. Серцевину теорії складають закони, що становлять її основу.

Історично перші алгоритми з’явилися разом з появою математики. Саме у IX-му столітті відомий середньоазіатський вчений, філософ, математик Мухаммед ал-Хорезмі у своїх наукових трактатах детально пояснив правила виконання арифметичних дій. При перекладі цих наукових робіт на латину вперше з’явився термін «алгоритм» (ал-Хорезмі – Algorithmi) і використовувався він спершу для визначення правил обчислень у десятковій позиційній системі числення.

Поняття алгоритму належить до основних понять математики. ***Під алгоритмом розуміють скінчену***

послідовність чітко визначених дій, виконання яких призводить до розв'язання всіх задач даного типу.

Варто зазначити, що сформульоване поняття є неточним математичним означенням, оскільки у ньому не визначено, що розуміється під «діями», що таке «задачі даного типу». Самі ж математики, розв'язуючи задачі, будуючи алгоритми, що дозволяли знаходити їх розв'язки, не уточнювали саме поняття алгоритму. В цьому на той час просто не було необхідності. Під алгоритмом завжди розумілася деяка процедура, послідовність дій, яка дозволяла шляхом виконання елементарних кроків отримувати однозначний результат або ж, навпаки, за скінчену кількість кроків дійти висновку, що розв'язку сформульованої задачі не існує.

Довгий час розглядалися алгоритми, що виконували арифметичні операції, операції перевірки рівностей, нерівностей або інших подібних відношень. Але на початку 20-х років ХХст. з'явилися складніші об'єкти, якими оперували алгоритми: матриці, множини, функції тощо. Постали питання щодо трактування поняття елементарності кроків, тлумачення однозначності алгоритму, виникла думка, що не для всяких математичних задач можна знайти алгоритм розв'язання за скінчений проміжок часу. Виникла ідея існування алгоритмічно нерозв'язних проблем.

Таким чином у 30-х роках ХХ століття виникла теорія алгоритмів. Теорія алгоритмів досліджує питання побудови конкретних алгоритмічних моделей, кожна з яких містить конкретний набір елементарних кроків, способів визначення наступного кроку. Завданням теорії алгоритмів є також дослідження питання про існування чи не існування ефективних алгоритмів розв'язання окремих задач. Найбільшу цінність представляють моделі, які одночасно були б і універсальними, і простими.

Згодом бурхливий розвиток обчислювальної техніки, використання її в дослідженнях багатьох наук призвів до розробки великої кількості різноманітних алгоритмів в різних прикладних галузях. Зрозуміло, що всякому алгоритму відповідає задача, яку він призначений розв'язувати, але разом з

тим може існувати не один алгоритм, що розв'язує дану задачу. Такі алгоритми називаються *еквівалентними* і, зрозуміло, постає питання вивчення ефективності цих алгоритмів.

Надалі розглянемо питання ознак ефективності еквівалентних алгоритмів, визначення критеріїв і обчислення цієї ефективності.

Складність алгоритму

В загальній теорії алгоритмів окремо виділяють ***дескриптивну*** складову, що займається лише питаннями існування або відсутності алгоритмів, що приводять до заданої мети та способах задання цих алгоритмів, і ***метричну*** складову, що займається оцінюванням складності процесів обчислення протягом виконання алгоритму.

Складність алгоритму – це кількісна характеристика. Вона визначається часом, за який виконується алгоритм (часова складність), та об'ємом пам'яті комп'ютера, необхідного для його виконання (ємкісна складність). Тому про складність алгоритму логічно вести мову стосовно саме машинних алгоритмічних моделей. Оскільки подолання обмежень на пам'ять комп'ютера – технічна проблема, що вирішується на рівні його вдосконалення майже що півроку, то часова складність алгоритму, яка в більшій мірі залежить від обраної алгоритмічної моделі, методу реалізації задачі – це творча, евристична проблема.

Саме оцінці часової складності алгоритмів і буде присвячена левова частина матеріалу цього посібника, оскільки майстерність та високий фаховий рівень алгоритміста визначається саме його умінням розробити найоптимальніший за часовою складністю алгоритм розв'язання поставленої замовником задачі.

Класи алгоритмів

Визначивши основне поняття алгоритму, можна зробити висновок, що будь-який алгоритм є послідовністю деяких вказівок. Але не кожна така послідовність може називатися алгоритмом.

Сучасне поняття терміну «алгоритм» більш широке. Воно для багатьох співзвучне зі словами метод, спосіб, процедура, програма. Можна сказати, що алгоритм – це чітко сформульована інструкція, а інструкції зустрічаються практично у всіх сферах нашого життя.

Алгоритм є фундаментальним поняттям інформатики. Науковці виділяють три основних класи алгоритмів: ***обчислювальні, інформаційні та керуючі.***

Обчислювальні алгоритми – це такі, в ході виконання яких проводяться обчислення. Обчислювальні алгоритми працюють з числами або з їх наборами — векторами, матрицями, множинами. Переважно, це алгоритми, які опрацьовують математичну інформацію.

Інформаційні алгоритми працюють з великими об'ємами інформації. Прикладом може слугувати алгоритм пошуку необхідної числової або символьної інформації, що відповідає певним ознакам. Ефективність роботи цих алгоритмів залежить від організації даних, їх структури, представлення. Прикладом є відомі сьогодні всім бази даних, інформаційні та експертні системи тощо.

Керуючі алгоритми характерні тим, що дані до них надходять від зовнішніх процесів, якими вони керують. Результатами роботи цих алгоритмів є вироблення своєчасного необхідного керуючого сигналу як реакції на швидку зміну вхідних даних. Саме тому значення даних у ході виконання керуючих алгоритмів змінюються, подекуди навіть досить швидко, і алгоритм повинен своєчасно правильно прореагувати на це, тобто видати потрібний керуючий сигнал у потрібний момент часу. Примітивними і найпростішими прикладами таких керуючих алгоритмів є ті, що використовуються у побутовій техніці, на виробництві для автоматизації процесів на них для звільнення від використання ручної праці.

Основні властивості алгоритму.

Дискретність – будь-який алгоритм зображується у вигляді окремих вказівок.

Виконання команд алгоритму повинно бути послідовним, з точною фіксацією моментів завершення виконання однієї команди і початку виконання наступної.

Скінченність – виконання алгоритму завершується після завершення кінцевої кількості кроків.

У математиці існують обчислювальні процедури, які мають алгоритмічний характер, але не володіють властивістю скінченності. Прикладом такої процедури може бути обчислення значення числа π. Така процедура описує нескінченний процес і ніколи не завершиться. Але якщо обмежитися певною кількістю знаків після коми, то ми отримаємо алгоритм обчислення числа π з заданою точністю.

Визначеність – кожний крок алгоритму повинен бути чітко і недвозначно визначений, не повинен допускати довільного трактування виконавцем.

Тобто алгоритм розрахований на механічне виконання. Якщо один і той самий алгоритм доручити для виконання різним виконавцям, то вони повинні дійти одного і того ж результату.

Зрозумілість – формулювання дій алгоритму повинно бути орієнтоване на конкретного виконавця.

Якщо виконавець є некомпетентною людиною в питаннях, що вирішуються даним алгоритмом, то необхідно вибрати доступнішу для його формулювання форму, якою є словесний спосіб. Наприклад, якщо алгоритм передбачає обчислення коренів деякого квадратного рівняння, то для учня початкових класів необхідно самим детальним чином описати всі виконувані дії на рівні, що відповідає його знанням математики: додавання, віднімання, множення, ділення, добування квадратного кореня за допомогою калькулятора тощо. Для учня старших класів цей алгоритм буде містити математичну формулу для обчислення коренів рівняння та аналіз їх існування та відсутності.

Якщо ж виконання алгоритму буде запропоновано комп'ютеру, то алгоритм потрібно зобразити насамкінець відповідною мовою програмування.

Масовість – в алгоритмі повинна бути передбачена можливість виконання його для різних початкових значень.

Цим самим забезпечується його використання для розв'язування цілого класу однотипних задач.

Ця властивість відрізняє алгоритм від задачі, що розв'язується за допомогою калькулятора. Прикладом може слугувати наступний. На калькуляторі можна порахувати значення тільки, наприклад, конкретного виразу «2+2». У той самий час алгоритм задачі цього класу виглядатиме як програмування виразу «z=x+y», де змінні *x* та *y* можуть набувати будь-яких вхідних числових значень,а змінна*z*отримуватимиме вихідний результат розв'язання даної задачі.

Результативність – алгоритм повинен забезпечувати обов'язкове отримання результату після кінцевої кількості кроків.

Тобто кожна дія повинна бути достатньо простою, щоб її можна було виконати точно і за скінчений проміжок часу.

Способи представлення алгоритмів

Існує ***чотири способи запису алгоритмів***, вибір яких залежить від того, хто його складає або на кого він орієнтований:

- словесний спосіб запису алгоритмів;
- запис алгоритмів за допомогою схем;
- опис алгоритмів мовою псевдокодів;
- запис алгоритмів мовою програмування.

Словесний спосіб запису алгоритмів орієнтований на людину-виконавця.

Мабуть, поки що важко уявити собі інший спосіб запису. Це найбільш проста і доступна форма представлення алгоритму. Словесна форма зазвичай використовується для алгоритмів, орієнтованих на виконавця – людину і цей спосіб є найбільш доступним будь-кому незалежно від його спеціальної підготовки.

Повертаючись до історії трансформації поняття алгоритм, слід зазначити, що ще до 1950 року під цим словом частіше за

все розуміли викладений в «Елементах» Евкліда алгоритм Евкліда – процес знаходження найбільшого спільного дільника (НСД) двох натуральних чисел. Тому буде корисно навести для прикладу опис цього алгоритму.

1) Взяти два натуральних числа. Якщо вони рівні, то перше з них і є найбільшим спільним дільником, якщо ж ні, то перейти до пункту 2.
2) Порівняти два числа і вибрати більше з них.
3) Більше з двох чисел замінити різницею більшого і меншого.
4) Перейти до пункту 1.

Зверніть увагу, що у першому пункті конкретно сказано, яке саме число необхідно вибрати у випадку збігу двох чисел. Саме це і є характерною рисою алгоритму, виконавцем якого може бути навіть не підготовлена в даній галузі людина.

Наведений алгоритм застосуємо до конкретної пари чисел.

Нехай задані два числа: 45 та 12.

Продемонструємо процес знаходження НСД за наведеним алгоритмом у вигляді таблички, де на кожному кроці більше число, від якого віднімається друге (менше) число, виділятимемо жирним шрифтом.

1)	**45**	12	5)	**9**	3
2)	**33**	12	6)	**6**	3
3)	**21**	12	7)	3	3
4)	9	**12**			

Отже, за сім кроків одержано результат: **НСД(45,12)=3.**

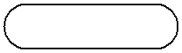
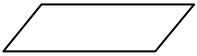
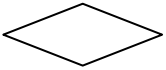
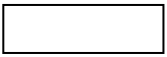
Перевірка роботи алгоритму є суттєвим кроком на шляху до його розуміння. Саме такий підхід допоможе найкраще зрозуміти роботу будь-якого алгоритму.

Схеми дозволяють зобразити алгоритм в наочній графічній формі.

Цей спосіб вже вимагає деяких знань. Вони полягають у знайомстві зі спеціальними стандартами графічних зображень блоків, в середину яких поміщаються команди алгоритму.

Наведемо таблицю найпростіших і найвживанішихблоків.

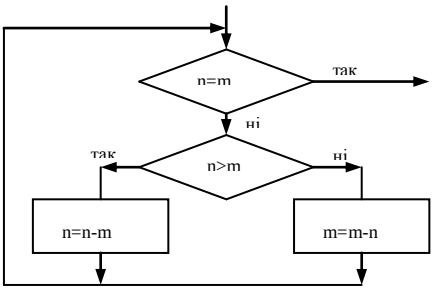
Таблиця 1

	Початок та кінець алгоритму
	Введення або виведення даних
	Вибір напрямку виконання алгоритму в залежності від виконання умови
	Виконання операцій, в результаті яких відбувається зміна значення даних

При створенні схеми алгоритму блоки із записаними в них командами з'єднуються між собою стрілками, які визначають черговість виконання дій алгоритму.

Для запису команд всередині блоків використовується природна мова з елементами математичної символіки. У результаті перевірки умови під час вибору напрямку виконання алгоритму виникають два можливі шляхи для його продовження. Ці шляхи зображуються стрілками з позначеннями «так» (+) та «ні» (–). Перехід по стрілці з позначенням «так» відбувається у випадку, коли умова виконується, а перехід по стрілці з позначенням «ні» – у протилежному випадку.

Блоки початку і кінця алгоритму використовуються при записові повного алгоритму задачі. Надалі вважатимемо, що алгоритми, які розглядаються в посібнику як базові, можуть бути самі використані в інших алгоритмах як їх фрагменти. Тому блоки початку і кінця алгоритму в таких випадках нами використовуватись не будуть. Для запису алгоритмів, з якими матимемо справу, даної інформації цілком достатньо. Зобразимо у визначених позначеннях



Мал.2

алгоритм Евкліда. Слід погодитися, що наочність схематичного представлення алгоритму (мал.2) має значні переваги.

Однак ця наочність швидко втрачається, коли зображується великий алгоритм. У таких випадках У схемі алгоритму виділяються і відокремлюються її окремі частини – модулі, основною умовою яких є один вхід і один вихід. Згодом вони включаються у схему алгоритму як окремі блоки. Такий підхід до складання алгоритму відображає ідею структурного програмуванняу вигляді створення функцій, заголовних файлів, допоміжних алгоритмів тощо.

Для запису алгоритмів за допомогою *мови псевдокодів* використовуються службові слова та спеціальні правила запису окремих дій.

У мові псевдокодів прийняті певні синтаксичні правила для запису команд, що полегшує запис алгоритму на стадії його проектування і дає можливість використання більш широкого набору команд, розрахованого на абстрактного виконавця.

Наприклад, у мові псевдокодів використовуються спеціальні службові слова та правила запису для вибору напрямку подальшого виконання алгоритму в залежності від виконання чи невиконання сформульованої умови **ЯКЩО**...**ТО**...**ІНАКШЕ**, повторення визначеної групи дій певну кількість разів **ПОКИ**...**ПОЧАТОК**...**КІНЕЦЬ** тощо.

Разом з тим мова псевдокодів дозволяє використовувати дещо довільну форму представлення математичних записів, умов тощо.

Розглянемо як приклад попередньо наведений алгоритм Евкліда. Мовою псевдокодів він може виглядати таким чином:

```
АЛГОРИТМ найбільший_спільний_дільник
ПОЧАТОК
ВВЕДЕННЯ «Задайте два натуральних числа», n,m
    ПОКИ n≠m
        ПОЧАТОК
            ЯКЩО n>m
                ТО n: =n-m
                ІНАКШЕ m: =m-n
            ВСЕ
        КІНЕЦЬ
    ВИВЕДЕННЯ „Найбільший спільний дільник заданих чисел: ”, n
КІНЕЦЬ
```

Зміст

Для нотаток

Навчальне видання

Караванова Тетяна Петрівна

ТЕОРІЯ АЛГОРИТМІВ
ЧАСТИНА 1. НЕОБЧИСЛЮВАЛЬНІ АЛГОРИТМИ

Навчальний посібник

Відповідальний за випуск – ***І. М. Черевко***

Літературний редактор – ***О. В. Лупул***
Технічний редактор – ***Т. П. Караванова***

Підписано до друку 14.09.2022. Формат 60x84/16.
Папір офсетний. Друк різографічний. Умов.-друк. арк.14,7.
Обл.-вид. арк. 15,8. Тираж 00. Зам. Н-000п.
Видавництво та друкарня Чернівецького національного університету.
58012, Чернівці, вул. Коцюбинського, 2.
e-mail: ruta@chnu.edu.ua

Свідоцтво суб'єкта видавничої справи ДК № 891 від 08.04.2002.