§ 54. Алгоритм, його властивості й форми подання

Вивчивши цей параграф, ми:

дізнаємося, що таке алгоритм;

з'ясуємо, чому виконання алгоритму можна доручити комп'ютеру;

познайомимося з різними формами подання алгоритму;

довідаємося, що таке алгоритмічні мови і для чого вони створюються.

====54.1. Алгоритм і його виконавець==========

Алгоритми відіграють велику роль у нашому житті, тому що вони акумулюють здобутий людством досвід ефективних способів діяльності.

У процесі виховання, навчання, фізичного тренування, набуття власного життєвого досвіду ми засвоюємо різні алгоритми — правила поведінки в повсякденному житті й у кризових ситуаціях, способи розв'язання математичних задач і виконання хімічних дослідів, виграшні стратегії ігор, інструкції для роботи з програмними засобами й різноманітними технічними пристроями та багато інших. Один раз засвоєним алгоритмом ми користуємося все своє життя, і це зберігає наші сили, час, дозволяє досягти успіхів у тій чи іншій діяльності.

Під алгоритмом розуміють опис послідовності дій, які потрібно виконати для досягнення поставленої мети або здобуття необхідного результату. Будь-який алгоритм складається з розрахунком на певного ***виконавця***, якому належить виконувати приписані дії. Кожна дія алгоритму спричиняється окремою *вказівкою*. Вказівки алгоритму називаються *командами*. Отже, виконавець алгоритму має бути здатним зрозуміти та виконати кожну з команд алгоритму. Так, наприклад, алгоритм розв'язання квадратного рівняння розрахований на виконавця, який знає, що таке дискримінант, як обчислити квадратний корінь тощо.

Сукупність команд, які виконавець може зрозуміти й виконати, називається ***системою команд виконавця.***

Якщо алгоритм орієнтовано на виконання людиною, то до формулювання команд жорсткі вимоги не висуваються. Достатньо, щоб людина точно зрозуміла, що саме від неї вимагається. Так, обидві команди: «знайди добуток чисел 5 і 7» і «помнож 5 на 7» людина зрозуміє і виконає однаково.

Якщо ж виконавцем алгоритму є технічний пристрій, зокрема комп'ютер, то в такому разі слова «зрозуміти команду» означають ***розпізнати*** (або *ідентифікувати*) команду. Ідентифікацією (від лат. identicus — тотожний і ficatio — роблю) називається ототожнення об'єкта з одним із відомих системі.

Комп'ютер має певний обмежений перелік точно визначених команд, які він може ідентифікувати й виконати. Ніякі відхилення від правил запису команд не припустимі. Комп'ютер у точності виконує надані йому команди й не коригує наших помилок. У разі неможливості ідентифікації або виконання команди він лише вказує на наявність помилки і припиняє роботу над алгоритмом.

Для виконання алгоритму виконавцеві звичайно недостатньо самого алгоритму. Виконати алгоритм означає застосувати його до розв'язання конкретної задачі, тобто здійснити заплановані дії стосовно певної сукупності вхідних даних. *Вхідними* даними називають такі, які надаються виконавцеві до початку його роботи над алгоритмом. На відміну від них дані, які виконавець видає як результат здійсненої роботи, називають *вихідними*. Проте в процесі роботи виконавець може виробляти й використовувати такі дані, які не є результуючими. Це *проміжні* дані.

Наприклад, для розв'язання квадратного рівняння виконавцю надаються вхідні дані — коефіцієнти рівняння, за якими він обчислює корені рівняння — вихідні дані. У ході виконання алгоритму було знайдено і використано значення дискримінанта — проміжне дане.

====54.2. Властивості алгоритму==========

Не будь-яка інструкція чи послідовність вказівок є алгоритмом. Алгоритм має певні характеристичні особливості, які називають *властивостями* алгоритму.

Першою властивістю, притаманною алгоритму, є ***дискретність***. Дискретність означає подання процесу розв'язання задачі у вигляді послідовності відокремлених простих дій, які виконуються одна за одною в певному порядку. Тільки після завершення виконання однієї дії можна перейти до наступної.

Другою властивістю алгоритму є ***скінченність***, або *результативність.* За скінченну кількість кроків алгоритм приводить до здобуття потрібного результату або до досягнення поставленої мети.

Третьою властивістю алгоритму є його ***зрозумілість*** виконавцеві. Алгоритм складається тільки з таких команд, які входять до системи команд його виконавця.

Важливою властивістю алгоритму є ***детермінованість*** (від лат. determinare — визначати), або *визначеність.* Кожна команда алгоритму точно й однозначно описує дії виконавця — що йому треба зробити на даному кроці та до якої команди перейти на наступному. У виконавця не може виникнути потреби в будь-якій додатковій інформації. Можливість самостійного прийняття виконавцем будь-яких рішень виключена. Таким чином, алгоритм приписує *точне*, тобто без будь-яких відхилень, і *формальне*, тобто без втручання у смисл, виконання його команд.

Із детермінованості алгоритму випливає, що для заданої сукупності вхідних даних будь-який його виконавець, аби він мав відповідну систему команд, неодмінно досягне одного й того самого результату шляхом виконання одного й того самого ланцюжка дій. Саме це й дозволяє використовувати автоматизовані пристрої, зокрема комп'ютер для виконання алгоритмів.

Ще однією властивістю алгоритму є ***масовість***. Алгоритм не складається для однієї конкретної задачі. Він розраховується на розв'язання всіх задач певного типу, які відрізняються між собою вхідними даними. Із смислу задачі, способу її розв'язання, реалізованого в алгоритмі, звичайно випливають певні обмеження на допустимі значення вхідних даних. Ці обмеження визначають можливості застосування алгоритму.

Отже, властивостями алгоритму є дискретність, скінченність, зрозумілість, детермінованість, масовість. Тепер, спираючись на ці властивості, визначимо, що таке алгоритм.

Алгоритм *— це точний і зрозумілий для виконавця припис про здійснення скінченної послідовності визначених дій з метою розв'язання задачі певного типу або досягнення поставленої мети.*

Прикладом алгоритму, який дає можливість продемонструвати всі його властивості, є алгоритм збирання автомобіля на конвеєрі. Дійсно, процес збирання розкладається на послідовність окремих операцій. Розпочати наступну операцію можна тільки після завершення попередньої. Послідовність операцій є точно встановленою, змінити їх порядок не можна (наприклад, спочатку здійснюється закріплення корпусу, і тільки потім встановлення двигуна). На кожному етапі збирання точно визначено, яка саме операція має виконуватися. На конвеєрі працює бригада робітників (виконавець). Вони розуміють, що треба робити, і можуть виконати потрібні операції. Будь-яких самостійних рішень приймати не треба — всі дії чітко визначені та сплановані. Бригада має комплект деталей (вхідні дані), необхідних для збирання конкретної моделі автомобіля. Ніякі інші деталі не знадобляться. За певний термін усі етапи збирання будуть завершені, і ми отримаємо результат виконання алгоритму — готовий автомобіль з'їжджає з конвеєра!

====54.3. Словесна й таблична форми подання алгоритму==========

Алгоритм може бути поданий у різних формах.

У ***словесній формі*** алгоритм записується як послідовність занумерованих словесних команд. Команди виконуються в порядку зростання їх номерів. Якщо потрібно змінити цей

порядок, застосовується спеціальна вказівка на перехід до виконання команди із заданим номером. Словесна форма алгоритму найчастіше використовується в людському спілкуванні, в інструкціях користувачам програмних засобів або побутових приладів тощо.

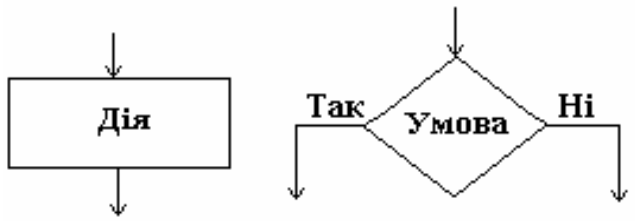
Записом алгоритму можна вважати *формулу*, тому що з неї випливає порядок здійснення обчислень для отримання числового результату. Якщо виконується серія розрахунків за однаковими формулами, то для запису алгоритму застосовується ***таблична форма***. За табличною формою в певних стовпцях таблиці розміщуються вхідні дані, а значення в наступних стовпцях — проміжні й вихідні — обчислюються за відповідними формулами. Таким чином, таблиця відбиває етапи виконання алгоритму. Ми використовували табличну форму подання алгоритму при роботі з електронними таблицями Excel.

====54.4. Блок-схема==========

Поширеною формою наочного подання алгоритму є ***блок-схема***.

Блок-схема складається з окремих геометричних фігур — блоків, які з'єднуються напрямленими лініями. Лінії показують послідовність переходу від одного блока до наступного.

Для подання алгоритму застосовуються блоки двох видів: ***функціональні*** й ***альтернативні*** (рис. 54.1).



а) б)

Рис. 54.1. Основні елементи блок-схеми алгоритму: а) функціональний блок; б) альтернативний блок

Функціональні блоки (рис. 54.1а) позначаються прямокутниками. Одна напрямлена лінія входить у прямокутник і одна виходить із нього. Всередині прямокутника розміщується команда, яка має бути виконаною.

Альтернативні блоки позначаються ромбами (рис. 54.1б). Слово «альтернативний» (від лат. alternare — чергуватися) означає «такий, що припускає одну з двох можливостей, які виключають одна одну».

Усередині ромба розміщується *умова*. Умовою називають будь-яке висловлювання, яке може мати тільки одне з двох значень — «істина» або «хибність». Наприклад, у ролі умов можуть бути висловлювання: «колір прапорця помаранчевий», «z — двозначне число», «зріст хлопця не нижче 170 см», «x = 7», «a >= b» тощо.

В альтернативний блок входить одна напрямлена лінія, а виходять з нього дві — з надписами «так» і «ні». Умова, записана всередині ромба, піддається перевірці. За результатом перевірки — «істина» («так») або «хибність» («ні») — здійснюється вибір відповідної напрямленої лінії, тобто вибір одного з двох можливих варіантів продовження виконання алгоритму.

а) б)

Рис. 54.2. Допоміжні елементи блок-схеми алгоритму:

Крім функціональних та альтернативних блоків, у блок-схемах застосовуються також ***допоміжні блоки*** для позначення початку або кінця алгоритму (рис. 54.2а) і введення або виведення даних (рис. 54.2б).

====54.5. Алгоритмічні мови==========

Блок-схеми є зручними для наочного подання алгоритмів, проте для складних алгоритмів вони стають громіздкими. Більш компактною формою подання алгоритму є його опис на *алгоритмічній мові.*

Алгоритмічною мовою *називається штучна мова, призначена для подання алгоритмів.*

Алгоритмічні мови дозволяють подати алгоритм у вигляді тексту, який складається за певними правилами із застосуванням спеціальних слів, які називаються *службовими*. Кількість таких слів обмежена. Кожне службове слово має точно визначений смисл, призначення і спосіб застосування. Службові слова добираються так, щоб їх смисл в алгоритмічній мові перекликався із звичайним значенням слова. Алгоритм, поданий алгоритмічною мовою, легко сприймається, тому що він читається як звичайний текст.

Можна виділити три види алгоритмічних мов. До першого відносимо мови, розраховані на подання алгоритму у вигляді, зручному для його усвідомлення людиною. Якщо ми відкриємо книги, присвячені алгоритмам, то переконаємося, що різні автори застосовують різні алгоритмічні мови, які вони вважають зручними для презентації алгоритмів читачам. Основною функцією алгоритмічних мов є відтворення логічної будови алгоритму, що є найважливішим для розробки алгоритму й найскладнішим для його усвідомлення. Подання логічної будови здійснюється за чітко визначеними правилами, і саме для цього вводяться й застосовуються службові слова. До подання дій у таких мовах звичайно не висувається жорстких вимог, припускається порівняно вільний їх опис.

Другий вид алгоритмічних мов складають мови, призначенням яких є подання алгоритму в формі, придатній для однозначного перетворення алгоритму на мову комп'ютерних команд. Такі мови є своєрідним компромісом між людиною й комп'ютером. Людина робить крок назустріч комп'ютеру, бо подає алгоритм, використовуючи обмежений набір засобів мови в точній відповідності до встановлених правил. Комп'ютер «навчається» розпізнавати інструкції алгоритму і трансформувати їх у комп'ютерні команди.

Алгоритмічна мова, конструкції якої однозначно перетворюються на команди комп'ютеру, називається ***мовою програмування*** (від грец. programma — розпорядження, припис, оголошення). *Текст алгоритму, записаний мовою програмування, називається* ***програмою***.

Історично склалося так, що мови програмування є англомовними в тому сенсі, що їх службові слова є словами або скороченнями слів англійської мови.

За роки використання комп'ютера було створено тисячі мов програмування, проте лише деякі з них отримали всесвітнє визнання, для них було затверджено стандарт мови. Одним з найважливіших наслідків створення стандартизованих мов програмування стала можливість для фахівців різних країн обмінюватися програмами. Це сприяло розвитку культури програмування, становленню технології розробки програм, формуванню фонду програмних продуктів.

Надзвичайна різноманітність мов програмування пояснюється тим, що кожна з них має свої переваги для подання алгоритмів того чи іншого призначення. Проте всім мовам програмування притаманна одна спільна властивість: вони надають можливість застосувати комп'ютер для вирішення будь-якої задачі, якщо складено алгоритм її розв'язування.

Алгоритмічні мови третього виду займають проміжну позицію між мовами першого і другого видів. Це мови навчального призначення. З одного боку, вони полегшують сприйняття алгоритму людиною й надають їй такі засоби подання алгоритму, які добре узгоджуються з природним процесом обмірковування потрібних дій. З іншого боку, такі мови є певною сходинкою до мов програмування, тому що конструктивно мають з ними багато схожих рис.

Алгоритмічна мова, створена для вивчення правил і прийомів складання алгоритмів, називається ***навчальною алгоритмічною мовою*** (*НАМ*). Службові слова навчальної алгоритмічної мови є словами або скороченнями слів української мови. Це спрощує як читання і розуміння алгоритмів, написаних мовою НАМ, так і складання алгоритмів засобами НАМ. Важливим достоїнством НАМ є те, що її службові слова, конструкції і правила запису алгоритму подібні до тих, які є характерними для поширених мов програмування. Завдяки такій схожості оволодіння НАМ є кроком до реального програмування.

Далі ми ознайомимося з НАМ і мовою програмування Паскаль.

ВИСНОВКИ

З алгоритмами ми зустрічаємося на кожному кроці. Вони вказують, яку послідовність дій необхідно виконати, щоб досягти потрібного результату. Кожний алгоритм розраховується на виконавця, який спроможний зрозуміти й виконати команди алгоритму. За алгоритмом процес розв'язання задачі постає як виконання скінченної послідовності окремих простих команд, кожна з яких точно визначає дії виконавця. Це розкриває шлях до створення технічних пристроїв, які сприймають і виконують алгоритми. Алгоритми подають у різних формах. Для наочного подання алгоритму використовуються блок-схеми. Подання алгоритму у вигляді тексту здійснюється за допомогою штучних мов. Навчальні алгоритмічні мови створюються для засвоєння основ складання алгоритмів. Мови програмування орієнтовані на їх «розуміння» комп'ютером, який у такому разі виступає виконавцем алгоритму.

Контрольні питання та вправи

1. Уставте слова, пропущені у наведеному нижче визначенні алгоритму.

«Алгоритм — це … і … припис про здійснення скінченної послідовності визначених дій з метою розв'язання задачі певного типу або досягнення поставленої мети».

а) чіткий;
б) точний;
в) зрозумілий для виконавця;
г) короткий;
д) простий;
е) доступний.

2. Властивостями алгоритму є:

а) зрозумілість;
б) повнота;
в) лаконічність;
г) масовість;
д) дискретність;
е) детермінованість;
є) формальність;
ж) скінченність;
з) правильність.

3. Зрозумілість алгоритму означає, що:

а) кожна його команда є зрозумілою для всіх;
б) кожна команда алгоритму є зрозумілою для його виконавця;
в) кожна його команда є простою для виконання;
г) виконавець розуміє й може виконати кожну команду алгоритму;
д) кожна команда алгоритму може бути виконаною.

4. Детермінованість алгоритму означає, що:

а) алгоритм дає виконавцю все, що йому потрібно для розв'язання завдання;
б) алгоритм обмежує дії виконавця;
в) алгоритм однозначно визначає всі дії виконавця;
г) алгоритм гарантує розв'язання задачі за скінченну кількість кроків.

5. Дискретність алгоритму означає, що він подає процес розв'язання задачі у вигляді:

а) сукупності окремих простих дій;
б) визначеної послідовності окремих простих дій;

в) певної множини зрозумілих дій;
г) скінченної множини дій, які виконавець може виконати.

6. Масовість алгоритму означає, що його можна застосовувати:

а) для розв'язання будь-яких задач;
б) для розв'язання будь-яких задач певного класу;
в) для розв'язання задач певного класу з будь-якими вхідними даними;
г) для розв'язання задач певного класу, які мають вхідні дані, що належать до кола допустимих для даного алгоритму.

7. Дайте відповіді на питання:

а) з яких елементів складається блок-схема;
б) що таке функціональний блок у блок-схемі, яке його призначення;
в) що таке альтернативний блок у блок-схемі, яке його призначення;
г) що таке умова;
д) які переваги й недоліки має подання алгоритму за допомогою блок-схеми?

8. Дайте відповіді на питання:

а) з якою метою створюються алгоритмічні мови, які переваги вони мають у порівнянні з блок-схемами;
б) які особливості притаманні навчальній мові;
в) чим відрізняються мови програмування від інших алгоритмічних мов?

9. Індійський математик Д. Капрекар довів, що будь-яке чотиризначне число, у якого не всі цифри однакові, в результаті перетворень за певним алгоритмом приводить до одного й того самого числа, яке отримало назву сталої Капрекара. Алгоритм має такий вигляд:

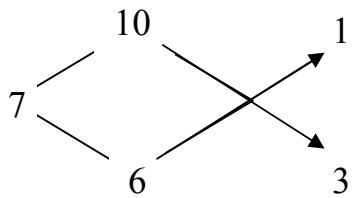
1) переставити цифри числа N за спаданням;
2) переставити цифри числа N за зростанням;
3) відняти від першого числа друге;
4) якщо отримана різниця R не дорівнює числу N, то повторити з нею дії 1)—3).

Через не більш як 7 повторень алгоритм приводить до числа, яке далі переходить само в себе. Це і є стала Капрекара. Знайдіть її.

10. Існує старовинний алгоритм визначення пропорції, у якій потрібно змішати дві речовини різних концентрацій для одержання речовини заданої концентрації. Алгоритм приписує виконання таких дій:

1) одну під одною запишіть концентрації розчинів, які маєте; ліворуч між ними — ту, що потрібна;
2) знайдіть різницю між більшою концентрацією та потрібною і запишіть її праворуч від меншої концентрації;
3) знайдіть різницю між потрібною концентрацією та меншою і запишіть її праворуч від більшої концентрації;
4) прочитайте відповідь: числа справа визначають пропорцію, у якій треба змішати відповідні розчини.

Наприклад, для одержання 7% розчину з наявних 10% і 6% маємо такий запис:



Отже, для отримання 7% розчину слід узяти 1 частину 10% та 3 частини 6% розчинів.

Користуючись наведеним алгоритмом, знайдіть відповіді на наведені нижче питання:

а) Як отримати 3 літри молока жирністю 3,2%, якщо маємо молоко 1,5% та 4,5% жирності?

б) Як треба змішати чаї двох сортів — ціною 64 та 82 грн. за кг, щоб утворити суміш ціною в 70 грн. за кг?

в) Скільки цукерок ціною по 12 грн. 50 коп. за 1 кг потрібно додати до 20 кг цукеркової суміші ціною по 9 грн. 15 коп. за 1 кг, щоб довести ціну суміші до 10 грн. за 1 кг?

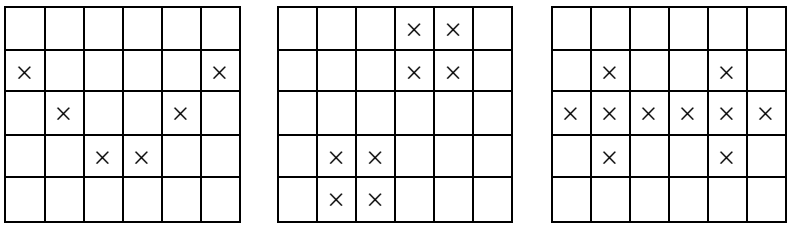
11. Джон Конвей запропонував алгоритм «Життя», за яким можна спостерігати поетапний розвиток «цивілізації» на клітинному просторі.

Спочатку у деяких клітинках простору зародилося життя. Кожний наступний етап розвитку «цивілізації» цілком визначається ситуацією, яка була на попередньому етапі, за такими правилами:

- клітинка виживає, якщо у неї два або три сусіди;
- клітинка гине, якщо у неї менше двох сусідів (від самотності) або більше трьох (від тісноти);
- життя зароджується в пустій клітинці, якщо вона має трьох сусідів.

Сусід — це *жива* клітинка, з якою є хоча б одна спільна точка дотику. Для підрахунку сусідів кожної клітинки треба розглядати 8 клітинок навколо неї.

Побудуйте етапи розвитку таких «цивілізацій»:



12. Складіть алгоритми розв’язання задач про монети:

1) серед 4 монет є одна фальшива, яка відрізняється від справжніх за вагою. За два виваження на терезах без важелів знайти фальшиву монету;

2) серед 1000 монет є одна фальшива. За найменшу кількість виважень на терезах без важелів визначити, легша чи важча фальшива монета від справжньої;

3) серед 27 монет є одна фальшива, яка легша за справжню. За три виваження на терезах без важелів знайти фальшиву монету.

алгоритм, алгоритмічна мова, альтернативний блок, блок-схема, виконавець алгоритму, детермінованість, дискретність, зрозумілість, масовість, мова програмування, навчальна алгоритмічна мова, програма, система команд виконавця, скінченність, словесна форма подання алгоритму, таблична форма подання алгоритму, умова, функціональний блок