Лекція №1

(лекція має невеликий обсяг, оскільки розглядається разом з вступною лекцією, яка знайомить студентів із структурою курсу; формалізацію поняття алгоритму висвітлено в лекції №2, яка також розглядається одночасно з лекцією №1)

План

1.1. Неформальне тлумачення алгоритму.

1.1.1. Історія поняття алгоритму.

1.1.2. Підходи до визначення поняття алгоритму.

1.1.3. Основні властивості алгоритму.

1.1.4. Параметри алгоритму.

1.1.5. Базові структури алгоритмів(алгоритмічні конструкції).

1.1.1. Історія поняття алгоритму.

Те, що зараз ми розуміємо під словом алгоритм, використовувалося в глибокій давнині, наприклад, теорема про залишки в Китаї (Китайська теорема), арифметичні операції в Індії. Але праці Евкліда і аль-Хорезмі для теорії складності алгоритмів мають особливе значення.

Мухамед ібн Муса з Хорезму, за арабським ім'ям – аль-Хорезмі (походженням з середньоазіатського міста Хорезм), видатний багдадський вчений, що працював у IX столітті н.е. У своїй книжці – трактаті "Про індійський рахунок" аль-Хорезмі описав десяткову систему числення і арифметичні операції " множення і ділення, сумування, віднімання та інші". Сьогодні збереглися лише переклади трактату. Перші з них відносяться до початку XII століття.

Далі ми наводимо цитати з "Книги про індійський рахунок", переклад якого був виконаний Ю.Копелевич з середньовічного латинського тексту, що зберігається у Кембриджському університеті.

Кожний розділ трактату, а іноді навіть абзац, починався словами "Сказав Альгорізмі…". Це словосполучення використовували у своїх лекціях і професори середньовічних університетів. Поступово ім'я аль-Хорезмі набуло звучання "алгоризм", "алгоритм" і навіть перетворилися у назву нової арифметики. Пізніше термін "алгоритм" почав означати регулярний арифметичний процес (Хр. Рудольф, 1525р.). І тільки наприкінці XVII ст. в роботах Лейбніца цей термін набув змістовності, яка не заперечує сучасному тлумаченню: "Алгоритм - це будь-який регулярний обчислювальний процес, що дозволяє за кінцеву кількість кроків розв'язувати задачі визначеного класу". Зауважимо, що за довгу еволюцію слова "алгоритм" було втрачено джерело його виникнення. І тільки у 1849 році сходознавець Ж. Рейно повернув нам ім'я аль-Хорезмі.

Зауважимо, що й слово “алгебра” бере свій початок з математичного трактату аль-Хорезмі “Книга відновлення і протиставлення”. Мухамеду ібн Мусі ще не були відомі від’ємні числа, тому в процесі обчислень він користувався операцією перенесення від’ємника з одної частини рівняння в іншу, де той стає доданком. Цю операцію Мухамеда ібн Муса називав “відновленням”. Слову “протиставлення” відповідає зміст збирання невідомих на одну сторону рівняння. Арабською “відновлення” – аль-джебр. Звідси походить слово “алгебра”. Слова “алгоритм” і “алгебра” на перших кроках розвитку математики було щільно пов’язані між собою і за змістом і за походженням. Вони виникли з одного джерела і разом пройшли багатовіковий шлях еволюції, зайняли провідні місця у сучасній науковій термінології.

Здавна найбільшу увагу приділяли дослідженням алгоритму з метою мінімізації обсягу досліджень – часовій складності розв’язання задач. Але зміст складності алгоритму не обмежується однією характеристикою. В ряді випадків не менше значення має складність логіки побудови алгоритму, різноманітність його операцій, зв’язаність їх між собою. Ця характеристика алгоритму називається програмною складністю. В теорії алгоритмів, крім часової та програмної складності, досліджуються також інші характеристики складності, наприклад, ємкісна, але найчастіше розглядають дві з них - часову і програмну. Якщо у кінцевому результаті часова складність визначає час розв’язання задачі, то програмна складність характеризує ступінь інтелектуальних зусиль, що потрібні для синтезу алгоритму. Вона впливає на витрати часу проектування алгоритму.

Вперше значення зменшення програмної складності продемонстрував аль-Хорезмі у своєму трактаті “Про індійський рахунок”. У часи аль-Хорезмі для розрахунків користувалась непозиційною римською системою числення. Її вузловими числами є I, V, X, L, C, D, M, всі решта чисел утворюються сумуванням і відніманням вузлових. аль-Хорезмі, мабуть, першим звернув увагу на складність римської системи числення у порівнянні з позиційною десятковою з точки зору простоти операцій, їх послідовного виконання та засвоєння. Він писав: “…ми вирішили розтлумачити про індійський рахунок за допомогою .IX. літер, якими вони виражали будь-яке своє число для *легкості і стислості*, *полегшуючи* справу тому, хто вивчає арифметику, тобто число найбільше і найменше, і все, що є в ньому від множення і ділення, сумування, віднімання та інше.”. Виокремленні слова – *легкість, стислість*, *полегшення* свідчать перш за все про те, що мова йде про програмну складність алгоритмів арифметичних операцій з використанням двох систем числення. Мабуть, ці слова аль-Хорезмі про складність алгоритмів при їх порівнянні були першими в історії арифметики.

Алгоритми реалізації арифметичних операцій, описані аль-Хорезмі у словесній формі, були першими у позиційній десятковій системі числення. Цікаво спостерігати, як точно і послідовно описує він алгоритм сумування, користуючись арабською системою числення і кільцем (нулем). Наведемо повністю цей алгоритм.

“Сказав Алгорізмі: Якщо ти хочеш додати число до числа або відняти число від числа, постав обидва числа в два ряди, тобто одне над другим, і нехай буде розряд одиниць під розрядом одиниць і розряд десятків під розрядом десятків. Якщо захочеш скласти обидва

числа, тобто додати одне до другого, то додай кожний розряд до розряду того ж роду, який над ним, тобто одиниці до одиниць, десятки до десятків. Якщо в якому-небудь із розрядів, тобто в розряді одиниць або десятків, або якому-небудь іншому набереться десять, став замість них одиницю і висувай її в верхній ряд, тобто, якщо ти маєш в першому розряді, який є розряд одиниць, десять, зроби з них одиницю і підніми її в розряд десятків, і там вона буде означати десять. Якщо від числа залишилось що-небудь, що нижче десяти, аби якщо саме число нижче десяти, залиши його в тому ж розряді. А якщо нічого не залишиться, постав кружок, щоби розряд не був пустим; але нехай буде в ньому кружок, який займе його, аби не сталося так, що якщо він буде пустим, розряди зменшаться і другий буде прийнятий за перший, і ти обманешся в своєму числі. Те ж саме ти зробиш у всіх розрядах. Подібним же чином, якщо збереться у другому розряді .Х., зробиш з них одиницю і піднімеш її в третій розряд, і там вона буде означати сто, а що лишається нижче .Х., залишиться тут. Якщо ж нічого в інших не залишається , ставиш тут кружок, як вище. Так ти зробиш в інших розрядах, якщо буде більше".

Можна бачити, що в цьому опису є всі параметри алгоритму. Це один з перших відомих у світі вербальних арифметичних алгоритмів.

Розглянемо логіку побудови арифметичних процедур з використанням римської та арабської систем числення з метою порівняння їх за програмною складністю. Розглянемо приклад операції сумування. Будемо користуватися алгоритмом на основі табличного методу. У арабській системі з порозрядними операціями розмір таблиці 10*10. Визначення суми чергових розрядів двох чисел, наприклад, 2 і 3 за таблицею дорівнює 5, або 7 і 9 дорівнює 16. Ці таблиці ми пам'ятаємо з дитинства.

Інша ситуація є з римською системою числення. Крім таблиці (I,II,…X)*(I,II,…X), що є еквівалентом таблиці 10*10 у арабській позиційній системі, додатково потрібно ще чотири таблиці з вузловими числами римської системи L,C,D,M та доповнення табличного методу логічними процедурами. Наприклад, для операції сумування двох чисел CMLIX + XCIV потрібні таблиці більшого об'єму, ніж 10*10 кожна. Один з варіантів рахунку полягає в представленні чисел, по-перше, розділеними на окремі цифри, по-друге, проведенням операцій віднімання і сумування окремих цифр з використанням таблиць, по-третє, об'єднанням цифр, що залишилися, в єдине число.

CMLIX + XCIV = (-C) + M +L +(-I) + X + (-X) + C +(-I) + V;

Оскільки –C + C = 0; –X + X = 0; – I – I = – II, V – II = III,

то: CMLIX + CIV = M + L + III = MLIII;

Як бачимо, для проведення розрахунків у римській системі необхідно виконувати більше типів операцій, ніж у десятковій арабській позиційній системі числення. Крім того, у римській системі потрібні додаткові логічні перетворення, що суттєво ускладнюють зв'язки між окремими операціями обчислювального процесу.

Часова складність операцій сумування і віднімання в десятковій системі визначається кількістю розрядів у взаємодіючих числах. У римській системі часова складність залежить

від порядку розташування цифр у числах. У тих випадках, коли не потрібно утворювати ланцюги логічних операцій, часова складність не перевищує часову складність операцій з десятковими числами. У протилежному випадку, як у розглянутому прикладі, зростання невелике.

Таким чином, за логікою побудови і різноманітністю операцій – програмною складністю, арабська система суттєво простіша за римську, що й довів аль-Хорезмі. За часовою складністю вони майже однакові.

Величезне революційне значення мало використання десяткової систем числення у Європі у всіх сферах життя – побуті, навчанні, торгівлі, суднобудуванні, техніці, географії, астрономії та багато інших. І те, що цей процес співпав з епохою Відродження, не є випадковим.

1.1.2. Визначення алгоритму.

Алгоритм – фундаментальне поняття математики, йому не можна дати точне математичне визначення із залученням інших фундаментальних понять. Відсутність такого визначення не заважає інтуїтивному розумінню його змісту. Але в літературних джерелах наводяться різні варіанти розуміння сутності алгоритму.

Далі наводяться кілька можливих визначень поняття «алгоритм» немає.

Алгоритм — скінчений набір правил, який визначає послідовність операцій для розв’язку конкретної множини задач та володіє наступними важливими рисами: скінченністю, визначеністю, вводом, виводом, ефективністю.
Алгоритм — система обчислень, що виконується за чітко визначеними правилами, яка після деякої кількості кроків приводить до розв’язку поставленої задачі.
Алгоритм — чітко детермінована послідовність дій, яка описує процес перетворення об’єкту із початкового стану в кінцевий, записана за допомогою зрозумілих виконавцю команд.
Алгоритм — послідовність дій, направлених на отримання кінцевого результату за скінчену кількість кроків.
Алгоритм — послідовність дій, яка або приводить до розв’язку задачі, або пояснює, чому такий розв’язок отримати не можливо.
Алгоритм — точна, однозначна, скінчена послідовність дій, яку необхідно виконати для досягнення конкретної мети за скінчену кількість кроків.
Алгоритм — це скінчена послідовність команд, які потрібно виконати над вхідними даними для отримання результату.

1.1.3. Основні властивості алгоритму.

Властивості алгоритму:

Скінченність. Алгоритм має завжди завершуватись після виконання скінченної кількості кроків.

Дискретність. Процес, що визначається алгоритмом, можна розділити на окремі елементарні етапи.

Визначеність. Кожен крок алгоритму має бути точно визначений.

Вхідні дані. Алгоритм має деяку кількість (можливо, нульову) вхідних даних.

Вихідні дані. Алгоритм має одне або декілька вихідних даних.

Ефективність. Алгоритм має завершуватися отриманням результату обчислень, що записуються у вихідні дані.

Масовість. Властивість алгоритму, яка полягає в тому, що алгоритм повинен забезпечувати розв'язання однотипних задач для будь-яких вхідних даних.

1.1.4. Параметри алгоритму.

(на основі публікацій проф. Черкаським М. В.)

Модель - це формалізоване, спрощене представлення реального об'єкту.

Будь-яка наукова модель має цільовий характер, тобто вона будується для конкретних цілей і для вирішення певних задач. Вона достатньо точно відображає ті сторони модельованого явища, які мають ключове значення для вирішення поставленої задачі. Укладена в моделі помилка рано чи пізно заважає вирішувати інші задачі, пов'язані з модельованим об'єктом. Вирішити нові задачі можна або шляхом уточнення, узагальнення первинної моделі, або шляхом побудови нової моделі.

Розглядаючи найзагальнішу модель алгоритму можна виділити (рис. 1.1) можна виділити: систему вхідних даних, систему проміжних результатів та систему кінцевих результатів

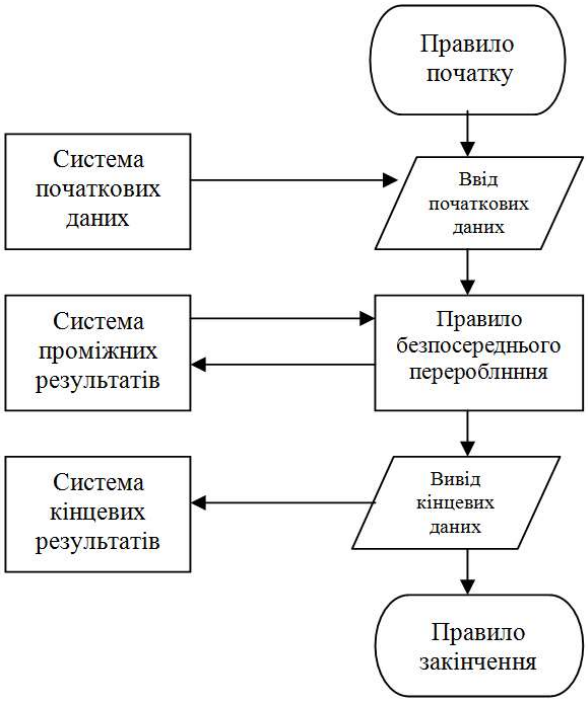

Рис. 1.1.

Тоді можна сформувати параметричну модель алгоритму(рис 1.2).

Рис. 1.2.

Тому загалом можна виділити такі параметри алгоритму:
1. Правило початку.

2. Правило вводу даних.

3. Система(потенційно нескінченна) вхідних даних.

4. Правило безпосереднього перероблення.

5. Система проміжних результатів.

6. Система кінцевих результатів.

7. Правило виводу.

8. Правило закінчення.

У своїх роботах проф. Черкаський також часто оперує поняттям задачі.

Задача – це сформульоване намагання, яке за набором вхідних даних і умов за допомогою деякого алгоритму дозволяє отримати результат, що повністю відповідає цьому набору вхідних даних і умов:

M: y1, y2,…ym ← x1, x2,…xn
, де M – умови; x1, x2,…xn – вхідні дані; y1, y2,…ym – результати; ← – невизначений алгоритм.

Тоді параметричну модель алгоритму можна доповнити до пари задача-алгоритм(рис 1.3).

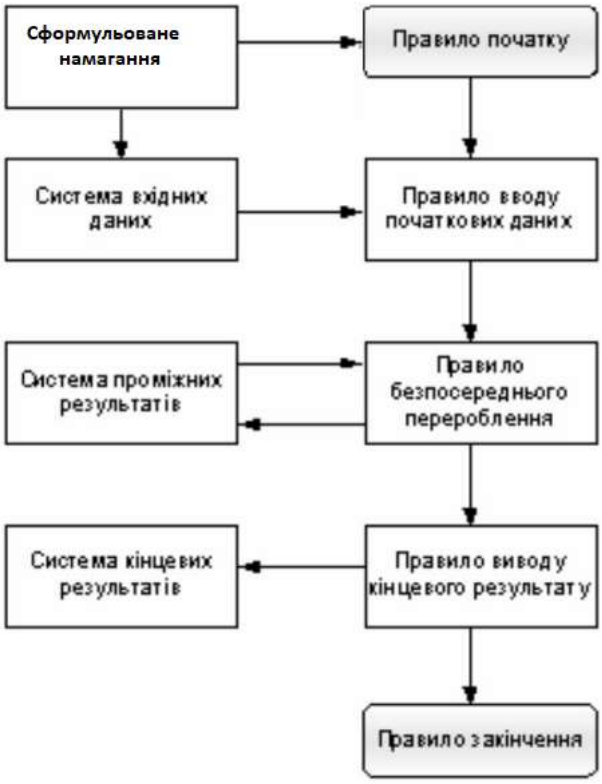

Рис. 1.3

1.1.5. Базові структури алгоритмів(алгоритмічні конструкції).

Можна виділити чотири базові структури алгоритмів:

· лінійні;

· розгалужені;

· циклічні;

· змішані.

Лінійні алгоритми

Лінійним називається алгоритм (фрагмент алгоритму), в якому окремі команди виконуються послідовно один за одни, не залежно від значень вхідних даних і проміжних результатів.:

Розгалужений алгоритм

Часто хід обчислювального процесу залежить від вихідних даних або проміжних

результатів; алгоритм реалізується в одному з декількох, заздалегідь передбачених (можливих) напрямків. Такі напрямки часто називаються гілками. Кожна гілка може бути будь-якого ступеня складності, а може взагалі не містити команд, тобто бути виродженою. Вибір тієї або іншої гілки здійснюється в залежності від результату перевірки умови з конкретними даними. У кожному випадку алгоритм реалізується тільки по одній гілці, а виконання інших виключається. Приклад розгалуженого алгоритму:

Циклічний алгоритм

Виконання деякої частини алгоритму багаторазово при різних значеннях деяких змінних називається циклічним обчислювальним процесом. Циклами називаються повторювані ділянки обчислень. Для організації циклів необхідно: задати початкове значення змінної, що визначає цикл (параметр циклу), змінять цю змінну перед кожним повторенням циклу, перевіряти умову продовження (закінчення) циклу. У циклічних алгоритмах виконання деяких операторів (групп операторів) здійснюється багаторазово з тими ж або модифікованими даними.

У залежності від способу організації кількості повторень циклу розрізняють три типи

циклів:

1) цикл із заданою умовою закінчення роботи (ЦИКЛ - ДО);

2) цикл із заданою умовою продовження роботи (ЦИКЛ - ПОКИ);

3) цикл із заданою умовою повторень роботи (ЦИКЛ З ПАРАМЕТРОМ).