Програма візуалізації рішення задачі про Ханойські вежі, використовуючи рекурсивний алгоритм

О.Ю. Головатюк, *студент,*
А.І. Приходькіна, *ст. викладач*
Кіровоградський національний технічний університет

Тема рекурсії досить складна і в багатьох викликає труднощі в процесі реалізації рекурсивних програм. Рекурсивні алгоритми складніше налагоджувати, але вони дозволяють дуже гнучко і красиво вирішити задачу. Відомо, що будь-який рекурсивний алгоритм можна замінити ітерацією, але це займає більше часу на реалізацію.

В програмуванні рекурсією називається конструкція, в якій функція викликає саму себе. Розрізняють просту (пряму) і складну (непряму) рекурсії. Функція називається простою рекурсивною, якщо містить у своєму тілі виклик самої себе. Якщо функція викликає іншу функцію, котра викликає першу, то така функція називається складною рекурсивною. Кількість вкладених викликів функції називається глибиною рекурсії.

Коли функція викликає сама себе, новий набір локальних змінних і параметрів розміщується в пам'яті в стеку, а код функції виконується з самого початку, причому використовуються саме ці нові змінні. За рекурсивного виклику функції нова копія її коду не створюється. Новими є лише значення, які використовує подана функція. При кожному поверненні з рекурсивного виклику старі локальні змінні і параметри вилучаються зі стеку, і відразу за рекурсивним викликом відновлюється робота функції.

Рекурсивні методи є базовими методами вирішення алгоритмічних задач. Одним із прикладів використання рекурсії є досить стара і добре відома задача про Ханойські вежі, яку придумав французький математик Едуард Люка в 1883 році. Спочатку її продавали як іграшку, і з нею пов'язано декілька кумедних легенд.

За умовою дано три стержня – початковий (A), додатковий (B) і кінцевий (C). На початковий нанизана деяка кількість дисків (будемо її вважати рівній n), причому вони відрізняються розміром, і лежать менший на більшому, як це показано на рисунку 1.

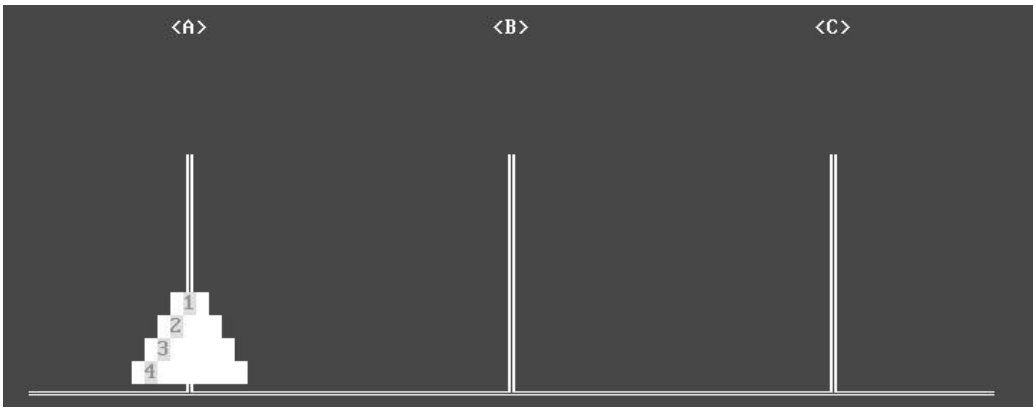

Рисунок 1 – Початкове розташування дисків

Задача полягає в тому, щоб перенести всі диски з початкового стержня на кінцевий за найменше число ходів. Тобто рішенням задачі є розташування дисків, показане на рисунку 2.

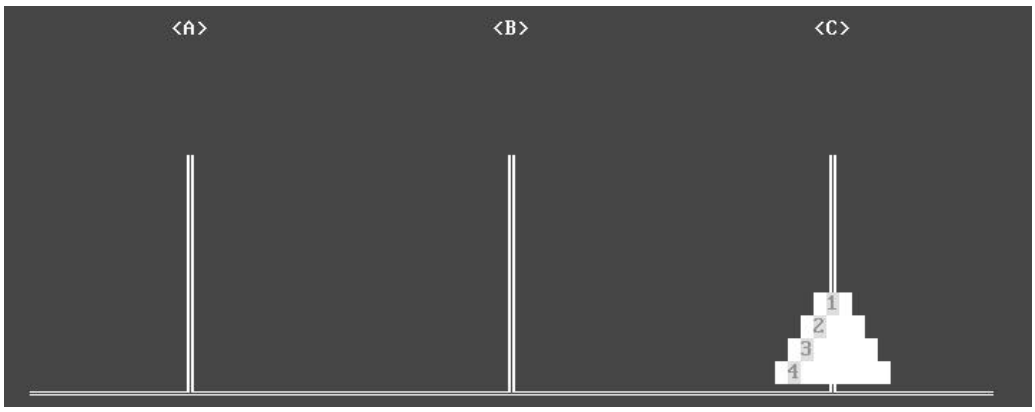

Рисунок 2 – Кінцеве розташування дисків

За один раз дозволяється переносити тільки один диск, причому не можна класти більший на менший.

Вибір тієї чи іншої стратегії переміщення дисків викликає питання необхідної кількості ходів для вирішення задачі. Очевидно, що кількість ходів зростає зі зростанням числа дисків, і, бажано, отримати деяку функцію $f(n)$, яка видавала б для даної стратегії кількість ходів, необхідних для розв'язання задачі в залежності від числа дисків.

Нам необхідно перенести n дисків зі стержня А на стержень С. Припустимо у нас є процедура перенесення n-1 диска, позначимо її П(n-1,Ст1,Ст2). Тоді задачу легко вирішити, для цього спочатку перенесемо n-1 диск з стержня А на В, застосовуючи процедуру П, потім перенесемо n-ий диск зі стержня А на С і нарешті перенесемо n-1 диск зі стержня В на С. Алгоритм в цьому випадку можна записати наступним чином:

1. П(n-1,A,B);
2. Перенести диск з А на С;
3. П(n-1,B,C).

Крім цього, потрібно розглянути окремий випадок, який не є рекурсивним. Якщо диск всього один, то можна відразу перенести його із стержня А на стержень С. Таким чином, процедура П представляється у вигляді блок-схеми, яка зображена на рисунку 3.

Повернемося до питання обчислення функції $f(n)$. Для згаданої вище рекурсивної процедури, представлення $f(n)$ досить елементарне:

$$f(1) = 1$$
$$f(n) = 2 * f(n-1) + 1$$

Явний вигляд функції $f(n)$ легко знайти з наведених вище співвідношень, користуючись принципом математичної індукції:

$$f(n) = 2^n - 1$$

Кількість переміщень в залежності від кількості дисків обчислюється за формулою 2^n-1, де n – кількість дисків.

Зауважимо ще одну властивість дисків, які переміщаються. Нехай наші диски пронумеровані від 1 до n, таким чином, що диск з найменшим діаметром має номер 1, другий за величиною діаметра диск має номер 2, і так далі, диск з найбільшим діаметром має номер n. Тоді можна помітити, що два диски з номерами однакової

парності ніколи не будуть лежати один на одному (тобто диск з номером 2 не буде лежати на дискові з номером 4, 6 або 8). Доведемо, що ця властивість виконується.

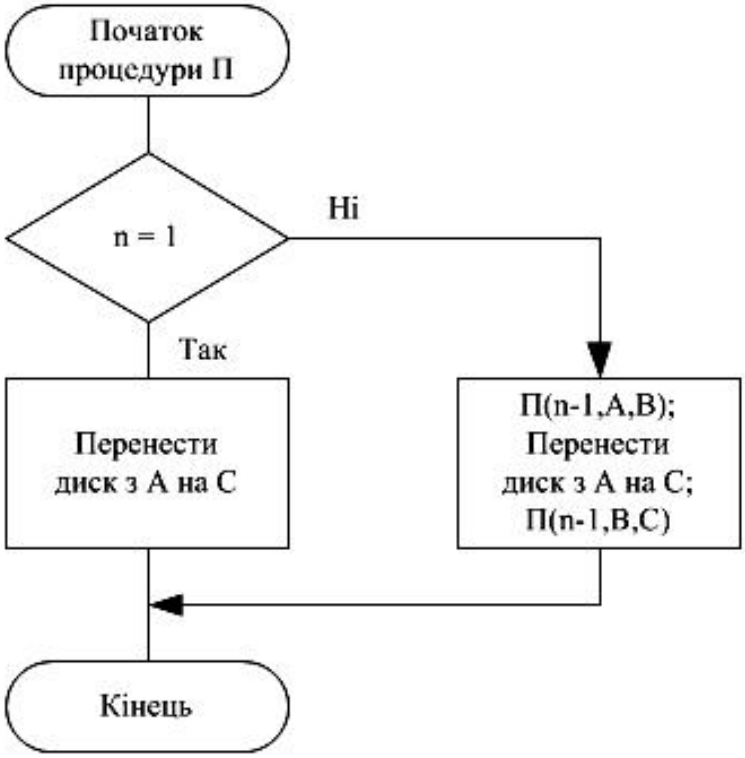

Рисунок 3 – Блок-схема процедури переміщення дисків

Доведення будемо проводити по індукції, припустимо, що властивість виконана для процедури П(p,Ст1,Ст2), де p < n (для p = 1 і П(1,Ст1,Ст2) властивість виконана автоматично), далі доведемо, що тоді властивість виконується і для П(n,Ст1,Ст2). Почнемо з перенесення n-1 дисків на стержень В. Поки не переміщений n-1 диск, жоден диск не кладеться безпосередньо на диск з номером n і це означає, що потрібна властивість виконана. Розглянемо випадок, коли n-2 дисків знаходяться на одному стержні, диски з номерами n-1 і n на іншому стержні, а третій стержень порожній. Переміщуємо диск з номером n-1. Тепер потрібно перемістити перші n-2 дисків на диск з номером n-1, тому диски будуть опинятися на диску з номером n. Якщо ми кладемо диск з номером k на диск з номером n, то для того щоб утворити піраміду дисків з номерами від k до 1 і мати можливість перемістити диск з номером k+1 на диск з номером n-1. Але оскільки потрібна властивість виконується для n-1 дисків, то парність k+1 диска не може збігатися з парністю n-1, отже вона збігається з парністю n, і звідси випливає, що парність n і k різна.

Головна перевага рекурсивних функцій – з їх допомогою спрощується реалізація багатьох алгоритмів і програма стає зрозуміліша. За допомогою рекурсивної функції можливо описати нескінченне обчислення без явних повторень блоків коду програми.

Розроблена програма може використовуватись при вивченні даної задачі з метою кращого розуміння роботи рекурсивного алгоритму.

Список літератури

1. Арсак Ж. Программирование игр и головоломок. – М.: Наука, 1990. – 224 с.
2. Фаронов В. В. Turbo Pascal. – СПб.: БХВ-Петербург, 2004. – 1056 с.