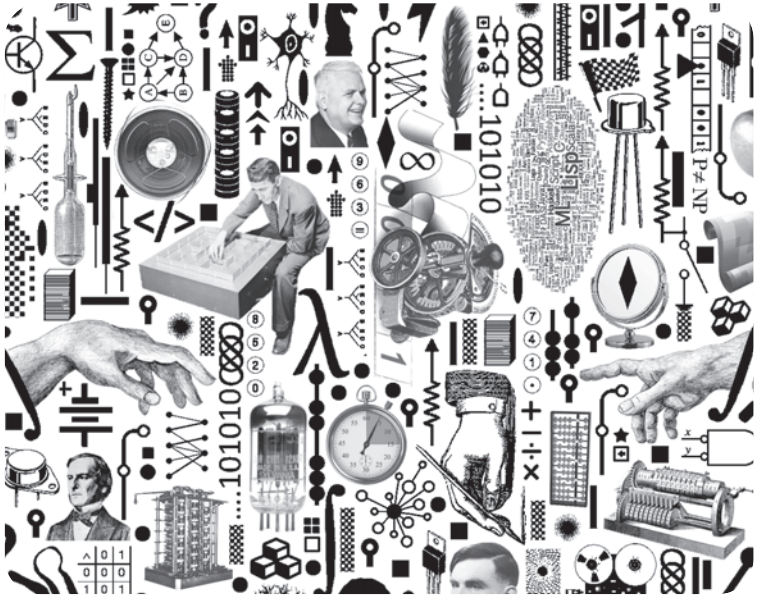
컴퓨터과학이 여는 세계

이광근 지음



세상을 바꾼 컴퓨터, 소프트웨어의 원천 아이디어 그리고 미래

인사이트
insight

『컴퓨터과학이 여는 세계』 미리보기

펴내면서

PREFACE

컴퓨터 이야기다. 근본이 무엇이고, 어떻게 탄생했고, 소프트웨어의 세계는 어떤 세계인지. 컴퓨터가 우리의 지능과 본능과 현실을 어떻게 확장시키는지. 그래서 우리가 지금 기대고 있는 디지털 세계를 어떻게 바라볼 수 있는지.

독자들은 궁금해 할 것이다. 이 책이 내게 어떤 새로운 도움이 될까? 이 책은 두 가지 성격으로 독자들과 만날 듯하다. 컴퓨터 세계의 근본이 궁금할 때 누구나 펼쳐 볼 수 있는 과학 교양서적으로, 혹은 컴퓨터 전문가의 기본을 준비시켜 주는 '예과' 입문서로서. 다음 네 가지 정도를 염두에 두고 썼기 때문이다.

- 밑거름

  컴퓨터 관련 기술은 대개 매스컴을 통해서 접하고, 우리들의 이해는 표면적인데 머물기 쉽다. 독자들이 컴퓨터과학기술의 핵심을 쉽게 익혀 긴 수명의 밑거름이 되기를 바라며 썼다.

• 안목

컴퓨터와 소프트웨어는 우리의 모든 일상을 움직이는 중요한 인프라다. 독자들이 그 원천 아이디어를 이해해서 미래에 가능한 응용을 창조하거나 예측할 수 있는 안목을 갖추는 데 도움이 되기를 바라며 썼다.

• 확장

컴퓨터과학은 모든 분야의 성과를 바탕으로 자라며 모든 분야(과학, 인문학, 공학, 사회과학, 예술, 교육, 경영, 의학, 법학 등)를 살찌우는 보편 학문의 성격을 점점 띠고 있다. 다른 세계를 상상하는 모두와 이 책이 손잡을 수 있기를 바라며 썼다.

• 기회

지금까지의 정보화 문명은 시작일 뿐이다. 컴퓨터과학기술이 미래에 만들어 갈 다양한 기회를 상기시키는 계기가 되기를 바라며 썼다.

특히 이 책에서는 컴퓨터과학에서 나온 원천적인 아이디어들에 집중했다. 컴퓨터과학 고유의 핵심 아이디어들이 펼쳐진다.

부담이 없지는 않았다. 한숨에 컴퓨터과학이란 분야를 제대로 짚을 순 있는 건지. 멀어지면 숨소리를 놓치고 가까우면 형체를 잃는 그런 줄타기였다. 균형추로 삼은 건 컴퓨터과학 고유의 원천 아이디어들이었다.

의외로 시중에서 그런 아이디어를 모아 소개하는 책은 찾기 어려웠다.

영어권도 마찬가지였다. 단편적이었다. 컴퓨터과학의 풍경 아래에 흐르는 원천 아이디어들. 이것들이 나온 이야기들. 그 연유와 의미. 그래서 그 진폭과 그 주파수로 생각이 진동한다면. 이 책이 독자들에게 그런 진동을 유도할 수 있다면 영광일 것이다.

그런 진동이 우리가 원하는 선진국형 원천지식을 만들어 낼 토양을 도탑게 다져줄 것이라고 본다. 컴퓨터과학에서 그런 지식이 지금까지 어떤 동기로 어떻게 나왔는지, 겁낼 필요 없는 모습을 소개한다.

그리고, 당연하지만 편안한 우리말로 컴퓨터과학을 쉽게 설명할 수 있을 거라 믿고 나섰다. 확인할 수 있어서 즐거웠는데, 독자들도 같은 생각일진 모르겠다.

사실 따스한 모국어로 깊이 있는 전공서적들이 모인 시리즈를 상상해 왔었다. 이 책이 그런 시리즈의 한 권이 될 수 있을까. 그래서 누구나 쉽게 들어설 수 있는 열린 울타리의 비옥한 토양, 그 한 귀퉁이를 이 책이 담당한다면 기쁠 것 같다.

전세계적으로 컴퓨터 소프트웨어 교육이 누구에게나 필수가 되고 있고 우리나라도 예외는 아니다. 이 교육의 목표는 시민들에게 우리를 둘러싼 디지털 세상을 바라보는 시각을 형성해주는 것이다. 마치 물리 교육이 우리를 둘러싼 자연을 바라보는 시각을 형성해 주듯이. 그런데 그 시각이란, 근본을 알아내려고 뒤로 물러날 때 더 넓은 부채꼴을 그리며 많은 것을 커버할 수 있을 것이다.

그런 넓은 시각을 형성해 줄 근본적인 내용을 가능한 한 쉽게 전달하는 콘텐츠. 이 책이 그런 한 수이길 바란다.

표기법

전문용어가 나타날 때 영어를 아래첨자*subscript*로 덧붙였다. 전문용어는 최대한 힘을 빼고 쉽고 직관적인 표현을 썼다. 지레 겁먹게 하는 불필요한 한문 단어는 피했다. 도저히 쉬운 말을 찾을 수 없을 땐 영어 발음대로 쓰기도 했다. 사용한 전문용어들의 색인은 책 뒤에 정리해 놓았다. 참고했던 자료는 참고문헌에 정리해 놓았다. 그 밖의 자료는 책에서 언급한 전문용어를 구글 검색창에 넣으면 쉽게 얻을 수 있을 것이다. 길목에 오르막 표시 ⛢가 있으면 조금 어려운 부분의 시작을 뜻한다. 내리막 표시 ⛢까지 건너뛰어도 전체를 조망하는 데는 지장이 없을 것이다.

그리고 단락 사이에 종종 시를 인용해 끼웠다. 이해를 돕는 촉매로, 혹은 한 컷의 삽화나 쉼표를 대신해서 시를 사용했다.

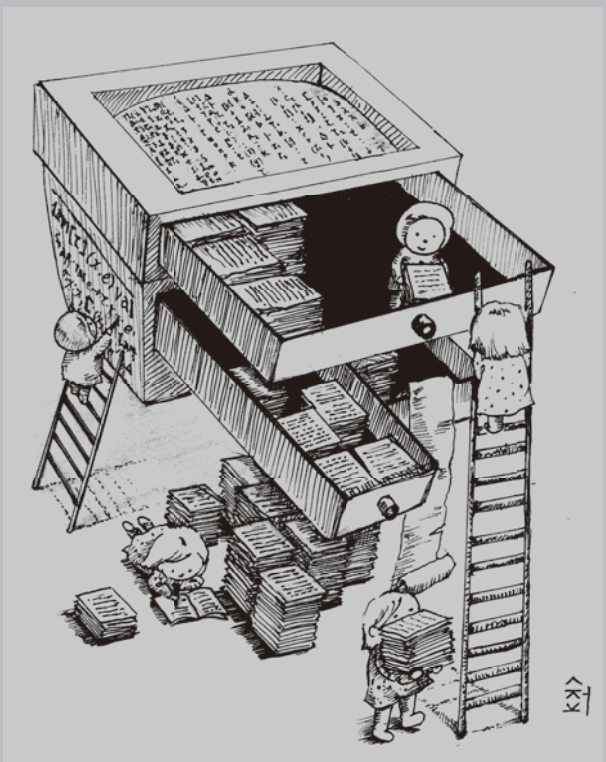
차례

CONTENTS

01

마음의 도구

컴퓨터과학이 여는 세계



『컴퓨터과학이 여는 세계』 미리보기

나는 매일 도구를 쓴다. 자고 일어난 침대, 이 닦는 칫솔, 세수할 때 상하수도와 비누, 밥먹는 그릇과 수저, 입는 옷, 쓰는 안경, 신는 신발, 출퇴근할 때 타는 엘리베이터와 지하철과 마을버스. 이 도구들 덕분에 나는 매일 새로운 기분으로 2만 보가 넘는 거리를 간편하게 이동하며 캠퍼스에 도착한다. 이 모든 도구를 사용할 수 없었다면 나는 그냥 동물일 것이다.

인류가 만든 모든 도구는 인간 능력의 한계를 늘 확장시켜왔지만, 컴퓨터라는 도구는 좀 특이하다. 인류가 발명한 대개의 도구는 물리적인 도구이고 다루려면 물리적인 근육이 필요하다. 하지만 컴퓨터는 '마음의 도구'이고 그 도구를 다루는 방법은 물리적인 근육이 아니라 언어다. 언어로 작성된 텍스트를 컴퓨터의 메모리에 실으면 컴퓨터는 그 텍스트가 표현한 할 일을 해 간다.

컴퓨터라는 도구는 그래서 할 수 있는 일의 한계가 없어 보인다. 컴퓨터가 읽는 텍스트(소프트웨어)는 우리가 무궁무진하게 지어낼 수 있기 때문이다. 그것대로 컴퓨터는 무궁무진한 일을 하기 때문이다. 컴퓨터가 읽는 텍스트는 우리가 읽는 문학과 같다. 읽고 있는 컴퓨터를 지배하고 한없이 많이 지어질 것이다.

이런 소프트웨어들로 마음의 도구를 능숙하게 다루며 인간은 놀랍게 확장하고 있다. 인간의 지능이 확장하고 있고, 인간의 본능이 확장하고 있고, 인간의 현실이 확장하고 있다. 예전에는 불가능했던 지식과 지능이 컴퓨터와 팀이 되어 발휘되고 있고, 예전에는 불가능했던 규모와 속도로 소통하고 놀 수 있게 되었고, 우리를 압박하던 시공간의 제약이 사라지기 시작했다. 우리는 각자의 뇌가 가진 지능의 한계를 탈출하게 되었고, 이별 있는 세상에서 이별 없는 세상에 놓이게 되었고, 상상 속의 거대한 놀이공원에

서 수많은 사람과 맘껏 마음을 맞춰 놀 수 있게 되었고, 여기 같이 있어야만 안심하고 할 수 있는 일들이 아무 때나 멀리서도 가능하게 되었다. 나는 새로운 스케일과 차원으로 확장한 나를 매일 만나게 되었다.

어떻게 이런 일이 가능한 걸까? 마음의 도구와 그 도구를 다루는 방법의 원천은 무엇일까? 컴퓨터과학은 어떻게 탄생한 것이고 지금까지 무엇을 밝혀냈을까?

앞으로 우리는 그 답을 살피러 떠나보겠다. 우리의 여행은 네 개의 코스로 구성되어 있다

- 400년의 축적

  컴퓨터는 특별하다. 컴퓨터 하나로 한없이 많은 일을 할 수 있기 때문이다. 그래서 컴퓨터를 보편만능의 기계*universal machine*라고 부른다. 누가 이런 놀라운 도구를 발명한 걸까? 컴퓨터는 20세기 수학자들의 큰 꿈이 철저히 좌절되는 과정에서 나온 부산물이었다. 좌절을 엄밀히 확인하는 과정에서 고안된 소품. 이것이 21세기 정보혁명의 주인공이 된다.

  컴퓨터(보편만능의 기계)의 디자인이 출현한 원조 논문의 배경과 내용이 펼쳐진다. 원조 밥집을 가보는 재미도 있지만 현란한 디지털 세계의 근본을 쉽게 파악하게 해줄 기초가 거기에 있다.

- 그 도구의 실현

  머릿속에서만 디자인 된 보편만능의 도구. 이를 실제 작동하는 물건으로 어떻게 만들 수 있었을까?

컴퓨터를 실현하기까지는 100여 년간의 색다른 축적 과정이 필요했다. 그런데 이 과정은 컴퓨터가 디자인되던 경로와는 별도로 마무리 되고 있었다. 다른 경로로 무르익은 또 다른 상상과 융합의 열매. 1854년과 1937년을 스치는 100여 년의 선분.

스위치 기술이 부울 논리*boolean logic*를 만나 날개를 달고, 스위치만으로 컴퓨터의 모든 것이 차곡차곡 만들어지는 과정이 펼쳐진다.

• 소프트웨어, 지혜로 짓는 세계

컴퓨터를 만능이게 하는 소프트웨어. 사람은 소프트웨어를 만들고, 컴퓨터는 소프트웨어를 실행한다. 컴퓨터라는 도구를 다루는 방법은 소프트웨어이고, 소프트웨어는 사람의 지혜를 통과하면서 짜여진다.

그렇다면 소프트웨어를 잘 만들 방법은 무엇인가? 이 목표를 위해 컴퓨터과학은 무엇을 밝혀냈는가?

이에 대한 답으로, 두 갈래에서 출현한 컴퓨터과학 고유의 원천 아이디어들이 소개된다. 그 두 갈래는 소프트웨어가 일하는 방도(알고리즘*algorithm*)와 그 소프트웨어를 표현하는 방식(언어*language*)에 대한 탐구다. 이 두 기둥에 기대어 조망하는 소프트웨어의 세계가 펼쳐진다.

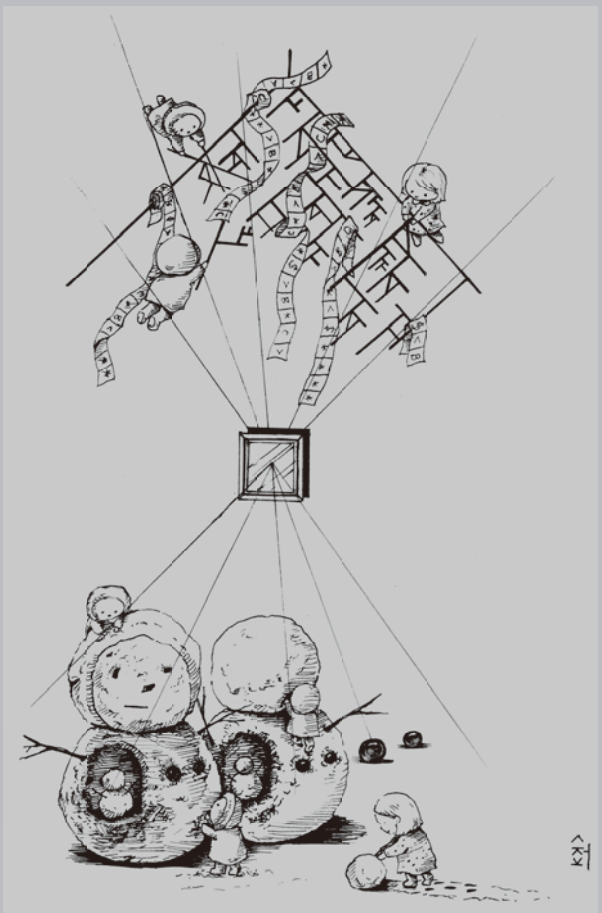
• 그 도구의 응용

인간의 능력을 확장시키는 다양함과 규모 면에서 컴퓨터는 다른 도구를 능가한다. 어느 정도로 우리를 확장시키고 있을까?

04

소프트웨어,
지혜로 짓는 세계

컴퓨터과학이 여는 세계



『컴퓨터과학이 여는 세계』 미리보기

컴퓨터는 마음의 도구이고 그 도구를 다루는 방법은 지혜와 언어다. 모든 도구는 다루는 방법이 함께 하는데, 인류가 발명한 대개의 도구는 물리적인 도구였고 다루려면 물리적인 근육이 필요했다. 하지만, 컴퓨터는 마음의 도구이고 그 도구를 다루는 방법은 지혜와 언어로 짜인 소프트웨어다.

소프트웨어는 사람의 지혜를 통과하면서 언어로 짜짓는다. 계획을 짜고 전략을 짜고 가구를 짜고 무늬를 짜듯이, 농사를 짓고 건물을 짓고 이름을 짓고 시를 짓듯이 소프트웨어는 사람이 짜고 짓는다.

소프트웨어는 무섭다. 구체적인 실천이기 때문이다. 실천은 틈새를 허용하지 않는다. 컴퓨터는 소프트웨어 그대로를 무심히 실행에 옮길 뿐, 빈틈이 있다면 고스란히 드러낸다.

컴퓨터를 만능이게 하는 소프트웨어. 사람은 소프트웨어를 만들고, 컴퓨터는 소프트웨어를 실행한다. 소프트웨어를 궁리하고 만드는 것은 사람의 일이고, 소프트웨어를 실행하는 일은 컴퓨터의 일이다.

4.1 그 도구를 다루는 방법

그렇다면 소프트웨어를 잘 만들 방법은 뭔가? 이 목표를 위해 컴퓨터과학은 무엇을 밝혀냈는가?

두 줄기로 번졌다. 일하는 방도와 표현하는 방식에 대해. 만들려는 소프트웨어(튜링기계)가 일하는 방도와, 그 소프트웨어(튜링기계)를 표현하는 방식에 대해.

일하는 방도는 알고리즘*algorithm*에 대한 탐구고, 표현하는 방식은 언어*language*에 대한 탐구다. 어떤 알고리즘으로 소프트웨어가 일을 하도록 해야 좋은지, 어떤 언어로 소프트웨어를 표현하는 게 좋은지. 푸는 솜씨와 담는 그릇, 혹은 말 달릴 때의 박차와 고삐 같이 소프트웨어 달릴 때의 박차(알고리즘)와 고삐(언어)에 대한 탐구다.

이 두 줄기에서 다듬어진 기둥이 소프트웨어 세계의 기초를 이룬다.

알고리즘

소프트웨어는 튜링기계다. 튜링기계는 어떤 일을 하고, 그 일은 어떤 문제를 푸는 것이다. 컴퓨터는 그 소프트웨어(튜링기계)를 실행하면서 문제 풀이를 자동으로 진행한다. "컴퓨터로 문제를 푼다"는 것이 이 뜻이다. 컴퓨터로 돌릴 소프트웨어를 만드는 것이다.

알고리즘이란 소프트웨어(튜링기계)로 만드는 문제 푸는 방도다. 따라서, 컴퓨터로 문제를 풀려면 우선 알고리즘을 찾고 그 해법이 맞는지 확인한 후, 그 알고리즘대로 작동할 소프트웨어를 만들게 된다.

이러면서 우리는 정글에 들어선다. 컴퓨터로 풀 수 있는 무한히 많은 문제들. 그 문제마다 많은 수준의 다양한 해법(알고리즘)들. 이것들이 모인 정글.

그리고 묻게 된다. 내가 만든 알고리즘은 이 정글의 어디쯤인가. 같은 문제를 푸는 더 싸고 빠른 방안은 있는가. 더 좋다면 얼마나 더 좋은가. 혹은 더 좋기는 불가능한가. 그리고, 결국 튜링기계가 현실적인 비용으로 해결할 수 있는 문제들의 경계는 어디까지인가. 그래서 내가 풀려는 문제는 그 경계의 안인가 바깥인가.

이런 질문에 대한 채비가 알고리즘 분야가 구축한 성과다. 문제와 해법들의 밀림을 정확히 이해할 수 있게 하는 장비들. 이에 기대어 탐험해 간 밀림의 풍경.

언어

소프트웨어를 표현하는 언어, 즉 튜링기계를 표현하는 언어, 언어가 필요하다.

튜링기계의 규칙표 방식은 원시적이다. 근본적으론 모든 소프트웨어가 그렇게 정의될 수 있지만, 현실은 그 이상을 원한다. 그렇게 원시적으로 머물 순 없다.

먼 길을 가기 위해선 차원이 다른 언어의 채비가 필요하다. 끝없이 정교하고 복잡한 일을 할 소프트웨어들. 그런 소프트웨어들을 제대로 만들어가야 할 먼 길.

그 채비의 요건은 사람에게서 온다. 사람에게 편해야 한다. 하고자 하는 일을 표현하기 편해야 한다. 그리고 그 표현에 실수가 없는지 확인하기 편해야 한다. 원하는 바를 실행하도록 소프트웨어가 구성돼 있는지, 소프트웨어가 사용되기 전에 사람이 쉽게 검증할 수 있어야 한다. 소프트웨어가 우리와 우리 주변의 모든 것을 책임져가는 상황에서, 소프트웨어 제작자가 내놓는 처방(소프트웨어)은 의사의 처방이 그래야 하는 것처럼 실수가 없어야 하기 때문이다.

이러한 요건을 갖춘 채비가 언어 분야가 축적하는 성과다. 그런 요건을 갖추려는 언어를 위해 논리와 짝을 이루어 구축한 풍경들. 정교한 논리로 가다듬어 가는 컴퓨터 언어와 주변 도구들.

4.2 푸는 솜씨, 알고리즘과 복잡도

알고리즘*algorithm*이란 컴퓨터가 따라할 문제풀이법이다. 자동으로 돌릴 수 있는 문제풀이법. 기계적인 방식으로 자동화되는 문제풀이법.[1]

복잡도*complexity*란 알고리즘을 실행에 옮길 때 드는 비용을 말한다. 컴퓨터가 알고리즘을 실행하며 답을 낼 때까지 드는 비용이 알고리즘의 복잡도다. 그 비용의 단위는 시간이나 메모리다. 알고리즘대로 컴퓨터가 실행할 때 시간이나 메모리를 얼마나 써야 답을 내는지. 이게 그 알고리즘의 복잡도다.

풍경

주어진 문제를 컴퓨터로 풀고 싶다면, 알고리즘(문제풀이법)을 찾고 그 해법이 맞는지 확인한 후, 그 복잡도(실행비용)가 견딜 만하다고 판단되면, 그 알고리즘대로 작동할 소프트웨어를 만들게 된다.

주어진 문제의 해법으로 찾아진 알고리즘은 대개 여러 가지다. 어떤 문제를 해결할 방도가 어찌 하나뿐일까. 문제마다 컴퓨터 풀이법은 여럿이기 마련이다. 아주 빨리 결과를 내는 알고리즘과 아주 늦게 결과를 내는 알고리즘. 아주 적은 메모리를 쓰는 알고리즘과 아주 많은 메모리를 소모하는 알고리즘. 당연히 빠르고(시간) 가볍게(메모리) 일을 마치는 알고리즘을 우리는 원한다.

1 모두 같은 말이다 : '컴퓨터로' = '자동으로' = '기계적으로' ; '풀 수 있는' = '계산 가능한' = '돌릴 수 있는' = '실행하는' ; 그래서 '컴퓨터로 풀 수 있는' = '자동으로 계산가능한' = '기계적으로 돌릴 수 있는'이 모두 같은 말이다

어떤 문제는 그 알고리즘의 복잡도가 너무 크다. 심지어 우주의 수명이 다할 때까지 기다려도 끝나지 않을 알고리즘도 있다. 진행은 하지만 아무리 기다려도 답을 아직 내놓지 못하는 알고리즘. 문제 중에는 이렇게 무시무시하게 비싼 알고리즘밖에는 더 좋은 방법을 아직 모르는 것들이 수두룩하다.

또 컴퓨터로는 절대 풀 수 없는 문제들도 있다. 예를 들어 멈춤 문제 *halting problem*가 있다. 모든 프로그램마다 그 프로그램이 과연 멈출지 멈추지 않을지를 정확히 예측할 수 있는 프로그램은 없다고 했었다. 그런데 멈춤 문제뿐이 아니다. 무수히 많다.

문제마다 다양한 해법(알고리즘)들이 모인 정글. 이 정글에는 온순한 문제, 이상한 문제, 또 무시무시한 문제들이 어슬렁거린다. 컴퓨터로 풀 수 있는 문제, 풀 수 없는 문제. 풀 수 있다고 해도 그 비용이 너무 큰 문제. 풀 수 있는 문제 중에서도 그 비용이 현실적으로 참을 만한 문제. 그런 문제의 알고리즘 중에서도 될 수 있으면 비용이 적었으면 하는 바람.

질문은 많다. 내가 만든 알고리즘이 더 싸고 빨라질 방안은 있는지. 아니면 더 좋기는 불가능한지. 문제마다 해법이 어느 정도로 현실적일 수 있는지. 아직 비현실적인 해법밖에 알려지지 않은 문제를 맞닥뜨린 건 아닌지. 그런 문제들에 현실적인 해법을 찾을 가능성은 있는지. 그리고 결국 컴퓨터가 현실적인 비용으로 해결할 수 있는 문제들의 경계는 어디인지.

알고리즘 예

- 책 읽기.

  첫 쪽부터 끝까지 차례로 읽는다. 총 n쪽의 책이라면 n번 쪽수를