가천대학교
Gachon University

01

Algorithm

과정 소개, 알고리즘 개요

2025-03-07

조윤실

목 차

※ 가천대학교 컴퓨터공학과 '기계 학습 프로그래밍' 과정

과정 소개

강의 개요

교과목명	알고리즘	이수	전선
강의시간	금1,금2,금3 (09:00 ~ 11:50) 금6,금7,금8 (14:00 ~ 16:50)	학점	3학점
강의실	AI공학관-302	연락처	031-750-5768
e-mail	yswoola@gachon.ac.kr	긴급연락처	010-2570-0166

본 강좌, 기계학습프로그래밍 과목은

- 기계학습 발전 과정과 다양한 기계학습 알고리즘에 대한 이론 설명과 실습을 병행하면서
- 기계학습의 주요 내용을 이해하고 프로그래밍하는 과정으로 진행됩니다.

강의 목표

- 문제 해결을 위해

1. 널리 알려진 기본 알고리즘의 원리 이해
2. 문제를 효율적으로 해결하기 위한 설계 전략을 예제로 이해
3. 알고리즘을 코드로 구현

선수과목

- 컴퓨터공학과 교육과정 : https://www.gachon.ac.kr/cs/5899/subview.do

3	1	전선	알고리즘	3	3
2	1	전필	자료구조	3	3
1	2	전선	이산수학	3	3
1	2	전필	C언어	3	3
1	1	계교	파이썬	3	3

강의 일정

주차	날짜	내용
1	03/07	개강, 알고리즘 개요
2	03/14	자료구조 기초
3	03/21	재귀 알고리즘
4	03/28	정렬 알고리즘1
5	04/04	정렬 알고리즘2
6	04/11	탐색 알고리즘
7	04/18	그래프 이론 기초
8	04/25	중간고사

주차	날짜	내용
9	05/02	휴강
10	05/09	그래프 알고리즘
11	05/16	억지 기법과 탐욕 전략
12	05/23	분할 정복
13	05/30	동적 계획법
14	06/06	백트랙킹
15	06/13	머신러닝/딥러닝 알고리즘
16	06/20	기말 프로젝트 발표, 종강

※ 수업내용 및 활동은 상황에 따라 변동될 수 있음

강의 시간

- 교육 시간

시간	금1, 금2, 금3	금6, 금7, 금8
1 교시	09:00 ~ 10:15 (75분)	14:00 ~ 15:15 (75분)
break time	10:15 ~ 10:30 (15분)	15:15 ~ 15:30 (15분)
2 교시	10:30 ~ 11:50 (80분)	15:30 ~ 16:50 (80분)

※ 사정에 따라 교육시간 배분은 변동될 수 있음

강의 진행방법

- 이론 설명 + 실습

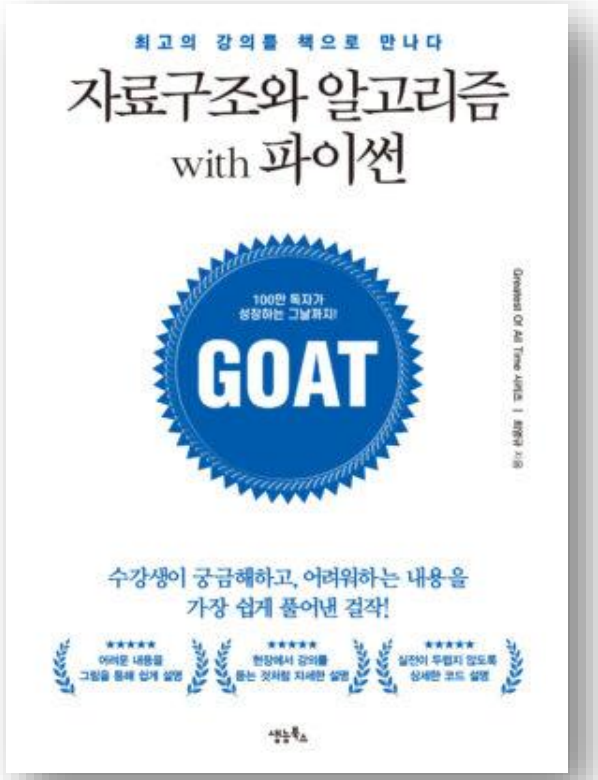
1. 알고리즘 관련 이론 & 원리 설명
2. 알고리즘 적용 사례 찾기
3. 코드 실습 진행

수업 준비물

- 교재 + 강의자료 + 노트북
  - 파이썬 3.10.x 이상 (Colab / 주피터 노트북 / Visual Studio Code)
  - 필요한 라이브러리는 수업 시 설치

강의 교재

- **주교재(강의자료) + 부교재**



저자: 최영규

출간: 2023.11

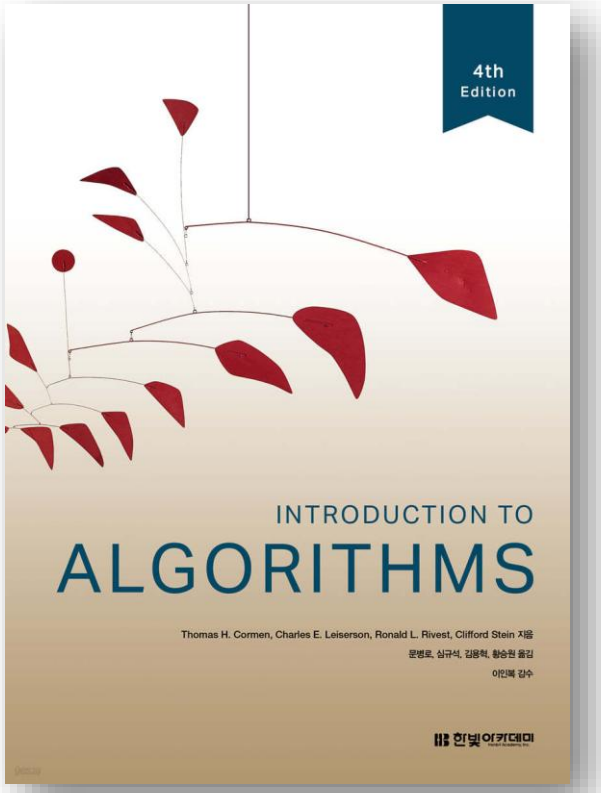
생능북스

저자: 문병로

출간: 2024-01(3판)

한빛아카데미

- **보충 자료**



저자: Robert Sedgewick (4판)

출간: 2024.07 **외**

한빛아카데미

성적 평가방법

평가요소	성적 평가방법	비율
출석	결석 4회 이상의 경우 F 처리, 지각 2회는 결석 1회로 간주	20%
과제	과제 4 회	20%
중간고사	필기 / 실기	30%
기말고사	개인 프로젝트	30%

- 출석체크 : 출결관리시스템(스마트출석부)
- 과제 : 3/6/10/13차시 전후 별도 공지
- 중간고사 : 강의시간에 배운 내용 + 관련지식 기반 필기/실기(프로그래밍) 출제 예정
- 기말고사 대체 : 다양한 알고리즘이 적용된 서비스 프로토타입 만들기

Q & A

※ 가천대학교 컴퓨터공학과 '알고리즘' 과정

알고리즘 개요

Quiz!

알고리즘이란?

알고리즘이란?

- 알고리즘(Algorithm) 정의

위키백과, 우리 모두의 백과사전.


다른 뜻에 대해서는 알고리즘 (동음이의) 문서를 참고하십시오.

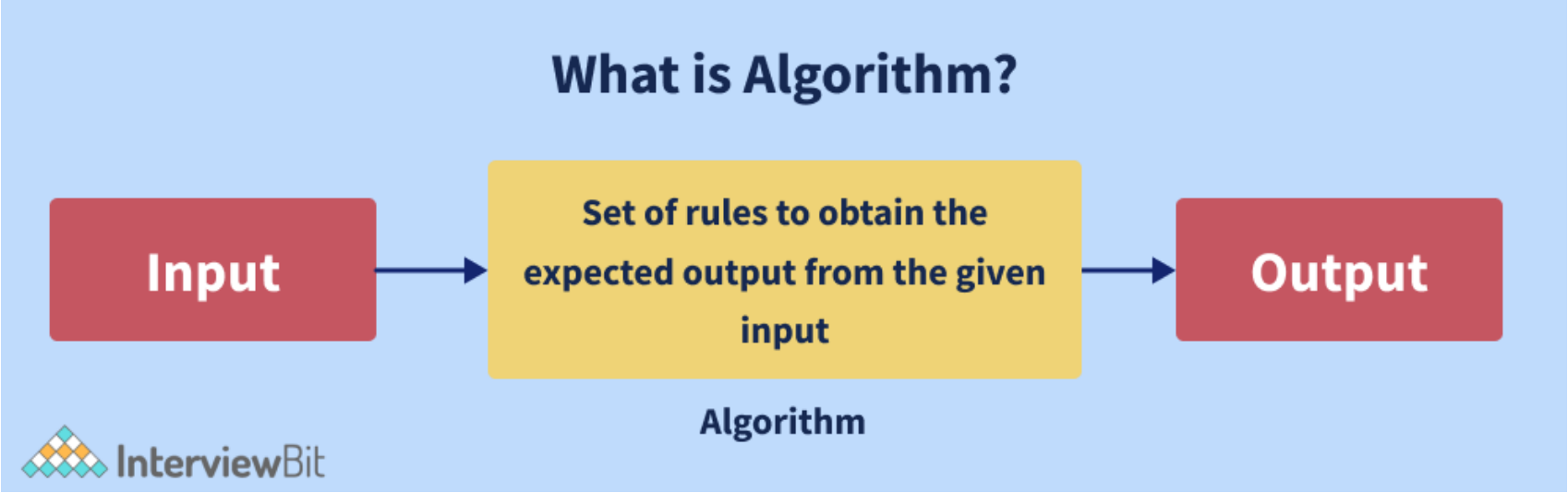
알고리즘(영어: algorithm)은 수학과 컴퓨터과학에서 사용되는, 문제 해결 방법을 정의한 '일련의 단계적 절차'이자 어떠한 문제를 해결하기 위한 '동작들의 모임'이다. 계산을 실행하기 위한 단계적 **규칙과 절차**를 의미하기도 한다. 즉, 문제 풀이에 필요한 계산 절차 또는 처리 과정의 순서를 뜻한다. 프로그램명령어의 집합을 의미하기도 한다.

알고리즘은 연산, 데이터 마이닝(기계 학습) 또는 자동화된 추론을 수행한다.

- 문제를 해결하기 위한 명확한 절차나 규칙의 집합

알고리즘이란?

- 알고리즘(Algorithm) 정의
  - 어떤 값이나 값의 집합을 입력(input)으로 받아 또 다른 값이나 값의 집합을 출력(output)하는 잘 정의된 계산 절차



이미지 출처: https://www.interviewbit.com/algorithm-interview-questions/

알고리즘이란?

- 알고리즘은 작업 과정의 묘사
  - 어떤 작업을 수행하기 위해 입력을 받아 출력을 만들어내는 과정을 애매하지 않게 기술한 것

문제

- 100명의 학생의 시험점수의 최대값을 찾으라

입력

- 100개의 점수

출력

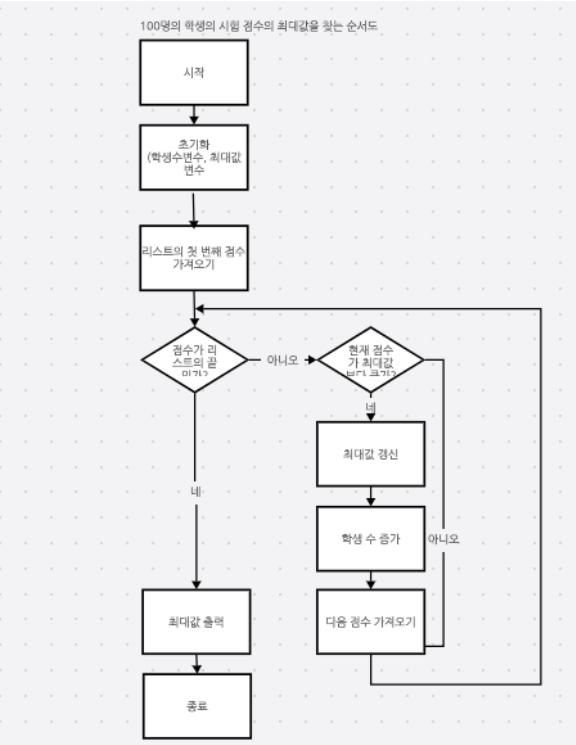
- 입력된 100개의 점수들 중 최댓값

알고리즘

maxScore $(x[], n)$:

$x[1 \ldots n]$의 값을 차례대로 보면서 최댓값을 계산한다

return 위에서 찾은 최댓값



알고리즘이란?

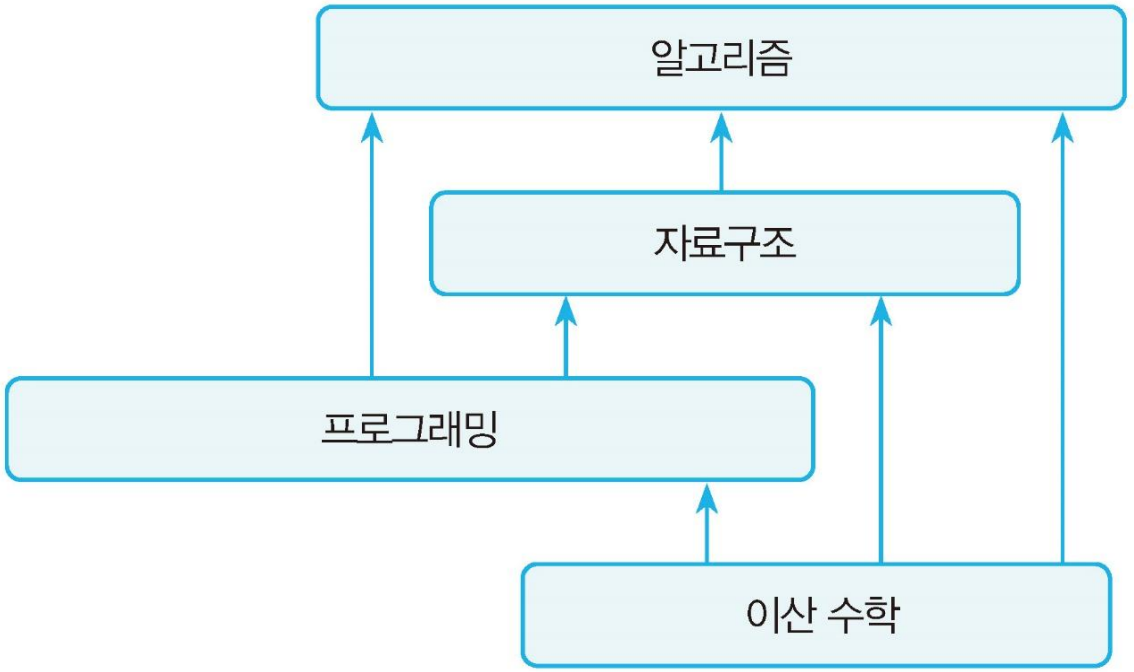
- 일상생활에서 알고리즘이 적용된 사례

알고리즘이란?

- 알고리즘은 생각하는 방법의 훈련
  - 문제 자체를 해결하는 알고리즘을 배운다
  - 그 과정에 깃든 '생각하는 방법'을 배우는 것이 더 중요하다
  - 미래에 다른 문제를 해결하는 생각의 빌딩블록을 제공한다

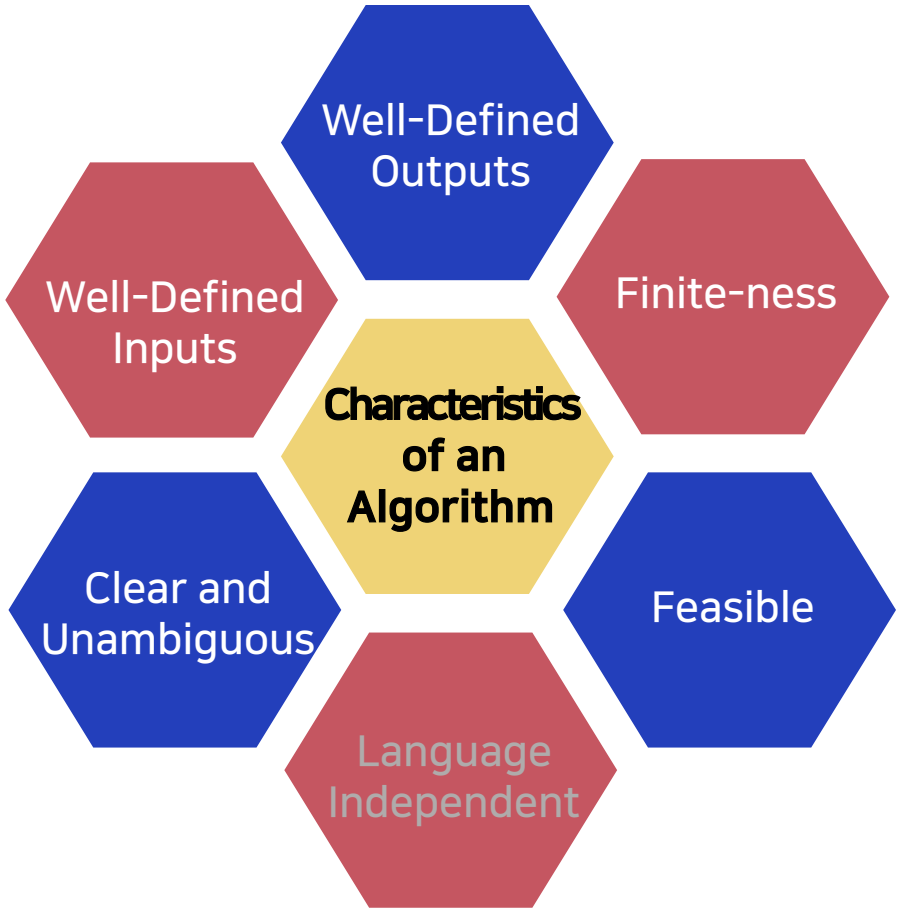
알고리즘이란?

- 알고리즘은 자료구조의 확장이다.
  - 선행과목

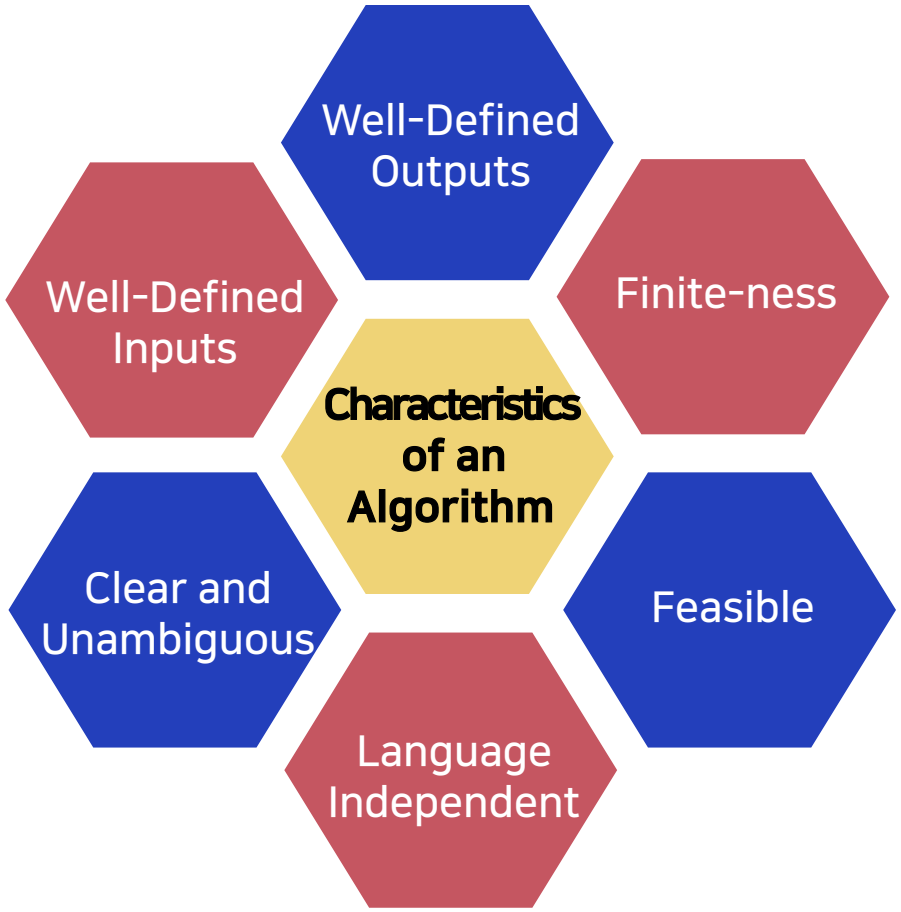


알고리즘, 자료구조, 프로그래밍, 이산 수학의 관계

알고리즘의 조건



바람직한 알고리즘



+

Efficiency

Generality

최소한의 자원을 사용하여

최대한 빠르게(시간)

원하는 결과를 얻어야 함

알고리즘 성능 분석 필요성

- 왜 알고리즘의 성능을 분석해야 할까?
  - 동일한 문제를 해결하는 다양한 알고리즘이 존재
  - 빠른 알고리즘을 선택하면 실행 속도를 크게 단축 가능
  - 컴퓨터 자원을 효율적으로 활용

알고리즘 성능 분석 방법

- 이론적 분석(Asymptotic Analysis)
  - 입력 크기가 매우 클 때의 성능 평가
  - 데이터의 크기에 따른 예측을 통해 객관적으로 비교할 수 있는 기준을 제시
  - 빅오(Big-O) 표기법 사용

- 실험적 분석(Empirical Analysis)
  - 실제 실행 시간을 측정하여 성능 비교
  - 하드웨어 및 환경의 영향을 받을 수 있음

알고리즘 성능 분석 기준

- 성능의 기준
  - **연산량** : 알고리즘이 얼마나 적은 연산을 수행하는가?

$$T : N \to C \ (N = \{n \mid n \geqslant 0,\ n \in \mathbb{N}\},\ C = \{c \mid c \geqslant 0,\ c \in \mathbb{N}\})$$

  - **메모리 사용량** : 얼마나 적은 메모리 공간을 사용하는가?

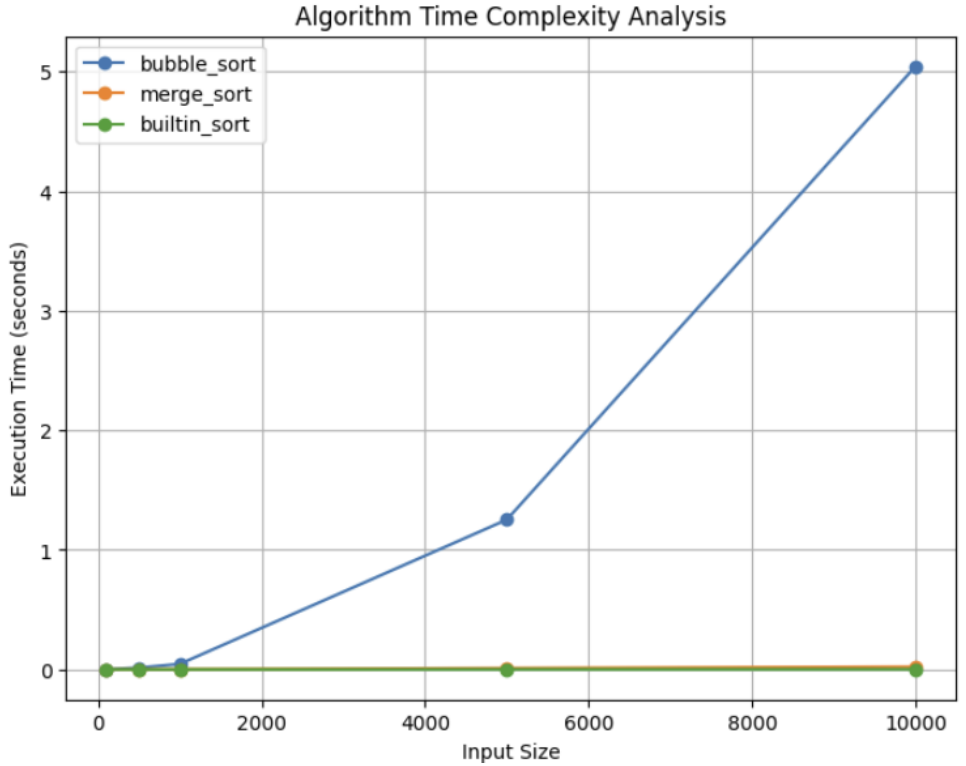
$$S : N \to V \ (N = \{n \mid n \geqslant 0,\ n \in \mathbb{N}\},\ V = \{v \mid v \geqslant 0,\ v \in \mathbb{N}\})$$

알고리즘 성능 분석

- 연산량 확인 방법 예 : (Python)
  1. 연산 횟수 직접 카운팅
  2. Big-O 분석을 위한 타이머 활용 (time)
  3. 프로파일링 도구 활용 (cProfile, line-profiler)
  4. CPU 사용량 확인 (psutil.cpu_percent)

실습문제 : 알고리즘 성능 분석(시간)

- 정렬 알고리즘의 실행 시간 성능 분석하여 그래프로 시각화하기.
  1) bubble_sort, merge_sort, builtin_sort 사용
  2) sizes = [100, 500, 1000, 5000, 10000] 사용 (임의의 정수)
  3) 실행 시간(time) 측정



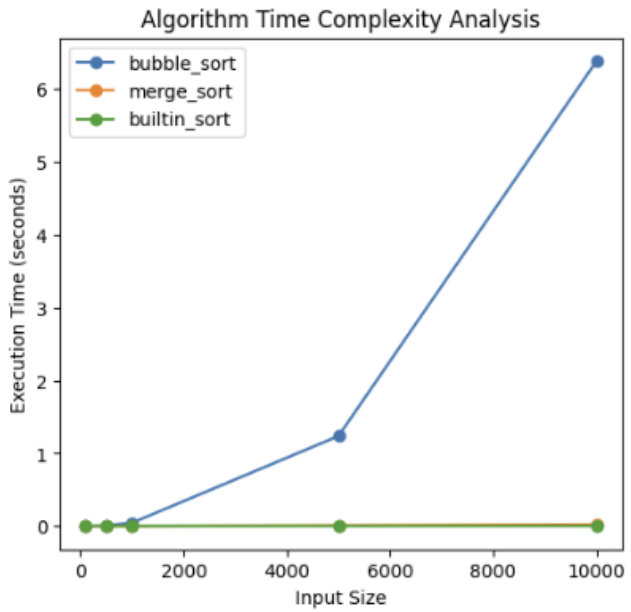
※본 강

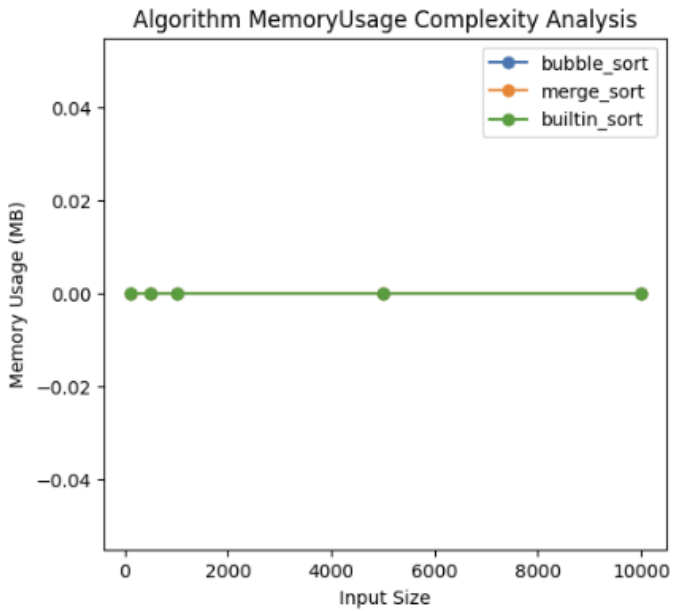
알고리즘 성능 분석

- 메모리 사용량 확인 방법 예 : (Python)
  1. 전체 프로세스 메모리 사용량 측정 (psutil.memory_info)
  2. 코드 라인별 메모리 사용량 측정 (memory_profiiler)
  3. 메모리 사용량 추적 (tracemalloc)
  4. (Garbage Collector) 사용하여 메모리 수거 확인 (gc)

실습문제 : 알고리즘 성능 분석(시간,메모리)

- 알고리즘의 실행 시간, 메모리사용량 성능 분석하여 그래프로 비교하기.
  1) bubble_sort, merge_sort, builtin_sort 사용
  2) sizes = [100, 500, 1000, 5000, 10000] 사용 (임의의 정수)
  3) 실행 시간(time) 측정, 전체 메모리 사용량(psutil.memory_info) 측정



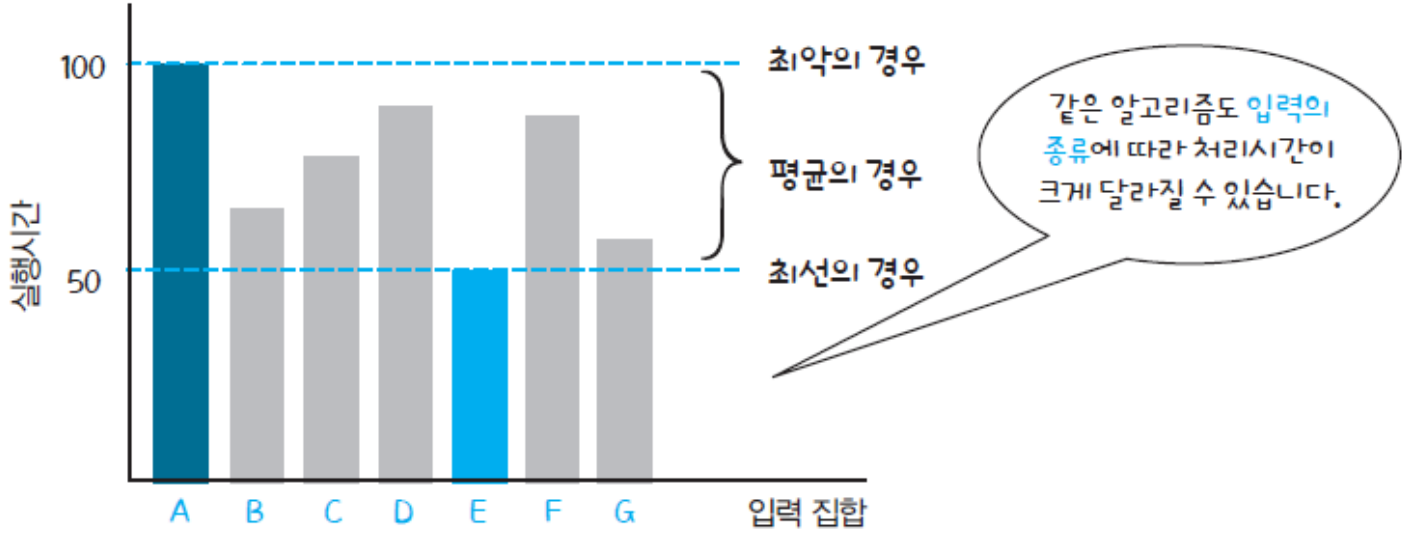


(정리) 알고리즘 성능 분석

- (Python) 성능 분석을 위한 주요 지표
  - 실행 시간 (time 모듈) → 더 빠른 알고리즘 선택
  - CPU 사용량 (psutil.cpu_percent) → CPU 효율 확인
  - 메모리 사용량 (psutil.memory_info, memory_profiler) → RAM 소모 최적화
  - 라인별 메모리 사용량 (memory_profiler) → 메모리 많이 사용하는 부분 분석

알고리즘 복잡도 분석

- 시간 복잡도 분석(Time complexity analysis)
  - 알고리즘이 실행되는 데 걸리는 연산 횟수 알고리즘의 자체 구조에만 영향을 받음
    - 최선의 경우(Best Case) : 빅 오메가(Big-Omega) 표기법
    - 평균적인 경우(Average Case) : 빅 세타(Big-Theta) 표기법
    - 최악의 경우(Worst Case) : 빅 오(Big-O) 표기법

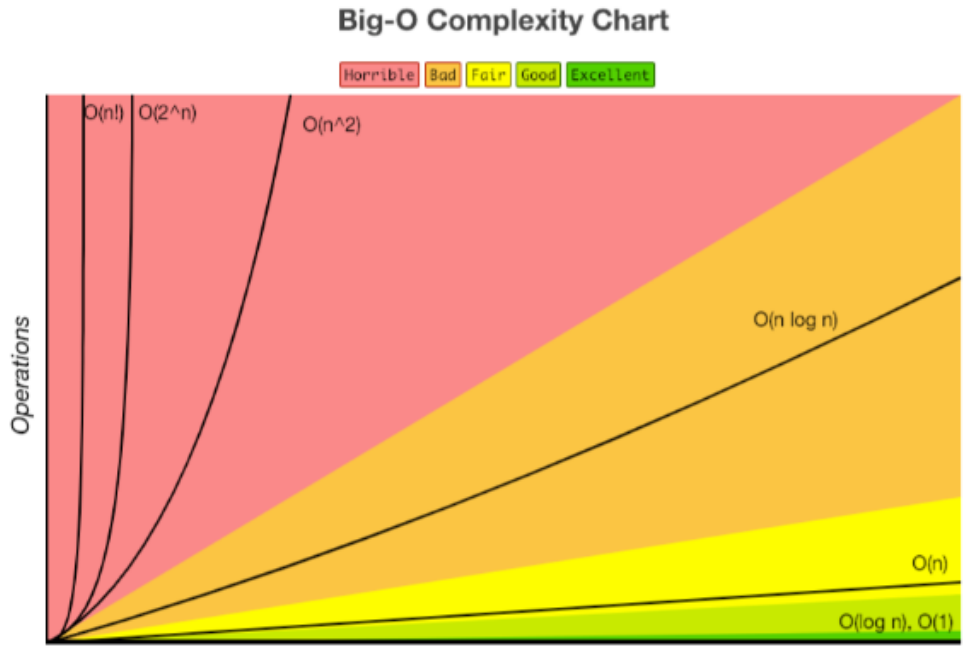


[그림 출처] : 자료구조와 알고리즘 with 파이썬 by 생능북스

빅오 표기법의 예

- 자주 사용되는 빅오 표기의 시간 복잡도

$$O(1) < O(\log n) < O(n) < O(n\log n) < O(n^2) < O(n^3) < O(2^n) < O(3^n) < O(n!)$$



http://bigocheatsheet.com/

빅오 표기법의 예

- 시간 복잡도와 알고리즘 예 :

시간 복잡도	알고리즘 예시	성능 특징
O(1)	해시 테이블 조회	입력 크기에 무관
O(log n)	이진 탐색	데이터가 커도 빠름
O(n)	선형 탐색	데이터 크기에 비례
O(n log n)	병합 정렬	비교적 효율적
O(n^2)	버블 정렬	데이터 크면 매우 느림
O(2^n)	피보나치(재귀)	지수적으로 증가

알고리즘의 표기법

▪ 알고리즘의 다양한 표기법

- **자연어를 이용한 서술적 표현**

  누구나 이해하기 쉬우나 일관성과 명확성이 떨어짐

- **순서도를 이용한 도식화**

  명령의 흐름을 쉽게 파악할 수 있지만 복잡한 알고리즘을 표현하는 데는 한계가 있음

- **프로그래밍 언어를 이용한 구체화**

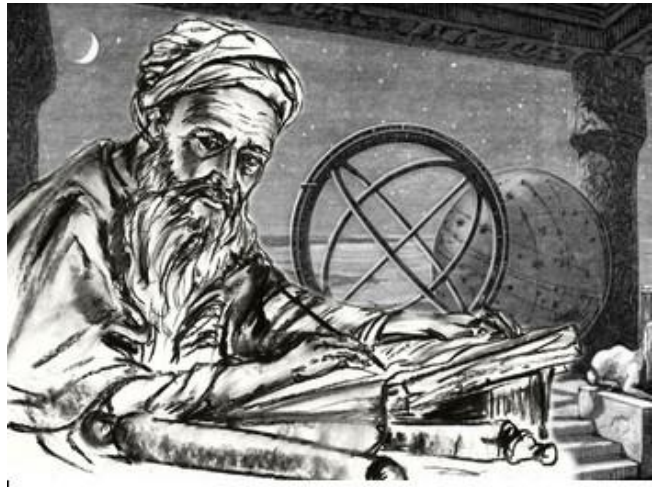
  추가로 구체화할 필요가 없으나 해당 언어를 모르면 이해하기 어려움

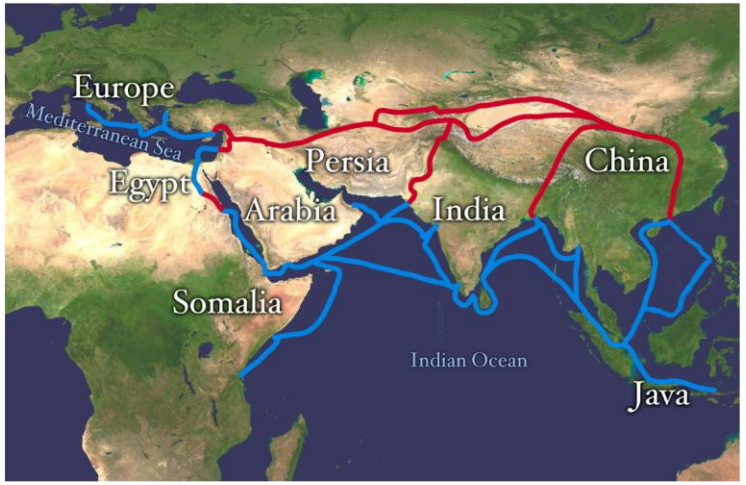
- **가상코드(Pseudo-code)를 이용한 추상화**

  가상코드는 직접 실행할 수는 없지만 일반적인 프로그래밍 언어와 형태가 유사해 프로그래밍 언어로 구체화하기가 쉬움

알고리즘 어원과 역사

알고리즘의 어원





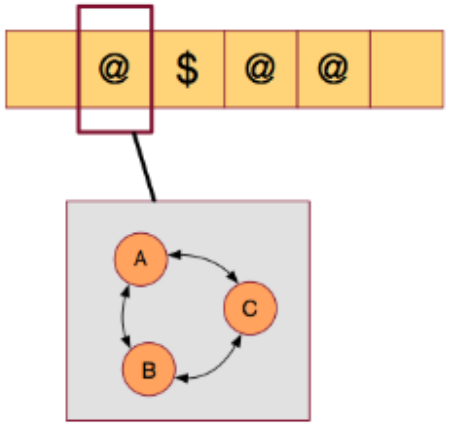
- 9세기 페르시아 당대 최대 과학자&수학자인 무함마드 알 콰리즈미의 이름에서 유래함.
- 유럽에서는 알 콰라지미가 전파한 대수를 '아라비아 수' 또는 '콰라즘에서 온 사람이 가르쳐 준 수(알 콰라즘)' 라고 부름,
- '알 콰라즘' 이 라틴어화 한 '알고리즘(algorithm)' 이라는 단어의 어원이 됨

알고리즘의 역사

시대	수학 기법	수학자	특징	예제
기원전 3000년경	기본 산술 연산	바빌로니아인	쐐기문자를 사용하여 덧셈,곱셈, 나눗셈 계산	바빌로니아 수학판
기원전 300년경	유클리드 알고리즘	유클리드	최대공약수를 구하는 알고리즘	GCD(36,24)=12
9세기	십진법 연산	알콰리즈미	대수학과 알고리즘 개념의 기초 확립	알고리즘(Algorithm)이라는 용어의 유래
17세기	**이진법**	**라이프니츠**	**컴퓨터 연산의 기초가 되는 이진법 개발**	**0과 1로 수 표현**
18세기	그래프 이론	오일러	쾨니히스베르크의 다리 문제를 해결하면서 그래프 이론 창시	오일러 경로와 오일러 서킷
19세기	차분기관	찰스 배비지	기계식 컴퓨터 개념 도입	프로그래밍 가능한 기계 설계
19세기	프로그래밍 개념	에이다 러브레이스	최초의 프로그래밍 알고리즘 설계	배비지의 해석기관을 위한 프로그램
20세기 초	**튜링 머신**	**앨런 튜링**	**이론적 컴퓨터의 개념 정립**	**튜링 기계를 사용한 문제 해결**
20세기 중반	**퀵 정렬**	**토니 호어**	**효율적인 분할 정복 정렬 알고리즘**	**O(n log n) 평균 수행 시간**
20세기 후반	**동적 계획법**	**리처드 벨만**	**최적 부분 구조를 활용한 최적화 기법**	**피보나치 수열 최적화**
21세기	**기계 학습 알고리즘**	**여러 연구자**	**딥러닝**	**강화학습을 통한 패턴 분석**

알고리즘의 역사

- 튜링 머신



Current State	Read Symbol	Write Symbol	Next Position	Next State
A	@	@	->	A
A	$	$	->	B
B	@	@	<-	C
C	$	@	->	C
C	@	$	->	B

- 튜링 머신(Turing machine)은 긴 테이프에 쓰여 있는 여러 가지 기호들을 일정한 규칙에 따라 바꾸는 기계를 의미
- 튜링 기계는 알고리즘적으로 풀 수 있는 문제의 한계를 정의하고, 계산 가능성(computability)을 연구하는 데 중요한 역할을 합
- 엘런 튜링이 제안한 머신으로 현대 컴퓨터의 초안이 됨

이미지 출처: https://norman3.github.io/papers/docs/neural_turing_machine.html

알고리즘의 역사

- 알고리즘의 현대적 의미
  - 알고리즘은 컴퓨터 프로그래밍 및 소프트웨어 개발에서 중심적인 역할을 하게 됨
  - 이는 문제를 해결하기 위한 단계별 절차를 설계하고 구현하는 데 필수적인 요소
  - 데이터 처리, 정보 검색, 인공 지능 개발 등 다양한 분야에서 활용됨
  - 알고리즘의 효율성, 속도, 정확성 등 다양한 특성은 기술의 발전과 함께 계속해서 중요한 연구 주제가 되고 있음

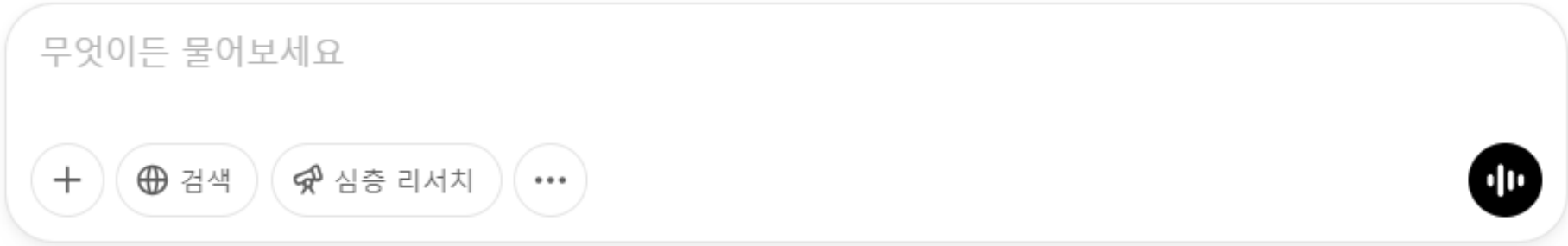
알고리즘 검색

미래를 바꾼 알고리즘

2013년 (국내 출판 2013년 5월)

2025년 현재 최고의 알고리즘은?

- Chatbot에게 물어봐
  - 현재(2025년)까지 개발자들이 가장 자주 사용한 알고리즘 Top10?
  - 현재 컴퓨터공학 전공 학생들이 알고 있어야 할 알고리즘 Top10?
  - 미래를 위해 컴퓨터공학 전공 학생들이 배워야 할 알고리즘 TOP10
  - 알고리즘을 적용하면 좋을 대표적인 OOO 분야 10개?



(필수) 알고 있어야 할 알고리즘 Top10 예 :

알고리즘 유형	대표적인 알고리즘	주요 활용 분야
정렬 알고리즘 (Sorting)	퀵 정렬, 병합 정렬, 힙 정렬	데이터 정렬, 검색 최적화
탐색 알고리즘 (Search)	이진 탐색, DFS, BFS	데이터 탐색, 경로 탐색, AI
동적 계획법 (Dynamic Programming, DP)	피보나치 수열, 최장 공통 부분 수열, 배낭 문제	최적화 문제 해결, 메모이제이션
그리디 알고리즘 (Greedy)	크루스칼, 프림, 허프만 코딩	최적해 탐색, 네트워크 구축
분할 정복 알고리즘 (Divide and Conquer)	병합 정렬, 퀵 정렬, 이진 검색	효율적인 문제 해결, 정렬 및 탐색
그래프 알고리즘 (Graph)	다익스트라, 벨만-포드, 플로이드-워셜	네트워크 경로 탐색, 최단 거리 계산
문자열 매칭 알고리즘 (String Matching)	KMP, 보이어-무어	텍스트 검색, 데이터 검증
해싱(Hashing)과 해시 테이블	해시 함수, SHA-256	데이터 저장 및 검색, 보안
정규 표현식 (Regular Expressions, Regex)	패턴 매칭, 문자열 처리	데이터 검증, 텍스트 분석
머신 러닝 및 AI 알고리즘 (Machine Learning & AI)	선형 회귀, 로지스틱 회귀, CNN, 트랜스포머	데이터 분석, 자연어 처리, 인공지능

알고리즘의 중요성

- 코딩 테스트에 자주 출제되는 문제 : https://programmers.co.kr/

출처: https://school.programmers.co.kr/learn/challenges?tab=algorithm_practice_kit

(진로 역량 향상) 알고 있으면 좋을 알고리즘 Top10 예 :

알고리즘 유형	주요 알고리즘	관련 고연봉 직군	활용 분야
그래프 알고리즘	다익스트라(Dijkstra), BFS/DFS, 최소 신장 트리(MST)	네트워크 엔지니어, 데이터 사이언티스트, ML 엔지니어	소셜 네트워크 분석, 최단 경로 문제, 추천 시스템
기계학습/딥러닝 알고리즘	경사 하강법(Gradient Descent), 역전파(Backpropagation), 신경망 최적화	AI 연구원, ML 엔지니어, 데이터 사이언티스트	예측 모델링, 패턴 인식, 자연어 처리, 컴퓨터 비전
분산 시스템 알고리즘	합의 알고리즘(Consensus), 일관성 알고리즘(Consistency), 장애 허용(Fault Tolerance)	클라우드 아키텍트, 시스템 설계자, 분산 시스템 엔지니어	대규모 분산 시스템, 클라우드 인프라, 고가용성 시스템
최적화 알고리즘	선형 프로그래밍, 유전 알고리즘, 시뮬레이티드 어닐링	금융 엔지니어, 양적 분석가(Quant), 운영 연구 전문가	자원 할당, 포트폴리오 최적화, 스케줄링, 경로 최적화
암호화 및 보안 알고리즘	공개키 암호화, 블록체인 합의, 제로지식 증명	보안 엔지니어, 블록체인 개발자, 암호화폐 전문가	데이터 보안, 개인정보 보호, 디지털 자산 관리

Q & A

Next Topic

- (알고리즘을 위한)자료 구조 기초

Keep learning, see you soon!