

## 17

생성형 인공지능을 기반으로 한  
알고리즘 학습 플랫폼 구현

소속 정보컴퓨터공학부

분과 B

팀명 CodeSphere

참여학생 김대욱, 김문경, 김진우

지도교수 조준수

## 과제 목표

## 동기

- 2025년, 초·중·고교 정보 과목 확대 및 코딩교육 의무화 시행.
- 513만 명의 학생이 프로그래밍을 학습하고, 필수 역량이 됨.
- 채용 시장에서 코딩테스트와 기본 프로그래밍 지식을 요구.
- 그러나, 교사들은 효과적인 코딩 교육에 어려움을 겪고 있음.
- 기존 학습 사이트는 단순 문제 풀이 도구에 지나지 않음.

## 목표

- RAG와 LLM을 활용한 맥락 기반 맞춤형 힌트 제공 기능을 구현.
- 학습 평가를 위한 AI 기반 알고리즘 문제 생성 기능을 구현.
- 사용자별 맞춤형 알고리즘 문제 추천 시스템을 구현.
- 유저 랭킹이나 학습 달력, 스톱워치 등 알고리즘 학습에 도움이 되는 부가기능을 포함한 편리한 UI 구현.

## 서비스 구조

## 문제 생성 시스템

#1000024. 숫자의 속박을 푸는 방정식  
math

문제 설명  
수로 계층이 나뉘는 세상에서, 각 개인의 사회적 지위는 그가 다루는 숫자의 크기에 달려 있습니다. 사회 고위층들은 복잡한 숫자를 다루며 권력을 휘두릅니다. 젊은 수학자인 루안은 이 숫자 사회에서 자신의 진정한 정체성을 찾기 위해 궁극적인 수식을 찾고자 합니다. 이 수식은 두 개의 숫자 사이의 가장 중요한 공통 점을 찾는 것입니다. 루안은 이 문제를 해결하여 사회의 속박에서 벗어나기를 희망합니다. 당신은 루안을 도와 두 숫자 사이의 가장 필수적인 공통점을 찾아야 합니다.

입력  
두 정수 a와 b가 한 줄에 공백으로 구분되어 주어집니다. (1 ≤ a, b ≤ 100,000)

출력  
a와 b의 가장 중요한 공통 점을 출력합니다. 이는 두 숫자를 모두 나눌 수 있는 가장 큰 값입니다.

예제  
예제 입력 1  
48 18  
예제 출력 1  
6  
예제 입력 2  
101 103  
예제 출력 2  
1

⚠️ 제약조건  
1 ≤ a, b ≤ 100,000

문제 정보

문제 ID  
#1000024  
레벨  
5  
테스트 케이스  
2개

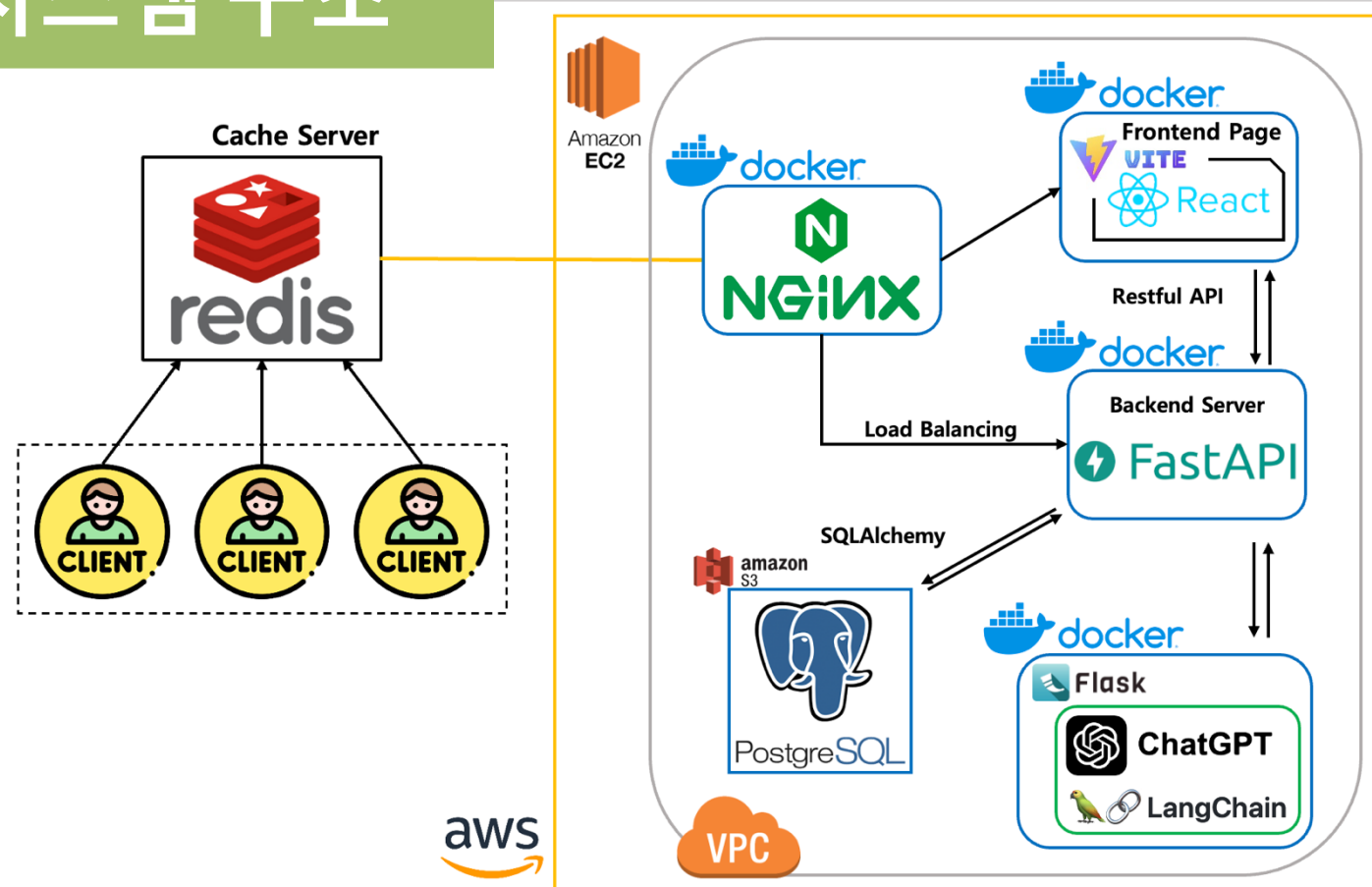
▶ 문제 풀이보기

새 문제 생성

## 알고리즘 문제 생성 과정 요약

1. 빈출 알고리즘 유형을 포함한 문제와 관련 메타 정보를 수집
  2. 문제 생성 패턴을 학습시켜 Few-shot Prompting 방식 문제 생성
  3. Tree-of-Thoughts(ToT)와 Beam Search 기반 다중 경로 탐색 구조를 채택하여 문제 생성 오류 최소화
  4. Self-Verification을 통해 형식 또는 논리적 오류를 2차 수정
- GPT-4o, GPT-5, claude-opus-4.1 등 최신 LLM 성능 교차 비교를 통해, **생성 속도 및 정확도가 우수한 모델 선정**
  - Chain-of-Thoughts(CoT) 대비 **ToT는 사고 경로를 분기·선택함**으로써 논리적 다양성과 오류 회복력을 강화함
  - 문제 유형별로 프롬프트 체인을 분리하여 재현성 높고 관리 가능한 구조로 설계

## 시스템 구조



## 시나리오

- 사용자는 서비스에 접속하고 로그인/회원가입을 진행
- 로그인 사용자의 요청은 NGINX를 거쳐 FastAPI 기반 서버로 전달
- 백엔드 서버는 PostgreSQL 데이터베이스에 사용자 문제 풀이 기록과 학습 데이터를 저장
- 문제 생성 요청 시, **LLM 생성 파이프라인**을 통해 신규 문제를 생성하고 자동 테스트케이스를 검증
- 힌트 요청 시, **RAG 파이프라인**으로 문제·풀이·사용자 코드 정보를 검색 후 LLM에 전달하여 단계별 힌트를 생성
- 생성된 문제와 힌트는 서비스 DB에 기록되며, 사용자는 웹 인터페이스를 통해 결과를 확인

## 알고리즘 플랫폼 구현

RGB거리  
dp

문제  
RGB거리에는 길이 N개 있다. 거리는 선분으로 나타낼 수 있고, 1번 집부터 N번 집이 순서대로 있다. 집은 빨강, 초록, 파랑 중 하나의 색으로 칠해야 한다. 각각의 집을 빨강, 초록, 파랑으로 칠하는 비용이 주어질때, 이때 구적을 만족하면서 모든 집을 칠하는 비용의 최솟값을 구해보자. 1번 집의 색은 2번 집의 색과 같지 않아야 한다. N번 집의 색은 N-1번 집의 색과 같지 않아야 한다. (2 ≤ i ≤ N-1) 번 집의 색은 i-1번, i+1번 집의 색과 같지 않아야 한다.

입력  
첫째 줄에 집의 수 N(2 ≤ N ≤ 1,000)이 주어진다. 둘째 줄부터 N개의 줄에는 각 집을 빨강, 초록, 파랑으로 칠하는 비용이 1번 집부터 한 줄에 하나의 주어진다. 집을 칠하는 비용은 1,000보다 작거나 같은 자연수이다.

출력  
첫째 줄에 모든 집을 칠하는 비용의 최솟값을 출력한다.

예제  
예제 입력 1  
3  
26 40 83  
49 60 57  
13 89 99  
예제 출력 1  
96  
예제 입력 2  
3  
1 100 100  
예제 출력 2  
3

Python 실행

코드 에디터 콘솔

1 import sys  
2 input = sys.stdin.readline  
3  
4 N = int(input())  
5 cost = [list(map(int, input().split())) for \_ in range(N)]  
6  
7 dp = [[0] \* 3 for \_ in range(N)]  
8 dp[0] = cost[0] # 첫 번째 집은 그대로 초기화  
9  
10 for i in range(1, N):  
11 dp[i][0] = cost[i][0] + min(dp[i-1][1], dp[i-1][2])  
12 dp[i][1] = cost[i][1] + min(dp[i-1][0], dp[i-1][2])  
13 dp[i][2] = cost[i][2] + min(dp[i-1][0], dp[i-1][1])  
14  
15 print(min(dp[N-1]))  
16

제출 결과  
10/10 통과한 테스트  
실행 시간 153ms  
메모리 7552

코드 에디터 콘솔

콘솔 출력

===== 실행 결과 =====  
결과 : PASS  
통과 : 5/5  
실행 시간 : 75 ms  
메모리 : 7680 KB  
===== 테스트 케이스 1 =====  
입력값 : 3  
26 40 83  
49 60 57  
13 89 99  
기댓값 : 96  
실제 출력 : 96  
결과 : PASS  
===== 테스트 케이스 2 =====  
입력값 : 3  
1 100 100  
100 1 100  
100 100 1  
기댓값 : 3  
실제 출력 : 3  
결과 : PASS  
===== 테스트 케이스 3 =====

RGB거리  
dp

문제  
RGB거리에는 길이 N개 있다. 거리는 선분으로 나타낼 수 있고, 1번 집부터 N번 집이 순서대로 있다. 집은 빨강, 초록, 파랑 중 하나의 색으로 칠해야 한다. 각각의 집을 빨강, 초록, 파랑으로 칠하는 비용이 주어질때, 이때 구적을 만족하면서 모든 집을 칠하는 비용의 최솟값을 구해보자. 1번 집의 색은 2번 집의 색과 같지 않아야 한다. N번 집의 색은 N-1번 집의 색과 같지 않아야 한다. (2 ≤ i ≤ N-1) 번 집의 색은 i-1번, i+1번 집의 색과 같지 않아야 한다.

입력  
첫째 줄에 집의 수 N(2 ≤ N ≤ 1,000)이 주어진다. 둘째 줄부터 N개의 줄에는 각 집을 빨강, 초록, 파랑으로 칠하는 비용이 1번 집부터 한 줄에 하나의 주어진다. 집을 칠하는 비용은 1,000보다 작거나 같은 자연수이다.

출력  
첫째 줄에 모든 집을 칠하는 비용의 최솟값을 출력한다.

예제  
예제 입력 1  
3  
26 40 83  
49 60 57  
13 89 99  
예제 출력 1  
96  
예제 입력 2  
3  
1 100 100  
100 1 100  
100 100 1  
예제 출력 2  
3

힌트 2/3

【1번째 힌트】  
코드 힌트: 학생의 코드는 전반적으로 올바른 방향으로 작성되어 있습니다. 그러나 문제가 발생할 수 있는 것은 입력을 읽어들이는 방법입니다. sys.stdin.readline을 줄 끝에 있는 개행 문자를 포함하므로, 이 점을 유의하여 지 문자열의 예상치 못한 오타가 발생할 수 있습니다. strip() 메소드를 활용하여 불필요한 개행 문자를 제거하고 입력을 받도록 하면 안전합니다. 입력에서 각 숫자를 잘못 읽어들이고 있는지 확인해보세요.  
2025-09-10 22:48:01  
【2번째 힌트】  
학생의 코드는 RGB거리 문제를 해결하기 위한 동적 계획법(DP) 접근을 잘 따라가고 있습니다. 다만, 지금 코드의 흐름은 정확히 맞으며 높은 효율성을 가지고 있습니다.  
코드 힌트: 현재 코드에서는 초기 조건 및 점화식 설정이 모두 잘 되어 있으므로, 특별한 한 논리적 오류는 발견되지 않습니다. 따라서, 지금과 같은 방식으로 DP 테이블을 채운 후 마지막 집에서의 최소 비용을 출력하는 부분까지 진행하면 됩니다. 이미 제외한 최종 결과를 출력할 수 있는 상태이므로 어떤 부분을 수정할 필요는 없어 보입니다. 혹시 출력값이 기대와 다르다면 입력값을 다시 확인해 보세요.  
2025-09-15 15:27:02

힌트 사용하기(2 / 3)