1. 스택 (3점)

스택(Stack)이 하나 있다. 스택은 자연수를 넣었다가 뺄 수 있는 자료 구조이며, 나중에 넣은 원소가 먼저 나온다는 특징이 있다.

처음에 스택은 비어 있었고, 이후 스택에 아래 연산을 순서대로 진행했다.

- 1, 2, 3, 4를 차례대로 스택에 넣었다.
- 스택에서 자연수 2개를 뺐다.
- 5, 6, 7, 8, 9을 차례대로 스택에 넣었다.
- 스택에서 자연수 3개를 뺐다.
- 10, 11을 스택에 넣었다.
- 스택에 있는 모든 자연수를 뺐다.

이때, 다음 보기 중 옳지 **않은** 것은?

» 스택에서 다섯 번째로 빠져나온 자연수는 7이다.

» 스택에서 여섯 번째로 빠져나온 자연수는 11이다.

» 스택에서 마지막에서 세 번째로 빠져나온 자연수는 6이다. (정답)

» 자연수 10은 자연수 6보다 먼저 스택에서 빠져나왔다.

» 스택에서 세 번째로 빠져나온 자연수와 네 번째로 빠져나온 자연수의 차는 1이다.

2. 달리기 시합 (3점)

네 로봇 A, B, C, D가 50미터 달리기 시합을 한다. 네 로봇은 각각 자신의 직선 주로를 따라 처음부터 끝까지 일정한 속력으로 달린다.

달리기 시합 결과, 아래와 같은 정보를 알 수 있었다.

- A가 결승선에 도착한 순간, B는 결승선까지 정확히 10미터 남아 있었다.
- B가 결승선에 도착한 순간, C는 결승선까지 정확히 15미터 남아 있었다.
- C가 결승선에 도착한 순간, D는 결승선까지 정확히 25미터 남아 있었다.

이때 A가 결승선에 도착한 순간, D는 결승선까지 정확히 몇 미터 남아 있었는가?

정답: 36

3. 마법 문자열 (5점)

마법사가 문자열을 저장하는 변수 S를 갖고 있다. S는 처음에 "1"이다.

마법사는 아래와 같은 두 종류의 주문을 통해 S를 변환할 수 있다.

- "1" 외치기: S의 맨 뒤에 '1'을 하나 붙인다.
- "2" 외치기: S 전체를 뒤집은 후, 맨 뒤에 "2"를 하나 붙인다.

마법사가 주문을 모두 외운 후 S는 "221211221121"이었다. 마법사가 외친 숫자들을 공백 없이 외친 순서대로 나열하라.

정답: 12122211221

4. 숫자 제거 (6점)

295168734에서 다섯 개의 숫자를 제거한 후 남은 네 개의 숫자를 순서대로 읽으면, 네 자리 자연수가 만들어진다.

예를 들어 295168734에서 9, 1, 6, 3, 4을 제거하면 네 자리 자연수 2587이 만들어진다.

이렇게 만들 수 있는 모든 네 자리 자연수 가운데, 49번째로 큰 수는?

정답: 5673

5. 19단의 자리수 (9점)

19단표는 행(가로줄)이 19개, 열(세로줄)이 19개인 표로, 1 이상 19 이하의 각 정수 x, y에 대해, x번째 줄의 y번째 칸에는 $x \times y$의 값이 적혀 있다. 즉, 19단표에는 $19 \times 19 = 361$개의 자연수가 적혀 있는 것이다.

19단표에 있는 자연수 중 한 자리 수의 개수를 A, 두 자리 수의 개수를 B, 세 자리 수의 개수를 C라고 하자.

$A + 2B + 3C$의 값은?

정답: 845

6. 투표 결과 (9점)

7명의 사람이 3명의 후보 A, B, C에게 투표를 한다.

각각의 사람은 독립적으로 세 후보 중 한 명에게 투표를 할 수 있다. 따라서 투표를 할 수 있는 모든 경우의 수는 3^7가지 있다.

모든 경우 가운데, A가 받은 득표수가 B가 받은 득표수보다 많은 경우의 수는 몇 가지인가?

정답: 897

7. 돌 가져가기 게임 (9점)

두 사람 A와 B가 다음과 같은 게임을 한다.

처음에 탁자 위에 n개의 돌이 있다.

A부터 시작해서 번갈아 가면서 게임을 진행한다. A는 자신의 차례에 13의 배수인 양수 개수의 돌만 가져갈 수 있다. B는 자신의 차례에 9의 배수인 양수 개수의 돌만 가져갈 수 있다. 자신의 차례에 돌을 가져갈 수 없는 사람이 진다.

A와 B 모두 항상 자신이 이기기 위해 최선을 다한다.

1 이상 1 000 이하의 모든 자연수 n 가운데, 위와 같은 게임에서 A가 이기도록 하는 n의 개수는?

정답: 684

8. 거리 점프 (10점)

수직선 위에 닫힌 구간 $[20, 25]$가 있다.

정수 t에 대해, $f(t)$의 값은, 처음 $x = t$ 위치에서 닫힌 구간 $[20, 25]$로 들어가기 위해 이동해야 하는 최소 거리로 정의한다. 즉:

- $t < 20$이면 $f(t) = 20 - t$이다.
- $20 \le t \le 25$이면 $f(t) = 0$이다.
- $t > 25$이면 $f(t) = t - 25$이다.

좌표 x_0에 있던 사람이 한 번의 **거리 점프**를 하면, 좌표 $f(x_0)$로 이동한다.

좌표 x_0에 있던 사람이 n번의 **거리 점프**를 했을 때 이동한 좌표를 $f^n(x_0)$ 라고 하자. 즉, $f^1(x_0) = f(x_0)$이고, $n \ge 2$이면 $f^n(x_0) = f(f^{n-1}(x_0))$이다.

$f^{10}(x) = 0$이도록 하는 정수 x의 값의 개수를 c, 합을 s라 하자. $c + s$의 값은?

정답: 1410

9. 홀수 평균 (11점)

다음 두 조건을 만족하는 서로 다른 세 정수의 순서쌍 (x, y, z)의 개수를 구하라.

- $1 \le x < y < z \le 30$
- 세 수의 평균인 $\frac{x+y+z}{3}$는 **홀수**$(1, 3, 5, 7, 9, 11, \dots)$이다.

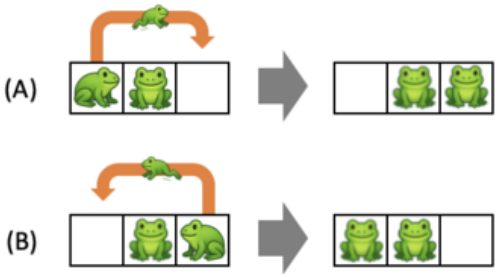
정답: 680

10. 개구리 점프 (14점)

아래와 같이 일렬로 배열된 24개의 칸이 있고, 각 칸은 비어 있거나 개구리 한 마리가 들어 있다.

처음에 아래와 같이 15개의 칸에만 개구리가 한 마리씩 놓여 있고 나머지 9개의 칸은 비어 있다.

각각의 개구리는 다음의 규칙에 의해 점프할 수 있다. 점프는 아래의 방식으로만 가능하며, 이외의 다른 이동은 허용되지 않는다.



- (A) 어느 개구리의 바로 오른쪽 칸에 다른 개구리가 있고, 오른쪽으로 두 칸 떨어진 칸이 비어 있으면, 이 개구리는 오른쪽으로 두 칸 떨어진 칸으로 점프할 수 있다.
- (B) 어느 개구리의 바로 왼쪽 칸에 다른 개구리가 있고, 왼쪽으로 두 칸 떨어진 칸이 비어 있으면, 이 개구리는 왼쪽으로 두 칸 떨어진 칸으로 점프할 수 있다.

개구리는 빈 칸으로만 점프할 수 있으며, 칸들 외의 영역으로는 이동할 수 없다.

위의 점프를 임의의 순서로 수행하여 얻을 수 있는 모든 배치를 고려할 때, 개구리들이 놓이게 되는 서로 다른 배치의 가짓수는 모두 몇 가지인가? (단, 개구리는 서로 구별하지 않는다.)

정답: 5005

11. 팀 나누기 (15점)

1번부터 12번까지의 번호가 붙은 총 12명의 선수가 있다. 이 선수들을 6명씩 두 팀(A팀, B팀)으로 정확히 나누려 한다.

두 팀은 미리 정해진 다음과 같은 순서로 선수를 한 명씩 선발한다: A $\rightarrow$ B $\rightarrow$ B $\rightarrow$ A $\rightarrow$ A $\rightarrow$ B $\rightarrow$ B $\rightarrow$ A $\rightarrow$ A $\rightarrow$ B $\rightarrow$ B $\rightarrow$ A

각 팀은 자신의 차례가 왔을 때, 아직 선발되지 않은 선수들 가운데 가장 선호하는 선수를 선발한다. 각 팀의 선수 선호 기준은 아래와 같다.

- **A팀:** 번호가 더 작은 선수를 선호한다.
- **B팀:** 선수들에 대한 선호 순위를 나타내는 $\{1, 2, \cdots, 12\}$의 순열 p를 가지고 있다. 각 i에 대해, i번 선수는 전체에서 p_i번째로 선호하는 선수이다. 즉, p_i의 값이 더 작은 선수 i를 선호한다.

가능한 모든 12!가지의 순열 p에 대해 위의 선발 규칙을 수행하여 두 팀(A, B팀)의 최종 구성을 결정할 때, 나올 수 있는 서로 다른 팀 구성 결과는 모두 몇 가지인가?

두 팀의 구성 결과가 서로 다르다는 것은, 소속팀이 다른 선수가 적어도 한 명 이상 존재한다는 것을 의미한다.

정답: 211

12. 연산의 합 (15점)

다음과 같은 수식을 생각해 보자: $1 \square 1 \square 1 \square 1 \square 1 \square 1 \square 1$

각 빈칸 $\square$에는 덧셈 연산자(+), XOR 연산자(^), AND 연산자(&), OR 연산자(|) 중 원하는 연산자를 독립적으로 선택하여 넣을 수 있다.

연산자의 우선순위는 아래와 같다. 아래의 우선순위는 프로그래밍 언어(C, C++, Java, Python)의 동작과는 다르다는 점에 유의하라.

- 비트 연산자인 ^, &, | 연산자의 우선순위는 서로 동일하다. 같은 우선순위를 가진 비트 연산자는 반드시 왼쪽에서 오른쪽으로 차례대로 계산한다.
- 덧셈 연산자(+)는 비트 연산자보다 우선순위가 낮으며, 항상 가장 마지막에 계산한다.

즉, 전체 수식을 계산할 때는 먼저 덧셈 연산자(+)를 기준으로 여러 개의 부분식으로 나눠서 각각의 값을 구한 다음, 이 부분식의 결괏값들을 더해 최종적으로 전체의 결괏값을 구한다. 각 부분식 내부는 비트 연산자로만 구성되며 왼쪽에서 오른쪽으로 차례대로 계산한다.

예시로, $1 \text{ ^ } 1 \text{ \& } 1 + 1 \text{ | } 1 \text{ \& } 1 + 1 = ((1 \text{ ^ } 1) \text{ \& } 1) + ((1 \text{ | } 1) \text{ \& } 1) + 1 = 2$이다.

모든 빈칸 $\square$에 연산자 4가지(+, ^, &, |)를 독립적으로 자유롭게 넣을 수 있으므로, 빈칸을 채워 수식을 완성할 수 있는 방법의 수는 총 $4^6 = 4096$가지 있다.

각각의 완성된 수식에 대해 결괏값을 계산한 후 모두 더한 값은 얼마인가?

정답: 7936

13. 트리 높이 줄이기 (8점)

아래 그림에는 트리가 있다. 트리는 정점들과 간선들로 이루어진 구조이다. 간선은 정점들을 부모-자식 관계로 연결하고 있다. 위쪽에 있는 정점은 부모, 아래쪽에 있는 정점은 자식이라고 부른다. 어떤 정점의 자식도 아닌 정점을 루트, 어떤 정점의 부모도 아닌 정점을 리프라고 부른다. 트리의 높이란 루트에서 리프까지의 경로 위 간선 길이의 합 중 최댓값을 뜻한다.

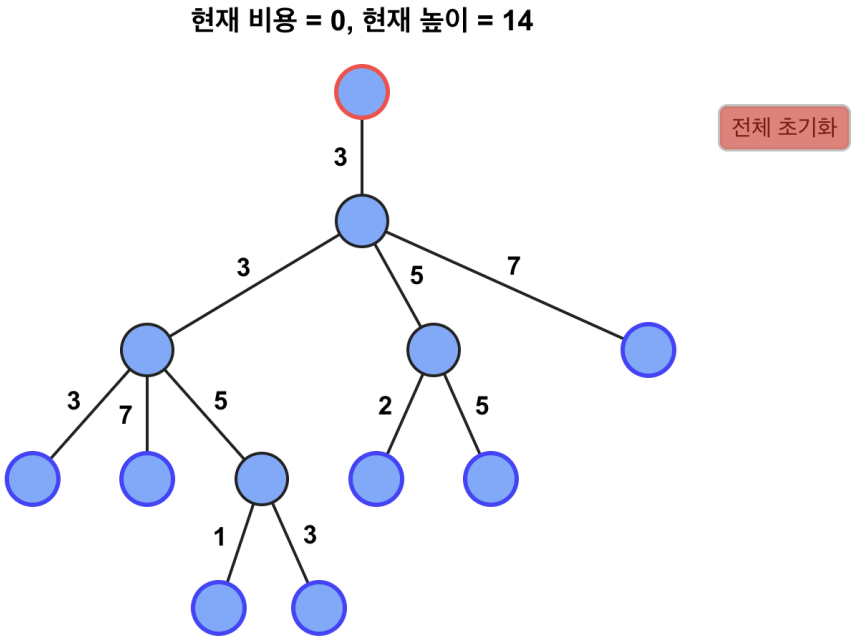
정점은 원으로 표시되어 있고, 각 간선 옆에는 해당 간선의 길이가 나타나 있다. 루트 정점은 빨간 윤곽선으로, 리프 정점은 파란 윤곽선으로 강조되어 있다.

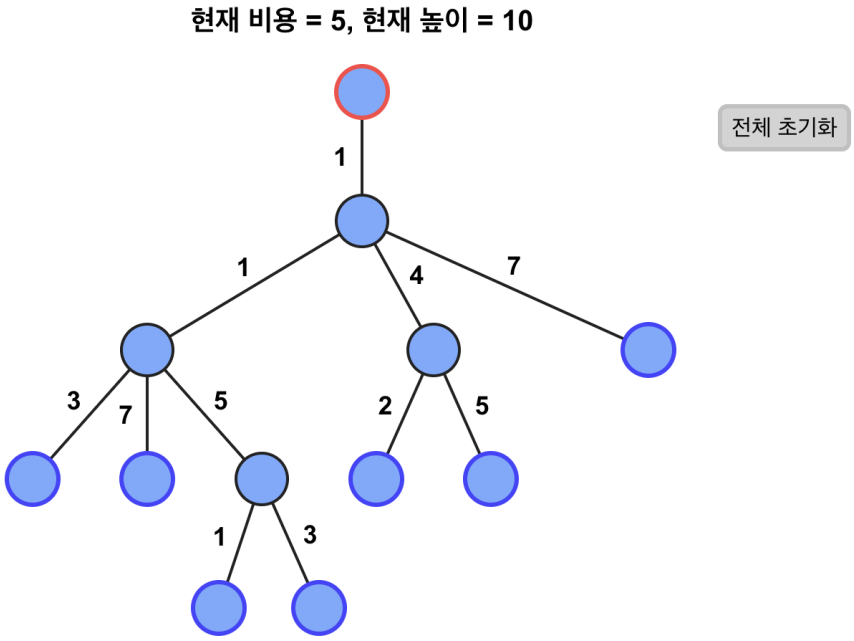
간선을 클릭한 다음 아래의 ▼ 버튼을 클릭하면 해당 간선의 길이를 1만큼 줄일 수 있으며, 초기화 버튼을 클릭하면 지금까지 줄인 이 간선의 길이를 초기화할 수 있다. 길이를 1만큼 줄이는 데에는 비용 1이 발생하며, 길이는 최소 1까지 줄일 수 있다. 간선의 길이는 실제 화면에 표시된 선의 길이와는 무관함에 유의하라. 화면의 상단에는 지금까지 사용한 비용이 표시되어 있다.

간선의 길이를 적절히 줄여 트리의 높이가 10 이하가 되도록 만들어야 한다. 또한, 길이를 줄이는 데에 드는 비용 또한 최소화하여야 한다.

채점 기준

트리의 높이가 10 이하이고, 비용이 가능한 최소 비용이면 전체 점수의 100%를 얻는다.





위의 방법 외에도 비용이 5인 방법으로 트리의 높이를 10 이하로 만들었다면 정답으로 처리된다.

14. 사탕 놓기 (8점)

8×8 격자 모양으로 사탕 바구니들이 놓여 있다. 격자의 맨 위쪽 행을 **1행**, 맨 왼쪽 열을 **1열**이라고 정의하며, 각 바구니는 그 좌표를 (i, j)와 같이 나타낸다. 이때, (i, j)는 **i행 j열에 위치한 사탕 바구니**의 좌표를 의미한다.

각 바구니에는 **사탕이 1개 들어 있거나, 아무것도 들어 있지 않다.** 하지만, 사탕이 정확히 어느 바구니에 들어 있는지는 알지 못한다.

다만, 사탕의 배치를 유추할 수 있는 정보가 아래와 같이 주어진다.

> $S(i, j)$를 $1 \le x \le i$, $1 \le y \le j$인 모든 자연수 순서쌍 (x, y) 중 사탕이 들어 있는 바구니의 개수로 정의하자. 단, i와 j는 각각 1 이상 8 이하의 자연수이다.

이때 $S(i, j)$의 값은 아래 표의 i행 j열에 적힌 값과 같다.

행/열	1	2	3	4	5	6	7	8
1	0	1	2	3	4	5	6	6
2	1	3	4	5	6	7	9	9
3	1	4	6	7	9	11	13	13
4	2	5	8	9	11	13	16	17
5	2	6	9	10	13	15	19	20
6	2	7	11	12	15	18	22	23
7	2	7	11	13	16	20	24	25
8	3	8	12	14	18	22	26	27

다시 말해, 표의 i행 j열에 적힌 값은 $1 \le x \le i,\ 1 \le y \le j$인 모든 (x, y) 중 사탕이 들어 있는 바구니의 개수와 같다.

즉, 표의 4행 2열에 5가 적혀 있으므로 $S(4, 2) = 5$이며, $(1, 1)$, $(2, 1)$, $(3, 1)$, $(4, 1)$, $(1, 2)$, $(2, 2)$, $(3, 2)$, $(4, 2)$ 중 5개의 바구니에 사탕이 들어 있다.

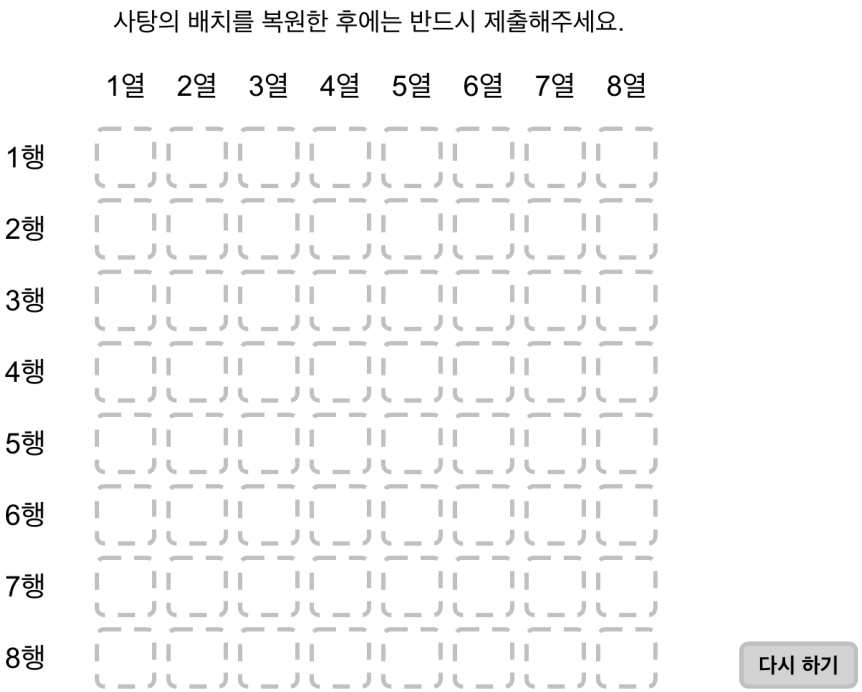
위 정보를 바탕으로, 각 바구니에 사탕이 실제로 어떻게 배치되어 있는지 복원해 보자.

조작 방법

- 각 바구니를 클릭하면 사탕을 넣거나, 이미 들어 있던 사탕을 제거할 수 있다.
  - 바구니의 테두리가 점선이라면, 바구니에 사탕이 없음을 의미한다. 이 경우, 클릭하면 사탕을 넣게 된다.
  - 바구니의 테두리가 실선이고 0가 적혀 있으면, 바구니에 사탕이 있음을 의미한다. 이 경우, 클릭하면 이미 들어 있던 사탕을 제거하게 된다.
- "다시 하기" 버튼을 누르면 처음부터 다시 시작할 수 있다.
- 사탕의 배치를 복원한 후에는 반드시 제출 버튼을 눌러야 한다.

채점 기준

- 사탕의 배치를 정확히 복원하면 점수의 100%를 받을 수 있다.



15. 포크 (9점)

당신은 화면에 주어진 정수 배열에서 포크를 이용해 수들을 선택하여, 그 합을 최대로 만들어야 한다.

이를 위해 당신은 아래와 같은 연산을 할 수 있다.

- **포크로 집기**: 배열의 `i`번째 수를 클릭하면, `i`번째 수와 `i+2`번째 수를 함께 선택한다. 단, 원소는 최대 한 번만 선택할 수 있으므로, 포크를 사용하려면 대상이 되는 두 원소(`i`번째와 `i+2`번째)가 모두 이전에 선택되지 않은 상태여야 한다. (단, 배열의 마지막 두 원소는 클릭할 수 없다.)

당신은 **선택한 수들의 총합**을 최대로 만들어야 한다.

조작 방법

- 수 위에 마우스 커서를 올리면, 선택될 두 수를 잇는 포크 모양이 미리 표시된다.
- 클릭을 통해 "포크로 집기" 연산을 수행할 수 있다.
- 언제든지 "다시 하기" 버튼을 눌러 처음부터 다시 시작할 수 있다.

채점 기준

- 선택한 수들의 총합이 가능한 최댓값이라면 만점을 받는다. 아니라면, 0점을 받는다.

원하는 만큼 포크를 사용한 후에는 반드시 제출해주세요. [현재 선택된 합: 0]

다시 하기

원하는 만큼 포크를 사용한 후에는 반드시 제출해주세요. [현재 선택된 합: 42]

다시 하기

16. 실 태우기 (10점)

길이가 100인 실이 일직선상에 놓여 있다. 편의상 이 실을 좌표 [0, 100] 구간으로 나타낸다.

이 실 위에는 불을 붙일 수 있는 20개의 지점이 있으며, 각 지점의 위치는 화면에 표시되어 있다.

여러분의 목표는 20개 지점 중 **정확히 4개 지점을 선택**하여 **실이 모두 타는 시각을 최소화**하는 것이다.

불타는 규칙

선택한 4개 지점에서 시각 0에 동시에 불을 붙인다. 불이 실을 태우는 방식은 다음과 같다:

- 각 지점에서 불을 붙이면, 불은 그 지점에서 **좌우 양방향으로 동시에** 번진다.
- 불은 **속도 1로** 번진다. 즉, 1초 후에는 양쪽으로 각각 거리 1만큼 번진다.
- 불은 실의 경계인 좌표 0과 100을 넘어서 번질 수 없다.
- 실의 각 부분은 가장 먼저 도달한 불에 의해 타며, 이미 탄 부분은 다른 불이 지나갈 수 없다.
- 불이 퍼지는 방향에 실이 없으면, 불은 그 방향으로 퍼져 나가기를 멈춘다.

실이 모두 타는 시각은 좌표 0부터 100까지의 실이 모두 타서 사라지는 시각을 의미한다.

조작 방법

화면에는 실과 20개의 지점이 표시되어 있다. 각 지점에는 해당 위치의 좌표가 함께 표시된다.

지점 선택하기:

- **지점을 클릭**하면 해당 지점을 선택하거나 선택 해제할 수 있다.
- 선택된 지점은 주황색으로, 선택되지 않은 지점은 청록색으로 표시된다.
- 정확히 4개 지점을 선택해야 한다.

시뮬레이션 실행하기:

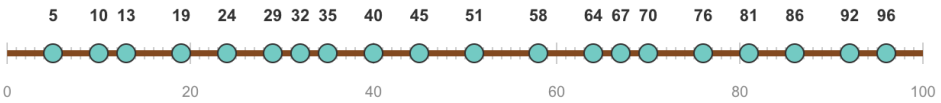
- 4개 지점을 모두 선택하면 “시뮬레이션 시작” 버튼이 활성화된다.
- 시뮬레이션을 실행하면 불이 어떻게 번지는지 시각적으로 확인할 수 있다.
- 각 지점에서 시작된 불은 서로 다른 색상으로 표시된다.

화면 상단에는 현재 선택된 지점의 개수와 상태 메시지가 표시된다. 언제든지 “다시 하기” 버튼을 눌러 처음부터 다시 시작할 수 있다.

채점 기준

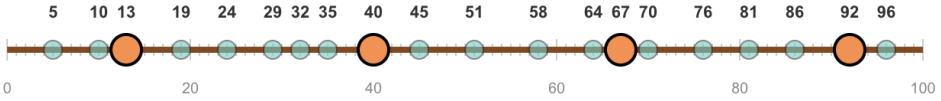
선택한 4개 지점에 불을 붙였을 때 실이 모두 타는 시각이 이론적 최솟값과 일치하면 정답으로 처리한다.

4개 지점을 선택해주세요



다시 하기 시뮬레이션 시작

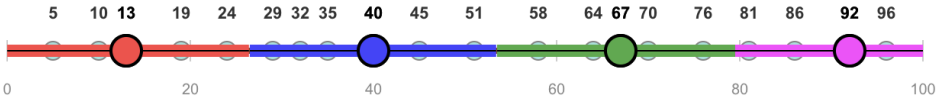
시뮬레이션을 시작할 수 있습니다
다른 지점을 선택하려면 기존 선택을 해제해주세요



다시 하기 시뮬레이션 시작

닫기

실이 모두 타는 시각: 13.5



다시 하기 시뮬레이션 시작

17. 올바른 괄호 문자열 (13점)

길이가 짝수이고, (또는)로 구성된 문자열이 주어질 때, 다음 행동을 **최소한의 횟수**로 수행해서 문자열이 올바른 괄호 문자열이 되도록 하려고 한다.

행동: 어느 문자 하나를 골라서 (를)로 바꾸거나,)를 (로 바꾼다.

올바른 괄호 문자열의 정의는 다음과 같다.

- ()는 올바른 괄호 문자열이다.
- X가 올바른 괄호 문자열이면, (X)도 올바른 괄호 문자열이다.
- X와 Y가 올바른 괄호 문자열이면, XY도 올바른 괄호 문자열이다.

문자열을 올바른 괄호 문자열로 바꾸더라도, 행동 횟수가 최소가 아니면 정답 처리가 되지 않음에 유의하라.

)) (( ) (( ) ( )) ( )) (( ))) (( )) ( )))) (

행동 횟수: 0

올바른 괄호 문자열이 아닙니다.

다시하기

((((((((( )) ( )) (( ))) (( )) ( )))))

행동 횟수: 6

완료했습니다! 반드시 제출 버튼을 눌러주세요.

다시하기

위의 방법 외에도 6회의 행동 횟수로 올바른 괄호 문자열을 만들었다면 정답 처리된다.

18. 버블 거울 정렬 (13점)

각각 정수가 적혀 있는 카드 12장이 나열되어 있다. 이 카드는 투명해서, 뒷면에서도 무슨 수가 쓰여 있는지 볼 수 있다. 초기에 모든 카드는 앞면이다.

인접한 두 카드를 선택해 좌우로 뒤집는 시행을 여러 번 할 수 있다. 이때, 두 카드의 위치가 서로 바뀌고, 두 카드 중 앞면인 카드는 뒷면이, 뒷면인 카드는 앞면이 된다.

최소한의 시행을 사용하여 카드에 적힌 수가 단조 증가하도록 카드를 정렬해야 한다. 최종적으로 모든 카드는 앞면이어야 한다.

뒷면인 카드는 적힌 수가 좌우 대칭되며, 밑줄을 그어 표시한다. 되돌리기 단추를 눌러 마지막 시행을 되돌릴 수 있다.

채점 기준

카드에 적힌 수가 단조 증가하고 뒤집어진 카드가 없다면, 전체 점수의 50%를 얻는다.

카드에 적힌 수가 단조 증가하고 뒤집어진 카드가 없으며, 최소 횟수의 시행을 사용했다면 전체 점수의 100%를 얻는다.

왼쪽에서부터 i번째에 위치한 "뒤집기" 버튼을 누르는 동작을 자연수 i로 나타냈을 때, 아래와 같이 진행하면 26회의 동작으로 카드를 정렬할 수 있다.

- 8, 7, 2, 1, 5, 4, 3, 2, 6, 5, 4, 3, 9, 8, 7, 6, 5, 4, 11, 10, 6, 7, 9, 8, 9, 11

이 방법 외에도 26회의 동작으로 카드를 정렬했다면 정답 처리된다.

19. 수식 최대화 (15점)

여러 개의 정수가 순서대로 나열되어 있고, 이 정수들 사이에 빼기(-) 기호가 있는 수식이 주어진다.

여러분의 목표는 이 수식에 **괄호를 적절히 추가하여** 수식의 값을 **최대화**하는 것이다.

예를 들어 수식이 $3-1-4$와 같을 때, 괄호를 추가하지 않고 계산하면 $3-1-4=-2$가 되지만, $3-(1-4)$와 같이 괄호를 추가하면 $3-(-3)=6$이 되어 더 큰 값을 얻을 수 있다.

여러분은 두 개의 부분문제를 해결해야 한다. 각 부분문제는 독립적이며, 각각 추가할 수 있는 괄호의 개수 제한이 있다.

- **부분문제 1**: 9개의 정수가 주어지며, 최대 10개의 괄호 쌍을 추가할 수 있다.
- **부분문제 2**: 18개의 정수가 주어지며, 최대 20개의 괄호 쌍을 추가할 수 있다.

조작 방법

"(수식의 값) = (수식)" 형태로 표시된다. 수식의 각 정수는 원 안에 표시되어 있고, 원들 사이에는 빼기 기호가 있다.

괄호 추가하기:

1. **시작 원을 클릭**하여 첫 번째 원을 선택한다.
2. **끝 원을 클릭**하여 두 번째 원을 선택한다.
3. 선택한 두 원과 그 사이의 모든 원들을 포함하는 괄호 쌍 (와)가 추가된다. 추가한 괄호가 짝이 맞아야만 추가할 수 있다.

괄호 제거하기:

- 기존 괄호 위에 마우스를 올리면, 짝이 맞는 두 괄호가 강조 표시된다.
- **괄호를 클릭**하면 강조된 해당 괄호 쌍이 제거된다.

화면 하단에는 현재 상황에 맞는 안내 메시지가 표시된다. 언제든지 "다시 하기" 버튼을 눌러 처음부터 다시 시작할 수 있다.

채점 기준

각 부분문제를 **해결했다**는 것은, 해당 부분문제에서 수식의 계산 결과가 이론적으로 가능한 최댓값과 같다는 것이다.

- 부분문제 1을 해결하면 전체 점수의 20%를 받는다.
- 부분문제 2를 해결하면 전체 점수의 80%를 받는다.
- 부분문제를 해결하지 못하면 해당 부분문제의 점수를 받지 못한다.

최종 점수는 각 부분문제에서 얻은 점수의 합이다.

위의 방법 외에도 괄호를 적절히 추가하여 계산한 수식의 값이 각각 34, 84라면 정답 처리된다.

20. 동전 뒤집기 (15점)

36개의 동전이 6개의 행, 6개의 열로 이루어진 격자 형태로 배치되어 있다. 각 동전의 앞면에는 `H`가, 뒷면에는 `T`가 적혀있다. 다시 말해, 현재 화면에 `H`가 표시되는 동전은 앞면이 위를, `T`가 표시되는 동전은 뒷면이 위를 향하고 있다.

당신은 다음 행동을 통해 동전 중 일부를 뒤집을 수 있다.

- 동전을 하나 선택하여 선택한 동전을 포함하여 같은 행 또는 같은 열에 있는 11개의 동전을 모두 뒤집는다.

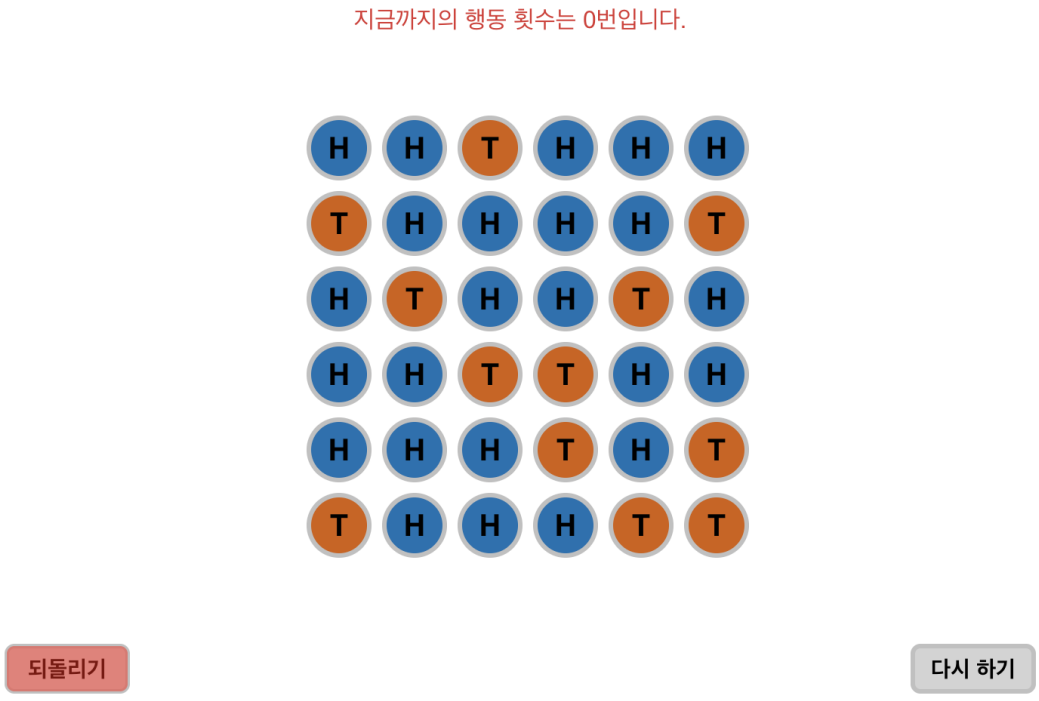
당신은 적절한 행동을 통해 모든 동전이 앞면을 위로 향하도록 하되, 행동 횟수를 최소화해야 한다.

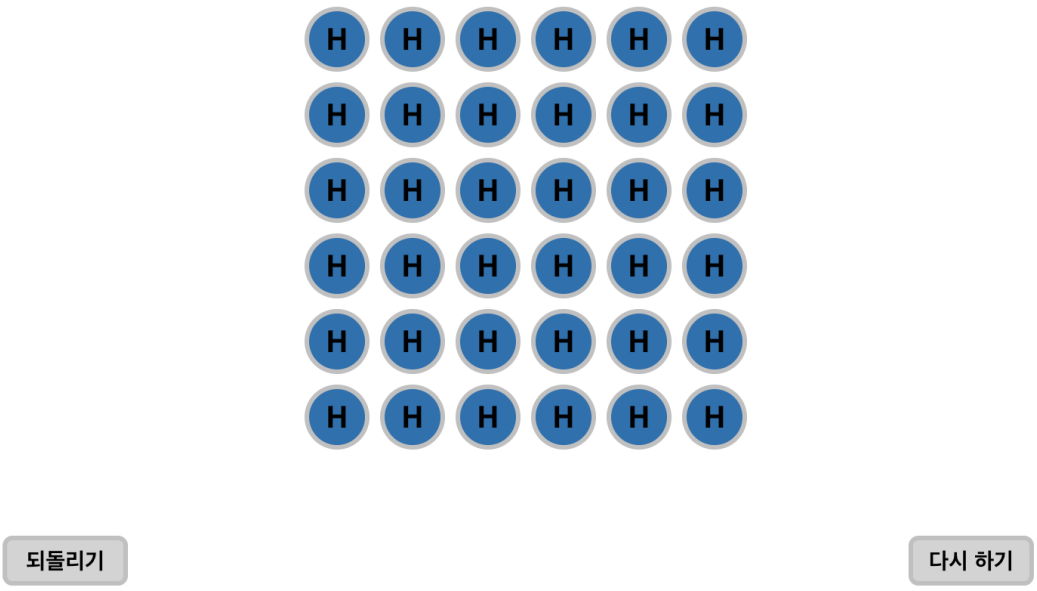
조작 방법

- 최대 100번까지 행동을 수행할 수 있다.
- "되돌리기" 버튼을 누르면 **가장 최근 행동**을 되돌리며, 행동 횟수도 함께 감소한다.
- "다시 하기" 버튼을 누르면 **초기 상태**로 돌아간다.

채점 기준

- 모든 동전이 앞면을 위로 향하도록 만들었지만 행동 횟수가 최소가 아니면 전체 점수의 30%를 받는다.
- 모든 동전이 앞면을 위로 향하도록 **최소 행동 횟수**로 만들었을 경우 전체 점수의 100%를 받는다.





i번째 줄의 j번째 동전을 뒤집는 행동을 (i, j)라고 표현하면, 아래와 같은 방법으로 16번의 행동으로 모든 동전이 앞면을 위로 향하도록 만들 수 있다.

- (1,1), (2,1), (2,2), (4,2), (5,2), (4,3), (6,3), (5,4), (1,4), (1,5), (3,5), (3,6), (4,6), (6,6), (6,4), (4,4)

이 방법 이외에도 16번의 행동으로 모든 동전이 앞면을 위로 향하도록 만들었다면 정답 처리된다.