SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

삼성 청년 SW 아카데미

APS기본

SAMSUNG SW ACADEMY FOR YOUTH

APS(Algorithm Problem Solving)

LIST 3

- 완전검색(Exhaustive Search)
- 그리디 (Greedy Algorithm)

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

완전 검색
(Exhaustive Search)

Baby-gin Game

설명

- 0~9 사이의 숫자 카드에서 임의의 카드 6장을 뽑았을 때, 3장의 카드가 연속적인 번호를 갖는 경우를 run이라 하고, 3장의 카드가 동일한 번호를 갖는 경우를 triplet이라고 한다.

- 그리고, 6장의 카드가 run과 triplet로만 구성된 경우를 baby-gin으로 부른다.

- 6자리의 숫자를 입력 받아 baby-gin 여부를 판단하는 프로그램을 작성하라.

Baby-gin Game

입력 예

- 667767은 두 개의 triplet이므로 baby-gin이다. (666, 777)
- 054060은 한 개의 run과 한 개의 triplet이므로 역시 baby-gin이다. (456, 000)
- 101123은 한 개의 triplet가 존재하나, 023이 run이 아니므로 baby-gin 이 아니다.
  (123을 run으로 사용하더라도 011이 run이나 triplet가 아님)

6자리의 숫자를 입력 받아 어떻게 Baby-gin 여부를 찾을 것인가?

완전 검색(Exhaustive Search)

- 완전 검색 방법은 문제의 해법으로 생각할 수 있는 모든 경우의 수를 나열해보고 확인하는 기법이다.
- Brute-force 혹은 Generate-and-Test 기법이라고도 불리 운다.
- 모든 경우의 수를 테스트한 후, 최종 해법을 도출한다.
- 일반적으로 경우의 수가 상대적으로 작을 때 유용하다.

완전 검색으로 시작하라

- 모든 경우의 수를 생성하고 테스트하기 때문에 수행 속도는 느리지만, 해답을 찾아내지 못할 확률이 작다.

- 자격검정평가 등에서 주어진 문제를 풀 때, 우선 완전 검색으로 접근하여 해답을 도출한 후, 성능 개선을 위해 다른 알고리즘을 사용하고 해답을 확인하는 것이 바람직하다.

완전 검색을 활용한 Baby-gin 접근

고려할 수 있는 모든 경우의 수 생성하기

- 6개의 숫자로 만들 수 있는 모든 숫자 나열 (중복 포함)
- 예) 입력으로 [2, 3, 5, 7, 7, 7]을 받았을 경우, 아래와 같이 순열을 생성할 수 있다.

```
2 3 5 7 7 7
2 3 7 5 7 7
2 3 7 7 5 7
   ...
7 7 7 5 3 2
```

<모든 경우의 순열 나열>

해답 테스트하기

- 앞의 3자리와 뒤의 3자리를 잘라, run와 triplet 여부를 테스트하고 최종적으로 baby-gin을 판단한다.
- 예)

순열을 어떻게 생성할 것인가

순열 (Permutation)

- 서로 다른 것들 중 몇 개를 뽑아서 한 줄로 나열하는 것
- 서로 다른 n개 중 r개를 택하는 순열은 아래와 같이 표현한다.

```
nPr
```

- 그리고 nPr은 다음과 같은 식이 성립한다.

```
nPr = n * (n-1) * (n-2) * ... * (n-r+1)
```

- nPn = n!이라고 표기하며 Factorial이라 부른다.

```
n! = n * (n-1) * (n-2) * ... * 2 * 1
```

단순하게 순열을 생성하는 방법

예) {1, 2, 3}을 포함하는 모든 순열을 생성하는 함수

- 동일한 숫자가 포함되지 않았을 때, 각 자리 수 별로 loop을 이용해 아래와 같이 구현할 수 있다.

```
for i1 from 1 to 3 {
    for i2 from 1 to 3 {
        if i2 != i1 {
            for i3 from 1 to 3 {
                if i3 != i1 and i3 != i2 {
                    print(i1, i2, i3)
                }
            }
        }
    }
}
```

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

탐욕 알고리즘
(Greedy Algorithm)

탐욕(Greedy) 알고리즘

- 탐욕 알고리즘은 최적해를 구하는 데 사용되는 근시안적인 방법

- 여러 경우 중 하나를 결정해야 할 때마다 그 순간에 최적이라고 생각되는 것을 선택해 나가는 방식으로 진행하여 최종적인 해답에 도달한다.

- 각 선택의 시점에서 이루어지는 결정은 지역적으로는 최적이지만,
  그 선택들을 계속 수집하여 최종적인 해답을 만들었다고 하여,
  그것이 최적이라는 보장은 없다.

- 일반적으로, 머릿속에 떠오르는 생각을 검증 없이 바로 구현하면 Greedy 접근이 된다.

탐욕 알고리즘의 동작 과정

- 1) 해 선택 : 현재 상태에서 부분 문제의 최적 해를 구한 뒤, 이를 부분해 집합(Solution Set)에 추가한다.

- 2) 실행 가능성 검사 : 새로운 부분해 집합이 실행 가능한지를 확인한다. 곧, 문제의 제약 조건을 위반하지 않는지를 검사한다.

- 3) 해 검사 : 새로운 부분해 집합이 문제의 해가 되는지를 확인한다. 아직 전체 문제의 해가 완성되지 않았다면 1)의 해 선택부터 다시 시작한다.

탐욕 알고리즘의 예

거스름돈 줄이기

- "어떻게 하면 손님에게 거스름돈으로 주는 지폐와 동전의 개수를 최소한으로 줄일 수 있을까?"

- 1) 해 선택 : 여기에서는 멀리 내다볼 것 없이 가장 좋은 해를 선택한다.
  단위가 큰 동전으로만 거스름돈을 만들면 동전의 개수가 줄어들므로 현재 고를 수 있는 가장 단위가 큰 동전을 하나 골라 거스름돈에 추가한다.

탐욕 알고리즘의 예

거스름돈 줄이기

- 2) 실행 가능성 검사 : 거스름돈이 손님에게 내드려야 할 액수를 초과하는지 확인한다. 초과한다면 마지막에 추가한 동전을 거스름돈에서 빼고, 1)로 돌아가서 현재보다 한 단계 작은 단위의 동전을 추가한다.

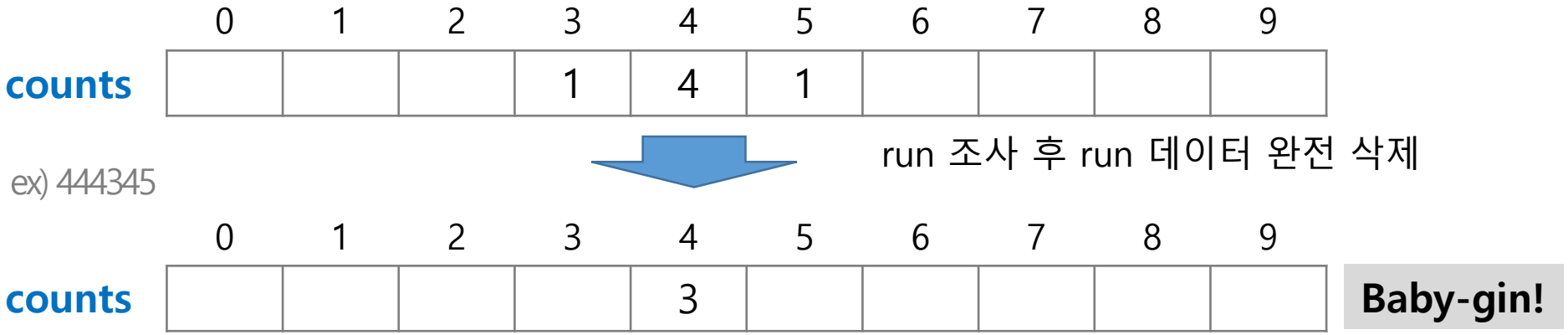
- 3) 해 검사 : 거스름돈 문제의 해는 당연히 거스름돈이 손님에게 내드려야 하는 액수와 일치하는 셈이다. 더 드려도, 덜 드려도 안되기 때문에 거스름돈을 확인해서 액수에 모자라면 다시 1)로 돌아가서 거스름돈에 추가할 동전을 고른다.

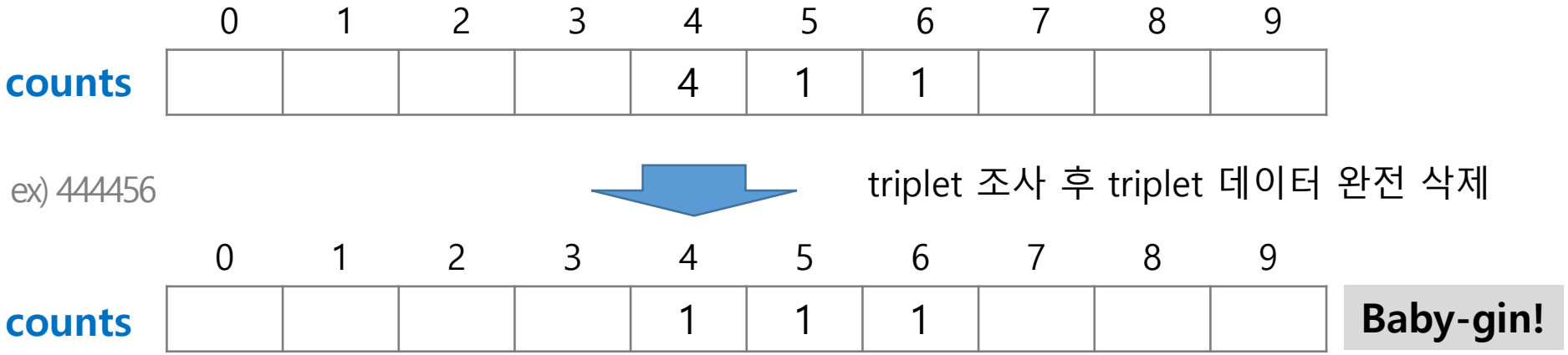
탐욕 알고리즘 풀이

- **Baby-gin을 완전검색 아닌 방법으로 풀어보자.**
  - 6개의 숫자는 6자리의 정수 값으로 입력된다.
  - Counts 배열의 각 원소를 체크하여 run과 triplet 및 baby-gin 여부를 판단한다.

탐욕 알고리즘의 예

풀이





자주 실수하는 오답

- 입력 받은 숫자를 정렬한 후, 앞뒤 3자리씩 끊어서 run 및 triplet을 확인하는 방법을 고려할 수도 있다.
  - 예) [6, 4, 4, 5, 4, 4]
    - 정렬하여 [4, 4, 4, 4, 5, 6]을 얻어내면 쉽게 baby-gin을 확인할 수 있다.
  - 예) [1, 2, 3, 1, 2, 3]
    - 정렬하면 [1, 1, 2, 2, 3, 3]로서, 오히려 baby-gin 확인을 실패할 수 있다.

- 위의 예처럼, 탐욕 알고리즘적인 접근은 해답을 찾아내지 못하는 경우도 있으니 유의해야 한다.

SAMSUNG SW ACADEMY FOR YOUTH

함께가요 미래로!
Enabling People

다음 방송에서 만나요!

삼성 청년 SW 아카데미