파이썬 코딩테스트 교재 1권 (입문)

일주일 개념 완성 + 바로 실전문제

구성: 개념 요약 → 예제 코드 → 연습문제 → 정답/해설

권장 학습 루틴(7일)

- 1–2일차: 입출력/자료형/조건문/반복문
- 3일차: 문자열/리스트/딕셔너리/정렬
- 4일차: 함수/재귀/디버깅
- 5일차: 코테 필수 패턴 맛보기
- 6–7일차: 연습문제 40선 + 해설 복기

목차

1. 코딩테스트 입출력 기본기

코테에서 가장 많이 터지는 부분은 알고리즘이 아니라 **입출력/형변환/공백 처리**입니다.

1.1 빠른 입력

입력이 많으면 sys.stdin.readline()을 사용합니다.

```
import sys
input = sys.stdin.readline

n = int(input().strip())
a, b = map(int, input().split())
arr = list(map(int, input().split()))
```

1.2 출력 포맷

print의 end, sep은 출력 포맷을 만들 때 유용합니다.

```
print(a, b, c)           # 기본: 공백 구분
print(a, b, c, sep=', ')  # 구분자 변경

for x in arr:
    print(x, end=' ')
print()                  # 마지막 줄바꿈
```

1.3 자주 쓰는 파싱 패턴

- 한 줄 문자열: s = input().rstrip()
- 공백 구분 정수들: list(map(int, input().split()))
- 여러 줄 n개: [int(input()) for _ in range(n)]
- 2차원 격자(정수): [list(map(int, input().split())) for _ in range(n)]
- 2차원 격자(문자): [list(input().rstrip()) for _ in range(n)]

2. 파이썬 자료형/연산

2.1 정수/실수/문자열

코테는 대부분 정수 연산입니다. 문자열은 슬라이싱/메서드를 자주 씁니다.

```
x = 10
y = -3
s = "hello"
print(s[0], s[-1])   # h o
print(s[1:4])        # ell
```

2.2 리스트/슬라이싱

슬라이싱은 새 리스트를 만든다는 점만 기억하면 됩니다.

```
arr = [1,2,3,4,5]
print(arr[::-1])     # 뒤집기
print(arr[1:4])      # [2,3,4]
arr.append(6)
arr.pop()
```

2.3 연산자 핵심

- //: 정수 나눗셈, %: 나머지
- **: 거듭제곱 (2**10 = 1024)
- and/or: 단축 평가를 함
- in: 포함 검사 (set/dict는 평균 O(1))

3. 조건문/반복문 패턴

3.1 조건문

경계값(=, 〈, 〉)을 정확히 처리하세요.

```
if x >= 0:
    print("non-negative")
else:
    print("negative")
```

3.2 반복문

for는 range, while은 조건. 코테는 for가 실수 적습니다.

```
for i in range(n):        # 0..n-1
    pass

for i in range(1, n+1):   # 1..n
    pass

for i in range(n-1, -1, -1):  # 역순
    pass
```

3.3 패턴 4종

- 누적 합: total += x
- 최댓값/최솟값 갱신: best = max(best, x)
- 카운팅: cnt += (조건)
- 중단/건너뛰기: break / continue

4. 문자열 다루기

문자열은 불변입니다. 바꾸는 연산은 새 문자열을 만듭니다.

4.1 메서드 TOP

- strip/rstrip: 공백 제거(입력 처리 필수)
- lower/upper: 대소문자 통일
- find: 위치(없으면 -1), count: 개수
- replace: 치환
- split/join: 분리/결합 (코테에서 매우 자주 사용)

```
s = "  AbC  "
print(s.strip().lower())    # abc

parts = ["A", "B", "C"]
print("-".join(parts))      # A-B-C
```

4.2 파싱 예시

```
line = "10,20,30"
nums = list(map(int, line.split(',')))
print(nums)
```

5. 리스트/튜플/집합/딕셔너리

5.1 리스트 vs 튜플

```
a = [1,2,3]
t = (1,2,3)
x, y = 5, 7
```

5.2 집합(set)

중복 제거/포함 검사에 강합니다.

```
s = set([1,1,2,3])
print(sorted(s))  # [1,2,3]
```

5.3 딕셔너리(dict) 카운팅

collections.defaultdict 또는 Counter를 쓰면 편합니다.

```
from collections import Counter
cnt = Counter("aabccc")
print(cnt["c"])  # 3
```

6. 함수와 재귀 기초

6.1 함수

함수로 분리하면 테스트/재사용이 쉬워집니다.

```
def clamp(x, lo, hi):
    if x < lo: return lo
    if x > hi: return hi
    return x
```

6.2 재귀

종료 조건 + 한 단계 줄이기.

```
def gcd(a, b):
    if b == 0:
        return a
    return gcd(b, a % b)
```

깊은 재귀는 반복문으로 바꾸는 편이 안전합니다.

7. 정렬과 key 사용

7.1 sorted vs sort

```
arr = [5,2,9,1]
print(sorted(arr))      # 새 리스트
arr.sort(reverse=True)  # 제자리
print(arr)
```

7.2 key 정렬

2차 기준 정렬의 정석은 key=lambda (튜플) 입니다.

```
items = [("kim", 90), ("lee", 75), ("park", 90)]
items.sort(key=lambda x: (-x[1], x[0]))
print(items)
```

8. 시간복잡도 감각(초간단)

아래 표는 '감'을 잡기 위한 최소 단위입니다. 정확한 정답이 아니라, 방향을 잡는 용도입니다.

입력 크기 n	대략 가능한 복잡도	비고
n <= 10^2	O(n^3)도 가능	삼중 루프 가능
n <= 10^5	O(n log n), O(n)	정렬/투포인터
n <= 10^6	O(n)	한 번 훑기

9. 연습문제 40선

입력/출력은 자유롭게 구성되어 있을 수 있으니, 문제에서 요구하는 형식을 그대로 구현하는 연습을 하세요.

〈b〉A01〈/b〉 두 정수 a,b가 주어진다. a+b, a-b, a*b를 한 줄에 공백으로 출력하라.

〈b〉A02〈/b〉 정수 n이 주어진다. 1부터 n까지의 합을 출력하라.

〈b〉A03〈/b〉 문자열 s(길이〈=100)가 주어진다. 공백을 모두 제거하여 출력하라.

〈b〉A04〈/b〉 문자열 s에서 'C' 또는 'c'의 개수와, 'cc'가 연속으로 등장하는 횟수를 출력하라.

〈b〉A05〈/b〉 정수 n이 주어진다. n의 각 자리수 합을 출력하라.

〈b〉A06〈/b〉 n개 정수가 주어진다. 최댓값과 최솟값을 출력하라.

〈b〉A07〈/b〉 n개 정수가 주어진다. 짝수만 골라 합을 출력하라.

〈b〉A08〈/b〉 한 단어(길이〈=20)가 주어진다. 뒤집어서 출력하라.

〈b〉A09〈/b〉 n개 정수가 주어진다. 오름차순 정렬하여 출력하라.

〈b〉A10〈/b〉 문자열 a,b가 주어진다. a+b(이어붙이기)를 출력하라.

〈b〉A11〈/b〉 정수 n이 주어질 때, 1..n 중 3의 배수의 개수를 출력하라.

〈b〉A12〈/b〉 정수 n이 주어진다. n이 소수이면 YES 아니면 NO를 출력하라.

〈b〉A13〈/b〉 n개의 정수가 주어진다. 중복을 제거하고 오름차순으로 출력하라.

〈b〉A14〈/b〉 문자열 s가 주어진다. 모음(a,e,i,o,u) 개수를 출력하라(대소문자 무시).

〈b〉A15〈/b〉 n과 k가 주어진다. nCk(조합)을 출력하라(파이썬 int 사용).

〈b〉A16〈/b〉 정수 n이 주어진다. 피보나치 n번째 수(0,1 시작)를 출력하라.

〈b〉A17〈/b〉 n개의 정수가 주어진다. 연속 부분합의 최댓값을 출력하라.

〈b〉A18〈/b〉 문자열 s가 주어진다. 괄호 문자열이 올바르면 YES 아니면 NO.

〈b〉A19〈/b〉 n개의 정수가 주어진다. 최빈값을 출력하라(동률이면 작은 값).

〈b〉A20〈/b〉 n개 정수와 질의 q개가 주어진다. 각 질의 [l,r] 구간합을 출력하라(누적합).

〈b〉A21〈/b〉 정수 n이 주어질 때, 1..n 중 3의 배수의 개수를 출력하라.

〈b〉A22〈/b〉 정수 n이 주어진다. n이 소수이면 YES 아니면 NO를 출력하라.

〈b〉A23〈/b〉 n개의 정수가 주어진다. 중복을 제거하고 오름차순으로 출력하라.

〈b〉A24〈/b〉 문자열 s가 주어진다. 모음(a,e,i,o,u) 개수를 출력하라(대소문자 무시).

〈b〉A25〈/b〉 n과 k가 주어진다. nCk(조합)을 출력하라(파이썬 int 사용).

〈b〉A26〈/b〉 정수 n이 주어진다. 피보나치 n번째 수(0,1 시작)를 출력하라.

〈b〉A27〈/b〉 n개의 정수가 주어진다. 연속 부분합의 최댓값을 출력하라.

〈b〉A28〈/b〉 문자열 s가 주어진다. 괄호 문자열이 올바르면 YES 아니면 NO.

〈b〉A29〈/b〉 n개의 정수가 주어진다. 최빈값을 출력하라(동률이면 작은 값).

〈b〉A30〈/b〉 n개 정수와 질의 q개가 주어진다. 각 질의 [l,r] 구간합을 출력하라(누적합).

〈b〉A31〈/b〉 정수 n이 주어질 때, 1..n 중 3의 배수의 개수를 출력하라.

〈b〉A32〈/b〉 정수 n이 주어진다. n이 소수이면 YES 아니면 NO를 출력하라.

〈b〉A33〈/b〉 n개의 정수가 주어진다. 중복을 제거하고 오름차순으로 출력하라.

〈b〉A34〈/b〉 문자열 s가 주어진다. 모음(a,e,i,o,u) 개수를 출력하라(대소문자 무시).

〈b〉A35〈/b〉 n과 k가 주어진다. nCk(조합)을 출력하라(파이썬 int 사용).

〈b〉A36〈/b〉 정수 n이 주어진다. 피보나치 n번째 수(0,1 시작)를 출력하라.

〈b〉A37〈/b〉 n개의 정수가 주어진다. 연속 부분합의 최댓값을 출력하라.

〈b〉A38〈/b〉 문자열 s가 주어진다. 괄호 문자열이 올바르면 YES 아니면 NO.

〈b〉A39〈/b〉 n개의 정수가 주어진다. 최빈값을 출력하라(동률이면 작은 값).

〈b〉A40〈/b〉 n개 정수와 질의 q개가 주어진다. 각 질의 [l,r] 구간합을 출력하라(누적합).

10. 정답/해설

아래 해설은 '핵심 아이디어 + 참고 코드' 형태로 제공합니다. 정답 코드를 그대로 베끼기보다, 아이디어를 이해하고 다시 구현해보세요.

A01~A03

입출력/문자열 처리 기본

```
# A01
a, b = map(int, input().split())
print(a+b, a-b, a*b)

# A02
n = int(input())
print(n*(n+1)//2)

# A03
s = input().rstrip('\n')
print(s.replace(' ', ''))
```

A04

카운팅 + 연속 등장 체크

```
s = input().rstrip()
t = s.lower()
cnt_c = sum(1 for ch in t if ch == 'c')
cnt_cc = 0
for i in range(len(t)-1):
    if t[i] == 'c' and t[i+1] == 'c':
        cnt_cc += 1
print(cnt_c)
print(cnt_cc)
```

A05

각 자리수 합

```
n = int(input())
print(sum(map(int, str(abs(n)))))  # 음수도 대응
```

A06~A09

리스트 기본 + 정렬

```
n = int(input())
arr = list(map(int, input().split()))
print(max(arr), min(arr))

# 정렬 출력 (A09)
arr.sort()
print(*arr)
```

A17

연속 부분합 최댓값(카데인)

```
n = int(input())
```

```
arr = list(map(int, input().split()))
best = cur = arr[0]
for x in arr[1:]:
    cur = max(x, cur + x)
    best = max(best, cur)
print(best)
```

A18

올바른 괄호(스택)

```
s = input().rstrip()
stack = []
ok = True
for ch in s:
    if ch == '(':
        stack.append(ch)
    else:
        if not stack:
            ok = False
            break
        stack.pop()
print("YES" if ok and not stack else "NO")
```

A20(구간합)

누적합으로 질의 처리

```
n, q = map(int, input().split())
arr = list(map(int, input().split()))
ps = [0]*(n+1)
for i in range(n):
    ps[i+1] = ps[i] + arr[i]

for _ in range(q):
    l, r = map(int, input().split())  # 1-indexed 가정
    print(ps[r] - ps[l-1])
```

나머지 문제들도 비슷한 패턴(카운팅/정렬/누적합/스택)으로 풀이할 수 있습니다. 2권에서 패턴을 본격적으로 다룹니다.