

PS 기초와 C++ 기초

190326(화) - 전현승

목차

- PS 기초
 - 문제 분석
 - 채점 결과
 - 시간복잡도 분석
 - 공간복잡도 분석
- C++ 기초
 - C++ 입출력 기초
 - C++ STL 기초 (자료구조 제외)



문제 분석

A+B ①

시간 제한	메모리 제한	제출	정답	맞은 사람	정답 비율
2 초	128 MB	132537	59797	44927	46.800%

문제 ③
두 정수 A와 B를 입력받은 다음, A+B를 출력하는 프로그램을 작성하시오.

입력 ④
첫째 줄에 A와 B가 주어진다. ($0 < A, B < 10$)

출력 ⑤
첫째 줄에 A+B를 출력한다.

예제 입력 1 복사 **예제 출력 1 복사** ⑥

```
1 2
3
```

힌트
여기를 누르면 1000번 예제 소스를 볼 수 있습니다.

① 낚시 또는 힌트


② 시간 메모리 정답자

③ 문제 잘읽기

④ 개수, 범위 주의

⑤ 형식 맞추기

⑥ 반드시 테스트 해보고 제출



문제 분석

- 제목 : 거의 도움 안됨
- 시간 제한, 메모리 제한 : 중요
- 정답자, 정답률 (대회 해당X) : 문제 선택의 지표

- 문제 : 잘 읽자
- 범위 (N제한) : 중요
- 입/출력 형식 : 반드시 문제에서 주어진 대로 입력받고, 출력
- 예제 : 해보고 제출하자



채점 결과

- AC – Accepted – 맞았습니다!!
- WA – Wrong Answer – 틀렸습니다
- TLE – Time Limit Exceeded – 시간 초과
- MLE – Memory Limit Exceeded – 메모리 초과
- RTE – Runtime Error – 런타임 에러
 - 배열 인덱스 초과 / return값이 0 이외 등
- CE – Compilation Error – 컴파일 에러
 - 채점 언어(컴파일러) 잘못 선택 / 문법 오류 등
 - 에러 메시지 볼 수 있음



Big-O Notation

- 자구, 알고시간에 한번씩 하는 내용
- 정확한, 엄밀한 정의는 생략
- $O(1)$, $O(N \lg N)$, $O(|V|E)$ 등의 표기법
- PS/알고리즘에서는 시간복잡도, 공간복잡도를 나타낼 때 사용
- 대략 이 알고리즘이 어느 정도 걸린다/메모리를 어느 정도 쓴다
- 계수는 떼고, 차수 가장 큰 항만 남긴다 $\rightarrow O(4N^2 + N \lg N) = O(N^2)$
- 변수 여러 개일 때는 같이 쓴다 $\rightarrow O(N+M)$



시간복잡도 분석

- 각 알고리즘들의 시간복잡도를 알아두는 것은 매우 중요 (기본)
- 자주 쓰이는 시간복잡도
 - $O(1)$: 단순 계산 (+, *, % 등), 배열 접근 등
 - $O(\lg N)$: 이분 탐색, 분할 정복 등 반으로 나뉘는 개념
 - $O(N)$: 반복문 1개, 선형탐색 등
 - $O(N \lg N)$: 정렬 (머지소트, 퀵소트 등)
 - $O(N^2)$, $O(N^3)$: 2, 3중 반복문, 완전탐색 등
 - $O(2^N)$: 크기가 N 인 집합의 부분집합 개수
 - $O(N!)$: 길이 N 짜리 순열 채우기



시간복잡도 분석

- 대략 1초에 1억 번 연산이라고 생각하면 편하다
 - 아주 대략적 수치 (모든 연산은 속도가 다르다)
 - 그래도 알아두면 편한 건 분명하다
- 예) 길이 100,000 배열을 정렬? (시간제한 1초)
 - $O(N^2)$ 정렬 알고리즘 : 0(100억) → TLE
 - $O(N \lg N)$ 정렬 알고리즘 : 0(1천만) → AC



공간복잡도 분석

- $O(N)$: N 크기의 1차원 배열 사용
- MLE는 좀 드물긴 함

- 128MB \rightarrow 5000*5000 2차원 배열이 한계
- 생각 없이 10000*10000 배열 같은걸 하면 안 된다

- 큰 변수들은 전역변수로 빼는 게 유리
 - 전역변수는 Heap에, 지역변수는 Stack에 선언됨
 - 프로세스당 기본 Stack 사이즈는 제한이 있다 (Heap은 제한 X)
 - 따라서 큰 변수를 지역변수에 선언하면 터질 수 있음



C++ 기초

- 정말 진짜 기초
- 이 정도만 알아도 기초 문제 푸는 데는 지장 X
- 나중에 따로 자료구조/STL 주제로 한 주 진행 예정
 - stack, queue, ... + pair, tuple, ... + binary_search, lower_bound 등



C++ 입출력

```
#include <iostream>
using namespace std;

int main() {
    int a, b;
    cin >> a >> b;
    cout << a + b << '\n';

    string s;
    cin >> s;
    cout << "Input: " << s << '\n';

    return 0;
}
```



C++ 입출력

- 빠른 입출력 (안 해서 TLE 날 수도)
- 이렇게 하면 scanf/printf/getchar 등과 C++ 입출력 같이 사용 불가

```
cin.tie(NULL);  
cout.tie(NULL);  
ios_base::sync_with_stdio(false);
```

- 개행으로 endl은 사용 지양 (느림)
 - 'Wn' 씁시다
- 기존 C 입력은 #include <cstdio> 하면 사용 가능



C++ STL

- 아주 기본적인 벡터, 소트 사용법만
- `vector<typename>` : 가변 배열
- 주의 : `insert/erase` $O(N)$

```
#include <vector>
using namespace std;

vector<int> v;
v.push_back(10);
v.push_back(20);
v.size(); // 2

v.push_back(30);
v.size(); // 3
v[0]; // 10
v.empty(); // false
v.clear();
```



C++ STL

- `sort(v.begin(), v.end(), comp)`
 - $O(N \lg N)$
 - 그냥 N 크기 배열에 `sort(arr, arr + N)`도 가능
 - 비교함수 따로 작성 가능

```
#include <algorithm>
using namespace std;
```

```
vector<int> v = {4, 2, 5, 3, 1};
for (int i = 0; i < v.size(); i++)
    cout << v[i] << ' '; // 4 2 5 3 1
sort(v.begin(), v.end());
for (int i = 0; i < v.size(); i++)
    cout << v[i] << ' '; // 1 2 3 4 5
```



참고

- 빅오 표기법(Big-O notation), 시간복잡도, 공간복잡도
 - <http://kks227.blog.me/220769859177>
- C++ 기초
 - <https://baactree.tistory.com/28>



끝

