바이러스

운영체제의 디렉토리 시스템을 구현해보자.

함수	기능
void init()	프로그램이 시작 전 단한번만 호출 된다.
int add(int id, int pid, int fileSize);	파일 또는 디렉토리를 아이디가 pid인 디렉토리에 추가한다. 이 때 파일 또는 디렉토리의 아이디는 id가 된다. fileSize가 0이면 디렉토리, 0보다 크면 파일이다. 추가 후 아이디가 pid인 디렉토리의 용량을 반환해야 한다.
int move(int id, int pid)	아이디가 id인 파일 또는 디렉토리을 찾아서 아이디가 pid인 디렉토리에 연결한다. 디렉토리의 경우 하위 디렉토리 전부 유지되어야 한다. 이동 후 아이디가 pid인 디렉토리의 용량을 반환해야 한다.
int infect(int id)	아이디가 id인 파일 또는 디렉토리를 찾는다. 파일인 경우 용량이 증가하고 디렉토리인 경우 하위 파일들이 전부 증가한다. 증가하는 양은 현재 존재하는 모든 파일용량의 평균이다. (나머지는 버림) 현재 존재하는 파일의 개수가 0이면 증가하지 않는다. 수행 후 아이디가 id인 파일 또는 디렉토리의 용량을 반환해야 한다.
int recover(int id)	아이디가 id인 파일 또는 디렉토리를 찾는다. 파일인 경우 원래 용량으로 돌아온다. 디렉토리인 경우 하위 파일들이 전부 원래 용량으로 돌아온다. Infect를 여러 번 받은 파일도 바로 원래대로 돌아온다. 수행 후 아이디가 id이 파일 또는 디렉토리의 용량을 반환해야 한다.
Int remove(int id)	아이디가 id인 파일 또는 디렉토리를 찾는다. 파일인 경우 파일을 지운다. 디렉토리의 경우 하위 디렉토리와 하위 파일들도 지워지고 현재 주어진 디렉토리도 지워진다. 수행하기 전 아이디가 id인 파일 또는 디렉토리의 용량을 반환해야 한다.

좀 더 상세한 부분은 아래를 참조한다.

추가사항

디렉토리의 용량은 하위 파일들의 합이다.

명령이 모순 되어 주어지는 경우는 없다.

- 지워진 파일이 감염되는 경우
- 부모 디렉토리가 자식 디렉토리 밑으로 이동하는 경우
- 등등 ..

Root 디렉토리가 처음에 존재하는데 Root 디렉토리의 아이디는 1000이며 Root 디렉토리가 삭제되거나 이동되는 경우는 없다.

제약사항

- 테스트 케이스 10개 단위로 각각 명령이 총 <=50, 1000, 5000, 10000개 있다.
- Id는 10^9이하의 양의 정수
- fileSize는 10^3이하의 양의 정수 또는 0
- 총 파일 사이즈는 int 범위를 넘어가지 않는다.
- 디렉토리의 높이는 30을 넘지 않는다.
- 40개 총합 2초를 넘으면 안된다. (입출력 포함)

입출력 예시

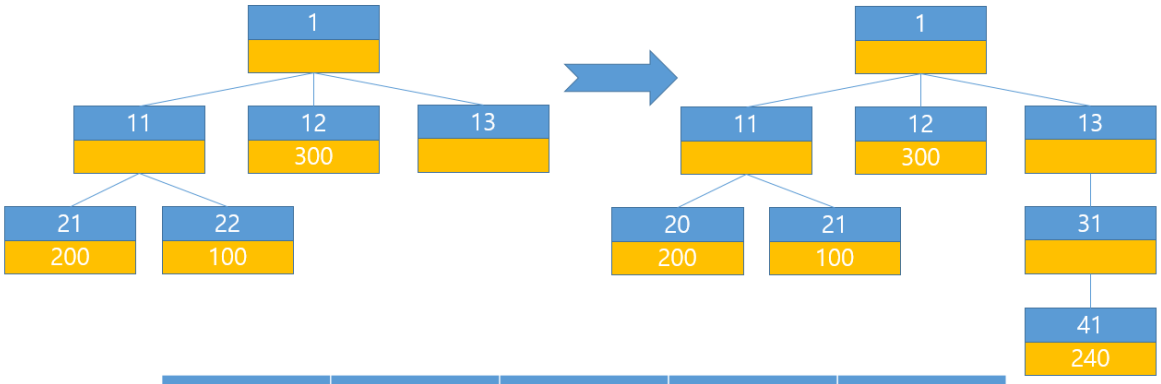
Input	Output
40 15 1 11000 1000 0 1 12000 1000 300 1 13000 1000 0 1 21000 11000 200 1 22000 11000 100 1 31000 13000 0 1 41000 31000 240 2 22000 31000 2 11000 13000 3 12000 3 13000 4 11000 4 22000 5 12000 5 11000	0 300 300 200 300 0 240 340 540 510 1326 200 100 510 200

Main code

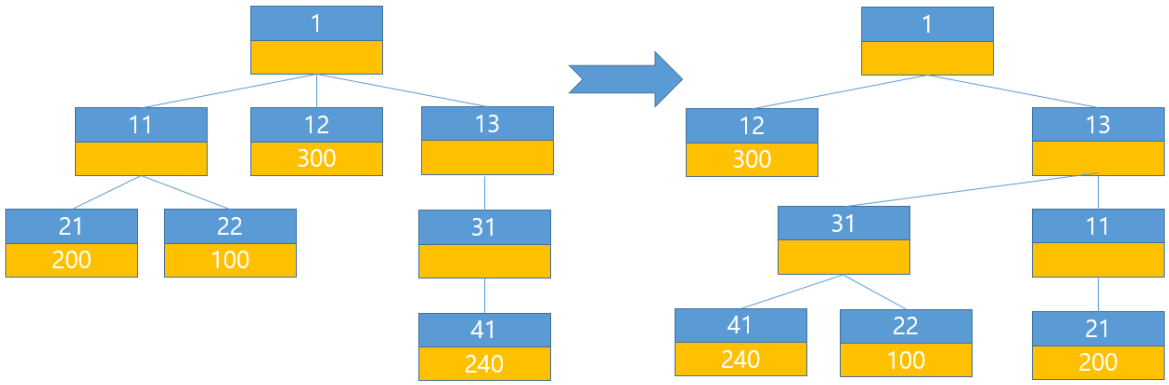
```
#include <cstdio>
using namespace std;

void init()
{

}

int add(int id, int pid, int fileSize)
{
    return 0;
}

int move(int id, int pid)
{
    return 0;
}

int infect(int id)
{
    return 0;
}

int recover(int id)
{
    return 0;
}

int remove(int id)
{
    return 0;
}

// do not touch
void do_test(int tc)
{
    int n; scanf("%d", &n);
    int cmd;
    int arg[3];

    init();
    while(n--)
    {
        scanf("%d", &cmd);
        if(cmd == 1)
        {
            scanf("%d %d %d", arg+0, arg+1, arg+2);
            printf("%d₩n", add(arg[0], arg[1], arg[2]));
        }
        else if(cmd == 2)
        {
            scanf("%d %d", arg+0, arg+1);
            printf("%d₩n", move(arg[0], arg[1]));
```

```
        }
        else if(cmd == 3)
        {
            scanf("%d", arg+0);
            printf("%dWn", infect(arg[0]));
        }
        else if(cmd == 4)
        {
            scanf("%d", arg+0);
            printf("%dWn", recover(arg[0]));
        }
        else if(cmd == 5)
        {
            scanf("%d", arg+0);
            printf("%dWn", remove(arg[0]));
        }
    }
}

int main()
{
    int tc; scanf("%d", &tc);
    for(int i=0; i<tc; i++)
    {
        do_test(i);
    }

    return 0;
}
```

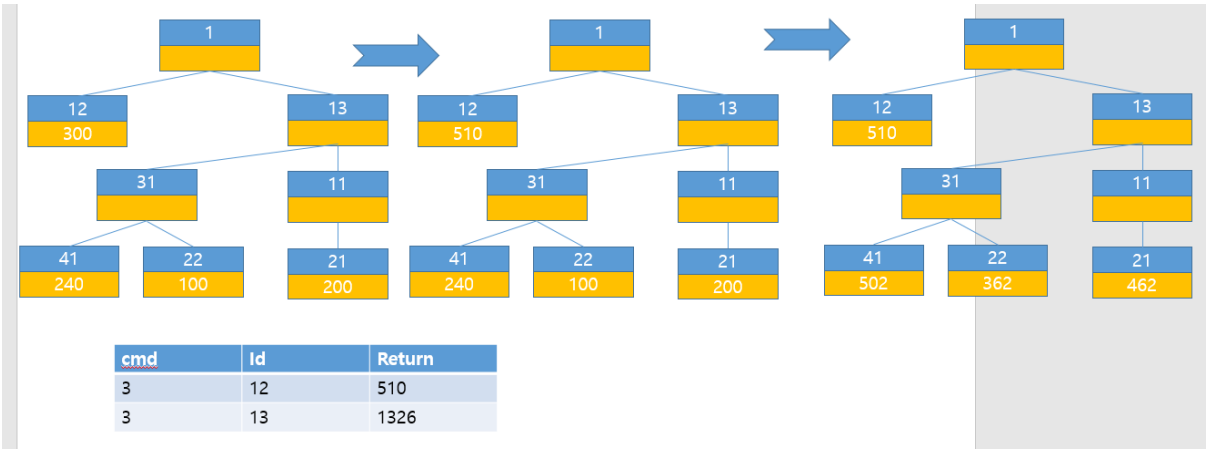
입출력 예제 참고 이미지



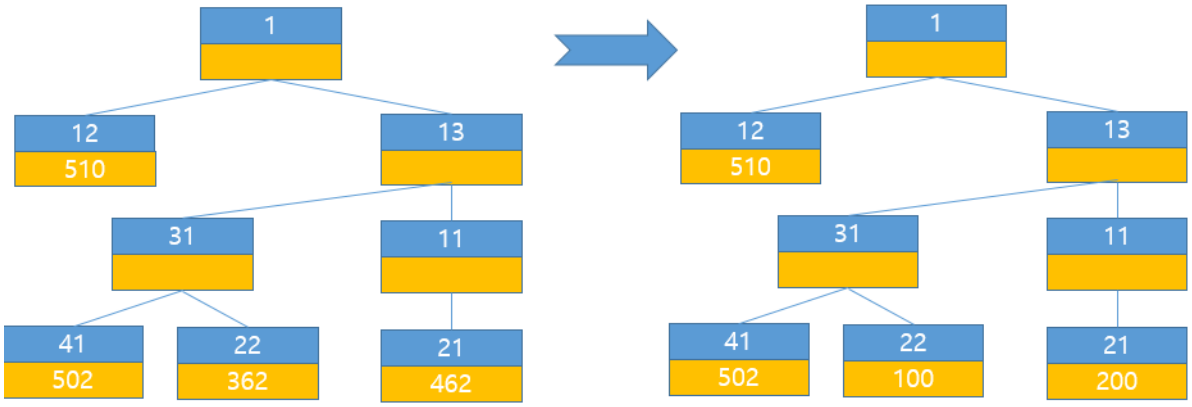
cmd	Id	Pid	Filesize	Return
1	31	13	0	0
1	41	31	240	240



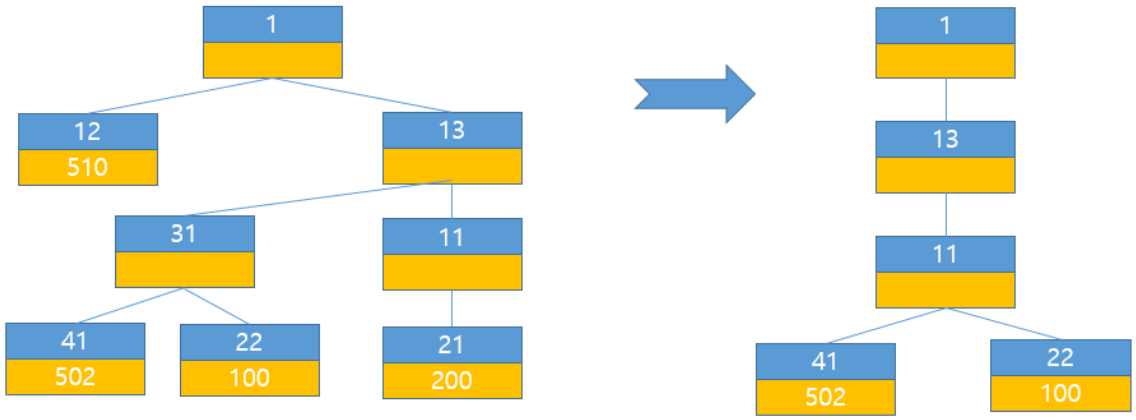
cmd	Id	Pid	Return
2	22	31	340
2	11	13	540



cmd	Id	Return
3	12	510
3	13	1326



cmd	Id	Return
4	11	200
4	22	100



cmd	id	Return
5	12	510
5	11	200