|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 8주차 - B 문제해결기법 보고서 | | | | | |
| 문제 | 아이스크림 케이크 | | | | |
| 분반 | 002 | 학번 | 12141540 | 이름 | 박영창 |
| 알고리즘 개요 | * 알림의 수 N를 입력한다 (0<T<=1000000) * D, O중 하나를 입력 받는다. * O를 입력 받았다면, templist 배열에 필요한 과일의 종류를 입력 받음과 동시에 해당 과일이 존재하는지 확인해 존재하지 않는다면 bool 변수 avail을 false로 변경한다. 그 다음, templist를 입력 받은 과일 개수만큼만 정렬하고, avail이 true라면 templist에 저장된 과일들을 1씩 감소시켜준다. 그리고 해당 주문번호인 orderlist[orderNum]을 -1로 설정하고 orderNum을 출력한다. Avail이 false라면 orderlist[orderNum]에 필요한 과일을 내림차순으로 모두 저장한다. (ex : 필요한 과일이 1 5 4 0 6이라면, 65410을 저장) avail을 확인하는 부분이 끝난 뒤에 orderNum을 1 증가시켜준다. * D를 입력 받았다면, 우선 입력 받은 과일의 개수를 fruit 배열에서 1 증가시켜준다. 그 다음 orderlist 배열을 모두 확인해, fruit 배열의 값이 변동됨으로써 판매가 가능해진 조합을 확인한다. 주문 순서 즉 인덱스 번호 순으로 탐색하다 가능한 값을 찾았을 때, 해당 orderlist[i]를 -1로 변경해주고, 그에 필요한 과일들의 개수를 1씩 감소시킨 뒤 i를 출력한다. * n를 입력 받는 것을 제외한 위 과정을 n-1번 반복한다. | | | | |
| 성능분석 | 배달(D)  주문의 개수 : N 해당 주문에서 필요한 과일의 개수 : M  시간 복잡도 : O(NM)  주문(O)  필요한 과일의 개수 : N  시간 복잡도 : O(N) | | | | |
| 실험결과 | 소요시간 : TIMELIMIT  사용된 메모리 공간 : 4.194MB | | | | |
| 비고 | 주문을 할 때는 해당 주문이 바로 처리될 수 있는 지만 확인하므로 오래 걸리지 않지만, 배달을 할 때는 주문 리스트 배열에 있는 모든 것을 확인해야 해서 오래 걸리는 것 때문에 문제를 해결하지 못했습니다… | | | | |