|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1주차 - B 문제해결기법 보고서 | | | | | |
| 문제 | 아르바이트 - 2 | | | | |
| 분반 | 002 | 학번 | 12141540 | 이름 | 박영창 |
| 알고리즘 개요 | * Testcase의 개수 T를 입력 받는다. (1<=T<=3000) * 물건의 가격 p, 동전의 종류 n을 차례대로 입력 받는다. (1<=p<=10000, 1<=n<=500) * 동전의 액면가 배열인 coin[500]에 coin[0] 부터 coin[n]까지 입력 받는다. * 손님이 10000원을 지불했을 때 거스름돈의 액수를 맞춰야 하므로, 물건의 가격 p를 이용하여 거스름돈을 구한다. => (p = 10000 – p) * 가격이 i일 때 필요한 가장 적게 드는 동전의 수가 들어있는 배열 dp[i]의 값을 모두 100000으로 초기화 시켜준다. (동전의 최소 개수를 구하는 코드이기 때문에 아무거나 큰 수로 초기화) * 동전은 오름차순으로 배열에 저장되어 있으므로, 액면가가 가장 작은 동전인 첫 번째 배열의 동전부터 아래의 식을 적용해 거스름돈 p의 동전 개수인 dp[p]까지 반복한다.   dp[i] = min(dp[i], dp[i-coin[j]] + 1)   * 위의 반복문을 통해 dp 배열에 첫번째 동전의 배수인 i값에 각각 dp/i 값이 들어가게 된다. (100000과 비교했을 때 (dp[i-coin[0]] +1) 을 계산한 값이 작은 수 이기 때문에) 같은 방법으로 coin[n]까지 반복하면, dp[p]에 거스름돈 p를 만들기 위한 동전의 최소 개수가 저장된다. * dp[p]가 100000 이상이면 0을 출력하고, 아니라면 dp[p]를 출력한다. * T를 입력 받는 것을 제외한 위 과정을 T-1번 반복한다. | | | | |
| 성능분석 | Dp 배열에서는 거스름돈 p까지, coin 배열에서는 동전의 종류 n까지를 사용해 반복문을 진행하므로, 시간 복잡도는 O(p\*n)이다. | | | | |
| 실험결과 | 소요시간 : 1.9s  사용된 메모리 공간 : 262.144KB | | | | |
| 비고 |  | | | | |