알고리즘

- HW07: Knapsack-

제 출 일	2017년 11월 15일
분 반	02반
담당교수	공은배
학 과	컴퓨터공학과
학 번	201302423
이 름	신종욱

1. 해결 방법

```
System.out.println("아이템 갯수와 최대 무게를 입력하세요");
Scanner scan = new Scanner(System.in);
int count = scan.nextInt();// 갯수저장하는 변수
int totalweight = scan.nextInt();// 총 가방크기를 저장
int[] value = new int[count + 1];// 가치와 무게를 저장
int[] weight = new int[count + 1];// 0번째 인덱스를 사용하면 계산시 복잡해서 사용하지않기로 했다.
System.out.println("아이템 갯수 : " + count + " 최대 무게 : " + totalweight);
System.out.println("데이터를 입력하세요");

ArrayList<Node> MaxValue = new ArrayList<>();// 가방에 들어간 가치가 가장 높은때
// 가치값과 OTPTable의 인덱스값을 저장하기위한 ArrayList

for (int i = 1; i < count + 1; i++) {
    value[i] = scan.nextInt();
    weight[i] = scan.nextInt();
} // 데이터를 입력받는다.
int[][] OPTtable = new int[count + 1][totalweight + 1];// 테이블생성
int[][][] WhatItem = new int[count + 1][totalweight + 1][count];// 각 테이블 마다 어떤 아이템이 선택되는지 알기위해 3차원배열로 선언
```

OPT테이블 작성하고 최대값을 추출하고 최대값일 때 어떤 아이템을 선택했는지 알아야한다.
그래서 OPT테이블을 위해 2차원 배열을 선언해야하고
최대값을 추출하기위해 ArrayList를 선언하여서 Node를 넣었는데 Node는 OPT테이블의 가치값과 인덱스를 저장한채 ArrayList에 더하는식으로 저장한다음
value로 정렬하면된다.

마지막으로 어떤 아이템을 선택했는지 알기 위해서는 OPTtable의 인덱스마다 어떤 아이템선택했는지 저장해야한다 그래서 3차원 배열을 선언하여서 저장하였다.

OPT테이블 작성과 List추가 아이템정보 추가하는 코드

```
for (int i = 1; i < count + 1; i++) {// 테이블과 최대값을
    int j;
    for (j = 0; j < totalweight + 1; j++) {
        if (weight[i] > j) {//일단 무게가 안될경우에는
            OPTtable[i][j] = OPTtable[i - 1][j];//이전 OPT값을 저장
            MaxValue.add(new Node(OPTtable[i][j], i, j));//여기서나온 인덱스와 값을 리스트에 저장
                WhatItem[i][j] = WhatItem[i - 1][j];
            //3번째 배열값들을 전부 복사
        } else if (OPTtable[i - 1][j] > value[i] + OPTtable[i - 1][j - weight[i]]) {
            OPTtable[i][j] = OPTtable[i - 1][j];
            MaxValue.add(new Node(OPTtable[i][j], i, j));
            WhatItem[i][j] = WhatItem[i - 1][j];
            //3번째 배열값들을 전부 복사
        } else {
            OPTtable[i][j] = value[i] + OPTtable[i - 1][j - weight[i]];
            MaxValue.add(new Node(OPTtable[i][j], i, j));
            int k = 0;
            WhatItem[i][j][k] = i;//i의 밸류의 값을 썼기때문에 i는 무조건이다
            for (k = 0; k < count - 1; k++) {
                WhatItem[i][j][k + 1] = WhatItem[i - 1][j - weight[i]][k];
            }//그이후는 Item은  그이전의 i-1의 OPT의 Item을 참고하면된다.
        }

    }

}
```

OPT테이블은 주어진 알고리즘으로 구현하였다.
구과정에서 가치와 인덱스를 저장해주고 새로운 아이템을 택할땐 3번째 경우
즉 Value[i]+OPTtable[i-1][j-weight[i]]일 때 인데 이때는 새롭게 아이템선택되는걸 선언해주어야한다.
새롭게 선언안 아이템 I와 나머지는 그이전 값들을 이용하여서 선언하였다.
그 외에는 i-1의 값을 그대로 들고와서 쓰면된다.

2. 실행 결과

```
<terminated> Knapsack [Java Application] C:₩Program Files₩Java₩jre1.8.0_144₩bin₩javaw.exe (2017. 11. 15. 오후 9:03:06)
아이템 갯수와 최대 무게를 입력하세요
5 11
아이템 갯수 : 5 최대 무게 : 11
데이터를 입력하세요
1 1
6 2
18 5
22 6
28 7
   0   0   0   0   0   0   0   0   0   0   0   0
   0   1   1   1   1   1   1   1   1   1   1   1
   0   1   6   7   7   7   7   7   7   7   7   7
   0   1   6   7   7  18  19  24  25  25  25  25
   0   1   6   7   7  18  22  24  28  29  29  40
   0   1   6   7   7  18  22  28  29  34  35  40
Max : 40
Item : 4 3
```

```
<terminated> Knapsack [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (2017. 11. 15. 오후 9:04:05)
아이템 갯수와 최대 무게를 입력하세요
8 104
아이템 갯수 : 8 최대 무게 : 104
데이터를 입력하세요
350 25
400 35
450 45
20 5
70 25
8 3
5 2
5 2
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20
   0   0   0   0   0  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20  20
   0   0   0   8   8  20  20  20  28  28  28  28  28  28  28  28  28  28  28  28  28  28  28  28
   0   0   5   8   8  20  20  25  28  28  33  33  33  33  33  33  33  33  33  33  33  33  33  33
   0   0   5   8  10  20  20  25  28  30  33  33  38  38  38  38  38  38  38  38  38  38  38  38
Max : 900
Item : 8 7 5 4 3 1
```

```
<terminated> Knapsack [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (2017. 11. 15. 오후 8:56:16)
아이템 갯수와 최대 무게를 입력하세요
10 165
아이템 갯수 : 10 최대 무게 : 165
데이터를 입력하세요
92 23
57 31
49 29
68 44
60 53
43 38
67 63
84 85
87 89
72 82
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  92   9
Max : 309
Item : 6 4 3 2 1
```

3. 느낀 점

테이블을 작성하는거는 어렵지 않았는데 어떤 아이템을 선택하였는지 저장하는게 어려웠다. 3차원 배열로 일단 단순하게 다 저장하는 식으로 했는데 더쉬운 방법이 있을까 궁금하다.