

0-1 Knapsack Problem

A complex network diagram with numerous nodes and edges, rendered in light blue and grey, serves as a background for the slide. The nodes are connected by thin lines, creating a web-like structure that spans the entire page.

Seoul National University of Science and Technology
Computer Science and Engineering

실습 예제 – 0-1 Knapsack Problem

배낭문제

Q. N개의 물건이 있고, 각 물건의 무게와 가치가 각각 다를 때, 정해진 무게만큼만 담을 수 있는 가방에 어떤 물건들을 담아야 이익을 최대화할 수 있는가?

(단, 물건을 쪼개거나 나눌 수 없음)

Input:

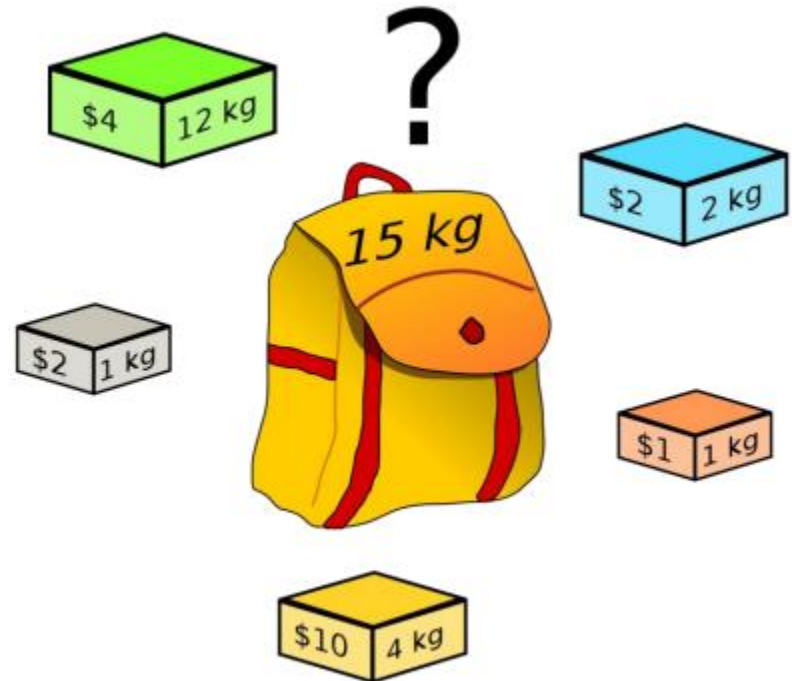
물건의 수량 (Objects)

가방의 최대 무게 (Capacity)

물건의 무게와 가치에 대한 리스트
(Value, Weight)

Output:

가방에 담긴 물건들의 가치의 총 합




실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

 가방의 용량이 처음엔 0이라고 가정하고 점차 늘려나감

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|---|---|---|---|---|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | | | | | | | | | | | |
| 2 | 40 | 4 | | | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

 처음엔 물건이 하나만 있다고 가정하며, 점차 고려할 물건의 수를 늘려나감

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별



가방의 용량이 0인 경우 → 담을 수 있는 물건이 없음 → 가방에 담긴 물건들의 가치의 총 합이 0

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|---|---|---|---|---|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | | | | | | | | | | |
| 2 | 40 | 4 | | | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

배낭의 용량이 1번 물건의 무게인 5가 되는 시점부터 연산 시작



| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|---|---|---|---|---|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | | | | | | |
| 2 | 40 | 4 | | | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

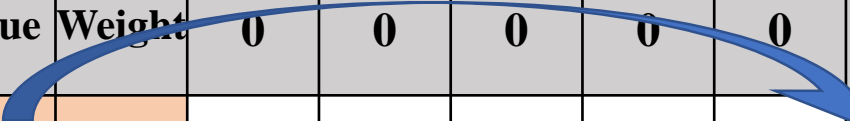
실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|----|---|---|---|---|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | | | | | |
| 2 | 40 | 4 | | | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |



실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | | | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|---|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | | | | | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|---|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 기존 배낭 상태에서의 가치의 합 = 새로 고려할 물건을 넣지 않은 경우의 가치의 합 | | | | | |
| 3 | 30 | 6 | 새로운 물건을 고려한 경우의 가치의 합 = 기존 배낭 상태 중 새 물건의 무게를 넣을 수 있을 만큼 여유가 있는 상태에서 새로운 물건을 넣었을 때의 가치의 합 | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | | | | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

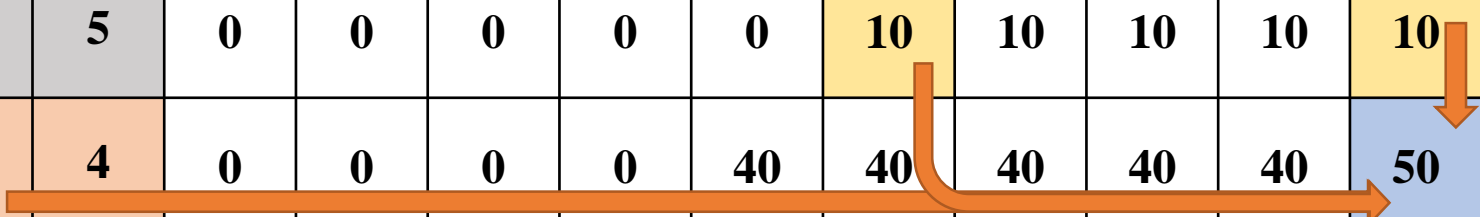
실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |



실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 50 |
| 3 | 30 | 6 | | | | | | | | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 50 |
| 3 | 30 | 6 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | | | | |
| 4 | 50 | 3 | | | | | | | | | | | |

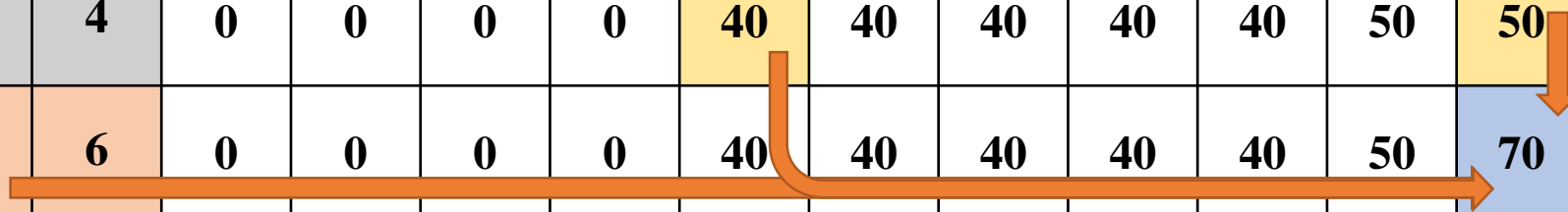
실습 예제 – 0-1 Knapsack Problem

배낭문제

고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|---|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 50 |
| 3 | 30 | 6 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 70 |
| 4 | 50 | 3 | | | | | | | | | | | |



실습 예제 – 0-1 Knapsack Problem

배낭문제

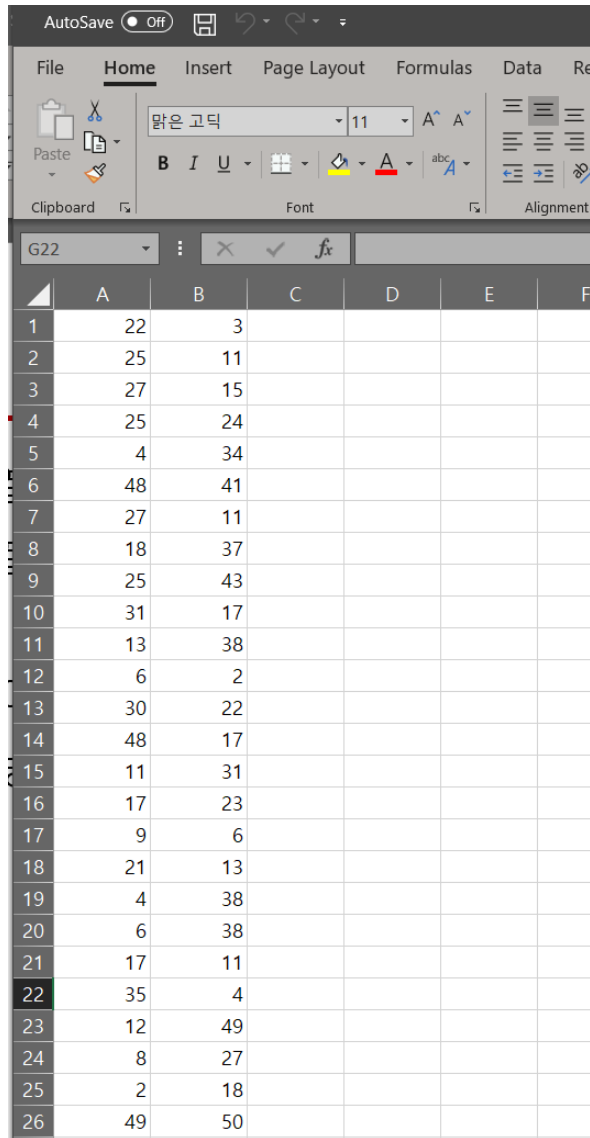
고려하는 물건과 가방의 용량을 점차 늘려가면서 이익을 계산

새로운 물건을 고려하는 경우, 특정 시점의 가방 용량 상황에서 해당 **물건을 넣는 것이 이득인지 아닌지**를 판별

| Capacity | | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|-------|--------|---|---|---|----|----|----|----|----|----|----|----|
| Object | Value | Weight | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 10 | 5 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 10 | 10 | 10 | 10 |
| 2 | 40 | 4 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 50 |
| 3 | 30 | 6 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 50 | 70 |
| 4 | 50 | 3 | 0 | 0 | 0 | 50 | 50 | 50 | 50 | 90 | 90 | 90 | 90 |

실습 방법 – 코드 샘플

quiz_4.csv



| | A | B | C | D | E | F |
|----|----|----|---|---|---|---|
| 1 | 22 | 3 | | | | |
| 2 | 25 | 11 | | | | |
| 3 | 27 | 15 | | | | |
| 4 | 25 | 24 | | | | |
| 5 | 4 | 34 | | | | |
| 6 | 48 | 41 | | | | |
| 7 | 27 | 11 | | | | |
| 8 | 18 | 37 | | | | |
| 9 | 25 | 43 | | | | |
| 10 | 31 | 17 | | | | |
| 11 | 13 | 38 | | | | |
| 12 | 6 | 2 | | | | |
| 13 | 30 | 22 | | | | |
| 14 | 48 | 17 | | | | |
| 15 | 11 | 31 | | | | |
| 16 | 17 | 23 | | | | |
| 17 | 9 | 6 | | | | |
| 18 | 21 | 13 | | | | |
| 19 | 4 | 38 | | | | |
| 20 | 6 | 38 | | | | |
| 21 | 17 | 11 | | | | |
| 22 | 35 | 4 | | | | |
| 23 | 12 | 49 | | | | |
| 24 | 8 | 27 | | | | |
| 25 | 2 | 18 | | | | |
| 26 | 49 | 50 | | | | |

A열 : 물건의 가치 (1 ~ 50)

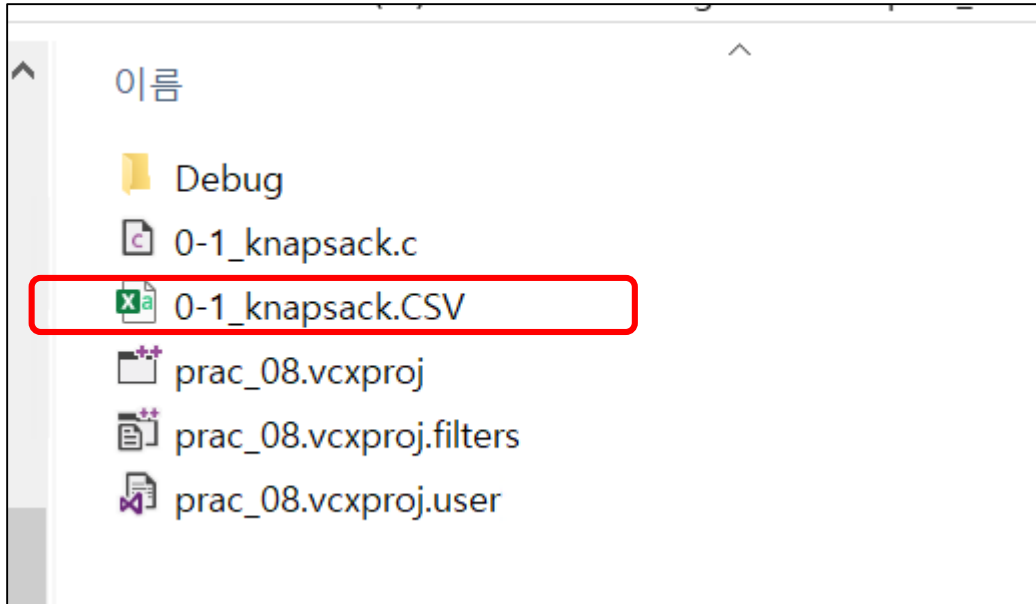
B열 : 물건의 무게 (1 ~ 50)

데이터의 개수 – 200

가방에 담을 수 있는 무게의 합 – 1,000

실습 방법 – 코드 샘플

Microsoft Visual Studio (2017)



0-1_knapsack.csv를 프로젝트 폴더로 이동

※.c 파일 이름은 학번으로 할 것 (예: 18520028.c or 18520028.cpp)

※ eclass 업로드 시 프로젝트 폴더를 압축해서 올리지 말고 .c(또는 .cpp) 파일만 업로드할 것

파일 입출력 설명

data.csv

C 0-1_knapsack.c x

```
1  #define _CRT_SECURE_NO_WARNINGS
2
3  #include <stdio.h>
4  #include <string.h>
5  #include <stdlib.h>
6
7  #define DATA_SIZE 1000
8  #define CAPACITY 10000
9  #define COLUMN_NUM 2
10
11 void read_data(int(*quiz)[COLUMN_NUM], const char* fileDir);
12
13 int main()
14 {
15     char* fileDir = "./0-1_knapsack.csv";
16     int data[DATA_SIZE][COLUMN_NUM];
17     int total_value = 0;
18
19     read_data(data, fileDir); // CSV data file read
20
21     // write your code here
22
23     printf("Total value: %d\n", total_value); // result format
24
25 }
26
27 void read_data(int(*quiz)[COLUMN_NUM], const char* fileDir)
```

전역변수 설명

DATA_SIZE: 데이터(물건)의 수

CAPACITY: 가방의 수용능력(무게)

COLUMN: 데이터의 컬럼 수 (2)

출력변수

total_value:

가방에 담긴 물건들의 가치의 총 합

Thank you

