BLG 336E - Analysis of Algorithms II
2020/2021 Spring
Final Project
Question 3

- Please submit your homework only through Ninova. Late submissions will not be accepted.
- Please do not forget to write your name and student ID on the top of each document you upload.
- You should write your code in C++ language and try to follow an object-oriented methodology with well-chosen variables, methods, and class names and comments where necessary.
- Your code should compile on Linux using g++. You can test your code through ITU's Linux Server.
- Because your codes will be processed with an automated tool, `Calico`, make sure that your output format matches the given sample output.
- You may discuss the problem addressed in the homework at an abstract level with your classmates, but you should not share or copy code from your classmates or any web sources. You should submit your individual homework. In case of detection of an inappropriate exchange of information, disciplinary actions will be taken.
- If you have any questions, please ask on related message board named **Final Exam Q3 Questions** for this question.

1 Finding Maximum Profit with Dynamic Programming (25 pts)

Implementation
In this question, you are a traveling merchant moving from city to city and try to gain max profit from your travels. For each city, there is an entry cost that you need to pay with special types of crystals and value of gold you can earn is expected. You have

limited number of crystals you can pay for the city entry costs. Therefore, you need to calculate maximum profit you can gain with the amount of crystals you have.

You are expected to implement Knapsack algorithm, which is used to find optimal combination for a set of items to have maximum gain within a limitation. Your algorithm should follow the equation 1 and pseudo-code in Algorithm 1 (Figure 1). Please check the textbooks [1], [2] and course slides for further details on this algorithm.

$$OPT(i, v) = \begin{cases} 0, & \text{if } i = 0 \\ OPT(\text{i - 1, w}), & \text{if } \text{w}_i > w \\ \max\{\ OPT(i-1, w),\ v_i + OPT(i-1, w-w_i)\}, & \text{otherwise} \end{cases} \quad (1)$$

```
Input: W, v1,…,vN, w1,…,wN

for w = 0 to W
    M[0, w] = 0

for i = 1 to N
    for w = 1 to W
        if wi > w
            M[i, w] = M[i - 1, w]
        else
            M[i, w] = max { M[i - 1, w], vi + M[i - 1, w - wi] }


max = M[N, W]


i = N, j = W

while i and w > 0
    if M[i, w] != M[i - 1, w]
        Add i to solutionSet
        w = w – wi
    i = i - 1

return {max, solutionSet}
```

Figure 1: Pseudo code for Knapsack algorithm

A skeleton code **q3_maxProfit_skeleton.cpp** and three test files **q3_maxProfit_test1.txt**, **q3_maxProfit_test2.txt** and **q3_maxProfit_test3.txt** are provided with the question. The skeleton code already includes method definitions, variable declarations, and output formatting. Therefore, you only need to fill the empty parts related to Knapsack algorithm.

Each test file includes information about the merchant and the cities. The first line indicates how many crystals merchant has. Each line starting with the second one, indicates each city's profit and entry cost, respectively.

Output of your code should contain 1) the filled dynamic programming table, 2) the max profit gained by the cities, and 3) cities visited for max profit. Your implementation will be graded as follow:

- Filling the dynamic programming table and maximum profit [15 pts]
- Finding the set of cities to visit [10 pts]

You can compile and test your implementation with the below commands. Figure 2, Figure 3, Figure 4 show the expected outputs for the given test cases.

```
g++ -std=c++11 -Wall -Werror q3_maxProfit.cpp -o q3_maxProfit.out
./q3_maxProfit.out
```

```
[onalug@ssh Final Exam]$ ./q3_maxProfit.out
Enter the name of the input file: q3_maxProfit_test1.txt
Dynaming Programming Table
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  7  7  7  7  7  7  7  7  7  7
  0  0  0  0  0  7  7  7  7  7  7  7 12 12 12
  0  0  0  3  3  7  7  7 10 10 10 10 12 12 12
  0  0  0 10 10 10 13 13 17 17 17 20 20 20 20
  0  0  2 10 10 12 13 13 17 17 19 20 20 22 22
Max profit is 22.
Cities visited: 1 3 4 5
```

Figure 2: Output for q3_maxProfit_test1.txt.

```
[onalug@ssh Final Exam]$ ./q3_maxProfit.out
Enter the name of the input file: q3_maxProfit_test2.txt
Dynaming Programming Table
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  7  7  7  7  7  7  7  7  7  7
  0  0  0  0  0  7  7  7  7  7 12 12 12 12 12
  0  3  3  3  3  7 10 10 10 10 12 15 15 15 15
  0  3  3 10 13 13 13 13 17 20 20 20 20 22 25
Max profit is 25.
Cities visited: 1 2 3 4
```

Figure 3: Output for q3_maxProfit_test2.txt.

```
[onalug@ssh Final Exam]$ ./q3_maxProfit.out
Enter the name of the input file: q3_maxProfit_test3.txt
Dynaming Programming Table
  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  7  7  7  7  7  7
  0  0  0 10 10 10 10 10 17 17 17
  0  0  2 10 10 12 12 12 17 17 19
Max profit is 19.
Cities visited: 1 2 3
```

Figure 4: Output for q3_maxProfit_test3.txt.

Bibliography

[1] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education India, 2006. 2

[2] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009. 2