Lab Test
Course: Algorithm Analysis Lab (IT-2202)

Write your name, roll number in a comment at the top of your program.

Problem 1:

You have a knapsack of weight capacity **B**. There are **n** types of objects. There is an infinite supply of objects of each type, that is, you can pack multiple objects of the same type in the knapsack. Each object in a given type has the same weight and the same cost (profit)—call these $\mathbf{w_i}$ and $\mathbf{c_i}$ for the i-th type. Your task is to pack objects in your knapsack such that the capacity constraint of the knapsack is not exceeded (that is, the total weight of the objects packed is $\leq$ B), and the total cost of the objects packed is as large as possible.

Devise a dynamic-programming algorithm to solve this variant of the knapsack problem. Write a C++ program ***knapsack.cpp*** to implement your algorithm. The program first reads **n, B, w1,w2,...,wn,** and **c1, c2,..., cn** from the user. The program then calculates and prints the optimal cost for these input values. It should also print the counts of the chosen objects of different types.

Sample input:

n = 5

B = 1000

Weights : 21 23 24 22 24

Costs : 23 25 24 24 22

Sample output:

Maximum cost = 1094

Weight = 1000

Counts : 40 6 0 1 0

Problem 2:

You are given two (unsorted) arrays $\mathbf{A = (a_0,a_1,a_2,...,a_{n-1})}$ and $\mathbf{B = (b_0,b_1,b_2,...,b_{n-1})}$, each consisting of n integers (positive, negative, or zero). You are also given a target sum **T** (a positive integer). Your task is to find disjoint subsets I, J of indices satisfying $\mathbf{I \cup J = \{0,1,2,...,n-1\}}$ such that

$\sum_{i\epsilon I} a_i + \sum_{j\epsilon J} b_j = T$

or to report that no such **I, J** exist.

Devise a dynamic-programming algorithm to solve this variant subset sum problem. Write a C++ program ***subset1.cpp*** to implement your algorithm. The program reads **n,** the elements of the arrays **A and B**, and the target **sum T** from the user.

Sample input:

n = 10

A = 79 -89 -85 94 74 12 -84 70 -21 22

B = -87 -10 62 -33 -39 23 15 30 72 48

T = 123

Sample Output:

Solution exists.

A : 79 -85 74 12 70 -21 22

B : -10 -33 15

Sum1 = 151, Sum2 = -28

Problem 3:

You are given two (unsorted) arrays $\mathbf{X} = (x_0, x_1, x_2, \ldots, x_{n-1})$ and $\mathbf{Y} = (y_0, y_1, y_2, \ldots, y_{n-1})$, each consisting of **n** positive integers. For each $i \in \{0,1,2,\ldots,n-1\}$, you make one of the two choices: $z_i = x_i$ **or** $z_i = -y_i$. Your task is to find out whether for some choices, we have

$$z_0 + z_1 + z_2 + \cdots + z_{n-1} = 0.$$

You solve this problem using a dynamic-programming algorithm. Write a C++ program ***subset2.cpp*** to implement your algorithm. The program read **n**, and the elements of the arrays **A** and **B** from the user.

Sample input:

n = 10

X = 34 31 26 11 58 39 83 62 81 95

Y = 53 72 43 27 46 6 32 55 56 37

Sample output:

Solution exists

Z = -53 -72 26 -27 58 -6 -32 62 81 -37

Sum of elements chosen from X is 227

Sum of elements chosen from Y is 227