Herein attached is a jflap model of a Deterministic Turing machine I created in my Theory of computation course.

This 4-tape Deterministic turing machine demonstrates a brute-force solution to the Knapsack problem, an NP-complete optimization problem.

The Turing machine takes input in the form of: C#n:w1,w2,…,wn, where

C is the knapsack capacity expressed in unary using the digit 1. For example, a capacity of 9kgs would be expressed as 111111111. *C is at least 1.*

n is the number of weights expressed in unary using the digit 0. For example, n=5 would be expressed as 00000. *n is atleast 1.*

w1 to wn are weight, in kgs, of each of the n weights separated by commas.

Some valid inputs to use are:

111111111#0000#111,1,11,11

1111#00#111,11

I have attached a link to Jfap 7.1.

To use ‘Knapsack.jff’, Follow below screenshots:

Graphical user interface, application

Description automatically generated

Find ‘knapsack.jff’ file.

To run ‘Knapsack.jff’:

A picture containing diagram

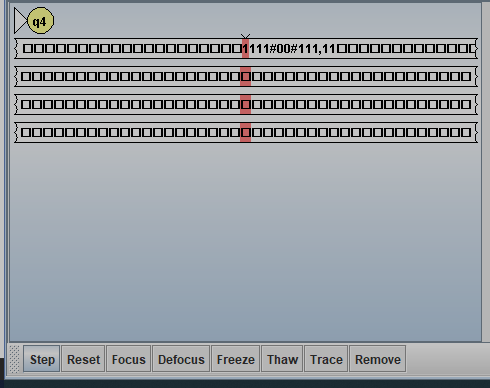
Description automatically generated

Place input(in the form given about) in input 1:

Graphical user interface, application

Description automatically generated

Click ‘ok’.



Click ‘Step’ to see the Turing Machine’s progression towards the answer, i.e. the particular subset of the weights given in the input which utilize the highest capacity of the Knapsack.

Example Result 1(check the third Tape, i.e. the third line):

Table

Description automatically generated

Example result 2(1st Tape=input; 3rd Tape=output):

Table

Description automatically generated