DSA 5113

Homework 2

Group 4

Dylan Steimel

Ramiro Angulo

Question 1: Grapes of Wrath, Inc. (38 points)

a) Grimes determines an upper bound for grapes used for the raisins product by using the following calculation.
First consider the following found in the Question/Problem:
• Grimes upper bound is 1,240,000 pounds
• 15% of the 6,200,000 pound crop was Grade "A" which is equal to 930,000 pounds
• From the 6,200,000-pounds the remaining crop was Grade "B" which is equal to 5,720,000 pounds
• "A" grapes averaged 9 points per pound
• B" grapes averaged 5 points per pound
• minimum average input quality for Raisins is 8 points per pound
Calculating the upper bound for Raisin grapes requires us to find the number of "A" grapes and "B" grapes that can lead to the Raisin 8 points per pounds average. Since "A" 9 points per pound is beyond "Raisins" 8 points per pound, it will all be used, but will need to be mixed with "B" 5 points per pound grape to bring down the average to 8 points per pound. Therefore, we are trying to find the amount of "B" pounds required to get an average of 8 points per pound.
ppp = Points per pound
A*9ppp + B*5ppp = A*8ppp + B*8ppp
A*9ppp – A*8ppp = B*8ppp – B*5ppp
A*1ppp = B*3ppp
(A*1ppp)/3ppp = B
(930,000*1)/3 = B
310,000 = B
This is the total number of "B" pounds needed to create the upper bound for Raisins. Add this to the total number of A pounds and you will get the answer below.
A + B = 930,000 pounds + 310,000 pounds = 1,240,000 pounds raw product

b) Bollman computes the fruit costs first getting each products point per pound ratio to tell how many of each grade "A" and "B" is required for each product to achieve its points per pound average minimum. Once Bollman arrives at the ratio using substitution they can calculate pounds of grade "A" and grade "B" per product. Once Bollman has the pounds of "A" and "B" they can find the cost by multiplying the pounds by the corresponding cost per pound.
ASSUMPTION: cost are in US dollars.
ppp = Points per pound
2-pack, 13 oz. Raisins
• Raisin fruit cost is $2.60
• Pounds per product from Table 2 = 6.5 pounds raw product
To get the ratio, recall from part a.) that A*1ppp = B*3ppp

In other words, A=3B
A + B = 6.5 pounds raw product
Now substitute A with 3B.
3B + B = 6.5 pounds of raw product
4B + = 6.5 pounds of raw product
B = 6.5/4
B = 1.625 pounds of raw product
Now find A by inputting newly found B value in the same equation.
A + B = 6.5 pounds of raw product
A + 1.62 = 6.5 pounds of raw product
A = 6.5 - 1.625
A= 4.875 pounds of raw product
Find cost
A * $0.45 = 4.875 * $0.45 = $2.19375
B * $0.25 = 1.625 * $0.25 = $0.40625
Cost A + Cost B = $2.19375 + $0.40625 =$2.60
24-Carton 10 oz. Juice
- Grape Juice fruit cost is $4.20
- Pounds per product from Table 2 = 14 pounds of raw product

A*9ppp + B*5ppp = A*6ppp + B*6ppp
A*9ppp – A*6ppp = B*6ppp – B*5ppp
A*3ppp = B*1ppp
Ratio
B=3A
Pounds
A + B = 14
Substitute B with 3A
A + 3A + 14
4A = 14
A = 14/4
A = 3.5
Now find B by inputting newly found A value in the same equation.
A + B = 14
3.5 + B = 14
B = 14 – 3.5
B = 10.5
Find cost
A * $0.45 = 3.5 * $0.45 = $1.575
B * $0.25 = 10.5 * $0.25 = $2.625
Cost A + Cost B = $1.575 + $2.625 =$4.20
6-pack 32 oz. Jelly
The cost of jelly is easy to calculating because it only involves grade B grapes and does not require a ratio between grade A and grade B grapes.
- Grape Juice fruit cost is $4.50
- Pounds per product from Table 2 = 18 pounds of raw product

Find cost
B * $0.25 = 18 * $0.25 = $4.50

c) Part C results below

i. Raisin: 54359, Juice: 190,000, Jelly: 177,037

ii. No grapes are leftover

iii. Jelly:$2.89 > Juice:$2.18 > Raisin:$0.52, Total Profit :$959,540

iv. Raisins: 8, Juice: 6, Jelly: 5

v. The shadow price of A-grade grapes is .07 (over the .45 current cost), so the shadow price is .52

vi. They should buy the grapes as the cost of .50 is greater than the shadow price of .52 and the upper bound on that shadow price will not bet met with those 300,000 pounds

vii. A Grade: .51 as the shadow price is .52

B Grade: .40 as the shadow price is .41

```
Make [*] :=
  Jelly  177037
  Juice  190000
Raisins   54359
;

Mix :=
Jelly   A         0
Jelly   B   3186670
Juice   A    665000
Juice   B   1995000
Raisins A    265000
Raisins B     88333.3
;

Profit = 2695540

:   supplies supplies.lb supplies.ub supplies.up supplies.down    :=
A   0.523661   -Infinity     930000    10490000       665000
B   0.410556   -Infinity    5270000     5863330      2083330
;
```

d) Part D results below

a. Raisin: 54359, Juice: 190,000, Jelly: 177,037. Thomas' profit contribution is the same at $959,540. The shadow price of "Grade A" grapes using Thomas' numbers is .24

```
Make [*] :=
  Jelly  177037
  Juice  190000
Raisins   54359
;

Mix :=
Jelly   A        0
Jelly   B   3186670
Juice   A    665000
Juice   B   1995000
Raisins A    265000
Raisins B     88333.3
;

Profit = 959540

production_cost [*] :=
  Jelly  11.54
  Juice  13.74
Raisins   6.89
;

:   supplies supplies.lb supplies.ub supplies.up supplies.down    :=
A   0.243661   -Infinity     930000    10490000       665000
B   0.130556   -Infinity    5270000     5863330      2083330
;
```

b. Raisin: 0, Juice: 172,857, Jelly: 210,000. Bollman's profit contribution is greater at $983,729. The shadow price of "Grade A" grapes using Bollman's numbers is .15

Jelly: $945,000

Juice: $725,999.4

Total: $1,670,999.4

Actual:$1,736,000

```
Make [*] :=
  Jelly  210000
  Juice  172857
Raisins       0
;

Mix :=
Jelly   A    325000
Jelly   B   3455000
Juice   A    605000
Juice   B   1815000
Raisins A         0
Raisins B         0
;

Profit = 983729

production_cost [*] :=
  Jelly  11
  Juice  14.02
Raisins   7.67
;

:   supplies supplies.lb supplies.ub supplies.up supplies.down    :=
A   0.155714   -Infinity     930000     1170000       496667
B   0.155714   -Infinity    5270000     5510000      2850000
;
```

c. Thomas and Bollman both assume the price of fruit/grade ratio before calculating their mixes and include it in their profit equation, resulting in wrong calculations.

When we observe the results using Bollman's numbers we see that we produce 0 raisin products. This is because the ratio of A:B grapes is lower in the other products and even though we are using more expensive grapes in the mix this is not being accounted for. Essentially her model lowers the cost of the product already purchased.

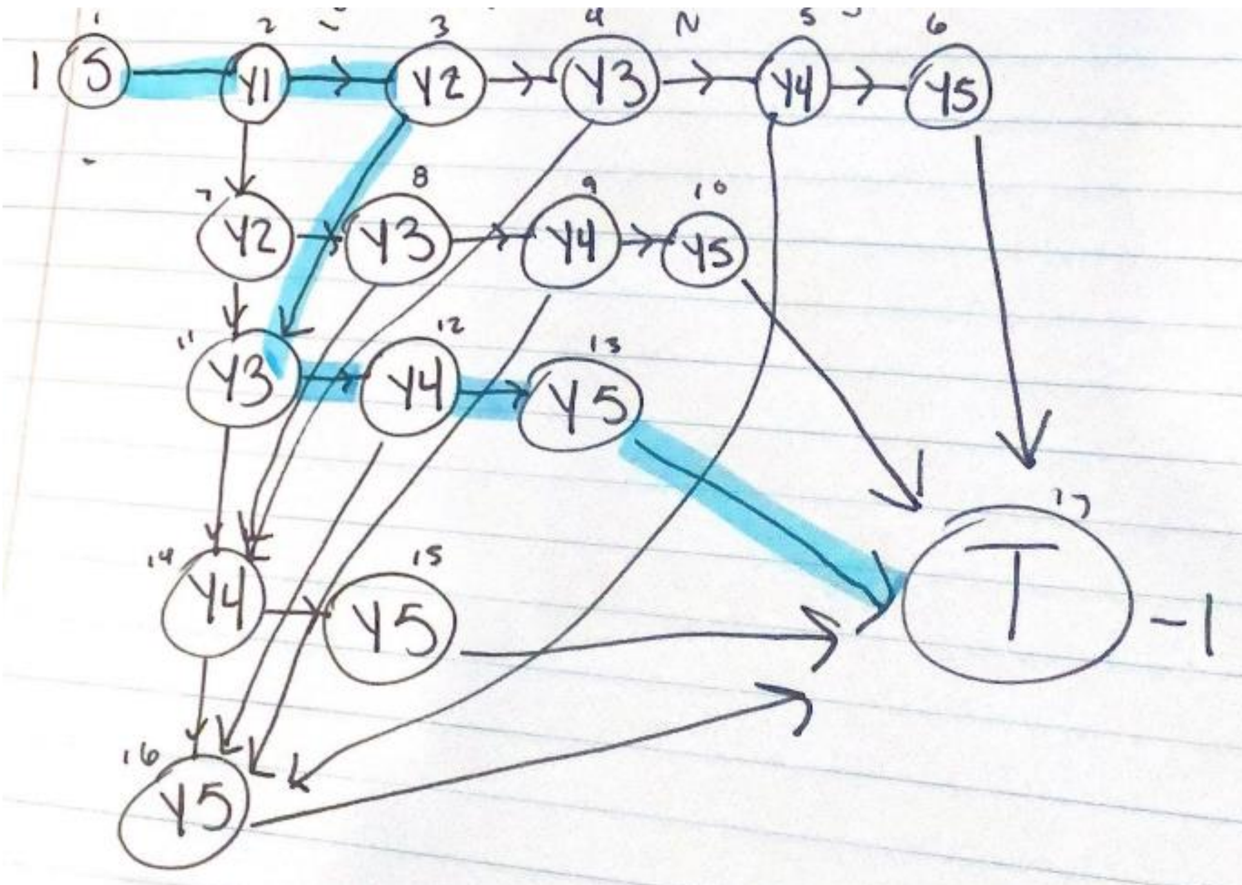
Observing Thomas' answers he gets the correct profit and product mix however because he assumes fruit price beforehand it throws off his values for the shadow price and cost of making products. So his model is not accurate for deciding purchase prices.

My model takes in to account cost of fruit but doesn't predetermine what mix will be used for each product resulting in a better and more accurate model for both ingredient allocation and purchase prices.

Problem 2 Visual

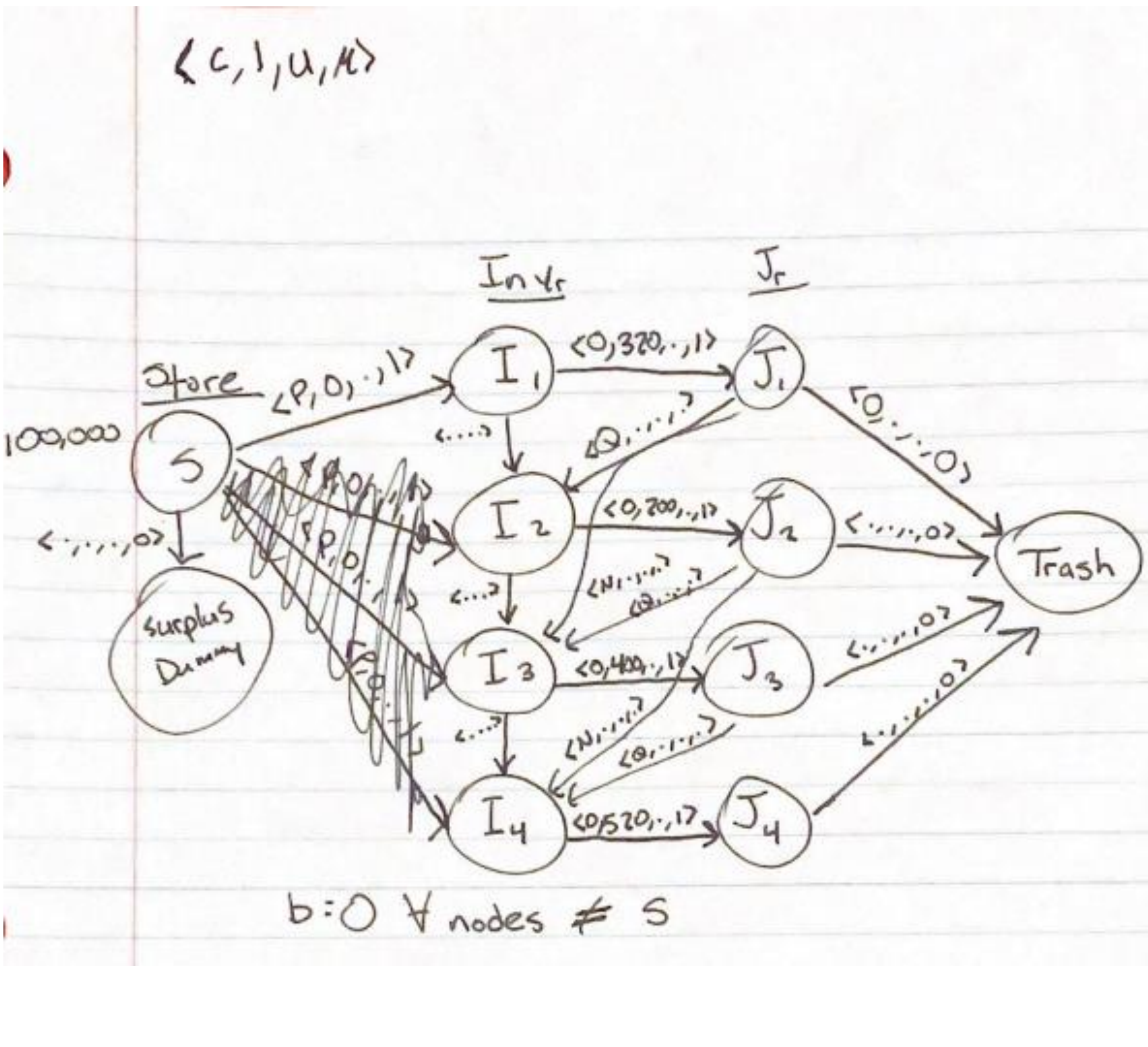
Keep your grill for 2 years, sell it for a second grill and keep that up to the fifth year. (Costs/bounds are in .dat file)

Each node represents the state of the grill at the beginning of each year. All grills listed vertically beneath and including node 2 represent a year old grill, purchased at the previous node. Moving horizontally to the right represents the keeping same grill from year to year. The “Y” labels represent years passed since the source node, so Y3 is 3 years from when they first bought a grill. Every arc a(i,j) represents the yearly cost associated with the decision at i and leads to j, who’s path length to any “Year 5” node is 5-(current year). This ensures we are finding the minimum cost for the given period.



Problem 3

a) Day 1 we can purchase as many tires as we want, we use a dummy node to handle the store's surplus. Moving vertically represents unused tires from the previous days inventory, which we assume has 0 cost. Each race day is represented by the J nodes and we use the numbers provided in the problem as lower bounds on our inventory to race nodes to ensure we have enough tires for each day. From those days we have service nodes leading to the next days inventory, with a cost of Q for quick service, and the day after's inventory with a cost of N for normal service. Finally, we have a trash node to ensure our flow is balanced.

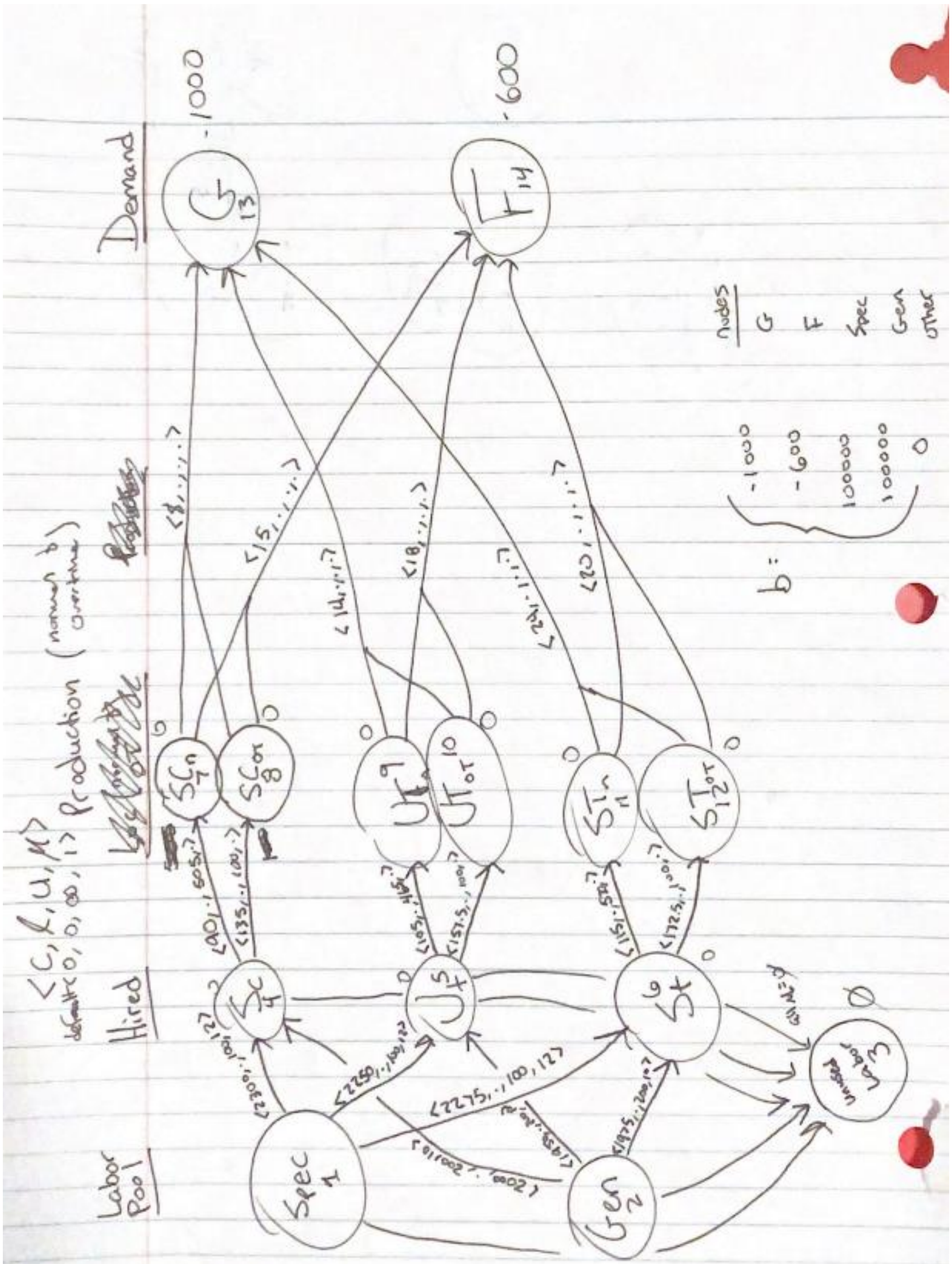


b) We observe that we only buy 520 tires, with 200 unused in day 1's race, 0 quick serviced, and all 320 that were used normal serviced for day 3. Day 2 we use the 200 unused tires from day 1, leaving 0 in inventory for day 3, 80 of which are quick serviced and the remaining 120 normal serviced for day 4. Day 3 we have 320 from day 1 normal service and 80 from day 2 quick service and all 400 are used in the race and quick serviced for day 4. Day 4 we have 400 quick service and 120 normal service tires that we use in the race.

```
x :=
i1 i2     200
i1 j1     320
i2 i3       0
i2 j2     200
i3 i4       0
i3 j3     400
i4 j4     520
j1 i2       0
j1 i3     320
j1 t        0
j2 i3      80
j2 i4     120
j2 t        0
j3 i4     400
j3 t        0
j4 t      520
s  i1     520
s  sd   99480
;
```

Problem 4:

Graph



Explanation: We start with the 2 worker supply nodes which feed all 3 plants at varied costs and abilities which we convert into supplies on these arcs. From there we feed the supplies to “Normal” production nodes for each plant with respective caps matching each plants limit, and “Overtime nodes” at higher costs which allow each plant to produce an extra 100 units. From there the products are distributed to the 2 companies to meet their demands. Note: We use a labor dummy node for unused labor.

Solution (Non-integer, assuming you don’t have to hire somebody an entire month)

In this solution we only hire specialized workers, maxing out all facilities normal production and giving the Scranton branch 60 units produced in overtime All of the Scranton production is shipped to Goggle along with 435 units from Utica. The remaining 30 Utica units and all units produced at Stamford go to Faceblock.

```
x [*,*] (tr)
:        1          2      4     5     6     7     8     9    10    11    12    :=
3    99866.7      1e+05    0     0     0     .     .     .     .     .     .
4       47.0833     0      .     .     .     .     .     .     .     .     .
5       38.75       0      .     .     .     .     .     .     .     .     .
6       47.5        0      .     .     .     .     .     .     .     .     .
7        .          .    505     .     .     .     .     .     .     .     .
8        .          .     60     .     .     .     .     .     .     .     .
9        .          .      .   465     .     .     .     .     .     .     .
10       .          .      .     0     .     .     .     .     .     .     .
11       .          .      .     .   570     .     .     .     .     .     .
12       .          .      .     .     0     .     .     .     .     .     .
13       .          .      .     .     .   505    60   435     0     0     0
14       .          .      .     .     .     0     0    30     0   570     0
;
```

Integer Solution (A hire must work the whole month)

In this solution we hire 4 general workers (1 at Utica, 3 at Stamford) and the rest specialized workers. Again maxing out normal production at all 3 plants. However 1 unit of is produced in overtime at Utica while the remaing 59 overtime units are again produced in Scranton. Shipping remains the same (with the 1 overtime unit from Utica also going to Goggle)

```
x [*,*] (tr)
:      1       2      4     5     6     7     8     9    10    11    12    :=
3    99870   99996    0     0     0     .     .     .     .     .     .
4       47       0    .     .     .     .     .     .     .     .     .
5       38       1    .     .     .     .     .     .     .     .     .
6       45       3    .     .     .     .     .     .     .     .     .
7      .       .    505     .     .     .     .     .     .     .     .
8      .       .     59     .     .     .     .     .     .     .     .
9      .       .      .   465     .     .     .     .     .     .     .
10     .       .      .     1     .     .     .     .     .     .     .
11     .       .      .     .   570     .     .     .     .     .     .
12     .       .      .     .     0     .     .     .     .     .     .
13     .       .      .     .     .   505    59   435     1     0     0
14     .       .      .     .     .     0     0    30     0   570     0
;
```