MAE 5930 - Optimization

Fall 2019

Homework 4

Jared Hansen

Due: 11:59 PM, Thursday October 17, 2019

A-number: `A01439768`

e-mail: `jrdhansen@gmail.com`

Purpose: the problems assigned help develop your ability to

- recognize and formulate integer programs.
- convert formulations into code using MATLAB's `intlinprog`.

NOTE: please write or type your formulations clearly so that a reader can understand what you are doing. You are welcome to use the equivalent functions in Python.

1. You are the supervisor for ACME security, and you are responsible for assigning security officers a position. In the table below, seven officers' availability and cost are shown. That is, the table shows the times each officer comes to work, leaves work, and how much you must pay them to work that time. Your goal as the supervisor is to ensure that at least one person is always on duty between 1pm and 9pm while minimizing cost. You cannot pay for a partial shift.

Officers	Jenna	Doug	Tyler	Cindy	Brad	Ian	Ellie
Hours	1-5PM	1-3PM	4-7PM	4-9PM	6-9PM	5-8PM	8-9PM
Cost ($)	30	18	21	38	20	22	9

As an example, you could choose Jenna, Tyler and Brad. This would cost you $\$30 + \$21 + \$20 = \$71$.

(a) Formulate this into an optimization problem by defining the variables, objective, and constraints. Be sure to specify if the variables are integers or reals.

- **DEFINING VARIABLES:**
  - In this problem, the only thing the supervisor can control is deciding who does work and who doesn't work. Therefore, the values of our decision variables must be **binary values** (integers) of either 0 (indicating that the guard isn't assigned to work) or 1 (indicating that the guard is assigned to work).
  - Let $\boxed{\mathbf{x}^T = [j, d, t, c, b, i, e]^T}$ where:
    * $[j = 1$ if Jenna is assigned to work or $j = 0$ if Jenna isn't assigned to work $]$
    * $[d = 1$ if Doug is assigned to work or $d = 0$ if Doug isn't assigned to work $]$
    * $[t = 1$ if Tyler is assigned to work or $t = 0$ if Tyler isn't assigned to work $]$
    * $[c = 1$ if Cindy is assigned to work or $c = 0$ if Cindy isn't assigned to work $]$
    * $[b = 1$ if Brad is assigned to work or $b = 0$ if Brad isn't assigned to work $]$
    * $[i = 1$ if Ian is assigned to work or $i = 0$ if Ian isn't assigned to work $]$
    * $[e = 1$ if Ellie is assigned to work or $e = 0$ if Ellie isn't assigned to work $]$
  - To describe this mathematically, we can use an indicator function. (Feel free to skip this bit if the above definition is satisfactory.)
    * Define A to be the set of guards we end up assigning to work. Thus, $\left[A \subseteq \mathbf{x}\right]$.

      Using the example solution in the prompt we'd have $\left[A = \{j, t, b\}\right]$.
    * We can define the indicator function as $\mathbb{1}_A(x) := \left\{ \begin{array}{ll} 1 & \text{if } x_p \in A \\ 0 & \text{if } x_p \notin A \end{array} \right\}$ where $x_p \in \{j, d, t, c, b, i, e\}$
    * Using the example given in the prompt where $\left[A = \{j, t, b\}\right]$ would yield:

      $\left[[\mathbb{1}_A(j) = 1], [\mathbb{1}_A(d) = 0], [\mathbb{1}_A(t) = 1], [\mathbb{1}_A(c) = 0], [\mathbb{1}_A(b) = 1], [\mathbb{1}_A(i) = 0], [\mathbb{1}_A(e) = 0]\right]$

      giving: $\left[\mathbf{x}^T = [1, 0, 1, 0, 1, 0, 0]^T\right]$ for the example solution given in the prompt.

- **OBJECTIVE:**
  - In words, our objective is to minimize the cost (while always having a guard on duty: see constraints for this part).
  - In math we have $\left[\min_{\mathbf{x}} \mathbf{c}^T\mathbf{x}\right] = \left[30j+18d+21t+38c+20b+22i+9e\right] = \left[\min_{\mathbf{x}} \Big([30,18,21,38,20,22,9][\mathbf{x}]\Big)\right]$
  - NOTE: $\left[\mathbf{c} = [30,18,21,38,20,22,9]\right]$ which is a vector (bolded) of costs for each guard, while c is a binary indicator (not-bolded) of whether or not Cindy is assigned to work.

- **CONSTRAINTS:**
  - **The optimum for all decision variables in x must be binary.**
    * We have already done this in the **defining variables** section above by specifying that each variable is only allowed to take on a value of either 0 or 1.
    * In order for the code below to make more sense, here is how we encode this into MATLAB:
      · We set the lower and upper bounds for $\mathbf{x}$ as:
      $$\left[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T \leq \begin{bmatrix}j^* & d^* & t^* & c^* & b^* & i^* & e^*\end{bmatrix}^T \leq \begin{bmatrix}1 & 1 & 1 & 1 & 1 & 1 & 1\end{bmatrix}^T\right]$$
      · We specify that all 7 indices of $\mathbf{x}$ must be integers, seen below as `intcon = (1:7) ;`
      · Since we've specified that all 7 variables in our vector $\mathbf{x}$ must be integers AND are bounded at 0 and 1, this leaves only integer values of 0 or 1 as possibilities for the output vector $\mathbf{x}^*$.
  - **At least one guard is always on duty.**
    We can create a binary matrix indicating guard availability, multiply that by our $\mathbf{x}$ vector, and make sure that each time period (e.g. 1-2PM, ..., 8-9PM) is at least 1 in the resulting vector output (representing how many guards are on duty during that hour of the day).
  - Here we have $\left[A_1\mathbf{x} \geq \mathbf{b}_1\right]$. To get this into the correct form with $\leq$ inequalities we can say that $\left[-A_1 = A_2\right]$ and $\left[-\mathbf{b}_1 = \mathbf{b}_2\right]$ giving $\left[A_1\mathbf{x} \geq \mathbf{b}_1\right] = \left[-A_1\mathbf{x} \leq -\mathbf{b}_1\right] = \left[A_2\mathbf{x} \leq \mathbf{b}_2\right]$
    which is now in the $\leq$ inequality form that we want.
  - Now let's define the matrix A_1. We encode guard availability with hours of the day (1-2PM, 2-3PM, ..., 8-9PM) as the rows, and the columns as an individual guard's availability during those hours (in order: j, d, t, c, b, i, e).
    For example, the only two guards available during the 1-2PM hour are Jenna and Doug, so the first row of the matrix A_1 looks like: $\begin{bmatrix}1 & 1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}$
    Another example: Jenna is available during four hours of the day, 1-2PM, 2-3PM, 3-4PM, and 4-5PM. Therefore the "j" column (1st column) of A_1 looks like: $\begin{bmatrix}1 & 1 & 1 & 1 & 0 & 0 & 0\end{bmatrix}^T$

    Altogether, we have:
    $$\left[A_1\mathbf{x} \geq b_1\right] : \begin{bmatrix}1&1&0&0&0&0&0\\1&1&0&0&0&0&0\\1&0&0&0&0&0&0\\1&0&1&1&0&0&0\\0&0&1&1&0&1&0\\0&0&1&1&1&1&0\\0&0&0&1&1&1&0\\0&0&0&1&1&0&1\end{bmatrix}\begin{bmatrix}j\\d\\t\\c\\b\\i\\e\end{bmatrix} \geq \begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}$$
    which signifies that at least one (hence the $\geq \mathbf{1}$) guard is on duty for each hour in 1-2PM, ..., 8-9PM.

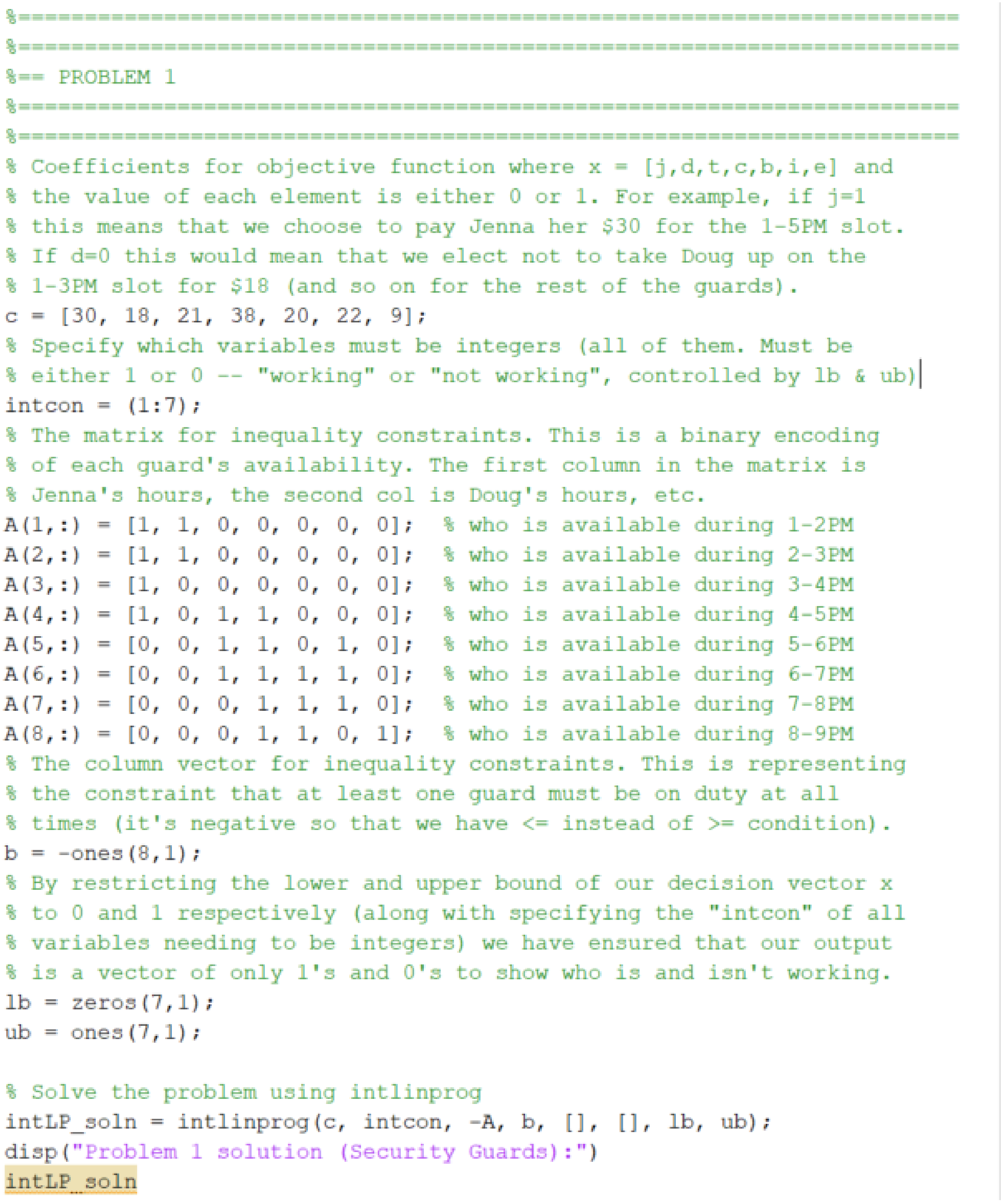
– Now, to get this in $\leq$ inequality form we use the logic described above, giving:

$$\left[A_2\mathbf{x} \leq b_2\right] : \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & -1 & -1 & -1 & 0 \\ 0 & 0 & 0 & -1 & -1 & -1 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} j \\ d \\ t \\ c \\ b \\ i \\ e \end{bmatrix} \leq \begin{bmatrix} -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \\ -1 \end{bmatrix}$$

Just as above, this signifies that at least one (hence the $\leq$ **-1**) guard is on duty for each hour in 1-2PM, 2-3PM, …, 8-9PM. But now we have it in a form that is congruent with standard inequality form, and can be entered into MATLAB.

(Although in MATLAB I just multiplied $(-1)(A_1)$ and $(-1)(b_1)$ instead of typing all the negative signs like I have here. The result is identical though, just less typing.)

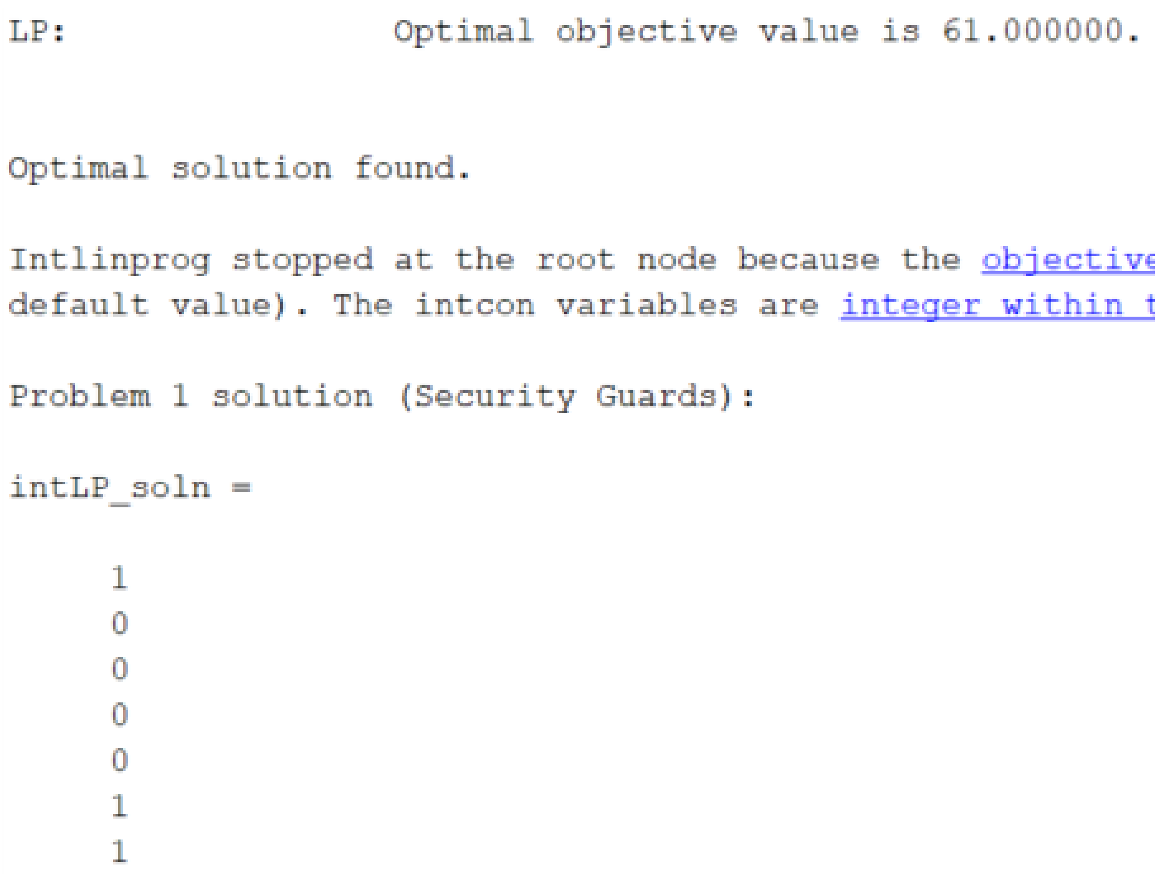
(b) Solve the problem using MATLAB's `intlinprog`. Please provide your code and output showing the correct answer.



```
%==========================================================================
%==========================================================================
%== PROBLEM 1
%==========================================================================
%==========================================================================
% Coefficients for objective function where x = [j,d,t,c,b,i,e] and
% the value of each element is either 0 or 1. For example, if j=1
% this means that we choose to pay Jenna her $30 for the 1-5PM slot.
% If d=0 this would mean that we elect not to take Doug up on the
% 1-3PM slot for $18 (and so on for the rest of the guards).
c = [30, 18, 21, 38, 20, 22, 9];
% Specify which variables must be integers (all of them. Must be
% either 1 or 0 -- "working" or "not working", controlled by lb & ub)
intcon = (1:7);
% The matrix for inequality constraints. This is a binary encoding
% of each guard's availability. The first column in the matrix is
% Jenna's hours, the second col is Doug's hours, etc.
A(1,:) = [1, 1, 0, 0, 0, 0, 0];  % who is available during 1-2PM
A(2,:) = [1, 1, 0, 0, 0, 0, 0];  % who is available during 2-3PM
A(3,:) = [1, 0, 0, 0, 0, 0, 0];  % who is available during 3-4PM
A(4,:) = [1, 0, 1, 1, 0, 0, 0];  % who is available during 4-5PM
A(5,:) = [0, 0, 1, 1, 0, 1, 0];  % who is available during 5-6PM
A(6,:) = [0, 0, 1, 1, 1, 1, 0];  % who is available during 6-7PM
A(7,:) = [0, 0, 0, 1, 1, 1, 0];  % who is available during 7-8PM
A(8,:) = [0, 0, 0, 1, 1, 0, 1];  % who is available during 8-9PM
% The column vector for inequality constraints. This is representing
% the constraint that at least one guard must be on duty at all
% times (it's negative so that we have <= instead of >= condition).
b = -ones(8,1);
% By restricting the lower and upper bound of our decision vector x
% to 0 and 1 respectively (along with specifying the "intcon" of all
% variables needing to be integers) we have ensured that our output
% is a vector of only 1's and 0's to show who is and isn't working.
lb = zeros(7,1);
ub = ones(7,1);

% Solve the problem using intlinprog
intLP_soln = intlinprog(c, intcon, -A, b, [], [], lb, ub);
disp("Problem 1 solution (Security Guards):")
intLP_soln
```

Figure 1: MATLAB code for solving the "security guard problem" formulated above by using `intlinprog`.



```
LP:                Optimal objective value is 61.000000.


Optimal solution found.

Intlinprog stopped at the root node because the objective
default value). The intcon variables are integer within t

Problem 1 solution (Security Guards):

intLP_soln =

     1
     0
     0
     0
     0
     1
     1
```

Figure 2: MATLAB output for the above code (see brief discussion of the output directly below).

- Based on the output of `intlinprog` we can conclude that the optimal solution is achieved by $\mathbf{x}^* = \begin{bmatrix} j^* \\ d^* \\ t^* \\ c^* \\ b^* \\ i^* \\ e^* \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$ which means that we will assign only Jenna, Ian, and Ellie to work.
- With $\mathbf{x}^*$ we know that $\left[\mathbf{c}^T\mathbf{x}^*\right] = \left[\begin{bmatrix}30 & 18 & 21 & 38 & 20 & 22 & 9\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 1 & 1\end{bmatrix}^T\right]$
$= \left[30(1) + 18(0) + 21(0) + 38(0) + 20(0) + 22(1) + 9(1)\right] = \left[30 + 22 + 9\right] = \left[\$61.00\right]$
This is why the MATLAB output says "`Optimal objective value is 61.00000.`"
- Therefore, with the given guard availability and the constraint of having at least one guard on duty at all times we conclude that:
only Jenna, Ian, and Ellie will be assigned to work, which will minimize the cost at $61.00.

2. You are the logistics engineer for an air transportation company. Your goal is to find out how to pack crates on a plane in an optimal way. The crates are of five possible types denoted, A, B, C, D, and E. The table below shows the mass, volume and value of each crate type.

Type	A	B	C	D	E
Mass (kg)	500	1500	2100	600	400
Volume (m^3)	25	15	13	20	16
Value ($)	50,000	60,000	90,000	40,000	30,000

The plane is divided into three segments - front, middle, and back. Each segment has a limited mass and volume capacity. These capacities are summarized in the table below.

Segment	Front	Middle	Back
Volume Capacity (m^3)	200	500	300
Mass Capacity (kg)	8000	20000	6000

For stability reasons during flight, the plane must also be balanced correctly. The balancing requirements are:

- mass of middle cargo $\geq$ (mass of front cargo + mass of back cargo)
- mass of middle cargo $\leq 2 \times$ (mass of front cargo + mass of back cargo)

Additionally there are only: (12 crates of type A), (8 of type B), (22 of type C), (15 of type D), and (11 of type E).

You need to decide how many crates of each type are going in what segment of the the plane to maximize the value delivered.

(a) Formulate this into an optimization problem by defining the variables, objective, and constraints. Be sure to specify if the variables are integers or reals.

- **DEFINING VARIABLES:**
  - In this problem, the only thing the logistics engineer can control is how many of each crate (A, B, C, D, E) to place in each segment (front, middle, back) of the plane.
  - Since there are five different types of crates and three segments of the plane, this leaves us with $5 \times 3 = 15$ decision variables.
  - I will denote each of these variables using $\left[\text{lowerCaseCrateType}_i\right]$ where $i \in \{1, 2, 3\}$ with 1 = front, 2 = middle, and 3 = back.
    For example, the number of type A crates placed in the middle would be represented by the variable a_2, the number of type C crates placed in the front segment would be represented by the variable c_1, the number of type D crates placed in the back segment would be represented by the variable d_3, etc.
  - Using this, let's define our decision variables as the vector $\mathbf{x}$. I organize them by segment as follows:
    $$\left[\mathbf{x}^T := \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & a_2 & b_2 & c_2 & d_2 & e_2 & a_3 & b_3 & c_3 & d_3 & e_3 \end{bmatrix}^T\right]$$
  - Since we cannot have "half of a crate" – or any other partial value of a crate – all values in our vector $\mathbf{x}$ must be integers. They must all also be ≥ 0 since we can't put a negative number of crates onto a plane.
  - Additionally, we cannot go over the maximum number for each crate (12 crates of type A), (8 of type B), (22 of type C), (15 of type D), and (11 of type E). We will handle this in the **constraints** section below.

- **OBJECTIVE:**
  - In words, our objective is to maximize the value of crates packed onto the plane (while obeying the given limitations: see **constraints** for this part).
  - In math we have $\left[\max_{\mathbf{x}}\left(\mathbf{c}^T\mathbf{x}\right)\right]$

    (NOTE: I use "k" at below as shorthand for the numeric value of 1,000, e.g. 50k $=$ 50000.

    where $\mathbf{c}^T\mathbf{x} = \Big[50k(a_1) + 60k(b_1) + 90k(c_1) + 40k(d_1) + 30k(e_1) + 50k(a_2) + 60k(b_2) + 90k(c_2) + 40k(d_2) + 30k(e_2) + 50k(a_3) + 60k(b_3) + 90k(c_3) + 40k(d_3) + 30k(e_3)\Big]$

    $= \begin{bmatrix}50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k\end{bmatrix}\left[\mathbf{x}\right]$
  - Therefore, our objective can be stated as:

    $\left[\max_{\mathbf{x}}\left(\begin{bmatrix}50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k\end{bmatrix}\left[\mathbf{x}\right]\right)\right]$

    $\boxed{\text{where } \left[\mathbf{c}^T = \begin{bmatrix}50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k & 50k & 60k & 90k & 40k & 30k\end{bmatrix}^T\right]}$
  - Finally, we must make use of the fact that $\left[\max_{\mathbf{x}}\left(f(\mathbf{x})\right)\right] \equiv \left[\min_{\mathbf{x}}\left(-f(\mathbf{x})\right)\right]$ to have our objective function in standard form, as well as a form which we can program into MATLAB (minimizing a function rather than maximizing it).

    $\boxed{\text{Thus, our final objective function is: } \left[\min_{\mathbf{x}} -\mathbf{c}^T\mathbf{x}\right] \text{ with } \mathbf{c} \text{ as defined in the list item directly above, and } \mathbf{x} \text{ as defined in the defining variables section above.}}$

- **CONSTRAINTS:**
  - **Constraints on total number of each CRATE TYPE in the aircraft:**
    - In general, $\big[0 \leq crate_i \leq max_i\big] \forall i \in \{A, B, C, D, E\}$
    - Crate A: $\Big[0 \leq \begin{bmatrix}1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 12\Big]$
    - Crate B: $\Big[0 \leq \begin{bmatrix}0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 8\Big]$
    - Crate C: $\Big[0 \leq \begin{bmatrix}0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 22\Big]$
    - Crate D: $\Big[0 \leq \begin{bmatrix}0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0\end{bmatrix} [\mathbf{x}] \leq 15\Big]$
    - Crate E: $\Big[0 \leq \begin{bmatrix}0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1\end{bmatrix} [\mathbf{x}] \leq 11\Big]$
  - **Constraints on total MASS for each segment in the aircraft:**
    - Front (mass): $\Big[\begin{bmatrix}500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 8000\Big]$
    - Middle (mass): $\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 20000\Big]$
    - Back (mass): $\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400\end{bmatrix} [\mathbf{x}] \leq 6000\Big]$
  - **Constraints on total VOLUME for each segment in the aircraft:**
    - Front (volume): $\Big[\begin{bmatrix}25 & 15 & 13 & 20 & 16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 200\Big]$
    - Middle (volume): $\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 25 & 15 & 13 & 20 & 16 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}] \leq 500\Big]$
    - Back (volume): $\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 25 & 15 & 13 & 20 & 16\end{bmatrix} [\mathbf{x}] \leq 300\Big]$
  - **Additional LOAD-BALANCING constraints for the aircraft:**
    - mass of middle cargo $\geq$ (mass of front cargo + mass of back cargo):

      $\Bigg(\Big[\begin{bmatrix}500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400\end{bmatrix} [\mathbf{x}]\Big] \leq$
      $\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}]\Big]\Bigg)$

      Now subtract the RHS from both sides of the inequality, factor out the $\mathbf{x}$ and we're left with:

      $\Big[\big[500, 1500, 2100, 600, 400, -500, -1500, -2100, -600, -400, 500, 1500, 2100, 600, 400\big] [\mathbf{x}] \leq 0\Big]$
    - mass of middle cargo $\leq 2 \times$ (mass of front cargo + mass of back cargo):

      $\Bigg(\Big[\begin{bmatrix}0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0\end{bmatrix} [\mathbf{x}]\Big] \leq$
      $2 \times \Big[\begin{bmatrix}500 & 1500 & 2100 & 600 & 400 & 0 & 0 & 0 & 0 & 0 & 500 & 1500 & 2100 & 600 & 400\end{bmatrix} [\mathbf{x}]\Big]\Bigg)$

      Now subtract the RHS from both sides of the inequality, factor out the $\mathbf{x}$ and we're left with:
      $\big[-1000, -3000, -4200, -1200, -800, 500, 1500, 2100, 600, 400, -1000, -3000, -4200, -1200, -800\big]\, \mathbf{x} \leq 0$

– **All constraints COMBINED into one system:**
Note that I wrote "00" as "h" (short for hundred) to get everything to fit on the page:

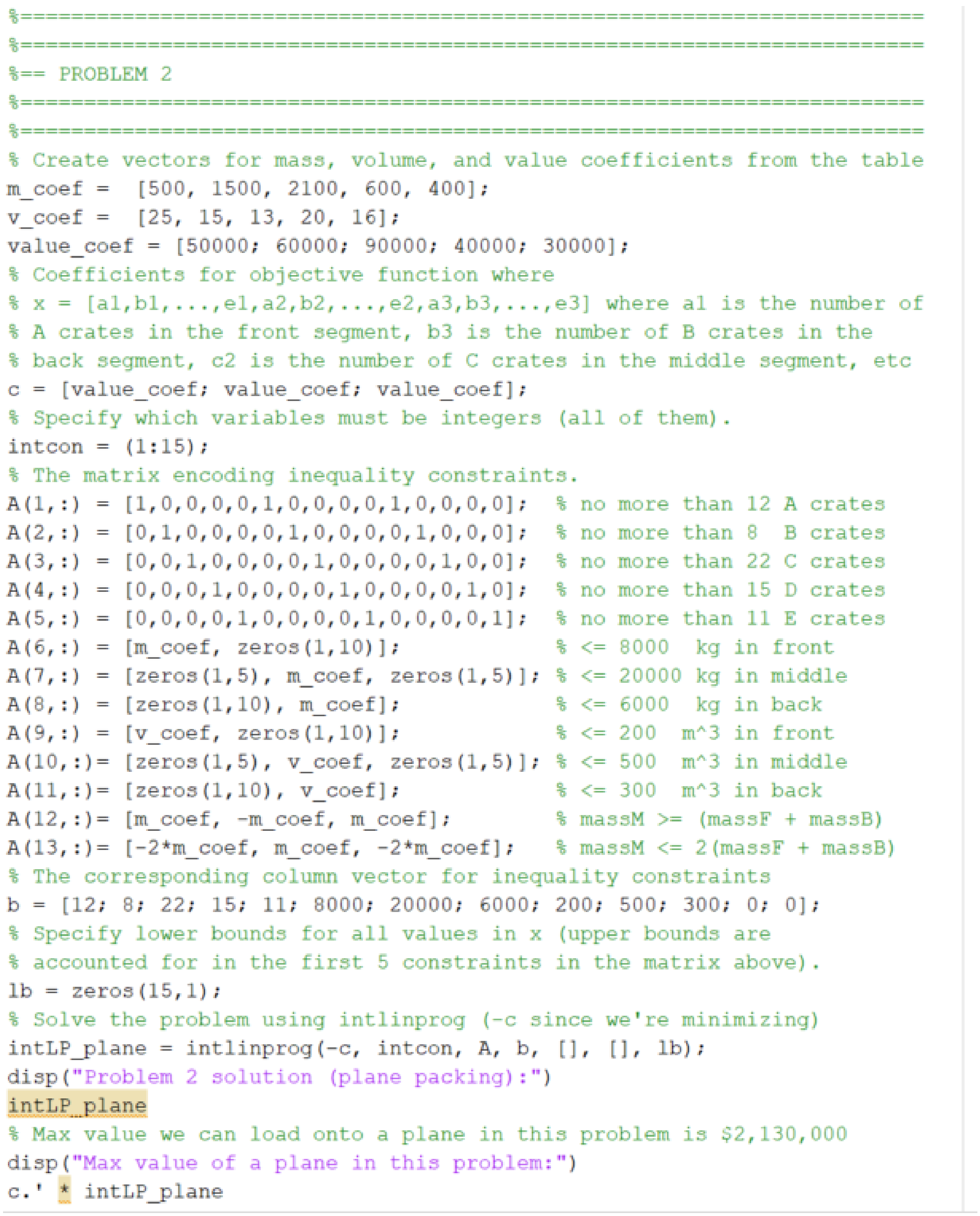
$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\
5h & 15h & 21h & 6h & 4h & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 5h & 15h & 21h & 6h & 4h & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5h & 15h & 21h & 6h & 4h \\
25 & 15 & 13 & 20 & 16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 25 & 15 & 13 & 20 & 16 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 25 & 15 & 13 & 20 & 16 \\
5h & 15h & 21h & 6h & 4h & -5h & -15h & -21h & -6h & -4h & 5h & 15h & 21h & 6h & 4h \\
-10h & -30h & -42h & -12h & -8h & 5h & 15h & 21h & 6h & 4h & -10h & -30h & -42h & -12h & -8h
\end{bmatrix}$$

Above is the matrix A (I ran out of room to put "$A =$ " in front of the matrix.

$$b = \begin{bmatrix} 12 \\ 8 \\ 22 \\ 15 \\ 11 \\ 8000 \\ 20000 \\ 6000 \\ 200 \\ 500 \\ 300 \\ 0 \\ 0 \end{bmatrix}$$

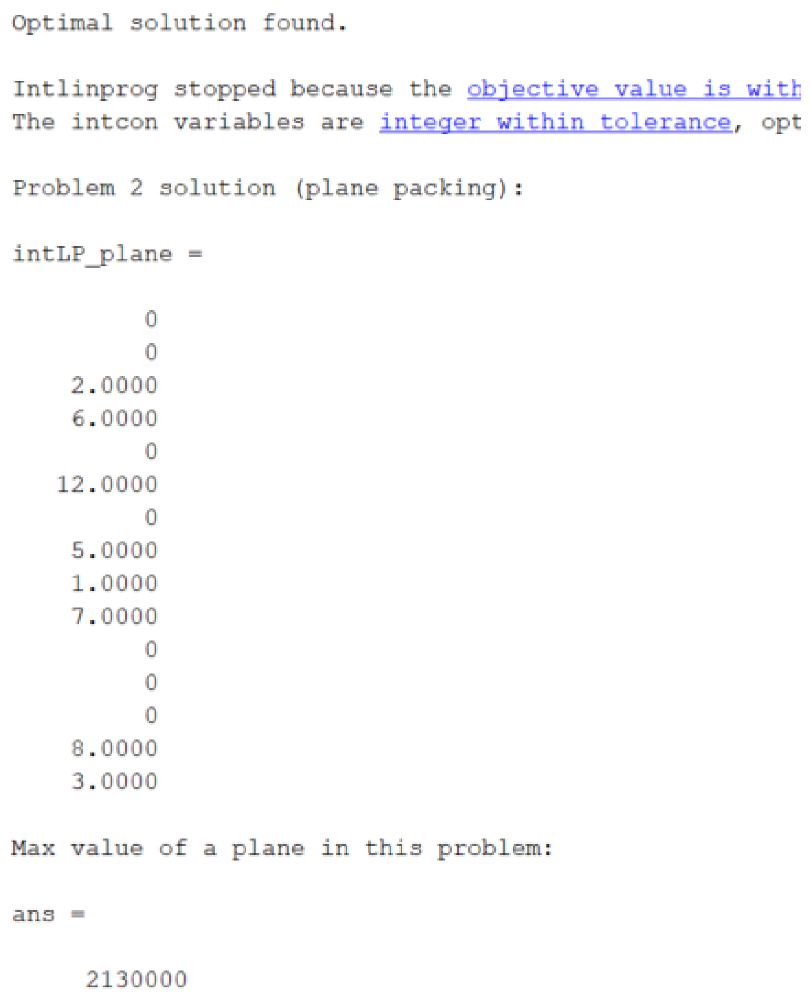
We have now defined all of our constraints in terms of a matrix A, a vector $\mathbf{b}$, and our vector of decision variables $\mathbf{x}$, combining them into the system $\left[A\mathbf{x} \leq \mathbf{b}\right]$.

(b) Solve the problem using MATLAB's `intlinprog`. Please provide your code and output showing the correct answer.



```
%=========================================================================
%=========================================================================
%== PROBLEM 2
%=========================================================================
%=========================================================================
% Create vectors for mass, volume, and value coefficients from the table
m_coef =  [500, 1500, 2100, 600, 400];
v_coef =  [25, 15, 13, 20, 16];
value_coef = [50000; 60000; 90000; 40000; 30000];
% Coefficients for objective function where
% x = [a1,b1,...,e1,a2,b2,...,e2,a3,b3,...,e3] where a1 is the number of
% A crates in the front segment, b3 is the number of B crates in the
% back segment, c2 is the number of C crates in the middle segment, etc
c = [value_coef; value_coef; value_coef];
% Specify which variables must be integers (all of them).
intcon = (1:15);
% The matrix encoding inequality constraints.
A(1,:) = [1,0,0,0,0,1,0,0,0,0,1,0,0,0,0];  % no more than 12 A crates
A(2,:) = [0,1,0,0,0,0,1,0,0,0,0,1,0,0,0];  % no more than 8  B crates
A(3,:) = [0,0,1,0,0,0,0,1,0,0,0,0,1,0,0];  % no more than 22 C crates
A(4,:) = [0,0,0,1,0,0,0,0,1,0,0,0,0,1,0];  % no more than 15 D crates
A(5,:) = [0,0,0,0,1,0,0,0,0,1,0,0,0,0,1];  % no more than 11 E crates
A(6,:) = [m_coef, zeros(1,10)];            % <= 8000  kg in front
A(7,:) = [zeros(1,5), m_coef, zeros(1,5)]; % <= 20000 kg in middle
A(8,:) = [zeros(1,10), m_coef];            % <= 6000  kg in back
A(9,:) = [v_coef, zeros(1,10)];            % <= 200  m^3 in front
A(10,:)= [zeros(1,5), v_coef, zeros(1,5)]; % <= 500  m^3 in middle
A(11,:)= [zeros(1,10), v_coef];            % <= 300  m^3 in back
A(12,:)= [m_coef, -m_coef, m_coef];        % massM >= (massF + massB)
A(13,:)= [-2*m_coef, m_coef, -2*m_coef];   % massM <= 2(massF + massB)
% The corresponding column vector for inequality constraints
b = [12; 8; 22; 15; 11; 8000; 20000; 6000; 200; 500; 300; 0; 0];
% Specify lower bounds for all values in x (upper bounds are
% accounted for in the first 5 constraints in the matrix above).
lb = zeros(15,1);
% Solve the problem using intlinprog (-c since we're minimizing)
intLP_plane = intlinprog(-c, intcon, A, b, [], [], lb);
disp("Problem 2 solution (plane packing):")
intLP_plane
% Max value we can load onto a plane in this problem is $2,130,000
disp("Max value of a plane in this problem:")
c.' * intLP_plane
```

Figure 3: MATLAB code for solving the "airplane packing problem" formulated above by using `intlinprog`.



```
Optimal solution found.

Intlinprog stopped because the objective value is with
The intcon variables are integer within tolerance, opt

Problem 2 solution (plane packing):

intLP_plane =

         0
         0
    2.0000
    6.0000
         0
   12.0000
         0
    5.0000
    1.0000
    7.0000
         0
         0
         0
    8.0000
    3.0000

Max value of a plane in this problem:

ans =

     2130000
```

Figure 4: MATLAB output for the above code (see brief discussion of the output directly below).

- Based on the output of `intlinprog` we can conclude that the optimal solution is achieved by $\mathbf{x}^{*T} = \begin{bmatrix} a_1{}^* & b_1{}^* & c_1{}^* & d_1{}^* & e_1{}^* & a_2{}^* & b_2{}^* & c_2{}^* & d_2{}^* & e_2{}^* & a_3{}^* & b_3{}^* & c_3{}^* & d_3{}^* & e_3{}^* \end{bmatrix}^T =$ $\begin{bmatrix} 0 & 0 & 2 & 6 & 0 & 12 & 0 & 5 & 1 & 7 & 0 & 0 & 0 & 8 & 3 \end{bmatrix}^T$
- In context, this means we pack the plan as follows:
  - In the front segment of the plane: 2 C crates, 6 D crates.
  - In the middle segment of the plane: 12 A crates, 5 C crates, 1 D crate, 7 E crates.
  - In the back segment of the plane: 8 D crates, 3 E crates.
- With $\mathbf{x}^*$ we know that $\left[\mathbf{c}^T\mathbf{x}^*\right] = \left[\$2,130,000.00\right]$ is the maximal value we can load onto the plane. I obtained this with the MATLAB code at the very bottom of the image above.
- **VERY IMPORTANT NOTE:** when I verified with some classmates regarding their answer, we came up with slightly different values for $\mathbf{x}^*$, but ended up with the same maximal value of $\$2,130,000.00$. This leads me to believe that variations in encoding, ordering constraints differently, different computer OS, different version of MATLAB, etc. result in different optimal $\mathbf{x}$ but the same optimal value $\left[\mathbf{c}^T\mathbf{x}\right]$ for how the maximal value of goods that we can fit onto the plane.

- **Verifying that the solution $\mathbf{x}^*$ satisfies constraints:**

  - Constraints on total number of each CRATE TYPE in the aircraft:
    $\begin{bmatrix} A_{optimal} & B_{optimal} & C_{optimal} & D_{optimal} & E_{optimal} \end{bmatrix} \leq \begin{bmatrix} A_{max} & B_{max} & C_{max} & D_{max} & E_{max} \end{bmatrix}$??

    $\boxed{\begin{bmatrix} 12 & 0 & 7 & 15 & 10 \end{bmatrix} \leq \begin{bmatrix} 12 & 8 & 22 & 15 & 11 \end{bmatrix} \text{ is true. } \textbf{SATISFIED}}$

  - Constraints on total MASS for each segment in the aircraft:
    - Mass for front segment: $\left[\left(mass_{front} = (2)(2100) + (6)(600) = 7800\right) \leq 8000\right]$ is true. **SATISFIED**.
    - Mass for middle segment: $\left[\left(mass_{middle} = (12)(500) + (5)(2100) + (1)(600) + (7)(400) = 19900\right) \leq 20000\right]$ is true. **SATISFIED**.
    - Mass for back segment: $\left[\left((8)(600) + (3)(400) = 6000\right) \leq 6000\right]$ is true. **SATISFIED**.

  - Constraints on total VOLUME for each segment in the aircraft:
    - Volume for front segment: $\left[\left(mass_{front} = (2)(13) + (6)(20) = 146\right) \leq 200\right]$ is true. **SATISFIED**.
    - Volume for middle segment: $\left[\left(mass_{middle} = (12)(25) + (5)(13) + (1)(20) + (7)(16) = 497\right) \leq 500\right]$ is true. **SATISFIED**.
    - Volume for back segment: $\left[\left((8)(20) + (3)(16) = 208\right) \leq 300\right]$ is true. **SATISFIED**.

  - First Additional LOAD-BALANCING constraints for the aircraft:
    mass of middle cargo $\geq$ (mass of front cargo + mass of back cargo)

    $\left[\left(mass_{middle} = 19900\right) \geq \left(mass_{front} + mass_{back} = 7800 + 6000 = 13800\right)\right]$ is true. **SATISFIED**.

  - Second Additional LOAD-BALANCING constraints for the aircraft: mass of middle cargo $\leq 2 \times$ (mass of front cargo + mass of back cargo)

    $\left[\left(mass_{middle} = 19900\right) \leq \left(2 \times (mass_{front} + mass_{back}) = (2)(7800 + 6000) = 27600\right)\right]$ is true. **SATISFIED**.

  - $\boxed{\text{THEREFORE, ALL CONSTRAINTS ARE SATISFIED for } \mathbf{x}^*}$