This LP is called `ALL_INTEGER_3D_LP` :

$$\begin{aligned} \max \quad & 1x_1 + 2x_2 + 4x_3 \\ \text{s.t.} \quad & 1x_1 + 0x_2 + 0x_3 \leq 6 \\ & 1x_1 + 0x_2 + 1x_3 \leq 8 \\ & 0x_1 + 0x_2 + 1x_3 \leq 5 \\ & 0x_1 + 1x_2 + 1x_3 \leq 8 \\ & x_1, x_2 \geq 0 \end{aligned}$$

In [ ]:
```
lp = gilp.examples.ALL_INTEGER_3D_LP # get LP example
gilp.lp_visual(lp).show() # visualize it
```

The 3D feasible region is shown on the left. Hold and drag the mouse to examine it from different angles. Next, click on a constraint to un-mute it. Each constraint is a gray plane in 3D space. Un-mute the constraints one by one to see how they define the 3D feasible region. Move the objective slider to see the isoprofit planes. The isoprofit plane is light gray and the intersection with the feasible region is shown in red. Like the 2D visualization, you can hover over corner points to see information about that point.

Q5: Use the objective slider to solve this LP graphically. Give an optimal solution and objective value. (Hint: The objective slider shows the isoprofit plane for some objective value in light gray and the intersection with the feasible region in red.)

A:

When it comes to LPs with 4 or more decision variables, our graphical approaches fail. We need to find a different way to solve linear programs of this size.

Part II: The Simplex Algorithm for Solving LPs

Dictionary Form LP

First, let's answer some guiding questions that will help to motivate the simplex algorithm.

Q6: Does there exist a unique way to write any given inequality constraint? If so, explain why each constraint can only be written one way. Otherwise, give 2 ways of writing the same inequality constraint.

A: No, there are multiple ways of writing inequality constraints. For example, $x \leq 4$ can also be written as $0 \leq 4 - x$.

Q7: Consider the following two constraints: $2x_1 + 1x_2 \leq 20$ and $2x_1 + 1x_2 + x_3 = 20$ where all x are nonnegative. Are these the same constraint? Why? (This question is tricky!)

A: I think they are the same constraint, given properly constrained x_3. If $x_3 \leq 0$, then this is the same constraint.

Q8: Based on your answers to **Q6** and **Q7**, do you think there exists a unique way to write any given LP?

A: No, there should be many ways.

You should have found that there are many ways to write some LP. This begs a new question: are some ways of writing an LP harder or easier to solve than others? Consider the following LP:

$$\begin{aligned} \max \quad & 56 - 2x_3 - 1x_4 \\ \text{s.t.} \quad & x_1 = 4 - 1x_3 + 1x_4 \\ & x_2 = 12 + 1x_3 - 2x_4 \\ & x_5 = 3 + 1x_3 - 1x_4 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{aligned}$$

Q9: Just by looking at this LP, can you give an optimal solution and its objective value. If so, explain what property of the LP allows you to do this. (Hint: Look at the objective function)

A: Set $x_3 = x_4 = 0$, then the objective value of the optimal solution is 56, and the optimal solution is $x_1 = 4$, $x_2 = 12$, and $x_5 = 3$.

The LP above is the same as `ALL_INTEGER_2D_LP` just rewritten in a different way! This rewitten form (which we found is easier to solve) was found using the simplex algorithm. At its core, the simplex algorithm strategically rewrites an LP until it is in a form that is "easy" to solve.

The simplex algorithm relies on an LP being in **dictionary form**. Recall the following properties of an LP in dictionary form:

- All constraints are equality constraints
- All variables are constrained to be nonnegative
- Each variable only appears on the left-hand side (LHS) or the right-hand side (RHS) of the constraints (not both)
- Each constraint has a unique variable on the LHS
- The objective function is in terms of the variables that appear on the RHS of the constraints only.
- All constants on the RHS of the constraints are nonnegative

Q10: Rewrite the example LP `ALL_INTEGER_2D_LP` in dictionary form. Show your steps!

$$\begin{aligned} \max \quad & 5x_1 + 3x_2 \\ \text{s.t.} \quad & 2x_1 + 1x_2 \leq 20 \\ & 1x_1 + 1x_2 \leq 16 \\ & 1x_1 + 0x_2 \leq 7 \\ & x_1, x_2 \geq 0 \end{aligned}$$

A: We want to write each constraint in the form $0 \leq \ldots$, so we multiply each one by -1 and then add the left-hand side to the right hand side. We get:

$$\begin{aligned} \max \quad & 5x_1 + 3x_2 \\ \text{s.t.} \quad & 0 \leq 20 - 2x_1 - 1x_2 \\ & 0 \leq 16 - 1x_1 - 1x_2 \\ & 0 \leq 7 - 1x_1 - 0x_2 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Now, let's assign variables to replace the $0 \leq \ldots$'s, and we have our answer.

$$\begin{aligned} \max \quad & z = 5x_1 + 3x_2 \\ \text{s.t.} \quad & x_3 = 20 - 2x_1 - 1x_2 \\ & x_4 = 16 - 1x_1 - 1x_2 \\ & x_5 = 7 - 1x_1 - 0x_2 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{aligned}$$

Most Limiting Constraint

Once our LP is in dictionary form, we can run the simplex algorithm! In every iteration of the simplex algorithm, we will take an LP in dictionary form and strategically rewrite it in a new dictionary form. Note: it is important to realize that rewriting the LP **does not** change the LP's feasible region. Let us examine an iteration of simplex on a new LP.

$$\begin{aligned} \max \quad & 5x_1 + 3x_2 \\ \text{s.t.} \quad & 1x_1 + 0x_2 \leq 4 \\ & 0x_1 + 1x_2 \leq 6 \\ & 2x_1 + 1x_2 \leq 9 \\ & 3x_1 + 2x_2 \leq 15 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Q11: Is this LP in dictionary form? If not, rewrite this LP in dictionary form.

A: That is not in dictionary form. The dictionary form is:

$$\begin{aligned} \max \quad & z = 5x_1 + 3x_2 \\ \text{s.t.} \quad & x_3 = 4 - 1x_1 - 0x_2 \\ & x_4 = 6 - 0x_1 - 1x_2 \\ & x_5 = 9 - 2x_1 - 1x_2 \\ & x_6 = 15 - 3x_1 - 2x_2 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

Q12: Recall from **Q9** how you found a feasible solution (which we argued to be optimal) just by looking at the LP. Using this same stratagy, look at the LP above and give a feasible solution and its objective value for this LP. Describe how you found this feasible solution. Is it optimal? Why?

A: Let $x_1 = x_2 = 0$, and then $x_3 = 4, x_4 = 6, x_5 = 9, x_6 = 15$. That makes $z = 0$, and if we are trying to maximize z, it's probably not optimal.

From **Q12** we see that every dictionary form LP has a corresponding feasible solution. Furthermore, there are positive coefficents in the objective function. Hence, we can increase the objective value by increasing the corresponding variable. In our example, both x_1 and x_2 have positive coefficents in the objective function. Let us choose to increase x_1.

Q13: What do we have to be careful about when increasing x_1?

A: We have to make sure it doesn't violate its constraints.

Q14: After choosing a variable to increase, we must determine the most limiting constraint. Let us look at the first constraint $x_3 = 4 - 1x_1 - 0x_2$. How much can x_1 increase? (Hint: what does a dictionary form LP require about the constant on the RHS of constraints?)

A: $4 - 1x_1 - 0x_2$ must be greater than or equal to 0.

Q15: Like in **Q14**, determine how much each constraint limits the increase in x_1 and identify the most limiting constraint.

A: The second constraint doesn't limit x_1, the third constraint limits x_1 to a max of $9/2$, and the last constraint limits x_1 to a max of 5. That makes $x_3 = 4 - 1x_1 - 0x_2$ the most limiting constraint here.

If we increase x_1 to 4, note that x_3 will become zero. Earlier, we identified that each dictionary form has a corresponding feasible solution acheived by setting variables on the RHS (and in the objective function) to zero. Hence, since x_3 will become zero, we want to rewrite our LP such that x_3 appears on the RHS. Furthermore, since x_1 is no longer zero, it should now appear on the LHS.

Q16: Rewrite the most limiting constraint $x_3 = 4 - 1x_1 - 0x_2$ such that x_1 appears on the left and x_3 appears on the right.

A: $x_1 = 4 - x_3$

Q17: Using substitution, rewrite the LP such that x_3 appears on the RHS and x_1 appears on the LHS. (Hint: Don't forget the rule about which variables can appear in the objective function)

A:

$$\begin{aligned} \max \quad & z = 20 - 5x_3 - 3x_2 \\ \text{s.t.} \quad & x_1 = 4 - 1x_3 - 0x_2 \\ & x_4 = 6 - 1x_2 \\ & x_5 = 1 - 2x_3 - 1x_2 \\ & x_6 = 3 - 3x_3 - 2x_2 \\ & x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{aligned}$$

Q18: We have now completed an iteration of simplex! What is the corresponding feasible solution of the new LP?

A: Let $x_3 = x_2 = 0$. Then $z = 20$, and $x_1 = 4, x_4 = 6, x_5 = 1, x_6 = 3$.

Now that we have seen an iteration of simplex algebraically, let's use GILP to visualize it! The LP example we have been using is called `LIMITING_CONSTRAINT_2D_LP`. To visualize simplex, we must import a function called `simplex_visual()`.

In [1]:
```
lp = gilp.examples.LIMITING_CONSTRAINT_2D_LP # get the LP example
gilp.simplex_visual(lp, initial_solution=np.array([[0],[0]])).show() #
show the simplex visualization
```

```
---------------------------------------------------------------------
-------
NameError                                 Traceback (most recent cal
l last)
<ipython-input-1-32545b9e92c4> in <module>
----> 1 lp = gilp.examples.LIMITING_CONSTRAINT_2D_LP # get the LP ex
ample
      2 gilp.simplex_visual(lp, initial_solution=np.array([[0],[0]])
).show() # show the simplex visualization

NameError: name 'gilp' is not defined
```

This visualization is much the same as the previous one but we now have an addtional slider which allows you to toggle through iterations of simplex. Furthermore, the corresponding dictionary at every iteration of simplex is shown in the top right. If you toggle between two iterations, you can see the dictionary form for both the previous and next LP at the same time.

Q19: Starting from point (0,0), by how much can you increase x_1 before the point is no longer feasible? Which constraint do you *hit* first? Does this match what you found algebraically?

A: We hit $x_1 = 4$ first, which does match what we found algebraically earlier.

Q20: Which variable will be the next increasing variable and why? (Hint: Look at the dictionary form LP at iteration 1)

A: It will be x_2 because it has the highest most limiting constraint.

Q21: Visually, which constraint do you think is the most limiting constraint? How much can x_2 increase? Give the corresponding feasible solution and its objective value of the next dictionary form LP. (Hint: hover over the feasible points to see information about them.)

A: $x_6 = \ldots$ is the most limiting constraint, because then x_2 can increase up to $3/2$. For some reason, hovering did not give me any information (nothing popped up).

Q22: Move the slider to see the next iteration of simplex. Was your guess from **Q21** correct? If not, describe how your guess was wrong.

A: Yes, my guess was correct with x_2 's constraints.

Q23: Look at the dictionary form LP after the second iteration of simplex. What is the increasing variable? Identify the most limiting constraint graphically and algebraically. Show your work and verify they are the same constraint. In addition, give the next feasible solution and its objective value.

A: x_3 is the variable that's being increased. $x_2 = 1 + 2x_3 - 1x_5$ is the most limiting constraint, which limits x_3 to $1/2$. Graphically, it's apparent because constraint (3) is the upper bound for this iteration of simplex, and the line only goes up from the previous iteration by $1/2$.

Q24: Is the new feasible solution you found in **Q23** optimal? (Hint: Look at the dictionary form LP)

A: It's not optimal, because x_3 still has a positive coefficient in the objective function.

Q25: In **Q21** and **Q23**, how did you determine the most limiting constraint graphically?

A: You can drag the slider and see where it stops between iterations. Otherwise, you can look at the equations as we've done before to see what the most limiting constraint is, and verify it with the slider on the graph.

(BONUS): In 2D, we can increase a variable until we hit a 2D line representing the most limiting constraint. What would be the analagous situation in 3D?

A: We would hit a plane in x_i, x_j that would represent the most limiting constraint.

Part III: Geometrical Interpretation of the Dictionary

We have seen how the simplex algorithm transforms an LP from one dictionary form to another. Each dictionary form has a corresponding dictionary defined by the variables on the LHS of the constraints. Furthermore, each dictionary form has a corresponding feasible solution obtained by setting all non-dictionary variables to 0 and the dictionary variables to the constants on the RHS. In this section, we will explore the geometric interpretation of a dictionary.

In [ ]:
```
lp = gilp.examples.ALL_INTEGER_2D_LP # get LP example
gilp.simplex_visual(lp, initial_solution=np.array([[0],[0]])).show() #
visualize it
```

Recall, we can hover over the corner points of the feasible region. **BFS** indicates the feasible solution corresponding to that point. For example, (7,0,6,9,0) means $x_1 = 7, x_2 = 0, x_3 = 6, x_4 = 9$, and $x_5 = 0$. **B** gives the indices of the variables "being defined" in that dictionary – that is, the variables that are on the LHS of the constraints. For simplicity, we will just say these variables are *in the dictionary*. For example, if **B** = $(1, 3, 4)$, then x_1, x_3, and x_4 are in the dictionary. Lastly, the objective value at that point is given.

Q26: Hover over the point (7,6) where $x_1 = 7$ and $x_2 = 6$. What is the feasible solution at that point ?

A: $x_1 = 7, x_2 = 6, x_3 = 0, x_4 = 3,$ and $x_5 = 0$

We have a notion of *slack* for an inequality constraint. Consider the constraint $x_1 \geq 0$. A feasible solution where $x_1 = 7$ has a slack of 7 in this constraint. Consider the constraint $2x_1 + 1x_2 \leq 20$. The feasible solution with $x_1 = 7$ and $x_2 = 6$ has a slack of 0 in this constraint.

Q27: What is the slack in constraint $1x_1 + 1x_2 \leq 16$ when $x_1 = 7$ and $x_2 = 6$?

A: 3

Q28: Look at the constraint $2x_1 + 1x_2 \leq 20$. After rewriting in dictionary form, the constraint is $x_3 = 20 - 2x_1 - 1x_2$. What does x_3 represent?

A: The slack of the constraint.

Q29: What do you notice about the feasible solution at point (7,6) and the slack in each constraint?

A: They are equal.

It turns out that each decision variable is really a measure of slack in some corresponding constraint!

Q30: If the slack between a constraint and a feasible solution is 0, what does that tell you about the relationship between the feasible solution and constraint geometrically?

A: That we have increased the solution as much as possible; we can't push any further.

Q31: For (7,6), which variables are **not** in the dictionary? For which constraints do they represent the slack? (Hint: The **B** in the hover box gives the indices of the variables in the dicitonary)

A: x_1, x_2, and x_4; they represent the slack in constraints 3, 4, and 5.

Q32: For (7,6), what are the values of the non-dictionary variables? Using what you learned from **Q30**, what does their value tell you about the feasible solution at (7,6)?

A: $x_3 = x_5 = 0$, and that tells me that the feasible solution at (7, 6) is locally optimal (for this dictionary).

Q33: Look at some other corner points with this in mind. What do you find?

A: Their objective values are locally optimal solutions, and the optimal solution is the one of these with the highest objective value.

Now, let's look at a 3 dimensional LP!

In [1]:
```
lp = gilp.examples.ALL_INTEGER_3D_LP # get LP example
gilp.lp_visual(lp).show() # visualize it
```

```
---------------------------------------------------------------------
-------
NameError                                 Traceback (most recent cal
l last)
<ipython-input-1-b24657dc113b> in <module>
----> 1 lp = gilp.examples.ALL_INTEGER_3D_LP # get LP example
      2 gilp.lp_visual(lp).show() # visualize it

NameError: name 'gilp' is not defined
```

Q34: Hover over the point (6,6,2) where $x_1 = 6$, $x_2 = 6$, and $x_3 = 2$. Note which variables are not in the dictionary. Toggle the corresponding constraints on. What do you notice?

A: The planes intersect on this line.

Q35: Look at some other corner points and do as you did in Q34. Do you see a similar pattern? Combining what you learned in Q33, what can you say about the relationship between the variables not in the dictionary at some corner point, and the corresponding constraints?

A: It looks again like the planes intersect on a line containing the corner point. It tells me that the variables in the dictionary describe a point of intersection, where the non-constraint variables are equal.

Q36: What geometric feature do feasible solutions for a dictionary correspond to?

A: These feasible solutions correspond to the region inside the boundaries of the graphed constraints, just as in 2D, but with an added dimension.

Part IV: Choosing an Increasing Variable

The first step in an iteration of simplex is to choose an increasing variable. Sometimes, there are multiple options since multiple variables have a positive coefficent in the objective function. Here, we will explore what this decison translates to geometrically.

In this section, we will use a special LP commonly referred to as the Klee-Minty Cube.

$$\begin{aligned} \max \quad & 4x_1 + 2x_2 + x_3 \\ \text{s.t.} \quad & x_1 \leq 5 \\ & 4x_1 + x_2 \leq 25 \\ & 8x_1 + 4x_2 + x_3 \leq 125 \\ & x_1, x_2, x_3 \geq 0. \end{aligned}$$

Furthermore, we will use an optional parameter called `rule` for the `simplex_visual()` function. This rule tells simplex which varaible to choose as an increasing variable when there are multiple options.

In [ ]:
```
klee_minty = gilp.examples.KLEE_MINTY_3D_LP
```

In [ ]:
```
gilp.simplex_visual(klee_minty, rule='dantzig', initial_solution=np.ar
ray([[0],[0],[0]])).show()
```

Q37: Use the iteration slider to examine the path of simplex on this LP. What do you notice?

A: The line goes along the outer rim of the shape, in a zig-zag fashion.

Above, we used a rule proposed by Dantzig. In this rule, the variable with the *largest* positive coefficient in the objective function enters the dictionary. Go through the iterations again to verify this.

Let us consider another rule proposed by Bland, a professor here at Cornell. In his rule, of the variables with positive coefficents in the objective function, the one with the smallest index enters. Let us examine the path of simplex using this rule! Again, look at the dictionary form LP at every iteration.

```
In [2]: gilp.simplex_visual(klee_minty, rule='bland', initial_solution=np.arra
        y([[0],[0],[0]])).show()
```

```
---------------------------------------------------------------------
-------
NameError                                 Traceback (most recent cal
l last)
<ipython-input-2-bebcadccb214> in <module>
----> 1 gilp.simplex_visual(klee_minty, rule='bland',
initial_solution=np.array([[0],[0],[0]])).show()

NameError: name 'gilp' is not defined
```

Q38: What is the difference between the path of simplex using Dantzig's rule and Bland's rule?

A: It takes a much shorter path along the shape, seeming to choose the shortest edge at each point.

Can you do any better? By setting `rule='manual'`, you can choose the entering variable explicitly at each simplex iteration.

Q39: Can you do better than 5 iterations? How many paths can you find? (By my count, there are 7)

A: I found a solution in one iterations, which was just to do 3 first.

```
In [ ]: gilp.simplex_visual(klee_minty,rule='manual', initial_solution=np.arra
        y([[0],[0],[0]])).show()
```

Q40: What does the choice of increasing variable correspond to geometrically?

A: It corresponds to traveling in that direction on the path to the optimal solution.

Q41: Are there any paths you could visualize taking to the optimal solution that `rule='manual_select'` prevented you from taking? If yes, give an example and explain why it is not a valid path for simplex to take. (Hint: Look at the objective value after each simplex iteration.)

A: Going 1, 2, 3, 4, 5 was not allowed but when I did it anyway, I noticed that x_4 did not have the highest maximum constraint in the objective function for iteration 4. That's why it's not a valid path to take; it violates the constraints.

Part V: Creating LPs in GILP (Optional)

We can also create our own LPs! Let us create the following LP.

$$\begin{aligned} \max \quad & 3x_1 + 2x_2 \\ \text{s.t.} \quad & 2x_1 + 1x_2 \leq 6 \\ & 0x_1 + 1x_2 \leq 2 \\ & x_1, x_2 \geq 0 \end{aligned}$$

We will create this LP by specifying 3 arrays of coefficents. We define the NumPy arrays `A`, `b`, and `c` and then pass them to the `LP` class to create the LP.

In [ ]:
```
A = np.array([[2,1],  # LHS constraint coefficents
              [0,1]])
b = np.array([6,2])  # RHS constraint coefficents
c = np.array([3,2])  # objective function coefficents
lp = gilp.LP(A,b,c)
```

Let's visualize it!

In [ ]:
```
gilp.lp_visual(lp).show()
```

... and solve it!

In [ ]:
```
gilp.simplex_visual(lp, initial_solution=np.array([[0],[0]])).show()
```