Lesson 8 Supplement: Excel and Multiple Optimal Solutions

1 Today...

- read and write to Excel from Python using xlwings
- test for multiple optimal soluitons

2 Main lesson

The main lesson for today is in a Jupyter notebook. We will just collect some useful ideas here for easy access.

3 xlwings: commonly used commands

- Import xlwings:

```
import xlwings as xw
```

- Open an existing Excel workbook called "my_workbook.xlsx":

```
wb = xw.Book('my_workbook.xlsx')
```

Note: This only works if the Python file (Jupyter notebook) and Excel file are in the same directory. Otherwise, we need the full path to the Excel file in quotes.

- Open a sheet called "my_sheet" in the Workbook called "wb"

```
sh = wb.sheets('my_sheet')
```

- Add a sheet called "my_sheet" in the Workbook called "wb"

```
sh = wb.sheets.add('my_sheet')
```

- To run the previous two commands without errors, we need to know whether or not the sheet "my_sheet" exists in Workbook "wb". This code allows me to try to add the sheet, and if that doesn't the sheet already exists and I can just open it.

```
try:                                        # try to add sheet
    sh = wb.sheets.add('my_sheet')
except:                                     # otherwise, open the sheet
    sh = wb.sheets('my_sheet')
```

- Read the data in from cells A1 to D15 in sheet "sh":

```
my_data = sh.range('A1:D15').value
```

Note: `my_data` will be a list of 15 lists (rows 1-15), each with 4 elements (columns A-D).

- Write a list (of lists) called 'my_double_list' beginning at cell A1 and forming a rectangle of the same dimensions:

```
sh.range('A1').value = my_double_list
```

4 Other Python reminders

- We can use a `range()` object in `for` loops. For example,

```
for i in range(3):
```

loops over $i = 0, 1, 2$.

- Recall that list indexing always starts at 0.
- To index over the list of lists,

```
mylist = [[a,b,c],[d,e,f]],
```

the item at position

```
mylist[0][1]
```

is b (list 0, position 1).

5 Multiple Optimal Solutions

Sometimes we might want to explore different optimal solutions to the same integer or linear program. This means, a solution that has the same objective value, but different values for the variables. In order to accomplish this, we can take the following steps:

1. Solve the model the first time
2. Add a constraint to the model that prevents the same optimal solution from occuring
   - We have to be careful not to make the constraint too restrictive, or it may prevent other optimal solutions from occuring.
3. Solve the model again and see what happens
   - If we get another optimal solution with the same objective value, then the model has multiple optimal solutions. We can add yet another constraint to check for a third optimal solution, etc.
   - If we get infeasible or an optimal solution with a worse objective value, then there was only the one optimal solution.
   - Is it possible to add a constraint to the model and get a better better optimal objective value?