Homework Assignment 1

1CV00 - Fundamentals of Operations Management

February 9, 2021

Please note

- The grade for this assignment determines 15% of your final grade.
- You need to work on the assignments in a groups of two students (you may also decide to work by yourself).
- You should submit your python files as GroupName.py (Write Group and your Group Number, such as Group130.py) by Tuesday 2 March, 23:59.
- You will have a function for each question in the GroupName.py. You have to call the first function *question1* and the second function *question2*. We will use unit testing to test your functions. Therefore, it is very important that you name your python code and the functions as stated. If our unit testing does not work on your file due to wrong naming (or other reasons), you will get a 0 grade.
- You may use the GroupName.py file on CANVAS as your template.
- A simplified version of the testing file that will be used in grading is also shared on CANVAS. However, only the first parameter sets for each question are provided and the remaining eight parameter sets (four for each) are not. You need to prepare your code to be able to run for different parameter sets. You can use this file to make sure that your code will not give an unexpected error during unit testing. Do not forget to change the path in the testing script to the path of the homework file, GroupName.py.
- Hint 1: Don't simply start programming, but first think about it. In general, this means that you design the software / algorithm you want to program, and then translate it to actual code that you implement. In this course, it means that you should first really understand the problem with its objective function and constraints, then write down the LP / MILP model on paper, and then translate that to PuLP. When asking questions on your model in PuLP, we may very well ask you to show us your model on paper. If you can't do that, you should probably work on that first, before you go back to PuLP.

- Hint 2: In both questions you need to use a so-called *big M*. There is a lot you can find on this online, for example in the part on Fixed Charges and Set-up Costs at `https://drive.google.com/file/d/1HHV3xFgoGsyuhY06GoDTxI9F3Z-02Sxc/edit`. The key idea is that you can link something with binary values, i.e., a value of either 0 or 1, to something with more possible values. For example: In the facility location problem, we only need to open a facility and pay opening costs if there are demands for that facility. In other words, we pay either 0 or 1 times the opening costs: 0 if there is no demand at the facility and 1 if there is more than 0 demand, which can be any value. If the total demand for the facility is D, then we may introduce F as the fixed costs to open the facility and O as the variable that indicates whether or not we open the facility. We may then add a constraint that states $D \leq M \times O$, while we add to the cost function $O \times F$.

Question 1

We consider a lot sizing problem, i.e., we need to determine in which period(s) to produce and how much to produce in any period in which we produce. We consider a set of periods $T = \{1, 2, \ldots, H\}$, with H being the time horizon. We further consider a set of products $P = \{1, 2, \ldots, N\}$, with N being the number of products. For each period $t \in T$ and each product $p \in P$, we are given the demand $D(t, p)$, which we have to fulfill in that period. We further know the fixed setup costs A and the inventory holding costs h per unit of inventory per period for inventory on-hand at the end of the period. In addition, there are product specific setup costs $a(p)$. That means that if in a certain period we produce products p_1 and p_2, we incur fixed setup costs $A + a_1(p_1) + a_2(p_2)$. (Notice that there is unlimited production capacity, as in the standard lot sizing problem.)

- The function needs to have 7 inputs (Time horizon: H, number of products: N, holding cost: h, fixed setup cost: A, product specific setup cost list: a, demand list of lists: D, and big M: M)
- The function needs to return the objective value.

```
def question1(H,N,h,A,a,D,M):

    #YOUR CODE HERE

    return #RETURN THE OBJECTIVE VALUE
```

- You can use below values to test your function. The output must be 195.5.

```
H=10
N=2
h=0.5
A=10
a=[5,9]
D=[[12,10,13,14,13,15,17,20,19,14],[9,17,11,20,10,16,13,22,15,12]]
```

```
M=1000
question1(H,N,h,A,a,D,M)
```

Question 2

We consider a lot sizing problem, i.e., we need to determine in which period(s) to produce and how much to produce in any period in which we produce. We consider a set of periods $T = \{1, 2, \ldots, H\}$, with H being the time horizon. We further consider a set of products $P = \{1, 2, \ldots, N\}$, with N being the number of products. For each period $t \in T$ and each product $p \in P$, we are given the demand $D(t, p)$, which we have to fulfill in that period. We further know the fixed setup costs A and the inventory holding costs h per unit of inventory per period for inventory on-hand at the end of the period. In addition, there are product specific setup costs $a(p)$. That means that if in a certain period we produce products p_1 and p_2, we incur fixed setup costs $A + a_1(p_1) + a_2(p_2)$. In addition to this standard lot sizing problem, the factory has a limited production capacity C on the total production amount. Moreover, demand values are sometimes entered as negative numbers. Such values must be taken as zero, since negative demand would not be possible.

- The function needs to have 8 inputs (Time horizon: H, number of products: N, holding cost: h, fixed setup cost: A, product specific setup cost list: a, demand list of lists: D, factory capacity: C, and big M: M)
- The function needs to return the objective value.

```
def question2(H,N,h,A,a,D,M,C):

    #YOUR CODE HERE

    return #RETURN THE OBJECTIVE VALUE
```

- You can use below values to test your function. The output must be 224.0.

```
H=10
N=2
h=0.5
A=10
a=[5,9]
D=[[12,-10,13,14,13,15,17,20,19,14],[-9,17,11,20,10,16,13,22,15,12]]
M=1000
C=35
question2(H,N,h,A,a,D,M,C)
```