ASEAN Journal of Science and Engineering Education

Journal homepage: http://ejournal.upi.edu/index.php/AJSEE/

Teaching Programming to Chemical Engineering Students

Riezqa Andika[1], Zulfan Adi Putra[2*]*

[1]Department of Chemical Engineering, Universitas Indonesia, Depok 16424, Indonesia
[2]PETRONAS Group Technical Solutions (Process Simulation and Optimization), Kuala Lumpur 50088, Malaysia
[*]Correspondence: E-mail: r.andika@che.ui.ac.id; zulfan.adiputra@petronas.my

ABSTRACTS

Programming (including coding, data science, and artificial intelligence) is one of the important top skills for chemical engineering undergraduate students, especially in the current era of Industrial Revolution 4.0 (IR 4.0). Despite its importance, there is almost no dedicated course for programming in the traditional chemical engineering curriculum in many universities. In this article, a future chemical engineering curriculum that includes a programming course is discussed and its integration into traditional courses is also proposed. This study proposes to use three software i.e., Microsoft Excel, GAMS, and Python to make students comfortable to programming so they can learn further by themselves. A representative case of classic logistics problem is presented as a case study. It is expected that this teaching strategy and the proposed example will improve students' cognition in programming.

ARTICLE INFO

Article History:
Submitted/Received 29 Apr 2021
First revised 21 Jun 2021
Accepted 19 Jul 2021
First available online 21 Jul 2021
Publication date 01 Ma 2022

Keyword:
Engineering,
GAMS,
Microsoft Excel,
Programming,
Python

1. INTRODUCTION

Programming skill is essential in the current era of Industrial Revolution 4.0 (IR 4.0). A study by Kamaruzaman et al. (2019) concludes that programming is one of the skill gaps between engineering and skills needed in the IR 4.0. This is an important matter because graduates from higher education institutions may not have this necessary skill required by the industry. At the same time, this matter can propagate the skill gaps and unemployment issues among engineering graduates.

In the recent years, the application of programming skill in many fields including engineering is shifted towards data science and Artificial Intelligence (AI). Several higher education institutions start to teach data science and AI to their students. One of them is TU Delft in the Netherlands that plans to educate all their students on AI.

Currently, open-source programming languages such as Python, Julia, and R are popular among academia, mainly for research and industry. Although they are widely used, these programming languages have a steep learning curve, especially for students who never work with Matlab or VBA (Visual Basic for Applications).

This article discusses the integration of programming and/or data science to the current chemical engineering undergraduate curriculum. As an example, one problem is presented and solved using Microsoft Excel, GAMS, and Python in this paper. This example can be broadly applied to any chemical engineering course such as numerical computation, chemical process simulation, or chemical engineering modelling. The discussion in this article is expected to be a useful contribution to restructuring the future chemical engineering undergraduate curriculum and the integration of programming into the traditional curriculum.

2. PROGRAMMING AND CHEMICAL ENGINEERING CURRICULUM

A study by the World Chemical Engineering Council on digitalization in chemical engineering education, particularly in the Asia-Pacific region, concluded that the future engineering undergraduate program curriculum should cover today's chemical engineering content accompanied by a graduate course on digital subjects such as mathematical modelling, programming tools and software, digital twins, algorithms and data structures, control systems and advanced sensors, data analytics and predictive maintenance, cyber security, and digital business models (Feise & Schaer, 2021).

The aforementioned study is supported by another study by Feise & Schaer (2021). They concluded that higher education institutions should regularly develop their teaching methodologies and content to meet changes in the materials and energy transformation industries. According to their study, numerical methods and AI are getting more and more important. Future engineers will not only need to be comfortable with commercial codes but also able to develop new ones.

A study by dos Santos et al. (2018) proposes a curriculum as depicted in **Figure 1** where "Advanced Programming for Chemical Process" course is implemented with two basic characteristics which are taught at the end of the undergraduate year and as elective course. The same proposal was also expressed that "Data Science for Engineers" course is taught at the end of the undergraduate year as an elective course (**Figure 2**).

If the current curriculum does not have a dedicated programming or data science course and it cannot be restructured, we propose an easy integration to teach programming inserted into the existing courses.

Open

DOI: http://dx.doi.org/10. 17509/xxxxt.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

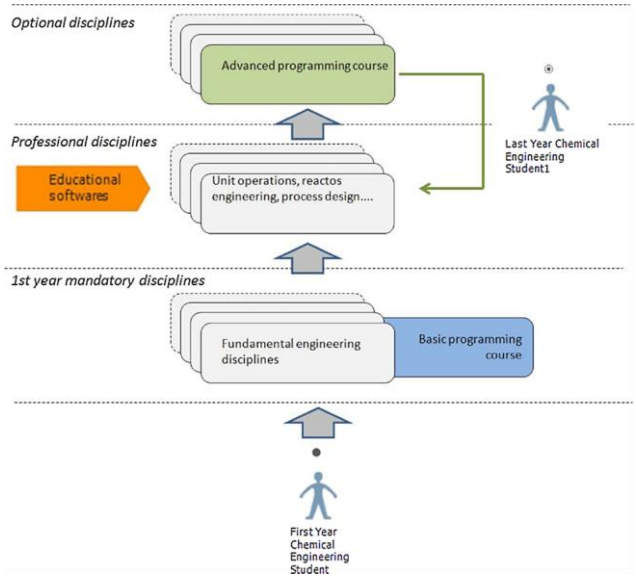

Figure 1. The chemical engineering program and the time when programming/ software course is normally offered (dos Santos et al., 2018).

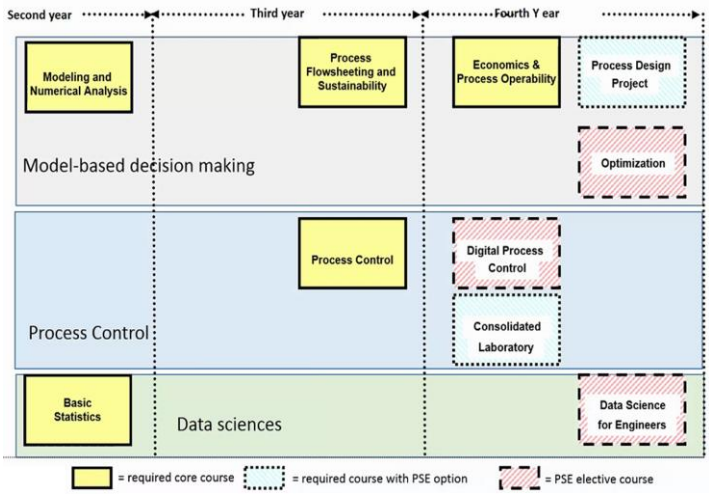

Figure 2. A feasible plan for basic and advanced Process Systems Engineering (PSE) courses.

Usually, programming is part of Process Systems Engineering (PSE) courses in the graduate program. Therefore, it is easier to integrate programming in PSE courses. For example, in chemical engineering modelling or optimization courses, students are taught linear programming in one session using Microsoft Excel, GAMS, and Python. As familiarity with a particular programming language takes time, students can begin to introduce themselves with the logical reasoning by using different program.

Microsoft Excel is proposed as the first software to use because it is intuitive software that it only takes relatively short time for students to be able to use it adequately. Thus, it does not take a long time to teach the software. Microsoft Excel can be used to solve simple to medium level numerical computation problems. Secondly, GAMS is selected because it is a high-level modelling system for mathematical programming and optimization. It means that the syntax is simple and easy to understand. Lastly, Python is chosen because it is one of the

DOI: http://dx.doi.org/10.17509/xxxx.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

most popular and widely used programs. Although it is the hardest among the three programs, it can be used further to deal with data science. In the next chapter, we will use a modified classic problem and show how to solve a simple case study using these three programs.

3. PROBLEM

The problem was modified from a classic logistics problem taken from posted in the GAMS website (https://www.gams.com/). The original problem was solved. The classic problem was modified to give a clear picture of the problem to Indonesian students. The modified problem is as follows.

An operation director for company P wants to minimize the cost of transporting goods from two factories in Tegal and Tasikmalaya to three markets in Jakarta, Kebumen, and Bandung (**Figure 3**). The transportation cost per item per kilometer (f) is IDR 2,000. Information on distance, market demand, and inventory at the three factories is summarized in **Table 1**. Solutions to find the minimum transportation cost are presented as follows.

4. SOLUTIONS

4.1. Microsoft Excel

The problem can be solved using Microsoft Excel by employing a similar method as carried out by Mohamad et al. (2021). First, students need to make a problem definition such as decision variable, constraint, and objective function as depicted in **Figures 4a** and **4b**. After that, the students can use a solver to solve the problem using one of three available solvers, Simplex LP, because it is a linear programming problem. This step is depicted in **Figure 5.**

Figure 3. Locations of factories and markets of the problem.

Table 1. Distance between factories and markets.

Distance (km)		Market			Supply
		Jakarta	Kebumen	Bandung	
Plant	Tegal	250	170	160	350
	Tasikmalaya	250	180	140	600
Demand		325	300	275	

Open

DOI: http://dx.doi.org/10. 17509/xxxxt.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

	A	B	C	D	E	F	G
1	**Logistics**						
2							
3							
4	Tegal to Jakarta	0	goods	Decision Variable			
5	Tegal to Kebumen	0	goods	Decision Variable			
6	Tegal to Bandung	0	goods	Decision Variable			
7	Tasikmalaya to Jakarta	0	goods	Decision Variable			
8	Tasikmalaya to Kebumen	0	goods	Decision Variable			
9	Tasikmalaya to Bandung	0	goods	Decision Variable			
10							
11	**Constraint**			Maximum			
12	Supply in Tegal	0	<=	350	goods		
13	Supply in Tasikmalaya	0	<=	600	goods		
14	Demand in Jakarta	0	=	325	goods		
15	Demand in Kebumen	0	=	300	goods		
16	Demand in Bandung	0	=	275	goods		
17							
18	**Transportation Cost**			**Total Transportation Cost**			
19	Tegal to Jakarta	IDR 0		IDR 0	Objective Function		
20	Tegal to Kebumen	IDR 0					
21	Tegal to Bandung	IDR 0					
22	Tasikmalaya to Jakarta	IDR 0					
23	Tasikmalaya to Kebumen	IDR 0					
24	Tasikmalaya to Bandung	IDR 0					
25							

Figure 4a. Design of Microsoft Excel file to solve the problem.

	A	B	C	D	E	F	G
1	**Logistics**						
2							
3							
4	Tegal to Jakarta	0	goods	Decision Variable			
5	Tegal to Kebumen	0	goods	Decision Variable			
6	Tegal to Bandung	0	goods	Decision Variable			
7	Tasikmalaya to Jakarta	0	goods	Decision Variable			
8	Tasikmalaya to Kebumen	0	goods	Decision Variable			
9	Tasikmalaya to Bandung	0	goods	Decision Variable			
10							
11	**Constraint**			Maximum			
12	Supply in Tegal	=SUM(B4:B6)	<=	350	goods		
13	Supply in Tasikmalaya	=SUM(B7:B9)	<=	600	goods		
14	Demand in Jakarta	=(B4+B7)	=	325	goods		
15	Demand in Kebumen	=B5+B8	=	300	goods		
16	Demand in Bandung	=B6+B9	=	275	goods		
17							
18	**Transportation Cost**			**Total Transportation Cost**			
19	Tegal to Jakarta	=B4*250*2000		=SUM(B19:B24)	Objective Function		
20	Tegal to Kebumen	=B5*170*2000					
21	Tegal to Bandung	=B6*180*2000					
22	Tasikmalaya to Jakarta	=B7*250*2000					
23	Tasikmalaya to Kebumen	=B8*180*2000					
24	Tasikmalaya to Bandung	=B9*140*2000					
25							

Figure 4b. Microsoft Excel formula to solve the problem.

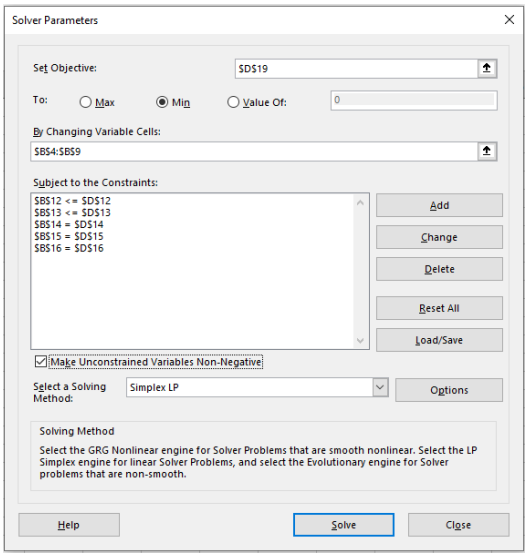

Figure 5. Solver to solve the problem.

DOI: http://dx.doi.org/10.17509/xxxx.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

4.2. GAMS

In GAMS, a model is a collection of statements delivered in the GAMS Language. Typically, a model needs to have sets, data, variables, equations, and model and solution statements. Different from Microsoft Excel, GAMS can solve complex problems such as renewable energy system by using a wide range of available solvers (Andika et al., 2019). By writing in GAMS, students are expected to be more comfortable with programming. The code to build GAMS model is shown as follows.

```
SET
   i 'factories'      / Tegal, Tasikmalaya /
   j 'markets'        / Jakarta, Kebumen, Bandung /;

PARAMETER
   a(i) 'capacity of factory i in cases'
        / Tegal       350
          Tasikmalaya 600 /

   b(j) 'demand at market j in cases'
        / Jakarta     325
          Kebumen     300
          Bandung     275 /;

TABLE
   d(i,j) 'distance in kilometer'
                Jakarta  Kebumen  Bandung
   Tegal          250      170      180
   Tasikmalaya    250      180      140;

SCALAR
   f 'freight in rupiah per case per kilometer' / 2000 /;

PARAMETER
   c(i,j) 'transport cost in rupiah per case';
   c(i,j) = f*d(i,j);

VARIABLE
   x(i,j) 'shipment quantities in cases'
```


DOI: http://dx.doi.org/10. 17509/xxxxt.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

```
   z      'total transportation costs in rupiah';

POSITIVE VARIABLE
   x;

EQUATION
   cost      'define objective function'
   supply(i) 'observe supply limit at factory i'
   demand(j) 'satisfy demand at market j';

cost..       z =e= sum((i,j), c(i,j)*x(i,j));
supply(i).. sum(j, x(i,j)) =l= a(i);
demand(j).. sum(i, x(i,j)) =g= b(j);


Model transport / all /;

Solve transport using lp minimizing z;

Display x.l, x.m ;
```

4.3. Python

To solve the problem by using Python, the students need to install Pyomo (Pyomo, 2021). Pyomo is a Python-based package that supports a diverse set of optimization capabilities for formulating, solving, and analyzing optimization models. The code to build the Python model and its comparison with GAMS are shown as follows.

```
# Import
from pyomo.environ import *

# Creation of a Concrete Model
model = ConcreteModel()

## Define sets ##
#  Sets
#       i   factories        / Tegal, Tasikmalaya /
#       j   markets          / Jakarta, Kebumen, Bandung / ;
model.i = Set(initialize=['Tegal','Tasikmalaya'], doc='Factories')
model.j = Set(initialize=['Jakarta','Kebumen', 'Bandung'], doc='Markets')

## Define parameters ##
#   Parameters
```

Open

DOI: http://dx.doi.org/10.17509/xxxx.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

```
#       a(i)  capacity of factory i in cases
#         /    Tegal       350
#              Tasikmalaya 600  /
#       b(j)  demand at market j in cases
#         /    Jakarta     325
#              Kebumen     300
#              Bandung     275  / ;
model.a = Param(model.i, initialize={'Tegal':350,'Tasikmalaya':600},
doc='Capacity of factory i in cases')
model.b = Param(model.j,
initialize={'Jakarta':325,'Kebumen':300,'Bandung':275}, doc='Demand at
market j in cases')
#  Table d(i,j)  distance kilometer
#                     Jakarta        Kebumen      Bandung
#      Tegal            250            170          180
#      Tasikmalaya      250            180          140  ;
dtab = {
    ('Tegal',     'Jakarta') : 250,
    ('Tegal',     'Kebumen') : 170,
    ('Tegal',     'Bandung') : 180,
    ('Tasikmalaya','Jakarta'): 250,
    ('Tasikmalaya','Kebumen'): 180,
    ('Tasikmalaya','Bandung'): 140,
    }
model.d = Param(model.i, model.j, initialize=dtab, doc='Distance in
kilometer')
#  Scalar f  freight in rupiah per case per kilometer  /2000/ ;
model.f = Param(initialize=2000, doc='Freight in rupiah per case per
kilometer')
#  Parameter c(i,j)  transport cost in rupiah per case ;
#            c(i,j) = f * d(i,j) ;
def c_init(model, i, j):
  return model.f * model.d[i,j]
model.c = Param(model.i, model.j, initialize=c_init, doc='Transport cost in
rupiah per case')

## Define variables ##
#  Variables
#       x(i,j)  shipment quantities in cases
#       z       total transportation costs in rupiah ;
#  Positive Variable x ;
model.x = Var(model.i, model.j, bounds=(0.0,None), doc='Shipment quantities
in case')

## Define constraint ##
# supply(i)    observe supply limit at factory i
# supply(i) .. sum (j, x(i,j)) =l= a(i)
def supply_rule(model, i):
  return sum(model.x[i,j] for j in model.j) <= model.a[i]
model.supply = Constraint(model.i, rule=supply_rule, doc='Observe supply
limit at factory i')
# demand(j)   satisfy demand at market j ;
# demand(j) .. sum(i, x(i,j)) =g= b(j);
def demand_rule(model, j):
  return sum(model.x[i,j] for i in model.i) >= model.b[j]
model.demand = Constraint(model.j, rule=demand_rule, doc='Satisfy demand at
market j')

## Define Objective and solve ##
#  cost        define objective function
#  cost ..     z  =e=  sum((i,j), c(i,j)*x(i,j)) ;
#  Model transport /all/ ;
```

Open

DOI: http://dx.doi.org/10. 17509/xxxxt.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

```
#  Solve transport using lp minimizing z ;
def objective_rule(model):
  return sum(model.c[i,j]*model.x[i,j] for i in model.i for j in model.j)
model.objective = Objective(rule=objective_rule, sense=minimize,
doc='Define objective function')


## Display of the output ##
# Display x.l, x.m ;
def pyomo_postprocess(options=None, instance=None, results=None):
  model.x.display()
```

4.4. Results

The results of all simulations are the same because the problem is meant to be solved with simple linear programming. The minimum value of the objective function obtained is IDR 341,500,000. The value of decision variables is shown in **Table 2**.

Table 2. Optimal value of supply of goods from factories to markets.

		Market		
		Jakarta	**Kebumen**	**Bandung**
Plant	**Tegal**	50	300	0
	Tasikmalaya	275	0	275

5. CONCLUSION

A profound comprehension of programming is essential for chemical engineering undergraduate students in the IR 4.0 era. This paper is intended to highlight the integration of programming in the chemical engineering curriculum. A classic logistics problem was taken as an example. This example is expected to facilitate students' understanding of the nature of programming and its underlying mathematical models and logic along with the means to solve them.

6. AUTHORS' NOTE

The authors declare that there is no conflict of interest regarding the publication of this article. The authors have confirmed that the paper is free of plagiarism.

7. REFERENCES

Kamaruzaman, F.M., Hamid, R., Mutalib, A.A., and Rasul, M.S. (2019). Comparison of engineering skills with IR 4.0 Skills. *International Journal of Online and Biomedical Engineering*, *15*(10), 15-28.

Feise, H.J., and Schaer, E. (2021). Mastering digitized chemical engineering. *Education for Chemical Engineers*, *34*, 78-86.

dos Santos, M.T., Vianna Jr., A.S., and Le Roux, G.A.C. (2018). Programming skills in the industry 4.0: are chemical engineering students able to face new problems?. *Education for Chemical Engineers*, *22*, 69-76.

DOI: http://dx.doi.org/10.17509/xxxx.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815

Mohamad, M.A., Putra, Z.A., Bilad, M.R., Nordin, N.A.H.M., and Wirzali, M.D.H. (2021). An excel based tool development for scheduling optimization. *ASEAN Journal of Science and Engineering Education*, *1*(1), 7-14.

Andika, R., Kim, Y., Yun, C.M., Yoon, S.H. Lee, M. (2019). Design of a renewable energy system with battery and power-to-methanol unit. *Korean Journal of Chemical Engineering*, *36*, 12-20.

Open

DOI: http://dx.doi.org/10. 17509/xxxxt.vxix
p- ISSN 2775-6793 e- ISSN 2775-6815