HW 10

Q1.1

$$
\begin{aligned}
&\min x_1 + x_2 + x_3 \\
&\text{s.t.} \begin{bmatrix} 10 \\ 0 \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 7 \\ 0 \end{bmatrix} x_2 + \begin{bmatrix} 0 \\ 0 \\ 5 \end{bmatrix} x_3 = \begin{bmatrix} 15 \\ 30 \\ 20 \end{bmatrix} \\
&x_1, x_2, x_3 \geqslant 0 \\
&\bar{x}_1 = 1.5 \\
&\bar{x}_2 = 4.286 \\
&\bar{x}_3 = 4 \\
&B = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 5 \end{bmatrix} \quad c_B = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \\
&B^{-1} = \begin{bmatrix} 1/10 & 0 & 0 \\ 0 & 1/7 & 0 \\ 0 & 0 & 1/5 \end{bmatrix} \\
&\hat{y}^\top = c_B^\top B^{-1} = [1/10, 1/7, 1/5]
\end{aligned} \tag{1}
$$

Q1.2

In [27]:

```python
import cvxpy as cp
import numpy as np
import array_to_latex as a2l
from IPython.display import display, Markdown

n = 3
x = cp.Variable(n)
b = np.array([15, 30, 20])
cb = np.array([1, 1, 1])
A = np.array([
    [10, 0, 0],
    [0, 7, 0],
    [0, 0, 5]
])

prob = cp.Problem(
    cp.Minimize(cb.T @ x),
    [
        A @ x == b,
        x >= np.zeros(n)
    ]
)

prob.solve()

y_hat = cb.T @ np.linalg.inv(A)
x_bar = x.value
```

```
print("Optimal solution x:")
print("-----------------------------")
display(Markdown(a2l.to_ltx(x.value, print_out=False)))

print("Optimal basis B:")
print("-----------------------------")
display(Markdown(a2l.to_ltx(A, print_out=False)))

print("Optimal basis inverse B^-1:")
print("-----------------------------")
display(Markdown(a2l.to_ltx(np.linalg.inv(A), print_out=False)))

print("Optimal dual solution y_hat:")
print("-----------------------------")
display(Markdown(a2l.to_ltx(y_hat, print_out=False)))
```

```
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 1.50 & 4.29 & 4.00 \end{bmatrix}$$

```
Optimal basis B:
-----------------------------
```

$$\begin{bmatrix} 10.00 & 0.00 & 0.00 \\ 0.00 & 7.00 & 0.00 \\ 0.00 & 0.00 & 5.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} 0.10 & 0.00 & 0.00 \\ 0.00 & 0.14 & 0.00 \\ 0.00 & 0.00 & 0.20 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.10 & 0.14 & 0.20 \end{bmatrix}$$

Q1.3

The knapsack problem:

$$\begin{aligned} \hat{Z} = \max \frac{1}{10}a_1 + \frac{1}{7}a_2 + \frac{1}{5}a_3 \\ \text{s.t. } 7a_1 + 11a_2 + 16a_3 \leq 80 \\ a_1, a_2, a_3 \geq 0, \text{ integers} \end{aligned} \tag{2}$$

Q1.4

In [28]:

```
a = cp.Variable(n, integer=True)
w = np.array([7, 11, 16])
```

```python
W = 80

prob_cost = cp.Problem(
    cp.Maximize(y_hat.T @ a),
    [
        w.T @ a <= W,
        a >= np.zeros(n)
    ]
)

prob_cost.solve()
new_pattern = a.value
Z_hat = round(prob_cost.value, 2)
print(f"Z_hat = {round(Z_hat, 2)}")
print(f"The minimum reduced cost is negative and equal to {round(1 - Z_hat, 2)}")
print("The new generated pattern is:")
display(Markdown(a2l.to_ltx(new_pattern, print_out=False)))
```

```
Z_hat = 1.1
The minimum reduced cost is negative and equal to -0.1
The new generated pattern is:
```

$$\begin{bmatrix} 11.00 & 0.00 & 0.00 \end{bmatrix}$$

Q1.5

In this part, I need to create a function with the column generation algorithm. I will print the previous solution but the algorithm should go all the way up to the optimal.

In [29]:

```python
from math import isclose

def cutting_stock(roll_width: float,
                  width_types: np.ndarray,
                  quantities: np.ndarray,
                  initial_pattern: np.ndarray):
    a_tol = 1e-5
    sep = "------------------------------"
    Z_hat = 10
    i = 0
    A = initial_pattern
    while not isclose(1 - Z_hat, 0.0, abs_tol=a_tol):
        # Print current iteration
        i += 1
        print(f"Iteration {i}")
        print(sep)
        # Solve the RMP
        # Get the number of variables from the columns in the basis
        n = A.shape[1]
        x = cp.Variable(n)
        cb = np.ones(n)

        prob = cp.Problem(
            cp.Minimize(cb.T @ x),
            [
                A @ x == b,
                x >= np.zeros(n)
            ]
```

```
)

prob.solve()
x_bar = x.value
sols_nonzero = ~np.isclose(x_bar, np.zeros(n), atol=a_tol)
# Getting the optimal basis based on the non-zero solutions
B = A[:, sols_nonzero]
B_inv = np.linalg.inv(B)
y_hat = np.ones(B.shape[0]).T @ B_inv

# Print required iteration information
print("Optimal solution x:")
print(sep)
display(Markdown(a2l.to_ltx(x_bar, print_out=False)))

print("Optimal basis B:")
print(sep)
display(Markdown(a2l.to_ltx(B, print_out=False)))

print("Optimal basis inverse B^-1:")
print(sep)
display(Markdown(a2l.to_ltx(B_inv, print_out=False)))

print("Optimal dual solution y_hat:")
print(sep)
display(Markdown(a2l.to_ltx(y_hat, print_out=False)))

# Solving the pricing problem
n_pricing = len(quantities)
a = cp.Variable(n_pricing, integer=True)
w = width_types
W = roll_width

prob_cost = cp.Problem(
    cp.Maximize(y_hat.T @ a),
    [
        w.T @ a <= W,
        a >= np.zeros(n_pricing)
    ]
)

prob_cost.solve()

new_pattern = a.value
Z_hat = prob_cost.value
print(f"Z_hat = {round(Z_hat, 2)}")
if 1 - Z_hat < 0:
    print(f"The minimum reduced cost is negative and equal to "
          f"{round(1 - Z_hat, 2)}")
    print("The new generated pattern is:")
    display(Markdown(a2l.to_ltx(new_pattern, print_out=False)))
    print(sep)
    print(sep)
    print(sep)
    # Add the new pattern to the list
    A = np.c_[A, new_pattern]
else:
    print(f"The minimum reduced cost is non-negative and equal to "
          f"{round(1 - Z_hat, 2)}")
    print("The column generation algorithm has finished")
```

```python
    print("Finished")

    return x_bar, B, A
```

In [30]:

```python
w = np.array([7, 11, 16])  # Types of width
b = np.array([15, 30, 20])  # The quantities of the width types
W = 80  # The width of the big roll
# The start point
A = np.array([
    [10, 0, 0],
    [0, 7, 0],
    [0, 0, 5]
])
x_opt, B_opt, A_opt = cutting_stock(W, w, b, A)
```

```
Iteration 1
-----------------------------
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 1.50 & 4.29 & 4.00 \end{bmatrix}$$

```
Optimal basis B:
-----------------------------
```

$$\begin{bmatrix} 10.00 & 0.00 & 0.00 \\ 0.00 & 7.00 & 0.00 \\ 0.00 & 0.00 & 5.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} 0.10 & 0.00 & 0.00 \\ 0.00 & 0.14 & 0.00 \\ 0.00 & 0.00 & 0.20 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.10 & 0.14 & 0.20 \end{bmatrix}$$

```
Z_hat = 1.1
The minimum reduced cost is negative and equal to -0.1
The new generated pattern is:
```

$$\begin{bmatrix} 11.00 & 0.00 & 0.00 \end{bmatrix}$$

```
-----------------------------
-----------------------------
-----------------------------
Iteration 2
-----------------------------
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 4.29 & 4.00 & 1.36 \end{bmatrix}$$

```
Optimal basis B:
```

```
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.00 & 11.00 \\ 7.00 & 0.00 & 0.00 \\ 0.00 & 5.00 & 0.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.14 & 0.00 \\ 0.00 & 0.00 & 0.20 \\ 0.09 & 0.00 & 0.00 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.09 & 0.14 & 0.20 \end{bmatrix}$$

```
Z_hat = 1.04
The minimum reduced cost is negative and equal to -0.04
The new generated pattern is:
```

$$\begin{bmatrix} 2.00 & 6.00 & 0.00 \end{bmatrix}$$

```
-----------------------------
-----------------------------
-----------------------------
Iteration 3
-----------------------------
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.00 & 4.00 & 0.45 & 5.00 \end{bmatrix}$$

```
Optimal basis B:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 11.00 & 2.00 \\ 0.00 & 0.00 & 6.00 \\ 5.00 & 0.00 & 0.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.00 & 0.20 \\ 0.09 & -0.03 & 0.00 \\ 0.00 & 0.17 & 0.00 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.09 & 0.14 & 0.20 \end{bmatrix}$$

```
Z_hat = 1.02
The minimum reduced cost is negative and equal to -0.02
The new generated pattern is:
```

$$\begin{bmatrix} 9.00 & 0.00 & 1.00 \end{bmatrix}$$

```
-----------------------------
-----------------------------
-----------------------------
Iteration 4
-----------------------------
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.00 & 3.89 & 0.00 & 5.00 & 0.56 \end{bmatrix}$$

```
Optimal basis B:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 2.00 & 9.00 \\ 0.00 & 6.00 & 0.00 \\ 5.00 & 0.00 & 1.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} -0.02 & 0.01 & 0.20 \\ 0.00 & 0.17 & 0.00 \\ 0.11 & -0.04 & 0.00 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.09 & 0.14 & 0.20 \end{bmatrix}$$

```
Z_hat = 1.01
The minimum reduced cost is negative and equal to -0.01
The new generated pattern is:
```

$$\begin{bmatrix} 6.00 & 2.00 & 1.00 \end{bmatrix}$$

```
-----------------------------
-----------------------------
-----------------------------
Iteration 5
-----------------------------
Optimal solution x:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 0.00 & 3.81 & 0.00 & 4.69 & 0.00 & 0.94 \end{bmatrix}$$

```
Optimal basis B:
-----------------------------
```

$$\begin{bmatrix} 0.00 & 2.00 & 6.00 \\ 0.00 & 6.00 & 2.00 \\ 5.00 & 0.00 & 1.00 \end{bmatrix}$$

```
Optimal basis inverse B^-1:
-----------------------------
```

$$\begin{bmatrix} -0.04 & 0.01 & 0.20 \\ -0.06 & 0.19 & 0.00 \\ 0.19 & -0.06 & 0.00 \end{bmatrix}$$

```
Optimal dual solution y_hat:
-----------------------------
```

$$\begin{bmatrix} 0.09 & 0.14 & 0.20 \end{bmatrix}$$

```
Z_hat = 1.0
The minimum reduced cost is non-negative and equal to 0.0
The column generation algorithm has finished
Finished
```

Q1.6 - Solution explanation

In the previous step the algorithm took 5 iterations to solve the problem.

In [31]:

```python
x_lat = a2l.to_ltx(x_opt, print_out=False)
B_lat = a2l.to_ltx(B_opt, print_out=False)
A_lat = a2l.to_ltx(A_opt, print_out=False)

display(Markdown(f"The optimal solution is \n {x_lat}"))
display(Markdown(f"The patterns constraints were \n {A_lat}"))
display(Markdown(f"From which the optimal basis is \n {B_lat}"))
display(Markdown(f"The optimal value is <b>{round(sum(x_opt), 2)}</b>"))
```

The optimal solution is

$$\begin{bmatrix} 0.00 & 0.00 & 3.81 & 0.00 & 4.69 & 0.00 & 0.94 \end{bmatrix}$$

The patterns constraints were

$$\begin{bmatrix} 10.00 & 0.00 & 0.00 & 11.00 & 2.00 & 9.00 & 6.00 \\ 0.00 & 7.00 & 0.00 & 0.00 & 6.00 & 0.00 & 2.00 \\ 0.00 & 0.00 & 5.00 & 0.00 & 0.00 & 1.00 & 1.00 \end{bmatrix}$$

From which the optimal basis is

$$\begin{bmatrix} 0.00 & 2.00 & 6.00 \\ 0.00 & 6.00 & 2.00 \\ 5.00 & 0.00 & 1.00 \end{bmatrix}$$

The optimal value is **9.44**

The solution means that we need approximately:

- 4 units of pattern **[0, 0, 5]** *(5 cuts of 16)*
- 5 units of pattern **[2, 6, 0]** *(2 cuts of 7 / 6 cuts of 11)*
- 1 unit of pattern **[6, 2, 1]** *(6 cuts of 7 / 2 cuts of 11 / 1 cut of 16)*