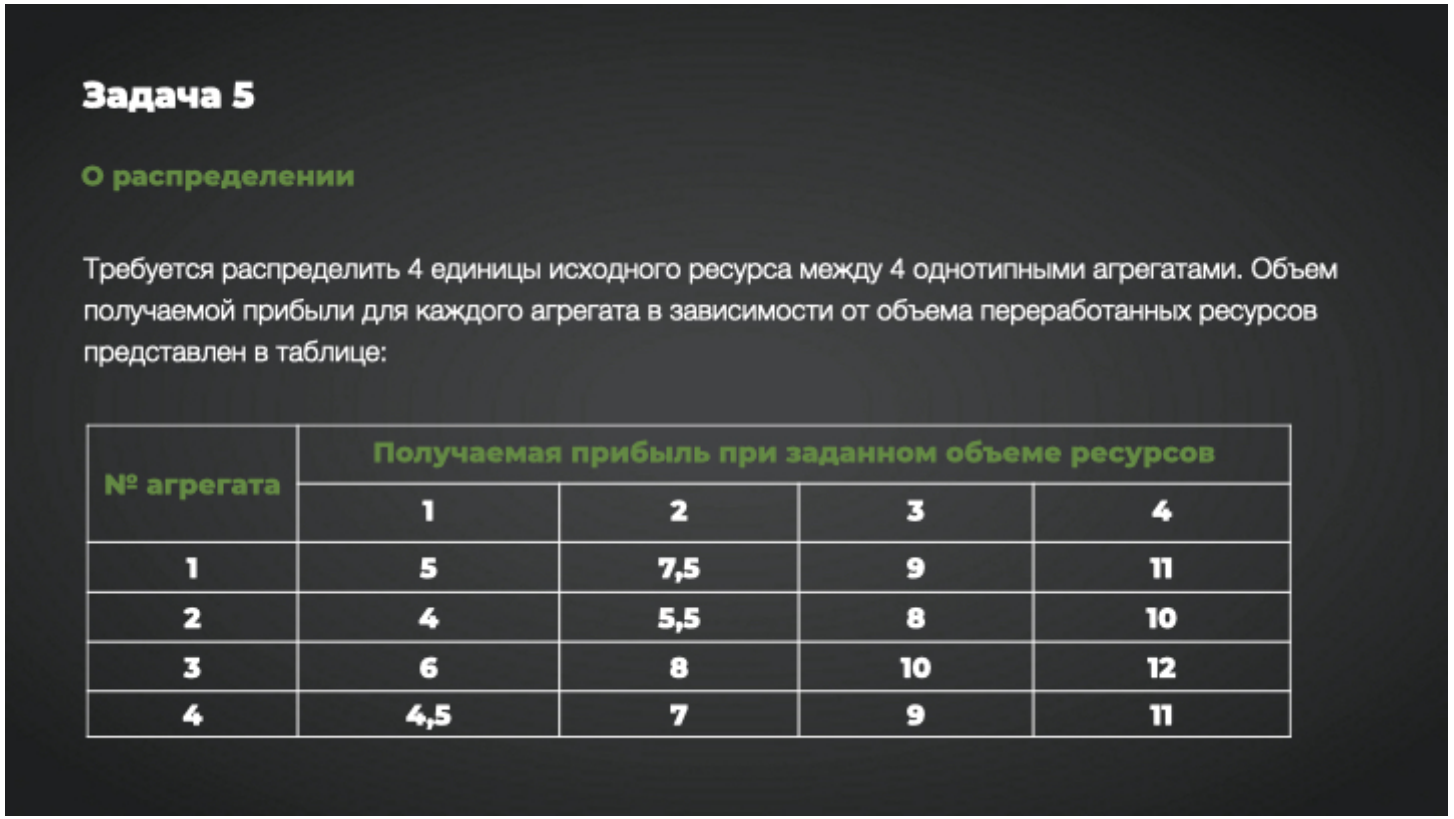
1. Проблема



Задача 5

О распределении

Требуется распределить 4 единицы исходного ресурса между 4 однотипными агрегатами. Объем получаемой прибыли для каждого агрегата в зависимости от объема переработанных ресурсов представлен в таблице:

№ агрегата	Получаемая прибыль при заданном объеме ресурсов			
	1	2	3	4
1	5	7,5	9	11
2	4	5,5	8	10
3	6	8	10	12
4	4,5	7	9	11

2. Содержательная постановка задачи

Составить план работы агрегатов для получения максимальной прибыли. Рассчитать максимальную прибыль при использовании 4 единиц ресурсов.

3. Формальная мат. модель

$X - ||x_{ij}||$ - матрица нулей и единиц, $x_{ij} = 1$ если выбрано ровно j ресурсов для i-го агрегата.
$C - ||c_{ij}||$ - матрица прибыли
Целевая функция - максимально возможная прибыль при распределении 4-х ресурсов:

$$\sum_{i=1}^{4}\sum_{j=1}^{4} x_{ij}c_{ij} \to \max$$

$$\forall i : \sum_{j=1}^{4} x_{ij} \le 1$$

$$\sum_{i=1}^{4}\left(\sum_{j=1}^{4} x_{ij} \cdot j\right) \le 4$$

$$X \in \{0, 1\}$$

Запишем ограничения на языке линейной алгебры:

$$\begin{pmatrix} x_{11} & \dots & x_{14} \\ x_{21} & \dots & x_{24} \\ x_{31} & \dots & x_{34} \\ x_{41} & \dots & x_{44} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{11} & \dots & x_{14} \\ x_{21} & \dots & x_{24} \\ x_{31} & \dots & x_{34} \\ x_{41} & \dots & x_{44} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \end{pmatrix} \le 4$$

4. Алгоритм и ПО

В качестве ПО будем использовать Python с подключенными модулями:

- numpy - для работы с линейной алгеброй
- cvxpy - для работы с линейным программированием

5. Решение задачи

Подключаем необходимые модули:

In [1]:
```
import numpy as np
import cvxpy
```

Функция для решения данного задания:

In [2]:
```
def solution(c):
    A = np.array([1,1,1,1])
    b = np.array([1,1,1,1])
    shp = np.shape(c)
    w = np.array([1,2,3,4]).reshape(-1,1)
    x = cvxpy.Variable(shape=c.shape, boolean = True)
    total_value = (cvxpy.multiply(c,x))
    constraints = [(x@A <= b),
                   (cvxpy.sum(x@w) <= 4),
                   (x>=0),
                   (x<=1)]
    problem = cvxpy.Problem(cvxpy.Maximize(cvxpy.sum(total_value)), constraints = constraints)
    print('Максимальная прибыль(в долларах)', problem.solve(), '\n')
    print(c*x.value)
```

6. Анализ

Проверим наш алгоритм на реальных данных:

In [3]:
```
c = np.array([[5,7.5,9,11],[4,5.5,8,10],[6,8,10,12],[4.5,7,9,11]])
solution(c)
```

```
Максимальная прибыль(в долларах) 19.5

[[5.  0.  0.  0. ]
 [4.  0.  0.  0. ]
 [6.  0.  0.  0. ]
 [4.5 0.  0.  0. ]]
```

Получили ожидаемый ответ. Из таблицы видно, что лучше всего в каждый агрегат загрузить по одному ресурсу.