Access Apple

In [1]:

```
# The imports

import numpy as np
import cvxpy as cp
```

Info

Space required to store (in cubic feet)

- **1 iPod cover :** `40 / 1000`
- **1 iPhone cover :** `45 / 1000`
- **1 iPad cover :** `210 / 1000`

Minimum Demand (per week)

- **iPods (say** `d1` **):** `5 000`
- **iPhones (say** `d2` **):** `0`
- **iPads (say** `d3` **):** `4 000`

Maximum Demand (per week)

- **iPods (say** `mD1` **):** `10 000`
- **iPhones (say** `mD2` **):** `15 000`
- **iPads (say** `mD3` **):** `8 000`

In [2]:

```
# Some given data

space_for_one = np.array([40/1000, 45/1000, 210/1000])
max_space = 6000

max_production = np.array([6000, 5000, 3000])

min_demand = np.array([5000, 0, 4000])
max_demand = np.array([10000, 15000, 8000])
profits = np.array([4, 6, 10])
```

(a) Problem 1 (Proportion of Time Spent)

In the current section, our decision variables will be the Proportions of time spent in making the 3 items.

The decision variables x1, x2 and x3 are defined as follows

- `x1` **: Proportion of time (per day) for iPod cover production.**
- `x2` **: Proportion of time (per day) for iPhone cover production.**
- `x3` **: Proportion of time (per day) for iPad cover production.**

The number of

- **iPods produced per day** `n1` **:** `x1 * 6000`
- **iPhones produced per day** `n2` **:** `x2 * 5000`

- **iPads produced per day** `n3` **:** `x3 * 3000`

Space required to store (in cubic feet)

- **x1 iPod covers :** `(x1 * 6000) * 40 / 1000`
- **x2 iPhone covers :** `(x2 * 5000) * 45 / 1000`
- **x3 iPad covers :** `(x3 * 3000) * 210 / 1000`

Profit (to Maximize)

`profit` **=** `5 * ((x1 * 6000)*4 + (x2 * 5000)*6 + (x3 * 3000)*10)`

Now, we're ready to write the constraints

Does not exceed the max production per day

- `x1 + x2 + x3` **<=** `1`

Does not exceed the max space

- `(x1 * 6000) * 40/1000` **+** `(x2 * 5000) * 45/1000` **+** `(x3 * 3000) * 210/1000` **<=** `6000`

Satisfy the Min demand

- `d1 <= 5 * (x1 * 6000)`
- `d2 <= 5 * (x2 * 5000)`
- `d3 <= 5 * (x3 * 3000)`

Does not exceed the Max Demand

- `5 * (x1 * 6000) <= mD1`
- `5 * (x2 * 5000) <= mD2`
- `5 * (x3 * 3000) <= mD3`

Now, we have a LP to maximize profit subject to the above-mentioned constraints

---

(b) Problem 2 (Number producted per week)

In the current section, our decision variables will be the number of items produced per week

---

The number of

- **iPods produced per week:** `y1`
- **iPhones produced per week:** `y2`
- **iPads produced per week:** `y3`

Space required to store (in cubic feet)

- **x1 iPod covers :** `(y1/5) * 40 / 1000`
- **x2 iPhone covers :** `(y2/5) * 45 / 1000`
- **x3 iPad covers :** `(y3/5) * 210 / 1000`

Profit (to Maximize)

`profit` **=** `(y1*4 + y2*6 + y3*10)`

Now, we're ready to write the constraints

Does not exceed the max production per day

- `(y1/5)/6000 + (y2/5)/5000 + (y3/5)/3000` **<=** `1`

Does not exceed the max space

- `(y1/5) * 40/1000` **+** `(y2/5) * 45/1000` **+** `(y3/5) * 210/1000` **<=** `6000`

Satisfy the Min demand

- `d1 <= y1`
- `d2 <= y2`
- `d3 <= y3`

Does not exceed the Max Demand

- `y1 <= mD1`
- `y2 <= mD2`
- `y3 <= mD3`

Now, we have a LP to maximize profit subject to the above-mentioned constraints

---

(c) Problem 3 (Number of Hours used for production per week)

The decision variables z1, z2 and z3 are defined as follows

- `z1` **: The number of hours devoted for iPod production (in one week)**
- `z2` **: The number of hours devoted for iPhone production (in one week)**
- `z3` **: The number of hours devoted for iPad production (in one week)**

The number of

- **iPods produced per day** `n1` **:** `(z1/40) * 6000`
- **iPhones produced per day** `n2` **:** `(z2/40) * 5000`
- **iPads produced per day** `n3` **:** `(z3/40) * 3000`

Space required to store (in cubic feet)

- **n1 iPod covers :** `((z1/40) * 6000) * 40 / 1000`
- **n2 iPhone covers :** `((z2/40) * 5000) * 45 / 1000`
- **n3 iPad covers :** `((z3/40) * 3000) * 210 / 1000`

Profit (to Maximize)

`profit` **=** `5 * ( ((z1/40) * 6000)*4 + ((z2/40) * 5000)*6 + ((z3/40) * 3000)*10 )`

Now, we're ready to write the constraints

Does not exceed the max production per day

- `z1 + z2 + z3` **<=** `8`

Does not exceed the max space

- `( (z1/40) * 6000) * 40/1000` **+** `((z2/40) * 5000) * 45/1000` **+** `((z3/40) * 3000) * 210/1000` **<=** `6000`

Satisfy the Min demand

- `d1 <= 5 * ((z1/40) * 6000)`
- `d2 <= 5 * ((z2/40) * 5000)`
- `d3 <= 5 * ((z3/40) * 3000)`

Does not exceed the Max Demand

- `5 * ((z1/40) * 6000) <= mD1`
- `5 * ((z2/40) * 5000) <= mD2`
- `5 * ((z3/40) * 3000) <= mD3`

(d) Relation between x1, x2, x3 amd z1, z2, z3

Recalling,

The decision variables x1, x2 and x3 are defined as follows

- `x1` **: Proportion of time (per day) for iPod cover production.**
- `x2` **: Proportion of time (per day) for iPhone cover production.**
- `x3` **: Proportion of time (per day) for iPad cover production.**

The decision variables z1, z2 and z3 are defined as follows

- `z1` **: The number of hours devoted for iPod production (in one week)**
- `z2` **: The number of hours devoted for iPhone production (in one week)**
- `z3` **: The number of hours devoted for iPad production (in one week)**

Equating the dialy production, we get

- `n1 = n1`
- `x1 * 6000 = (z1/40) * 6000`
- `x1 = z1/40`

Similarly,

- `x1 = z1/40`
- `x2 = z2/40`
- `x3 = z3/40`

(e) Implementation

In [3]:

```
def problem():

    # The proportions
    X = cp.Variable(3)

    # The number sold per day
    N = cp.multiply(X, max_production)

    # The profit, to be maximized (the objective)
    profit = cp.Maximize (5 * cp.sum(cp.multiply(N, profits)))

    # Constraints
    constraints = [
                    cp.sum(X) <= 1,   # max production per day
                    cp.sum(cp.multiply(N, space_for_one)) <= max_space,
                    min_demand <= 5 * N,
                    5 * N <= max_demand,
    ]

    # Formulating the problem and solving it
    prob = cp.Problem(profit, constraints)
    result = prob.solve()

    print("The proportion of time per day devoted for iPods, iPhones and iPads respective
ly")
    print("Optimum Solution:", X.value)
    print()

    print("The number of iPods, iPhones and iPads produced per week")
    print("Optimal Solution:", 5 * cp.multiply(X.value, max_production).value)
    print()

    print("Optimum Value:", result)


problem()
```

```
The proportion of time per day devoted for iPods, iPhones and iPads respectively
Optimum Solution: [0.16666667 0.31415449 0.51917884]

The number of iPods, iPhones and iPads produced per week
Optimal Solution: [5000.00000573 7853.86236644 7787.68257875]

Optimum Value: 145000.0000091098
```