Homework 9 (Question 2)

In [1]:

```
import sys
!{sys.executable} -m pip install pulp
```

```
Requirement already satisfied: pulp in c:\users\muntakim\appdata\local\programs\python\p
ython38-32\lib\site-packages (2.4)
Requirement already satisfied: amply>=0.1.2 in c:\users\muntakim\appdata\local\programs
\python\python38-32\lib\site-packages (from pulp) (0.1.4)
Requirement already satisfied: pyparsing in c:\users\muntakim\appdata\local\programs\pyt
hon\python38-32\lib\site-packages (from amply>=0.1.2->pulp) (2.4.7)
Requirement already satisfied: docutils>=0.3 in c:\users\muntakim\appdata\local\programs
\python\python38-32\lib\site-packages (from amply>=0.1.2->pulp) (0.17.1)
```

In [2]:

```
import pulp
from pulp import *
```

In [3]:

```
import csv
```

In [4]:

```
"""
Investment Portfolio Problem

Author : Muntakim Rahman 2021
"""
```

Out[4]:

```
'\nInvestment Portfolio Problem\n\nAuthor : Muntakim Rahman 2021\n'
```

Textbook Problem

2. Consider the following investments

- Canada bonds: Vanguard Canadian Aggregate Bond Index ETF (VAB)
- Canada stocks: Vanguard FTSE Canada All Cap Index ETF (VCN)
- Non-Canada stocks: Vanguard FTSE Global All Cap ex Canada Index ETF (VXC)
- Nixed 60% Stocks, 30% Canada: Vanguard Balanced ETF Portfolio (VBAL)

and their historical prices for the last year (in accompanying .csv file).

(a) Use PulP to compute the optimal portfolio with risk/return parameter $\mu = 0, 0.5, 1$ given the historical data. (Hint: see accompanying .ipynb file for a start, you will also need to upload the .csv file to syzygy)

(b) Repeat part (a) after adding the constrants appearing on the midterm: no more than 90% Stocks and no more than 80% Canada. How have the solutions changed?

Portfolio Selection

Let $R_j(t)$ denote the return on investment j over T monthly periods.

$$R = \sum_j x_j R_j \quad \dots (1) \tag{1}$$

Calculate the expected return (i.e. reward) with the formula :

$$ER = \sum_j x_j ER_j \quad \dots (2) \quad From\ (1) \tag{2}$$

For our purposes, assume that $R_j(t)$ is a random variable where values $t = 1 \dots T$ are taken with equal probability.

$$ER_j = \frac{1}{T} \sum_{t=1}^{T} R_j(t) \quad \dots (3) \tag{3}$$

Let the mean absolute deviation from the mean (MAD) represent the risk associated with a portfolio of investments.

For our purposes, the (MAD) is calculated as :

$$\begin{aligned} E|R - ER| &= E\left|\sum_j x_j R_j - \sum_j x_j ER_j\right| \quad \mathit{From}\ (1)\ \&\ (2) \\ &= E\left|\sum_j x_j (R_j - ER_j)\right| \\ &= \frac{1}{T}\sum_{t=1}^{T}\left|\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right]\right| \quad \ldots (4) \quad \mathit{From}\ (3) \end{aligned} \tag{4}$$

Linear Programming Problem

The LP Problem takes the positive risk/return parameter μ into account to form a linear combination between the risk and reward, as shown below.

$$\begin{aligned} &\text{Maximize} \quad \mu\, ER - E|R - ER| \\ &= \mu \sum_j x_j ER_j - \frac{1}{T}\sum_{t=1}^{T}\left|\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right]\right| \quad \mathit{From}\ (2)\ \&\ (4) \\ &= \frac{\mu}{T}\sum_{t=1}^{T}\sum_j x_j R_j(t) - \frac{1}{T}\sum_{t=1}^{T}\left|\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right]\right| \quad \mathit{From}\ (3) \\ &= \frac{\mu}{T}\sum_{t=1}^{T}\sum_j x_j R_j(t) - \frac{1}{T}\sum_{t=1}^{T} y_t \quad \mathit{Where}\ y_t = \left|\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right]\right| \end{aligned}$$

$$\begin{aligned} \text{Subject to} \quad -y_t \le \sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right] \le y_t \\ \mathit{Which\ can\ also\ be\ written\ as:} \\ y_t \ge +\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right], \quad y_t \ge -\sum_j x_j\left[R_j(t) - \frac{1}{T}\sum_{t=1}^{T} R_j(t)\right] \\ x_j \ge 0 \quad j = 1\ldots n, \quad y_t \ge 0 \quad t = 1\ldots T \\ \sum_j x_j = 1, \end{aligned}$$

At high value, μ maximizes reward regardless of risk. At low values, risk is minimized.

Import Data From CSV File

In [5]:

```
dates = []
investments = ['VAB', 'VCN', 'VXC', 'VBAL']

prices = {}
for j in range(len(investments)) :
    prices[investments[j]] = {}

## Print Prices Dictionary -> Mainly for Debugging Purposes.
print(prices)
```

```
{'VAB': {}, 'VCN': {}, 'VXC': {}, 'VBAL': {}}
```

In [6]:
```
with open('HW9P2.csv') as historical_prices :
    csv_reader = csv.reader(historical_prices, delimiter = ',')
    line_count = 0
    column_names = []

    row_count = 0
    for row_contents in csv_reader :
        if (row_count == 0) :
            for column_contents in row_contents :
                column_names.append(str(column_contents))
        else :
            # Column Count should be Equal to Total Number of Investments.
            n = 0
            for column_contents in row_contents :
                if (n == 0) :
                    current_date = str(column_contents)
                    dates.append(current_date)
                else :
                    j = investments[n - 1]
                    prices[j][current_date] = float(column_contents)

                n += 1
        row_count += 1

    # Number of Monthly Intervals in Which Returns are Calculated.
    T = row_count - 2

## Print CSV File Data to Screen -> Mainly for Debugging Purposes.
print(f'Columns Are \n{column_names}\n\n')

print(f'Dates : \n{dates}\n')

print('Prices :')
for j, price in prices.items() :
    print(f'Investment : {j}')
    print(f'Price : \n{price}\n')

print(f'Calculated T Value : {T}\n')
```

```
Columns Are
['Date', 'Vanguard Canadian Aggregate Bond Index ETF (VAB)', 'Vanguard FTSE Canada All C
ap Index ETF (VCN)', 'Vanguard FTSE Global All Cap ex Canada Index ETF (VXC) ', 'Vanguar
d Balanced ETF Portfolio (VBAL)']


Dates :
['2020-04-01', '2020-05-01', '2020-06-01', '2020-07-01', '2020-08-01', '2020-09-01', '20
20-10-01', '2020-11-01', '2020-12-01', '2021-01-01', '2021-02-01', '2021-03-01']

Prices :
Investment : VAB
Price :
{'2020-04-01': 25.73, '2020-05-01': 27.08, '2020-06-01': 27.06, '2020-07-01': 27.43, '20
20-08-01': 27.82, '2020-09-01': 27.38, '2020-10-01': 27.35, '2020-11-01': 27.19, '2020-1
2-01': 27.34, '2021-01-01': 27.34, '2021-02-01': 26.94, '2021-03-01': 26.13}

Investment : VCN
Price :
{'2020-04-01': 26.05, '2020-05-01': 29.41, '2020-06-01': 30.74, '2020-07-01': 31.54, '20
20-08-01': 32.75, '2020-09-01': 33.36, '2020-10-01': 32.41, '2020-11-01': 31.49, '2020-1
2-01': 35.09, '2021-01-01': 35.32, '2021-02-01': 35.3, '2021-03-01': 36.92}
```

```
Investment : VXC
Price :
{'2020-04-01': 32.93, '2020-05-01': 36.56, '2020-06-01': 38.49, '2020-07-01': 39.56, '20
20-08-01': 40.42, '2020-09-01': 41.68, '2020-10-01': 41.22, '2020-11-01': 40.58, '2020-1
2-01': 44.22, '2021-01-01': 45.45, '2021-02-01': 45.51, '2021-03-01': 46.42}

Investment : VBAL
Price :
{'2020-04-01': 23.24, '2020-05-01': 25.2, '2020-06-01': 25.97, '2020-07-01': 26.46, '202
0-08-01': 27.11, '2020-09-01': 27.4, '2020-10-01': 27.14, '2020-11-01': 26.74, '2020-12-
01': 28.45, '2021-01-01': 28.74, '2021-02-01': 28.78, '2021-03-01': 28.95}

Calculated T Value : 11
```

In [7]:

```python
# Calculate Monthly Returns
returns = {}

for j in investments :
    returns[j] = {}

    ## Print Returns Dictionary -> Mainly for Debugging Purposes.
    print(f'Returns Before Assignment : \n{returns}\n')

    for t in range(1, T + 1) :
        returns[j][t] = prices[j][dates[t]] / prices[j][(dates[t - 1])]

    ## Print Returns for Investment -> Mainly for Debugging Purposes.
    print(f'Returns for Investment {j} : \n{returns[j]}\n')
```

```
Returns Before Assignment :
{'VAB': {}}

Returns for Investment VAB :
{1: 1.0524679362611737, 2: 0.999261447562777, 3: 1.0136733185513673, 4: 1.01421800947867
3, 5: 0.9841840402588066, 6: 0.9989043097151206, 7: 0.9941499085923218, 8: 1.00551673409
34166, 9: 1.0, 10: 0.9853694220921727, 11: 0.9699331848552337}

Returns Before Assignment :
{'VAB': {1: 1.0524679362611737, 2: 0.999261447562777, 3: 1.0136733185513673, 4: 1.014218
009478673, 5: 0.9841840402588066, 6: 0.9989043097151206, 7: 0.9941499085923218, 8: 1.005
5167340934166, 9: 1.0, 10: 0.9853694220921727, 11: 0.9699331848552337}, 'VCN': {}}

Returns for Investment VCN :
{1: 1.128982725527831, 2: 1.0452227133628018, 3: 1.026024723487313, 4: 1.038363982244768
6, 5: 1.0186259541984732, 6: 0.9715227817745803, 7: 0.9716136994754706, 8: 1.11432200698
6345, 9: 1.0065545739526929, 10: 0.9994337485843714, 11: 1.0458923512747877}

Returns Before Assignment :
{'VAB': {1: 1.0524679362611737, 2: 0.999261447562777, 3: 1.0136733185513673, 4: 1.014218
009478673, 5: 0.9841840402588066, 6: 0.9989043097151206, 7: 0.9941499085923218, 8: 1.005
5167340934166, 9: 1.0, 10: 0.9853694220921727, 11: 0.9699331848552337}, 'VCN': {1: 1.128
982725527831, 2: 1.0452227133628018, 3: 1.026024723487313, 4: 1.0383639822447686, 5: 1.0
186259541984732, 6: 0.9715227817745803, 7: 0.9716136994754706, 8: 1.114322006986345, 9:
1.0065545739526929, 10: 0.9994337485843714, 11: 1.0458923512747877}, 'VXC': {}}

Returns for Investment VXC :
{1: 1.110233829334953, 2: 1.0527899343544858, 3: 1.027799428422967, 4: 1.021739130434782
5, 5: 1.0311726867887183, 6: 0.9889635316698656, 7: 0.9844735565259582, 8: 1.08969935929
02908, 9: 1.0278154681139757, 10: 1.0013201320132012, 11: 1.0199956053614592}

Returns Before Assignment :
{'VAB': {1: 1.0524679362611737, 2: 0.999261447562777, 3: 1.0136733185513673, 4: 1.014218
```

```
009478673, 5: 0.9841840402588066, 6: 0.9989043097151206, 7: 0.9941499085923218, 8: 1.005
5167340934166, 9: 1.0, 10: 0.9853694220921727, 11: 0.9699331848552337}, 'VCN': {1: 1.128
982725527831, 2: 1.0452227133628018, 3: 1.026024723487313, 4: 1.0383639822447686, 5: 1.0
186259541984732, 6: 0.9715227817745803, 7: 0.9716136994754706, 8: 1.114322006986345, 9:
1.0065545739526929, 10: 0.99943374858437l4, 11: 1.0458923512747877}, 'VXC': {1: 1.110233
829334953, 2: 1.0527899343544858, 3: 1.027799428422967, 4: 1.0217391304347825, 5: 1.0311
726867887183, 6: 0.9889635316698656, 7: 0.9844735565259582, 8: 1.0896993592902908, 9: 1.
0278154681139757, 10: 1.0013201320132012, 11: 1.0199956053614592}, 'VBAL': {}}

Returns for Investment VBAL :
{1: 1.0843373493975903, 2: 1.0305555555555554, 3: 1.0188679245283019, 4: 1.0245653817082
387, 5: 1.0106971597196606, 6: 0.9905109489051096, 7: 0.985261606484893, 8: 1.0639491398
653702, 9: 1.0101933216168717, 10: 1.001391788448156, 11: 1.0059068797776234}
```

In [8]:

```python
# Calculate Mean Monthly Returns
mean_returns = {}

for j in investments :

    mean_returns[j] = 0
    for t in range(1, T + 1) :
        mean_returns[j] += returns[j][t]
    mean_returns[j] /= T

    ## Print Mean Returns for Investment -> Mainly for Debugging Purposes.
    print(f'Mean Returns for Investment {j} : {mean_returns[j]}\n')
```

```
Mean Returns for Investment VAB : 1.0016071192237332

Mean Returns for Investment VCN : 1.0333235691699487

Mean Returns for Investment VXC : 1.032363878391878

Mean Returns for Investment VBAL : 1.0205670050915792
```

Solve LP Problem

In [9]:

```python
LP_Prob = LpProblem(name = 'Investment_Portfolio_Problem', sense = LpMaximize)

decision_variables_investment = {}
for j in range(len(investments)) :
    decision_variables_investment[investments[j]] = LpVariable(name = 'x_' + str(invest

decision_variables_deviation = {}
for t in range(1, T + 1) :
    decision_variables_deviation[t] = LpVariable(name = 'y_' + str(t), lowBound = 0)

    LP_Prob += decision_variables_deviation[t] >= 0, f'Low_Bound_Deviation_{t}'

for j in investments :
    LP_Prob += decision_variables_investment[j] >= 0, f'Low_Bound_Investment_{j}'

# Print Decision Variables -> Mainly for Debugging Purposes.
print(f'Investment Decision Variables : {decision_variables_investment}')
print(f'Deviation Decision Variables : {decision_variables_deviation}')
```

```
Investment Decision Variables : {'VAB': x_VAB, 'VCN': x_VCN, 'VXC': x_VXC, 'VBAL': x_VBA
L}
```

```
Deviation Decision Variables : {1: y_1, 2: y_2, 3: y_3, 4: y_4, 5: y_5, 6: y_6, 7: y_7,
8: y_8, 9: y_9, 10: y_10, 11: y_11}
```

Risk/Return Parameter of 1

In [10]:

```
mu = 1

expected_investment_returns = lpSum([mu * mean_returns[j] * decision_variables_investme
monthly_deviation = lpSum([decision_variables_deviation[t] for t in range(1, T + 1)])/T

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Returns
```

In [11]:

```
# The Constraints are Added to 'LP_Prob'
for t in range(1, T + 1) :
    monthly_deviation = lpSum([(returns[j][t] - mean_returns[j]) * decision_variables_i
    LP_Prob += decision_variables_deviation[t] >= -monthly_deviation, f'Deviation_Inves
    LP_Prob += decision_variables_deviation[t] >= monthly_deviation, f'Deviation_Invest

LP_Prob += lpSum([decision_variables_investment[j] for j in investments]) == 1, 'Probab
```

In [12]:

```
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
MAXIMIZE
1.0016071192237332*x_VAB + 1.0205670050915792*x_VBAL + 1.0333235691699487*x_VCN + 1.0323
63878391878*x_VXC + -0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.0909090909
0909091*y_11 + -0.09090909090909091*y_2 + -0.09090909090909091*y_3 + -0.0909090909090909
1*y_4 + -0.09090909090909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 +
-0.09090909090909091*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0

Low_Bound_Deviation_6: y_6 >= 0

Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0
```

```
Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0
```

```
Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [13]:

```python
LP_Prob.writeLP('OriginalPortfolioProblemParameter1.lp')
```

Out[13]:

```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
 y_8,
 y_9]
```

In [14]:
```
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[14]: 1

In [15]:
```
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 0.0
x_VCN = 0.0
x_VXC = 1.0
x_VBAL = 0.0


Optimal Value : Z = 1.0040672442191507
```

Risk/Return Parameter of 0.5

In [16]:
```
mu = 0.5

expected_investment_returns = lpSum([mu * mean_returns[j] * decision_variables_investme
monthly_deviation = lpSum([decision_variables_deviation[t] for t in range(1, T + 1)])/T

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Returns
```

```
c:\users\muntakim\appdata\local\programs\python\python38-32\lib\site-packages\pulp\pulp.
py:1537: UserWarning: Overwriting previously set objective.
  warnings.warn("Overwriting previously set objective.")
```

In [17]:
```
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
MAXIMIZE
0.5008035596118666*x_VAB + 0.5102835025457896*x_VBAL + 0.5166617845849744*x_VCN + 0.5161
81939195939*x_VXC + -0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.0909090909
0909091*y_11 + -0.09090909090909091*y_2 + -0.09090909090909091*y_3 + -0.0909090909090909
1*y_4 + -0.09090909090909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 +
-0.09090909090909091*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0
```

```
Low_Bound_Deviation_6: y_6 >= 0

Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0

Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0
```

```
Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [18]:

```
LP_Prob.writeLP('OriginalPortfolioProblemParameter0.5.lp')
```

Out[18]:
```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
 y_8,
 y_9]
```

In [19]:
```
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[19]:
```
1
```

In [20]:
```
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 0.20554895
x_VCN = 0.0
x_VXC = 0.39448954
x_VBAL = 0.3999615


Optimal Value : Z = 0.4890212764694815
```

Risk/Return Parameter of 0

In [21]:
```
mu = 0

expected_investment_returns = lpSum([mu * mean_returns[j] * decision_variables_investme
monthly_deviation = lpSum([decision_variables_deviation[t] for t in range(1, T + 1)])/T

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Returns
```

In [22]:
```
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
```

```
MAXIMIZE
-0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.09090909090909091*y_11 + -0.09
090909090909091*y_2 + -0.09090909090909091*y_3 + -0.09090909090909091*y_4 + -0.090909090
90909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 + -0.0909090909090909
1*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0

Low_Bound_Deviation_6: y_6 >= 0

Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0

Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
```

```
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
```

```
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [23]:

```
LP_Prob.writeLP('OriginalPortfolioProblemParameter0.lp')
```

Out[23]:

```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
 y_8,
 y_9]
```

In [24]:

```
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[24]:

```
1
```

In [25]:

```
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 1.0
x_VCN = 0.0
x_VXC = 0.0
x_VBAL = 0.0


Optimal Value : Z = -0.014445003809090912
```

Change Portfolio Composition (Upper Limit 90% Stocks,

80% Canada)

Risk/Return Parameter of 1

In [26]:
```
mu = 1

expected_investment_returns = lpSum([mu * mean_returns[j] * decision_variables_investme
monthly_deviation = lpSum([decision_variables_deviation[t] for t in range(1, T + 1)])/T

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Returns
```

In [27]:
```
stocks = {'VCN' : 1, 'VXC' : 1, 'VBAL' : 0.6}
canada_investments = {'VAB' : 1, 'VCN' : 1, 'VBAL' : 0.3}

# The New Constraints are Added to 'LP_Prob'
LP_Prob += lpSum([(percent * decision_variables_investment[stock]) for stock, percent i

LP_Prob += lpSum([(percent * decision_variables_investment[investment]) for investment,
```

In [28]:
```
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
MAXIMIZE
1.0016071192237332*x_VAB + 1.0205670050915792*x_VBAL + 1.0333235691699487*x_VCN + 1.0323
63878391878*x_VXC + -0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.0909090909
0909091*y_11 + -0.09090909090909091*y_2 + -0.09090909090909091*y_3 + -0.0909090909090909
1*y_4 + -0.09090909090909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 +
-0.09090909090909091*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0

Low_Bound_Deviation_6: y_6 >= 0

Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0
```

```
Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0
```

```
Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

Stocks_Limit: 0.6 x_VBAL + x_VCN + x_VXC <= 0.9

Canada_Limit: x_VAB + 0.3 x_VBAL + x_VCN <= 0.8

VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [29]:

```python
LP_Prob.writeLP('NewPortfolioProblemParameter1.lp')
```

Out[29]:

```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
```

```
y_8,
y_9]
```

In [30]:

```python
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[30]: 1

In [31]:

```python
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 0.1
x_VCN = 0.0
x_VXC = 0.9
x_VBAL = 0.0


Optimal Value : Z = 1.002868054329609
```

Risk/Return Parameter of 0.5

In [32]:

```python
mu = 0.5

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += mu*expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Retu
```

In [33]:

```python
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
MAXIMIZE
0.5008035596118666*x_VAB + 0.5102835025457896*x_VBAL + 0.5166617845849744*x_VCN + 0.5161
81939195939*x_VXC + -0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.0909090909
0909091*y_11 + -0.09090909090909091*y_2 + -0.09090909090909091*y_3 + -0.0909090909090909
1*y_4 + -0.09090909090909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 +
-0.09090909090909091*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0

Low_Bound_Deviation_6: y_6 >= 0
```

```
Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0

Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0
```

```
Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

Stocks_Limit: 0.6 x_VBAL + x_VCN + x_VXC <= 0.9

Canada_Limit: x_VAB + 0.3 x_VBAL + x_VCN <= 0.8

VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [34]:
```
LP_Prob.writeLP('NewPortfolioProblemParameter0.5.lp')
```

Out[34]:
```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
 y_8,
 y_9]
```

In [35]:
```
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[35]:
```
1
```

In [36]:
```
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 0.20554895
x_VCN = 0.0
x_VXC = 0.39448954
x_VBAL = 0.3999615


Optimal Value : Z = 0.4890212764694815
```

In [37]:
```
### Risk/Return Parameter of 0
```

In [38]:
```
mu = 0

expected_investment_returns = lpSum([mu * mean_returns[j] * decision_variables_investme
monthly_deviation = lpSum([decision_variables_deviation[t] for t in range(1, T + 1)])/T

# The Objective Function is Added to 'LP_Prob' First.
LP_Prob += expected_investment_returns - monthly_deviation, 'Maximize_Portfolio_Returns
```

In [39]:
```
print(LP_Prob)
```

```
Investment_Portfolio_Problem:
MAXIMIZE
-0.09090909090909091*y_1 + -0.09090909090909091*y_10 + -0.09090909090909091*y_11 + -0.09
090909090909091*y_2 + -0.09090909090909091*y_3 + -0.09090909090909091*y_4 + -0.090909090
90909091*y_5 + -0.09090909090909091*y_6 + -0.09090909090909091*y_7 + -0.0909090909090909
1*y_8 + -0.09090909090909091*y_9 + 0.0
SUBJECT TO
Low_Bound_Deviation_1: y_1 >= 0

Low_Bound_Deviation_2: y_2 >= 0

Low_Bound_Deviation_3: y_3 >= 0

Low_Bound_Deviation_4: y_4 >= 0

Low_Bound_Deviation_5: y_5 >= 0

Low_Bound_Deviation_6: y_6 >= 0

Low_Bound_Deviation_7: y_7 >= 0

Low_Bound_Deviation_8: y_8 >= 0

Low_Bound_Deviation_9: y_9 >= 0

Low_Bound_Deviation_10: y_10 >= 0

Low_Bound_Deviation_11: y_11 >= 0

Low_Bound_Investment_VAB: x_VAB >= 0

Low_Bound_Investment_VCN: x_VCN >= 0

Low_Bound_Investment_VXC: x_VXC >= 0

Low_Bound_Investment_VBAL: x_VBAL >= 0

Deviation_Investment_Negative_Relation_1: 0.0508608170374 x_VAB
 + 0.063770344306 x_VBAL + 0.0956591563579 x_VCN + 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Positive_Relation_1: - 0.0508608170374 x_VAB
 - 0.063770344306 x_VBAL - 0.0956591563579 x_VCN - 0.0778699509431 x_VXC + y_1
 >= 0

Deviation_Investment_Negative_Relation_2: - 0.00234567166096 x_VAB
 + 0.00998855046398 x_VBAL + 0.0118991441929 x_VCN + 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Positive_Relation_2: 0.00234567166096 x_VAB
 - 0.00998855046398 x_VBAL - 0.0118991441929 x_VCN - 0.0204260559626 x_VXC
 + y_2 >= 0

Deviation_Investment_Negative_Relation_3: 0.0120661993276 x_VAB
 - 0.00169908056328 x_VBAL - 0.00729884568264 x_VCN - 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Positive_Relation_3: - 0.0120661993276 x_VAB
 + 0.00169908056328 x_VBAL + 0.00729884568264 x_VCN + 0.00456444996891 x_VXC
 + y_3 >= 0

Deviation_Investment_Negative_Relation_4: 0.0126108902549 x_VAB
 + 0.00399837661666 x_VBAL + 0.00504041307482 x_VCN - 0.0106247479571 x_VXC
 + y_4 >= 0
```

```
Deviation_Investment_Positive_Relation_4: - 0.0126108902549 x_VAB
 - 0.00399837661666 x_VBAL - 0.00504041307482 x_VCN + 0.0106247479571 x_VXC
 + y_4 >= 0

Deviation_Investment_Negative_Relation_5: - 0.0174230789649 x_VAB
 - 0.00986984537192 x_VBAL - 0.0146976149715 x_VCN - 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Positive_Relation_5: 0.0174230789649 x_VAB
 + 0.00986984537192 x_VBAL + 0.0146976149715 x_VCN + 0.00119119160316 x_VXC
 + y_5 >= 0

Deviation_Investment_Negative_Relation_6: - 0.00270280950861 x_VAB
 - 0.0300560561865 x_VBAL - 0.0618007873954 x_VCN - 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Positive_Relation_6: 0.00270280950861 x_VAB
 + 0.0300560561865 x_VBAL + 0.0618007873954 x_VCN + 0.043400346722 x_VXC + y_6
 >= 0

Deviation_Investment_Negative_Relation_7: - 0.00745721063141 x_VAB
 - 0.0353053986067 x_VBAL - 0.0617098696945 x_VCN - 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Positive_Relation_7: 0.00745721063141 x_VAB
 + 0.0353053986067 x_VBAL + 0.0617098696945 x_VCN + 0.0478903218659 x_VXC
 + y_7 >= 0

Deviation_Investment_Negative_Relation_8: 0.00390961486968 x_VAB
 + 0.0433821347738 x_VBAL + 0.0809984378164 x_VCN + 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Positive_Relation_8: - 0.00390961486968 x_VAB
 - 0.0433821347738 x_VBAL - 0.0809984378164 x_VCN - 0.0573354808984 x_VXC
 + y_8 >= 0

Deviation_Investment_Negative_Relation_9: - 0.00160711922373 x_VAB
 - 0.0103736834747 x_VBAL - 0.0267689952173 x_VCN - 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Positive_Relation_9: 0.00160711922373 x_VAB
 + 0.0103736834747 x_VBAL + 0.0267689952173 x_VCN + 0.0045484102779 x_VXC
 + y_9 >= 0

Deviation_Investment_Negative_Relation_10: - 0.0162376971316 x_VAB
 - 0.0191752166434 x_VBAL - 0.0338898205856 x_VCN - 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Positive_Relation_10: 0.0162376971316 x_VAB
 + 0.0191752166434 x_VBAL + 0.0338898205856 x_VCN + 0.0310437463787 x_VXC
 + y_10 >= 0

Deviation_Investment_Negative_Relation_11: - 0.0316739343685 x_VAB
 - 0.014660125314 x_VBAL + 0.0125687821048 x_VCN - 0.0123682730304 x_VXC
 + y_11 >= 0

Deviation_Investment_Positive_Relation_11: 0.0316739343685 x_VAB
 + 0.014660125314 x_VBAL - 0.0125687821048 x_VCN + 0.0123682730304 x_VXC
 + y_11 >= 0

Probability_Simplex: x_VAB + x_VBAL + x_VCN + x_VXC = 1

Stocks_Limit: 0.6 x_VBAL + x_VCN + x_VXC <= 0.9

Canada_Limit: x_VAB + 0.3 x_VBAL + x_VCN <= 0.8
```

```
VARIABLES
x_VAB Continuous
x_VBAL Continuous
x_VCN Continuous
x_VXC Continuous
y_1 Continuous
y_10 Continuous
y_11 Continuous
y_2 Continuous
y_3 Continuous
y_4 Continuous
y_5 Continuous
y_6 Continuous
y_7 Continuous
y_8 Continuous
y_9 Continuous
```

In [40]:

```
LP_Prob.writeLP('NewPortfolioProblemParameter0.lp')
```

Out[40]:

```
[x_VAB,
 x_VBAL,
 x_VCN,
 x_VXC,
 y_1,
 y_10,
 y_11,
 y_2,
 y_3,
 y_4,
 y_5,
 y_6,
 y_7,
 y_8,
 y_9]
```

In [41]:

```
# The Problem is Solved Using PuLP's Choice of Solver.
LP_Prob.solve()
```

Out[41]: 1

In [42]:

```
print(f'Status: {LpStatus[LP_Prob.status]} \n')

for j, decision_variable in decision_variables_investment.items() :
    print(f'{decision_variable.name} = {decision_variable.varValue}')
print('\n')

if (LpStatus[LP_Prob.status] == 'Optimal') :
    print(f'Optimal Value : Z = {value(LP_Prob.objective)}')
else :
    print(f'No Optimal Value. Status Code : {value(LP_Prob.objective)}')
```

```
Status: Optimal

x_VAB = 0.71428571
x_VCN = 0.0
x_VXC = 0.0
x_VBAL = 0.28571429
```

```
Optimal Value : Z = -0.016217919445454546
```