exercise3 (Score: 9.0 / 9.0)

1. Test cell (Score: 2.0 / 2.0)
2. Test cell (Score: 1.0 / 1.0)
3. Test cell (Score: 2.0 / 2.0)
4. Test cell (Score: 2.0 / 2.0)
5. Test cell (Score: 2.0 / 2.0)

Lab 3

1. 提交作業之前，建議可以先點選上方工具列的**Kernel**，再選擇**Restart & Run All**，檢查一下是否程式跑起來都沒有問題，最後記得儲存。
2. 請先填上下方的姓名(name)及學號(stduent_id)再開始作答，例如：

```
name = "我的名字"
student_id= "B06201000"
```

3. 演算法的實作可以參考lab-3 (https://yuanyuyuan.github.io/itcm/lab-3.html), 有任何問題歡迎找助教詢問。
4. **Deadline: 10/30(Wed.)**

In [1]:

```
name = "馬宗儀"
student_id = "b06201006"
```

Exercise 3

The price (in euros) of a magazine has changed as follows:

Nov. 87	Dec. 88	Nov. 90	Jan. 93	Jan. 95	Jan. 96	Nov. 96	Nov. 00
4.5	5.0	6.0	6.5	7.0	7.5	8.0	8.0

1. Use the interpolating polynomial of *degree 7* to estimate the price in February 1989, in April 1998 and in November 2002.

Part 0. Import libraries.

In [2]:

```
import matplotlib.pyplot as plt
import numpy as np
```

Part 1. Define the polynomial interpolation function.

Please refer part of polynomial interpolation function in " *lagrange.ipynb* ".

In [3]:

(Top)

```
def lagrange(points):

    # 請參考 hands-on 的 Largrange polynomial interpolation
    # ===== 請實做程式 =====
    def f_of_x(x):
        f_x=0
        n=len(points)

        for i in range(n):
            x_i, y_i=points[i]

            def g(i,n):
                g_x=1

                for j in range(n):
                    if i!=j:
                        x_j, y_j=points[j]
                        g_x*=(x-x_j)/float(x_i-x_j)
                return g_x

            f_x+=y_i*g(i,n)

        return f_x
    return f_of_x
    # ====================
```

In [4]:

interpolation_function (Top)

```
# Test
P = lagrange((
    (0, 0),
    (1, 1),
    (-1, 1)
))

print('P(2) =', P(2))

### BEGIN HIDDEN TESTS
P = lagrange((
    (0, 0),
    (1, 1),
    (-1, 1)
))

assert P(0) == 0, 'P(0) is wrong!'
assert P(1) == 1, 'P(1) is wrong!'
assert P(-2) == 4, 'P(-2) is wrong!'
assert P(3) == 9, 'P(3) is wrong!'
### END HIDDEN TESTS
```

```
P(2) = 4.0
```

Part 2. Transfer data to input points (x: dates, y: prices).

In [5]:

(Top)

```
# Hint: points = ((*, *), ..., (*, *))
# ===== 請實做程式 =====
points=((87+10/12, 4.5),(88+11/12, 5.0),(90+10/12, 6.0),(93, 6.5),(95, 7.0),(96, 7.5),(96+10/12, 8.0),(100+10/12, 8.0))

# ====================
```

In [6]:

points_date (Top)

```
print('points:', points)

### BEGIN HIDDEN TESTS
data = np.ndarray.flatten(np.array(points))
prices = [4.5, 5., 6., 6.5, 7., 7.5, 8.]

assert len(data) == 16, 'points is wrong!'
assert np.sum(np.isin(data, prices)) == 8, 'Wrong prices in points!'
### END HIDDEN TESTS
```

```
points: ((87.83333333333333, 4.5), (88.91666666666667, 5.0), (90.83333333333333, 6.0), (93, 6
.5), (95, 7.0), (96, 7.5), (96.83333333333333, 8.0), (100.83333333333333, 8.0))
```

Part 3-1. Estimate the price in February 1989.

In [7]:

(Top)

```
# ===== 請實做程式 =====
P=lagrange(points)
estimated_price=P(89+1/12)
# ====================
```

In [8]:

Feb_1989 (Top)

```
print("My estimated price in February 1989 is", estimated_price)

### BEGIN HIDDEN TESTS
assert abs(estimated_price - 5.09) < 5e-2, 'Estimated price is wrong!'
### END HIDDEN TESTS
```

```
My estimated price in February 1989 is 5.095083945259736
```

Part 3-2. Estimate the price in April 1998.

In [9]:

(Top)

```
# ===== 請實做程式 =====
P=lagrange(points)
estimated_price=P(98+3/12)
# ====================
```

In [10]:

April_1998 (Top)

```
print("My estimated price in April 1998 is", estimated_price)

### BEGIN HIDDEN TESTS
assert abs(estimated_price - 8.67) < 5e-2, 'Estimated price is wrong!'
### END HIDDEN TESTS
```

```
My estimated price in April 1998 is 8.676742602621617
```

Part 3-3. Estimate the price in November 2002.

In [11]:

(Top)

```
# ===== 請實做程式 =====
P=lagrange(points)
estimated_price=P(102+10/12)
# =====================
```

In [12]:

Nov_2002 (Top)

```
print("My estimated price in November 2002 is", estimated_price)

### BEGIN HIDDEN TESTS
assert abs(estimated_price - 11.24) < 5e-2, 'Estimated price is wrong!'
### END HIDDEN TESTS
```

```
My estimated price in November 2002 is 11.241257976731944
```

In [ ]: