In [141…

```python
import numpy as np
import pandas as pd
import datetime as dt

#pd.options.display.float_format = "{:,.2f}".format
```

For calculations of mortgage 1 and 2

In [159…

```python
def fixed_rate_mortgage(r_mort, T_mort, D_mort):
    # monthly payment (inclusive of interest)
    M_mort = D_mort*r_mort/(1-(1+r_mort)**(-T_mort))



    # print(f"Monthly payment: ${round(M_mort1, 2)}")
    #month
    # creating the data frame
    installments = list(range(T_mort+1))
    balance = [M_mort*(1-(1+r_mort)**(inst-T_mort))/r_mort for inst in installments]
    df_mort = pd.DataFrame(balance, columns =['UPB (Million $)'])

    # calculating the interest and principal
    df_mort['Interest ($)'] = df_mort['UPB (Million $)'].shift(1) * r_mort
    df_mort['Principal ($)'] = M_mort - df_mort['Interest ($)']
    df_mort['Installment Amount($)'] = M_mort
    #df_mort['Month of Installment'] = month
    df_mort = df_mort[['Installment Amount($)', 'Principal ($)', 'UPB (Million $)']].round(2).fillna(0) # reorder columns
    df_mort.loc[:0,'Installment Amount($)'] = 0
    return df_mort
```

Mortgage 1

In [160…

```python
r_mort1 = 0.04/12 # monthly interest rate
T_mort1 = 30*12 # number of installments
D_mort1 = 1000000 # initial debt/loan

df_mort_1 = fixed_rate_mortgage(r_mort1, T_mort1, D_mort1)
```

In [161…

```python
df_mort_1
```

Out[161]:

	Installment Amount($)	Principal ($)	UPB (Million $)
0	0.00	0.00	1,000,000.00
1	4,774.15	1,440.82	998,559.18
2	4,774.15	1,445.62	997,113.56
3	4,774.15	1,450.44	995,663.12
4	4,774.15	1,455.28	994,207.84
...	...	...	...
356	4,774.15	4,695.37	18,938.53
357	4,774.15	4,711.02	14,227.50
358	4,774.15	4,726.73	9,500.78
359	4,774.15	4,742.48	4,758.29
360	4,774.15	4,758.29	0.00

361 rows × 3 columns

Mortgage 2

In [162…

```python
r_mort2 = (2.5/100)/12 # monthly interest rate #fixed rate of 2.5%
T_mort2 = 20*12 # number of installments
D_mort2 = 1000000 # initial debt/loan

df_mort_2 = fixed_rate_mortgage(r_mort2, T_mort2, D_mort2)
```

In [163…

```python
df_mort_2
```

Out[163]:

	Installment Amount($)	Principal ($)	UPB (Million $)
0	0.00	0.00	1,000,000.00
1	5,299.03	3,215.70	996,784.30
2	5,299.03	3,222.39	993,561.91
3	5,299.03	3,229.11	990,332.80
4	5,299.03	3,235.84	987,096.97
...	...	...	...
236	5,299.03	5,244.17	21,086.18
237	5,299.03	5,255.10	15,831.08
238	5,299.03	5,266.05	10,565.03
239	5,299.03	5,277.02	5,288.01
240	5,299.03	5,288.01	0.00

241 rows × 3 columns

Mortgage 3

In [146…

```
def create_date_column():
    start = dt.date(1983, 1, 1)
    end = dt.date(2012, 12, 1)

    datex = pd.date_range(start, end, freq='MS')
    date_list = datex.astype(str).tolist()
    date_list.insert(0, '000000000')
    return date_list

date_list = create_date_column()
```

In [147…

```
# change the file path here
file = r'reference_data.csv'
file_x = pd.read_csv(file)
file_x
```

Out[147]:

	Months	Date	Annual interest rate
0	0	NaN	0.00
1	1	1-Jan-83	0.03
2	2	1-Feb-83	0.03
3	3	1-Mar-83	0.03
4	4	1-Apr-83	0.03
...	...	...	...
356	356	1-Aug-12	0.05
357	357	1-Sep-12	0.05
358	358	1-Oct-12	0.05
359	359	1-Nov-12	0.05
360	360	1-Dec-12	0.05

361 rows × 3 columns

In [148…

```
df_mort_3 = pd.DataFrame()
air_list = file_x['Annual interest rate'].tolist()
mir_list = [i/12 for i in air_list]
mir_list_ = mir_list
```

In [149…

```
def indices_for_final_values(fsy, oy, t1):
    initial_list = [t1, 360-fsy]
    second_value = 360-fsy
    while oy < second_value:
        second_value = second_value - oy
        initial_list.append(second_value)
    return initial_list

def ubi_(fsy, oy):
    z = int((((360-12) - fsy)/oy)+1)
    ubilist = [fsy + (oy * i) for i in range(0,z)]
    ubilist.insert(0, 0)
    return ubilist

def final_values(input_list):
    list_2 = []
    list_1 = [input_list[0]]*84

    for i in input_list:
        if i==input_list[0]:
            pass
        else:
            for e in range(12):
                list_2.append(i)
    final_ = list_1 + list_2
    return final_
```

In [150…

```
final_n_values = final_values(indices_for_final_values(12*7, 12, 360))
final_n_values.insert(0, 0)

ubi_final_values = final_values(ubi_(12*7, 12))
ubi_final_values.insert(0, 0)
```

In [151…

```
def mortgage_3_calculations(arm_mir, nl, il):
    unpaid_bal, principal_x, initial_bal, interest_x, installments_y = [1000000], [0], [1000000], [0], [0]
    ub_m = unpaid_bal[0]
    iter_value = 1

    while ub_m > 0 and iter_value < 361 :
        initial_bal.insert(iter_value, unpaid_bal[iter_value-1])
        interest_x.insert(iter_value, initial_bal[iter_value] * arm_mir[iter_value])
        a = arm_mir[iter_value]
        b = nl[iter_value]
        installments_y.insert(iter_value, initial_bal[il[iter_value]+1] * arm_mir[iter_value] / (1-pow(1+a,-b)))
        principal_x.insert(iter_value, installments_y[iter_value] - interest_x[iter_value])
        unpaid_bal.insert(iter_value, initial_bal[iter_value] - principal_x[iter_value])
        iter_value = iter_value+1
    ub_end = abs(round(unpaid_bal[360]))
    unpaid_bal[360] = 0

    return unpaid_bal, principal_x, interest_x, installments_y
```

In [152…

```
ub, pp, ip, ia = mortgage_3_calculations(mir_list, final_n_values, ubi_final_values)
```

In [153…

```
m3_data_frame = {'Date': date_list, 'Annual interest rate': air_list, 'Installment Amount': ia, 'Interest': ip, 'Principal':pp,
df_mort_3 = pd.DataFrame(m3_data_frame)
```

In [154…

```
df_mort_3
```

Out[154]:

	Date	Annual interest rate	Installment Amount	Interest	Principal	UPB - Million $
0	000000000	0.00	0.00	0.00	0.00	1,000,000.00
1	1983-01-01	0.03	4,216.04	2,500.00	1,716.04	998,283.96
2	1983-02-01	0.03	4,216.04	2,495.71	1,720.33	996,563.63
3	1983-03-01	0.03	4,216.04	2,491.41	1,724.63	994,839.00
4	1983-04-01	0.03	4,216.04	2,487.10	1,728.94	993,110.06
...	...	...	...	...	...	...
356	2012-08-01	0.05	6,298.33	130.87	6,167.46	24,930.47
357	2012-09-01	0.05	6,298.33	104.92	6,193.41	18,737.06
358	2012-10-01	0.05	6,298.33	78.85	6,219.48	12,517.58
359	2012-11-01	0.05	6,298.33	52.68	6,245.65	6,271.93
360	2012-12-01	0.05	6,298.33	26.39	6,271.93	0.00

361 rows × 6 columns