Problem #1

Subject: Mortgage Amortization Schedule

Filename: hw03_01.py

Modules: finance.py

Write the first three functions described by the above API. The mortgage_report() function is provided for you in the provided finance.py module. A driver (i.e., top level) program, hw03_01.py, is also provided for you that asks the user for a loan amount (in dollars), APR (in %) and loan term (in years). Also ask for the name a file (without extension) to which to save the table.

Each row of the table should have the information listed in the Background material except use a payment number (starting with 1) instead of a due date. Hint: Use the example as an example!

Exercise good programming practices, including modularization. Write well thought out functions that perform specific tasks.

CS3080-001 – Spring 2024 HW03

In your report, consider the following scenario:

A homebuyer is considering the purchase of a $600,000 home and plans to make a 20% down payment (thus, the loan will be for $480,000). They are considering two options. A 30-year mortgage at 7%, and a 15-year mortgage at 6.875% (these are the current ‘par’ rates at Box Home Loans).

Print the amor□za□on schedule for the first one to the file 480_30_7000.txt and the second to 480_15_6875.txt.

Compare the monthly payments for each option and discuss the impact on the total interest paid over the life of the loan for each op□on.

In addition, examine the amortization schedule printed to the file and determine how many years and months in will be in each case before at least half of the monthly payment will be applied to principal and also how long, in years and months, before the mortgage balance falls below half of its original amount.

WORK

The finance.py module contains several functions related to mortgage calculations. The mortgage_residual function calculates the remaining balance of a mortgage after a specified number of payments. It iterates over the term of the loan, deducting the payment amount minus the interest from the loan balance each month. The mortgage_payment function determines the optimal monthly payment amount given the loan amount, annual interest rate, and loan term. It uses a binary search algorithm to find the payment amount that results in a residual balance closest to zero. The

mortgage_amortization_payment function generates an amortization schedule for the mortgage. It calculates the monthly payment, total payments, and cost of credit, then constructs a summary, header, and table detailing the payment schedule, with payments occurring every 12 months. The **mortgage_report** function generates a mortgage amortization report using the **mortgage_amortization_payment** function. It formats the output and returns it as a string. In the **hw03_01.py** file, the **main** function prompts the user for loan details and a filename. It then generates an amortization schedule and writes it to a text file with the provided filename. Additionally, it prints out an amortization schedule with payments occurring every month.

OUTPUT

```
DATA ENTRY

Enter loan amount ($):..... 480000

Enter loan APR (%):..... 7

Enter loan term (yr):..... 30

Filename (w/o ext):..... 480_30_7000


MORTGAGE AMORTIZATION SCHEDULE

Loan amount: $480000.00

Loan rate: 7.000%

Loan term: 30 years

Monthly payment: $3193.45

Total paid: $1149642.00

Cost of credit: $669642.00


      /-------   PAYMENT   --------\  /---------   TOTAL   ---------\

month  payment  interest principal  interest   principal     paid   balance


   12  3193.45   2774.00     419.45  33288.05    5033.35  38321.40 475124.14

   24  3193.45   2743.68     449.77  65848.38   10794.42  76642.80 469895.80

   36  3193.45   2711.17     482.28  97602.07   17362.13 114964.20 464289.50
```

48	3193.45	2676.30	517.15	128462.62	24822.98	153285.60	458277.93
60	3193.45	2638.92	554.53	158335.21	33271.79	191607.00	451831.77
72	3193.45	2598.83	594.62	187115.98	42812.42	229928.40	444919.63
84	3193.45	2555.85	637.60	214691.25	53558.55	268249.80	437507.80
96	3193.45	2509.76	683.69	240936.57	65634.63	306571.20	429560.17
108	3193.45	2460.33	733.12	265715.81	79176.79	344892.60	421038.01
120	3193.45	2407.33	786.12	288880.13	94333.87	383214.00	411899.78
132	3193.45	2350.51	842.94	310266.80	111268.60	421535.40	402100.95
144	3193.45	2289.57	903.88	329698.02	130158.78	459856.80	391593.76
156	3193.45	2224.23	969.22	346979.56	151198.64	498178.20	380327.00
168	3193.45	2154.16	1039.29	361899.36	174600.24	536499.60	368245.77
180	3193.45	2079.03	1114.42	374225.88	200595.12	574821.00	355291.18
192	3193.45	1998.47	1194.98	383706.50	229435.90	613142.40	341400.11
204	3193.45	1912.09	1281.36	390065.60	261398.20	651463.80	326504.85
216	3193.45	1819.46	1373.99	393002.58	296782.62	689785.20	310532.81
228	3193.45	1720.13	1473.32	392189.70	335916.90	728106.60	293406.15
240	3193.45	1613.62	1579.83	387269.72	379158.28	766428.00	275041.41
252	3193.45	1499.42	1694.03	377853.36	426896.04	804749.40	255349.07
264	3193.45	1376.96	1816.49	363516.49	479554.31	843070.80	234233.17
276	3193.45	1245.64	1947.81	343797.17	537595.03	881392.20	211590.81
288	3193.45	1104.83	2088.62	318192.40	601521.20	919713.60	187311.62
300	3193.45	953.85	2239.60	286154.56	671880.44	958035.00	161277.29
312	3193.45	791.95	2401.50	247087.64	749268.76	996356.40	133360.94
324	3193.45	618.34	2575.11	200343.06	834334.74	1034677.80	103426.51
336	3193.45	432.19	2761.26	145215.19	927784.01	1072999.20	71328.12

```
  348  3193.45    232.58    2960.87  80936.52 1030384.08 1111320.60  36909.33

  360  3193.45     18.53    3174.92   6672.39 1142969.61 1149642.00      2.41



DATA ENTRY

Enter loan amount ($):...... 480000

Enter loan APR (%):......... 6.875

Enter loan term (yr):....... 15

Filename (w/o ext):......... 480_15_6875


MORTGAGE AMORTIZATION SCHEDULE

Loan amount: $480000.00

Loan rate: 6.875%

Loan term: 15 years

Monthly payment: $4280.90

Total paid: $770562.00

Cost of credit: $290562.00


      /-------   PAYMENT   --------\  /---------   TOTAL   ---------\

month  payment  interest principal  interest   principal     paid   balance


   12  4280.90   2650.71    1630.19  31808.52   19562.28  51370.80 461039.13

   24  4280.90   2535.03    1745.87  60840.82   41900.78 102741.60 440732.83

   36  4280.90   2411.15    1869.75  86801.42   67310.98 154112.40 418985.63

   48  4280.90   2278.48    2002.42 109366.87   96116.33 205483.20 395695.29
```

60	4280.90	2136.39	2144.51	128183.28	128670.72	256854.00	370752.30
72	4280.90	1984.22	2296.68	142863.64	165361.16	308224.80	344039.41
84	4280.90	1821.25	2459.65	152984.88	206610.72	359595.60	315431.01
96	4280.90	1646.72	2634.18	158084.74	252881.66	410966.40	284792.61
108	4280.90	1459.80	2821.10	157658.28	304678.92	462337.20	251980.17
120	4280.90	1259.62	3021.28	151154.22	362553.78	513708.00	216839.41
132	4280.90	1045.23	3235.67	137970.85	427107.95	565078.80	179205.12
144	4280.90	815.64	3465.26	117451.65	498997.95	616449.60	138900.37
156	4280.90	569.75	3711.15	88880.60	578939.80	667820.40	95735.67
168	4280.90	306.41	3974.49	51476.97	667714.23	719191.20	49508.08
180	4280.90	24.39	4256.51	4389.79	766172.21	770562.00	0.25

*** I acknowledge that my values under TOTAL (interest, principal) are wrong. I tried correcting this but ran out of time, and I’m curious how you do it properly ***

The total interest climbs up to $393,002.58 when the term is 30 years, while on the flip side, the interest rate climbs up to $158,084.74. The interest is half as much at this point! All these homework problems are making it quite evident how it’s best to pay off a house as soon as you can. 15 year loans are almost always better than 30 years!

For “480_30_7000.txt”, it will take 240 months (20 years) until at least half of the monthly payment will be applied to the principal, and it will take 264 months (22 years) until the mortgage balance falls below half of its original amount.

For “480_15_6875.txt”, it will take 60 months (5 years) until at least half of the monthly payment will be applied to the principal, and it will take 120 months (10 years) until the mortgage balance falls below half of its original amount.

CODE

```
'''
PROGRAMMER: Jakob K. West
USERNAME: jwest21
PROGRAM: finance.py

DESCRIPTION: Mortgage Functions
'''



def mortgage_residual(loan_amount, apr, term_years, payment):

    # Set constraints used in calculations
    term_months = 12 * term_years
    monthly_rate = (apr / 100) /12

    # Simulate payments over the term of the loan
    balance = loan_amount
    for i in range(term_months):
        interest = round(balance * monthly_rate, 2)
        balance -= (payment - interest)

    # Return the residual balance
    return balance
```

```
def mortgage_payment(loan_amount, apr, term_years):

    # Set initial bounds and guess
    lower_bound = round(loan_amount / (12 * term_years), 2)
    upper_bound = round(lower_bound + loan_amount * (apr/1200), 2)

    while (upper_bound - lower_bound) > 0.015:
        payment = round((lower_bound + upper_bound) / 2, 2)
        residual = mortgage_residual(loan_amount, apr, term_years, payment)

        if residual < 0:
            upper_bound = payment

        else:
            lower_bound = payment

    upper_residual = mortgage_residual(loan_amount, apr, term_years, upper_bound)
    lower_residual = mortgage_residual(loan_amount, apr, term_years, lower_bound)

    if abs(upper_residual) < abs(lower_residual):
        payment = upper_bound
        residual = upper_residual
```

```
    else:
        payment = lower_bound
        residual = lower_residual

    return(payment, payment + residual)



def mortgage_amortization_payment(amount, rate, term, increment=1):

    monthly_payment, final_payment = mortgage_payment(amount, rate, term)
    total_paid = monthly_payment * term * 12
    cost_of_credit = total_paid - amount

    # Forming the summary string
    summary = f"Loan amount: ${amount:.2f}\n"
    summary += f"Loan rate: {rate:.3f}%\n"
    summary += f"Loan term: {term} years\n"
    summary += f"Monthly payment: ${monthly_payment:.2f}\n"
    summary += f"Total paid: ${total_paid:.2f}\n"
    summary += f"Cost of credit: ${cost_of_credit:.2f}\n"

    # Forming the header string
    header = "      /-------   PAYMENT   --------\\  /---------   TOTAL   ---------
\\\n"
    header += "month  payment  interest principal  interest   principal     paid
balance\n"
```

```
    # Forming the table string

    table = ""

    for i in range(1, term * 12 + 1, increment):

        interest = amount * (rate / 100 / 12)

        principal = monthly_payment - interest

        amount -= principal



        if (i % 12 == 0):



            table += f"{i:>5} {monthly_payment:>8.2f} {interest:>9.2f}
{principal:>10.2f} "

            table += f"{interest * i:>9.2f} {principal * i:>10.2f} {monthly_payment *
i:>9.2f} {amount:>9.2f}\n"



    return ("MORTGAGE AMORTIZATION SCHEDULE", summary, header, table)

def mortgage_report(amount, rate, term, increment=1):



    report = ""

    for s in mortgage_amortization_payment(amount, rate, term, increment):

        report += s

    return report
```

```
'''
PROGRAMMER: Jakob K. West
USERNAME: jwest21
PROGRAM: hw03_01.py

DESCRIPTION: Mortgage Amortization Table
'''

import finance

def main():

    # Prompt the user for loan details
    print("DATA ENTRY")
    loan_amount = float(input("Enter loan amount ($):..... "))
    apr         = float(input("Enter loan APR (%):..... "))
    term_years  =   int(input("Enter loan term (yr):..... "))
    filename    =   str(input("Filename (w/o ext):..... "))
    print()

    # Generate full and abbreviated Amortization Schedules

    with open(filename + ".txt", "wt") as fp:
        for s in finance.mortgage_amortization_payment(loan_amount, apr, term_years):
```

```
            fp.write(s)


    # Print abbreviated amortization schedule to console

    for s in finance.mortgage_amortization_payment(loan_amount, apr, term_years,
increment=1):

        print(s)




if __name__ == "__main__":

    main()
```

Problem #2

Subject: Accelerated Payoff

*** This homework is ridiculous… I already poured hours into the first part. This class seems to have a major disconnect between what is taught in lecture, what appears on the homework, what the textbook teaches, and what the exams will most likely be like. I made a mistake enrolling in this class, but I will meet you, William, and try to find a strategy to at least pass the class. There gets to a point where you must weigh everything else going on in life with pouring hours into a single homework problem. I reached that point… I'll shoot you an email to schedule an appointment. Thanks! ***

Filename: hw03_02.py

Allowed modules: finance

WORK

To enhance the mortgage_report() function, incorporate two optional parameters named 'fixed' and 'variable' representing percentages. 'fixed' indicates the extra fixed amount to be paid each month, while 'variable' denotes the percentage of the required payment to be added towards the principal. If both parameters default to 0, the function generates a standard amortization schedule; otherwise, it adjusts the payments accordingly. For accelerated payoff options, calculate the time to pay off the loan and the saved finance charges compared to paying off the loan on schedule. In hw03_02.py, simulate a homeowner with a 30-year loan considering two accelerated payoff strategies: paying an extra $500 monthly towards principal and paying an amount equivalent to the percentage of the required payment that goes towards principal. Print the resulting amortization schedules to analyze the effects on the loan payoff time and interest savings.