Problem #1

Subject: Mortgage residual

Filename: hw01_01.py
Allowed modules: NONE

Write a program that asks user to enter a loan amount (in dollars), an APR (in percent), a term (in years), and a payment amount (in dollars). Run a payment simulation over the term of the loan, tracking the remaining balance as on-time payments are made and interest is collected. At the end, provide an analysis report that summarizes the loan and presents the residual amount due after the final payment is made. Using your program, determine the correct minimum payment for a $1,234,567.89 loan taken out for a term of 34 years at an interest rate of 6.283%. What is the final payment amount? This will involve running your program multiple times, trying different payment amounts. With some thought up front, you should be able to find the answer in twenty tries or fewer. Think about a good strategy to use – it will pay off in the next assignment.
Your write-up should describe the strategy you used to determine the correct minimum payment.

WORK

After receiving the loan amount, APR, term length, and monthly payment from the user, the monthly interest rate is found by dividing the APR by 1200. A for loop is then executed for each month in the term. The interest amount for each month is then added to the total and then the payment is made. Doing this for all months in the term will give us the final payment the user has to make, and the residual balance after this payment is made.

OUTPUT

DATA ENTRY

Enter loan amount ($): 1234567.89

Enter loan APR (%): 6.283

Enter loan term (yrs): 34

Enter monthly payment ($): 7335.12

ANALYSIS RESULTS

Loan Rate: 6.283%

Loan Term: ..... 34 years

Loan Amount: $1234567.89

Monthly Payment: ..... $ 7335.12

Residual Balance: .... $ -10.14

Final Amount: ..... $ 7345.26

CODE

'''

PROGRAMMER:..Christopher Colbert

USERNAME:....ccolbert

PROGRAM:.....hw01_01.py

DESCRIPTION: Simulate a loan given the amount, APR, term length, and minimum payment. Outputs the final payment amount, and the residual balance.

'''

```
#recieve user input
print("DATA ENTRY")
loan_amount = float(input("Enter loan amount ($): "))
loan_apr = float(input("Enter loan APR (%): "))
loan_term = int(input("Enter loan term (yrs): "))
loan_monthly_payment = float(input("Enter monthly payment ($): "))

#Convert APR to monthly interest and monthly payment amount
monthly_interest_rate= (loan_apr/100)/12
remaining_balance=loan_amount

#for each month
for month in range(1, loan_term*12 + 1):
  #subtracting interest from payment
  monthly_interest_amount= round(remaining_balance * monthly_interest_rate, 2)
  principal_payment = loan_monthly_payment - monthly_interest_amount
  #take payment - interest out of balance
  remaining_balance -= principal_payment

#Print results
print("ANALYSIS RESULTS")
print("Loan Rate: ..... %11.3f%%" % loan_apr)
print("Loan Term: ..... %11d years" % loan_term)
print("Loan Amount: ..... $%10.2f" % loan_amount)
print("Monthly Payment: ..... $%10.2f" % loan_monthly_payment)
print("Residual Balance: .... $%10.2f" % remaining_balance)
print("Final Amount: ..... $%10.2f" % (loan_monthly_payment - remaining_balance))
```

Problem #2

Subject: Value of pi

Filename: hw01_02.py
Allowed modules: random, math

Write a program that asks the user for the number of trials to make. For each trial, generate a random point within a square region and determine if that point is also within the circular region. From this data, calculate the estimated value of pi and report the percentage error in this estimate.

WORK

By using a 2x2 sized square, a circle will fit at the origin with a radius of 1. This means that at any point inside the circle, $x^2 + y^2$ will be less than or equal to 1. Using this sized circle also means that the area will be equal to pi. The ratio of the areas will be equal to $\frac{\pi}{4}$, thus multiplying this ration by 4 will estimate the value of pi.

OUTPUT

DATA ENTRY

Enter how many runs to make: 10000

ESTIMATE OF PI

Number of runs:..... 10000

Estimate of pi:..... 3.14920000

Error:.....%0.76073465

DATA ENTRY

Enter how many runs to make: 1000

ESTIMATE OF PI

Number of runs:..... 1000

Estimate of pi:..... 3.21200000

Error:.....%7.04073465

DATA ENTRY

Enter how many runs to make: 100000

ESTIMATE OF PI

Number of runs:..... 100000

Estimate of pi:..... 3.13448000

Error:.....%0.71126535

CODE

```
'''
PROGRAMMER:..Christopher Colbert
USERNAME:....ccolbert
PROGRAM:.....hw01_02.py

DESCRIPTION: Plots random points within a square region, and checks
if the points are inside of a circle centered in the square. The program
then estimates the value of pi, and calculates how far that value is from the
actual value.
'''
import random

#recieve user input
print("DATA ENTRY")
runs = int(input("Enter how many runs to make: "))
successful_attempts=0
```

```
#for number of runs user inputs
for trial in range(runs):
  #generate a random point within the 2x2 square centered at the origin
  x = random.uniform(-1,1)
  y = random.uniform(-1,1)

  if x**2 + y**2 <= 1:
    successful_attempts += 1

#estimate pi and find the percentage of difference between that and the real value of pi
pi = 3.1415926535
pi_estimate = float(4 * successful_attempts / runs)
percent_error = abs(pi_estimate - pi) * 100

print("ESTIMATE OF PI")
print("Number of runs:..... %-10d" %runs)
print("Estimate of pi:..... %-2.8f" %pi_estimate)
print("Error:.............%%%1.8f" %percent_error)
```

Problem #3

Subject: Average seek time

Filename: hw01_03.py
Allowed modules: random

Write a program that asks the user for the number of tracks on the disk and how many trials to make. For each trial, pick a starting and target track randomly and determine the distance moved (in tracks) and report the average number of tracks moved over the course of the simulation. Using your program, what is the average number of tracks moved per access for the following number of tracks: 1, 2, 100, 1,000,000?

WORK

After the user enters the number of tracks, and number of runs to make, we will pick a random start and end point within the number of tracks chosen. We will then determine the distance between them by subtracting the end from the start. Adding up the distance of each run and dividing by the total number of runs will simulate going from location to location within hard drive storage, and how long on average it would take.

OUTPUT

DATA ENTRY

Enter number of tracks:.....1

Enter number of runs to make:..1000

SIMULATION RESULTS

Number of tracks:.... 1

Number of runs:..... 1000

Average distance:.... 0.00 tracks

DATA ENTRY

Enter number of tracks:.....2

Enter number of runs to make:..1000

SIMULATION RESULTS

Number of tracks:.... 2

Number of runs:..... 1000

Average distance:.... 0.49 tracks

DATA ENTRY

Enter number of tracks:.....100

Enter number of runs to make:..1000

SIMULATION RESULTS

Number of tracks:.... 100

Number of runs:..... 1000

Average distance:.... 32.25 tracks

DATA ENTRY

Enter number of tracks:.....1000000

Enter number of runs to make:..1000

SIMULATION RESULTS

Number of tracks:.... 1000000

Number of runs:..... 1000

Average distance:.... 335824.47 tracks

CODE

```
'''
PROGRAMMER:..Christopher Colbert
USERNAME:....ccolbert
PROGRAM:.....hw01_03.py

DESCRIPTION: Simulate the average seek time of a hard drive
'''
import random

print("DATA ENTRY")
tracks = int(input("Enter number of tracks:........"))
runs = int(input("Enter number of runs to make:.."))
total_distance = 0

#for amount of runs user inputs
for trials in range(runs):
  #pick random start and end track to determine number of tracks moved
  start_track = random.randint(0, tracks - 1)
  end_track = random.randint(0, tracks - 1)

  distance_moved = abs(end_track - start_track)
  total_distance += distance_moved

#calculate average distance moved between all runs
average_distance = float(total_distance / runs)
```

```
print("SIMULATION RESULTS")
print("Number of tracks:.... %-6d" %tracks)
print("Number of runs:..... %-6d" %runs)
print("Average distance:.... %-4.2f tracks" %average_distance)
```