Two Similar Programs

```
i = 0
while i < 100000000:
    i = i + 1

month = 0
curPrin = input(”Init. Prin:”)
intRate = input(”Rate:”)
paymt = input(”Payment:”)
TOTAL = 0.0
while curPrin > 0.0:
    intMo = …
    curPrin = …
    TOTAL = TOTAL + paymt
    month = month + 1
    print month, curPrin, TOTAL

where intMo = curPrin * intRate
and curPrin = curPrin + intMo - paymt
```
Two Similar Programs

```python
i = 0
while i < 100000000:
    i = i + 1

month = 0
curPrin = input("Init. Prin:")
intRate = input("Rate:")
paymt = input("Payment:")
TOTAL = 0.0
while curPrin > 0.0:
    month = month + 1
    intMo = ...
    curPrin = ...
    TOTAL = TOTAL + paymt
    print(month, curPrin, TOTAL)
```

Controlled by a condition affected by the loop body

Where \( \text{intMo} = \text{curPrin} \times \text{intRate} \)
and \( \text{curPrin} = \text{curPrin} + \text{intMo} - \text{paymt} \)
Vocabulary

- **Values** like 1, 2, 33.3
- **Expressions** like $1 + 2$ express **operations** whose results are values.
- **Variables** - have values
- **Assignment** operation
- **operations** like `forward()` and `print`
- **loop** expressed by a **while** statement
- **condition** like $i < 10000$ in a **while** statement.
Design Programs AROUND VARIABLES

- Current principal
- Monthly interest rate
- Current month's interest
- Monthly payment
- Number of months into the loan
- Total payments so far

Believe it or not, making the VARIABLES the chief concerns is the lesson of today's successful OBJECT ORIENTED PROGRAMMING practices!
Explain the Loan Program

month = 0  # At the beginning of the loan, no (zero) # months have passed. We are at month 0 of the loan

curPrin = input("Init. Prin:"")
# The current amount of money we owe the bank starts # out with what we borrowed.

intRate  = input("Rate:")
# The interest rate is fixed, but it is convenient to # use a variable for it so that the SAME PROGRAM can # be used with loans with different interest rates.

paymt   = input("Payment:")
# Same idea to use a VARIABLE for the constant monthly # payment.
TOTAL  =  0.0

I did ask you to do this! Use a variable to
ACCUMULATE the TOTAL (sum) of payments we make to bank.

while curPrin > 0.0:
    #This CONDITION makes the LOOP continue as long as
    #the amount we owe at the beginning of a month is
    # greater than ZERO.

    month = month + 1
    # Count this month.

    intMo = curPrin * intRate
    # This variable is used for clarity and convenience.

    curPrin = curPrin + intMo - paymt
    # Change amount owed at beginning to end of month.
TOTAL = TOTAL + paymt
#Add the current month's payment to the
#TOTAL (sum) of payments.

print month,curPrin,TOTAL
#END OF LOOP BODY

Python feature: this print operation
prints the three numbers on the SAME
line because of the commas (,)
Team Exercise...Step 1 Make it into a function

def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
    print month, curPrin, TOTAL
Step 1.1 TEST IT!!!

def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
        print month, curPrin, TOTAL

a = input('Loan Amt:')
r = input('Rate:')
x = input('Payment:')
loanCalc(a, r, x)
Step 2 Make the function RETURN the number of months needed to pay the loan.

```python
def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
        print(month, curPrin, TOTAL)
    return month

a = input('Loan Amt:')
r = input('Rate:')
x = input('Payment:')
loanCalc(a, r, x)
```
Step 2.1 TEST IT!!!

```python
def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
        print month, curPrin, TOTAL

    return month

a = input('Loan Amt:')
r = input('Rate:')
x = input('Payment:')
n = loanCalc(a, r, x)
print 'That loan lasted ', n, 'months'
```
Step 3 Disable monthly printout..TEST IT!!!

```python
def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
        if True:
            print month, curPrin, TOTAL
    return month

a = input('Loan Amt:')
r = input('Rate:')
x = input('Payment:')
n = loanCalc(x, y, z)
print 'That loan lasted ', n, 'months'
```

Change `True` to `False` to stop the long printout.
Step 4: The interesting part...

```python
def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
    return month
```

```
a = input('Loan Amt:')  
r = input('Rate:')      
x = input('Payment:')   
nMonthsGoal = input('Months WANTED:')
```

Make the computer FIGURE OUT what monthly payment x [unknown!] WOULD make the loan last a given, desired number of months.
Step 4: The interesting part...

def loanCalc(curPrin, intRate, paymt):
    month = 0
    TOTAL = 0.0
    while curPrin > 0.0:
        month = month + 1
        curInt = curPrin * intRate
        curPrin = curPrin + curInt - paymt
        TOTAL = TOTAL + paymt
        if False:
            print month, curPrin, TOTAL
    return month

a = input('Loan Amt:')

r = input('Rate:')

x = input('Payment:')  # nMonthsGoal = input('Months WANTED:')

Begin exploring...Use $20,000, 0.0066667/month, and use this or your program BY HAND to figure the PAYMENT out for 3 months, 12 months, 120, 360, etc.
Why bother...?

- What if we want to figure out what speed and/or time the robot should move so it reaches a goal position?
- The position depends in a complicated way on speed, time and other factors.
- A computer Algorithm to pick the speed or time would be useful here. It's like picking the monthly payment.