Problem 2: Paying Debt Off in a Year

(15/15 points)

Now write a program that calculates the minimum **fixed** monthly payment needed in order pay off a credit card balance within 12 months. By a fixed monthly payment, we mean a single number which does not change each month, but instead is a constant amount that will be paid each month.

In this problem, we will *not* be dealing with a minimum monthly payment rate.

The following variables contain values as described below:

1. `balance` - the outstanding balance on the credit card
2. `annualInterestRate` - annual interest rate as a decimal

The program should print out one line: the lowest monthly payment that will pay off all debt in under 1 year, for example:

```
Lowest Payment: 180
```

Assume that the interest is compounded monthly according to the balance at the end of the month (after the payment for that month is made). The monthly payment must be a multiple of $10 and is the same for all months. Notice that it is possible for the balance to become negative using this payment scheme, which is okay. A summary of the required math is found below:

> ***Monthly interest rate*** *= (Annual interest rate) / 12.0*
> ***Monthly unpaid balance*** *= (Previous balance) - (Minimum fixed monthly payment)*
> ***Updated balance each month*** *= (Monthly unpaid balance) + (Monthly interest rate x Monthly unpaid balance)*

Test Cases to Test Your Code With. Be sure to test these on your own machine - and that you get the same output! - before running your code on this webpage!

Click to See Problem 2 Test Cases

Be sure to test these on your own machine - and that you get the same output! - before running your code on this webpage!

Test Cases:

1.

```
Test Case 1:
balance = 3329
annualInterestRate = 0.2

Result Your Code Should Generate:
-------------------
Lowest Payment: 310
```

2.

```
Test Case 2:
balance = 4773
annualInterestRate = 0.2

Result Your Code Should Generate:
-------------------
Lowest Payment: 440
```

3.

```
Test Case 3:
balance = 3926
annualInterestRate = 0.2

Result Your Code Should Generate:
-------------------
Lowest Payment: 360
```

The code you paste into the following box **should not** specify the values for the variables `balance` or `annualInterestRate` - our test code will define those values before testing your submission.

Test Case 1

```
balance = 3329; annualInterestRate = 0.2
```

Output:

```
Lowest Payment: 310
```

Test Case 2

```
balance = 4773; annualInterestRate = 0.2
```

Output:

```
    Lowest Payment: 440
```

Test Case 3

```
balance = 3926; annualInterestRate = 0.2
```

Output:

```
    Lowest Payment: 360
```

Randomized Test Case 1

```
balance = 38; annualInterestRate = 0.2
```

Output:

```
    Lowest Payment: 10
```

Randomized Test Case 2

```
balance = 974; annualInterestRate = 0.2
```

Output:

```
    Lowest Payment: 90
```

Randomized Test Case 3

```
balance = 344; annualInterestRate = 0.18
```

Output:

```
    Lowest Payment: 40
```

Randomized Test Case 4

```
balance = 930; annualInterestRate = 0.25
```

Output:

```
    Lowest Payment: 90
```

Randomized Test Case 5

```
balance = 3629; annualInterestRate = 0.2
```

Output:

```
Lowest Payment: 340
```

Randomized Test Case 6

```
balance = 3175; annualInterestRate = 0.18
```

Output:

```
Lowest Payment: 290
```

Randomized Test Case 7

```
balance = 4921; annualInterestRate = 0.04
```

Output:

```
Lowest Payment: 420
```

Randomized Test Case 8

```
balance = 3017; annualInterestRate = 0.2
```

Output:

```
Lowest Payment: 280
```

Randomized Test Case 9

```
balance = 4172; annualInterestRate = 0.04
```

Output:

```
Lowest Payment: 360
```

Randomized Test Case 10

```
balance = 3892; annualInterestRate = 0.04
```

Output:

```
Lowest Payment: 340
```

Randomized Test Case 11

`balance = 4804; annualInterestRate = 0.2`

Output:

```
Lowest Payment: 440
```

Randomized Test Case 12

`balance = 4886; annualInterestRate = 0.18`

Output:

```
Lowest Payment: 450
```

Hints

Hint: How to think about this problem?

- Start with $10 payments per month and calculate whether the balance will be paid off in a year this way (be sure to take into account the interest accrued each month).
- If $10 monthly payments are insufficient to pay off the debt within a year, increase the monthly payment by $10 and repeat.

Hint: A way of structuring your code

- If you are struggling with how to structure your code, think about the following:
  - Given an initial balance, what code would compute the balance at the end of the year?
  - Now imagine that we try our initial balance with a monthly payment of $10. If there is a balance remaining at the end of the year, how could we write code that would reset the balance to the initial balance, increase the payment by $10, and try again (using the same code!) to compute the balance at the end of the year, to see if this new payment value is large enough.
  - I'm still confused!
    A good way to implement this problem will be to use a loop structure. You may want to refresh your understanding of **while** loops. Think hard about how the program will know when it has found a good minimum monthly payment value - when a good value is found, the loop can terminate.
- Be careful - you don't want to overwrite the original value of `balance`. You'll need to save that value somehow for later reference!