Assignment 6
Due Date for Programs: October 26, 2016

☞

The parts of this assignment marked with the ”alone” icon, , should be completed individually by each student.

You may discuss any parts of the assignments with your classmates and tutors (or anyone else) but you are responsible for understanding all of the parts of the assignment that you submit. Any sharing or copying of the parts of the assignments marked with ”alone” icon, , will be considered cheating.

You should not use any features of Python that have not been covered in class or recitations. If you have doubt if you are allowed to use certain structures, just ask one of the instructors.

Problem 1 (10 points): Moon and toilet paper

Complete the worksheet at the end of this assignment. You should work on this with a partner.

Problem 2 (20 points): Powers

Write a program that computes powers from 0 to 10 of a number entered by the user. Use a `for` loop in your solution.

Your program should prompt the user for an integer and print the results in the format matching the sample outputs below (make sure that the ^ signs and the = signs are aligned.

Here a few sample runs of the program:

Output:

```
Enter a number 2
2 ^  0  =  1
2 ^  1  =  2
2 ^  2  =  4
2 ^  3  =  8
2 ^  4  =  16
2 ^  5  =  32
2 ^  6  =  64
2 ^  7  =  128
2 ^  8  =  256
2 ^  9  =  512
2 ^ 10  =  1024
```

Output:

```
Enter a number -8
-8 ^  0  =  1
-8 ^  1  =  -8
-8 ^  2  =  64
-8 ^  3  =  -512
-8 ^  4  =  4096
-8 ^  5  =  -32768
-8 ^  6  =  262144
-8 ^  7  =  -2097152
-8 ^  8  =  16777216
-8 ^  9  =  -134217728
-8 ^ 10  =  1073741824
```

Output:

```
Enter a number 0
0 ^  0  =  1
0 ^  1  =  0
0 ^  2  =  0
0 ^  3  =  0
0 ^  4  =  0
0 ^  5  =  0
0 ^  6  =  0
0 ^  7  =  0
0 ^  8  =  0
0 ^  9  =  0
0 ^ 10  =  0
```

Output:

```
Enter a number 111
111 ^  0  =  1
111 ^  1  =  111
111 ^  2  =  12321
111 ^  3  =  1367631
111 ^  4  =  151807041
111 ^  5  =  16850581551
111 ^  6  =  1870414552161
111 ^  7  =  207616015289871
111 ^  8  =  23045377697175681
111 ^  9  =  2558036924386500591
111 ^ 10  =  283942098606901565601
```

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else)

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `powers.py`. You only need to submit the source code for this problem.

Problem 3 (30 points): Interest Earned

In class we talked about two different bank accounts: one using simple interest, the other using compound interest. We computed the functions that determine the amounts in each account after the specific number of years:

SaveWithUs: $b(y) = 1000.00 + 100y$ it is a linear function

BetterSavings: $b(y) = 1000.00 \times (1 + 0.08)^y$ it is an exponential function

(Note that the above functions assume that the starting balance is $1000.00.)

Write a program that performs a similar calculation but based on a user-specified starting balance.

The program should prompt the user for an initial balance and then prduce a table that shows the balances after each year in the two accounts. Remember that (1) SaveWithUs pays a flat interest of 10% of the initial investment at the end of each year; (2) BetterSavings pays a varying interest of 8% of the year's balance at the end of each year. You will need to modify the above formulas to work with the user-specified beginning balance.

The output of your program should be in a form of a table like the ones shown in the sample output below. The results should inidate which account has a higher balance by prinining a star next to the amount. All numbers should be printed with exactly two decimal places and comma-separators.

THe program should validate the input. If the user enters a negative value for the balance, the program should print an error message and reprompt.

Your program should use a `for` loop.

Here is the run of the program (it should always look the same):

Output:

```
Enter your initial balance: 10000.00
```

```
Year   |    SaveWithUs      |    BestSavings
--------------------------------------------------
 1     |        11,000.00 * |        10,800.00
 2     |        12,000.00 * |        11,664.00
 3     |        13,000.00 * |        12,597.12
 4     |        14,000.00 * |        13,604.89
 5     |        15,000.00 * |        14,693.28
 6     |        16,000.00 * |        15,868.74
 7     |        17,000.00   |        17,138.24 *
 8     |        18,000.00   |        18,509.30 *
 9     |        19,000.00   |        19,990.05 *
 10    |        20,000.00   |        21,589.25 *
 11    |        21,000.00   |        23,316.39 *
 12    |        22,000.00   |        25,181.70 *
 13    |        23,000.00   |        27,196.24 *
 14    |        24,000.00   |        29,371.94 *
 15    |        25,000.00   |        31,721.69 *
 16    |        26,000.00   |        34,259.43 *
 17    |        27,000.00   |        37,000.18 *
 18    |        28,000.00   |        39,960.19 *
 19    |        29,000.00   |        43,157.01 *
 20    |        30,000.00   |        46,609.57 *
```

Output:

```
Enter your initial balance: -100
Error: initial balance cannot be negative.


Enter your initial balance: 5500.00


Year   |    SaveWithUs      |    BestSavings
--------------------------------------------------
 1     |         6,050.00 * |         5,940.00
 2     |         6,600.00 * |         6,415.20
 3     |         7,150.00 * |         6,928.42
 4     |         7,700.00 * |         7,482.69
 5     |         8,250.00 * |         8,081.30
 6     |         8,800.00 * |         8,727.81
 7     |         9,350.00   |         9,426.03 *
 8     |         9,900.00   |        10,180.12 *
 9     |        10,450.00   |        10,994.53 *
 10    |        11,000.00   |        11,874.09 *
 11    |        11,550.00   |        12,824.01 *
 12    |        12,100.00   |        13,849.94 *
 13    |        12,650.00   |        14,957.93 *
 14    |        13,200.00   |        16,154.56 *
 15    |        13,750.00   |        17,446.93 *
 16    |        14,300.00   |        18,842.68 *
 17    |        14,850.00   |        20,350.10 *
 18    |        15,400.00   |        21,978.11 *
 19    |        15,950.00   |        23,736.36 *
 20    |        16,500.00   |        25,635.26 *
```

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else)

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `interest.py`. You only need to submit the source code for this problem.

Problem 4 (20 points): Testing fairness of a die

If you roll an actual six-sided fair die the chances of rolling any specific value should be exactly the same. We would like that to be true for our Python simulations as well.

In this program you will test if the random number generator that we used in the past can simulate a fair six-sided die.

Write a program that prompts the user for the number of rolls and then simulates rolling a die that number of times. As your program simulates a die, it should count how many times each value was rolled (Hint: you will need six different counters for that).

Once the simulation is complete, the program should print the report specifying what percentage of the total number of rolls each value occured.

Here is a sample run of the program (notice that your program will never produce results identical to the sample results since there is randomness involved) :

Output:

```
Enter the number of rolls: 100
1 was rolled 14.000000% of times
2 was rolled 11.000000% of times
3 was rolled 22.000000% of times
4 was rolled 21.000000% of times
5 was rolled 14.000000% of times
6 was rolled 18.000000% of times
```

Output:

```
Enter the number of rolls: 10000000
1 was rolled 16.671420% of times
2 was rolled 16.645640% of times
3 was rolled 16.678590% of times
4 was rolled 16.669160% of times
5 was rolled 16.663400% of times
6 was rolled 16.671790% of times
```

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else)

Extra credit (Optional): Add a second part to the program that simulates the same number of rolls of two six-sided die. The report part of the program should have 11 lines (one for each values from 2 to 12). To do this right you have to think about how the simulation should be done to be realistic and represent actual chances of obtaining any of the values from 2-12. With two fair die, the chances of getting different values are no longer the same.

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `fair_die.py`. You only need to submit the source code for this problem.

Problem 5 (20 points): Drawing

Write a program that uses the Turtle graphics module to draw something. Be as creative or as boring as you wish. There are the following restrictions on your program:

- you have to use at least two colors
- you have to use a loop (either a for loop or a while loop)

You can use any elements covered in the modules 1-5.

Comment your source code by 1) briefly describing parts of your program 2) including your name, the date, and your class section at top of your file (above everything else)

What to Submit

This program should be named (i.e., the name of the file containing the program should be) `drawing.py`. You only need to submit the source code for this problem.

Grading

The only way to receive the credit for the worksheet problems is to hand them in before the end of the lab session in which they are given.

The programs are graded based on correctness, style of code, design and documentation.

What and how to submit?

You should submit the source code file for each program to NYU Classes by the due date stated above. Make sure that you get an email confirmation after you submit the assignment. You should keep that email until the grades are returned - it is your proof that the assignment was submitted! If you do not get an email confirmation, you should try to resubmit the assignment. If you do not get that email, it means that we did not get your assignment.

Name(s) and NetId(s):

The thickness of toilet paper is 0.001 inch. Pretty thin, isn't it.

- How thick is the "stack" of toilet paper after 1 fold? 2 folds? 5 folds? You are always folding the previously folded piece in half.

- Write a formula to calculate the thickness of the folded toilet paper after n folds.

- How many times would you have to fold it (note that this may not be possible in practice) to get 1-foot thick stack of toilet paper? You can estimate it or calculate it exactly, but do not guess. Use the function from the previous step to do this.

- Figure out how far the Moon is from Earth (google might come in handy).

- Which do you think is larger: the distance from the Moon to Earth or the thickness of the stack of toilet paper folded 50 times?

  (Would you have given the same answer before learning about the ”Rice and Chessboard Story”?)

- Calculate the approximate number of miles in the stack of toilet paper after 50 folds.