





## Assignment 8

Due Date for Programs: November 9, 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): Using `help()` Function

All programming languages come pre-packaged with a standard library of functions that are designed to make your job as a programmer easier. Some of these functions are built right into the "core" of Python (`print`, `input`, `range`, etc). Other more specialized functions are stored in a series of files called "modules" that Python can access upon request by using the `import` statement:

- `import random`
- `import math`
- `import turtle`
- ...

You can list the functions that exist in a particular module by using the `help()` function. The `help()` function takes one argument (a string that represents the name of the module) and returns the user manual for that module.

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

### Problem 2 (25 points): `what_power()` Function

Write a function that implements the `what_power` function that we discussed in the lecture. This function should not use any of the built-in logarithmic functions from Python. Your function should take `base` and `result` as arguments and compute the value of the exponent for which  $base^{exponent} = result$ . Your function needs to operate only on integers (i.e. all values of `base`, `result` and `exponent` can be assumed to be integers). If no such value can be found the function should return -1. You may want to review the exercises that we went over in class to remember the rules for the base and result for which the computations could not be performed.

The program should prompt the user for the value of a base and a result and either print the answer or print a message stating that the calculation was not successful.

Here are a few sample runs of the program:

#### Output:

```
I calculate exponents for base^??=result.
```

```
Enter the base (integer): 3
```

```
Enter the result (integer): 81
```

```
The answer is: 4
```

```
3 ^ 4 = 81
```



**Output:**

```
I calculate exponents for base^???=result.  
  
Enter the base (integer): -4  
Enter the result (integer): 64  
  
Sorry, I do not know how to calculate that
```

**Output:**

```
I calculate exponents for base^???=result.  
  
Enter the base (integer): 2  
Enter the result (integer): 15  
  
Sorry, I do not know how to calculate that
```

**Output:**

```
I calculate exponents for base^???=result.  
  
Enter the base (integer): 0  
Enter the result (integer): 5  
  
Sorry, I do not know how to calculate that
```

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) 3) **documenting all the functions following the IPO format**

**What to Submit**

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



**Problem 3 (25 points): Logarithms Tables**

Write a program that prompts the user to enter a base for a logarithm and then computes and prints the values of  $\log_{base}(x)$  for values of  $x$  ranging from 0.25 to 5.0 in increments of 0.25. If the user enters an invalid base, the program should print an error (the base of a logarithm cannot be less than or equal to zero and it cannot equal to 1). If the user input is not valid, the program should terminate (do not give the user another chance).

Here is a sample run of the program:

**Output:**

```
Enter a number: 2  
log base 2 of 0.25 is -2.000  
log base 2 of 0.50 is -1.000  
log base 2 of 0.75 is -0.415  
log base 2 of 1.00 is 0.000  
log base 2 of 1.25 is 0.322  
log base 2 of 1.50 is 0.585  
log base 2 of 1.75 is 0.807  
log base 2 of 2.00 is 1.000  
log base 2 of 2.25 is 1.170  
log base 2 of 2.50 is 1.322  
log base 2 of 2.75 is 1.459  
log base 2 of 3.00 is 1.585  
log base 2 of 3.25 is 1.700  
log base 2 of 3.50 is 1.807  
log base 2 of 3.75 is 1.907
```



```
log base 2 of 4.00 is 2.000
log base 2 of 4.25 is 2.087
log base 2 of 4.50 is 2.170
log base 2 of 4.75 is 2.248
log base 2 of 5.00 is 2.322
```

**Output:**

```
Enter a number: -3
This is not a valid base.
```

**Output:**

```
Enter a number: 56
log base 56 of 0.25 is -0.344
log base 56 of 0.50 is -0.172
log base 56 of 0.75 is -0.071
log base 56 of 1.00 is 0.000
log base 56 of 1.25 is 0.055
log base 56 of 1.50 is 0.101
log base 56 of 1.75 is 0.139
log base 56 of 2.00 is 0.172
log base 56 of 2.25 is 0.201
log base 56 of 2.50 is 0.228
log base 56 of 2.75 is 0.251
log base 56 of 3.00 is 0.273
log base 56 of 3.25 is 0.293
log base 56 of 3.50 is 0.311
log base 56 of 3.75 is 0.328
log base 56 of 4.00 is 0.344
log base 56 of 4.25 is 0.359
log base 56 of 4.50 is 0.374
log base 56 of 4.75 is 0.387
log base 56 of 5.00 is 0.400
```

**Output:**

```
Enter a number: 0.5
log base 0.5 of 0.25 is 2.000
log base 0.5 of 0.50 is 1.000
log base 0.5 of 0.75 is 0.415
log base 0.5 of 1.00 is -0.000
log base 0.5 of 1.25 is -0.322
log base 0.5 of 1.50 is -0.585
log base 0.5 of 1.75 is -0.807
log base 0.5 of 2.00 is -1.000
log base 0.5 of 2.25 is -1.170
log base 0.5 of 2.50 is -1.322
log base 0.5 of 2.75 is -1.459
log base 0.5 of 3.00 is -1.585
log base 0.5 of 3.25 is -1.700
log base 0.5 of 3.50 is -1.807
log base 0.5 of 3.75 is -1.907
log base 0.5 of 4.00 is -2.000
log base 0.5 of 4.25 is -2.087
log base 0.5 of 4.50 is -2.170
log base 0.5 of 4.75 is -2.248
log base 0.5 of 5.00 is -2.322
```

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) 3) **documenting all the functions following the IPO format**



### What to Submit

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



### Problem 4 (40 points): Dice Game

You are going to implement a dice game. The game has two players: the user of your program and the computer. The user starts by rolling three dice. The user can opt to roll again or stop (for the max of three rolls) - the last roll results are used in the game. The computer rolls only once. The winner is determined using the following rules (the rules are applied in the order stated):

- If one of the players rolls three identical values - that's the winner. If both players roll three identical values (for example: 2-2-2 and 5-5-5), then the one with higher total score is the winner (i.e., 5-5-5). If the total combined scores are the same, it is a tie.
- If one of the players rolls two identical values - that's the winner. If both players roll two identical values (for example: 2-6-2 and 1-3-3), then the one with higher total score is the winner (i.e., 2-6-2). If the total combined scores are the same, it is a tie.
- The player with the higher total combined score wins. If scores are the same, it is a tie.

You should be defining functions for this program:

- `roll_three()` function should generate and return values of three dice

- `three_equal(val1, val2, val3)` function should return `True` if all three values passed to it are identical and `False` otherwise

- `two_equal(val1, val2, val3)` function should return `True` if at least two of the three values passed to it are identical and `False` otherwise

You can have more functions if you wish.

Here are some runs of the program:

#### Output:

```
Your current roll is: 3 3 6 sum = 12
Do you wanto to roll again? [y / n]y

Your current roll is: 4 4 2 sum = 10
Do you wanto to roll again? [y / n]n

Computer's roll is: 6 2 3 sum = 11

*****
And the winner is: YOU
*****
```

#### Output:

```
Your current roll is: 3 3 4 sum = 10
Do you wanto to roll again? [y / n]y

Your current roll is: 1 3 4 sum = 8
Do you wanto to roll again? [y / n]y

Your current roll is: 1 6 4 sum = 11

Computer's roll is: 4 3 6 sum = 13

*****
And the winner is: Computer
*****
```

#### Output:

```
Your current roll is: 6 6 6 sum = 18
```



```
Do you wanto to roll again? [y / n]n
Computer's roll is: 4 4 4 sum = 12
*****
And the winner is: YOU
*****
```

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) 3) **documenting all the functions following the IPO format**

### What to Submit

This program should be named (i.e., the name of the file containing the program should be) `dice.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):

---

### **help () with modules**

- In the IDLE's command line (not in a program file), type the following two commands:

```
import math
help(math)
```

You are looking at the manual for the `math` module (well, at the end of it since the manual is long and it scrolls all the way to the bottom when you display it in IDLE). Scroll up until you see description of the `log` functions. List all of the different log functions that are there. Describe them briefly in your own words.

Find and describe another interesting/useful function and give example of its use in a very very short (2-3 lines) Python program.

- Use the same technique to open the manual page for the `random` module. Scroll up to the description of `randint`, `random` and `randrange`. Explain in your own words how the functions differ and what they do.