Part 1

a) Compute (you can use any tool or software to compute answers in this part – but if you do not know to perform this computation, please talk to me about your course prerequisites):

2^{11}

```
In [2]: 2**11
Out[2]: 2048
```

$(2^4)^4$

```
In [3]: (2**4)**4
Out[3]: 65536
```

4^4

```
In [4]: 4**4
Out[4]: 256
```

8^5

```
In [5]: 8**5
Out[5]: 32768
```

837 MOD 100 (MOD is the modulo operator, a.k.a. the remainder)

```
In [6]: 837 % 100
Out[6]: 37
```

842 MOD 20

```
In [7]: 842 % 20
Out[7]: 2
```

23 MOD 112

```
In [8]: 23 % 112
Out[8]: 23
```

112 MOD 23

```
In [9]: 112 % 23
Out[9]: 20
```

b) Given vectors V1 = (1, 1, 3) and V2 = (1, 2, 2) and a 3x3 matrix M = [(2, 1, 3), (1, 2, 1), (1, 0, 1)], compute:

V2 + V1 = (1+1, 1+2, 3+2) = ***(2, 3, 5)***

V1 - V1 = V1 + -V1 = (1 – 1, 1 – 1, 3 – 3) = ***(0, 0, 0)***

|V1| (Euclidean vector length, not the number of dimensions)
square root(1*1 + 1*1 + 3*3) = square root (11) = *3.31662479036*

|V2|
Square root(1*1 + 2*2 + 2*2) = square root(9) = ***3***

M * V2 (matrix times vector, transpose it as necessary)

$$\begin{bmatrix}2 & 1 & 3\\ 1 & 2 & 1\\ 1 & 0 & 1\end{bmatrix} * \begin{bmatrix}1\\ 2\\ 2\end{bmatrix} = \begin{bmatrix}2*1+ & 1*2 & +3*2\\ 1*1+ & 2*2 & +1*2\\ 1*1+ & 0*2 & +1*2\end{bmatrix} = \begin{bmatrix}\mathbf{10}\\ \mathbf{7}\\ \mathbf{3}\end{bmatrix}$$

M * M (or M^2)

$$\begin{bmatrix}2 & 1 & 3\\ 1 & 2 & 1\\ 1 & 0 & 1\end{bmatrix} * \begin{bmatrix}2 & 1 & 3\\ 1 & 2 & 1\\ 1 & 0 & 1\end{bmatrix} =$$

$$\begin{bmatrix}2*2+1*1+1*3 & 2*1+2*1+3*0 & 2*3+1*1+3*1\\ 1*2+2*1+1*1 & 1*1+2*2+1*0 & 1*3+2*1+1*1\\ 1*2+0*1+1*1 & 1*1+0*2+1*0 & 1*3+0*1+1*1\end{bmatrix} = \begin{bmatrix}\mathbf{8} & \mathbf{4} & \mathbf{10}\\ \mathbf{5} & \mathbf{5} & \mathbf{6}\\ \mathbf{3} & \mathbf{1} & \mathbf{4}\end{bmatrix}$$

M^4

$$\begin{bmatrix}8 & 4 & 10\\ 5 & 5 & 6\\ 3 & 1 & 4\end{bmatrix} * \begin{bmatrix}8 & 4 & 10\\ 5 & 5 & 6\\ 3 & 1 & 4\end{bmatrix} = \begin{bmatrix}\mathbf{114} & \mathbf{62} & \mathbf{144}\\ \mathbf{83} & \mathbf{51} & \mathbf{104}\\ \mathbf{41} & \mathbf{21} & \mathbf{52}\end{bmatrix}$$

c) Suppose we are flipping a coin with Head (H) and Tail (T) sides. The coin is <u>not</u> balanced with 0.4 probability of H coming up (and 0.6 of T). Compute the probabilities of getting:

HTHH = ***0.4 * 0.6 * 0.4 * 0.4 = 0.0384***
THTT = ***0.6 * 0.4 * 0.6 * 0.6 = 0.0864***
Exactly 1 Head out of a sequence of 3 coin flips. ***= 0.6 * 0.4 * 0.4 = 0.096***
Exactly 2 Tails out of sequence of 3 coin flips. = ***0.4 * 0.4 * 0.6 = 0.096***

d) Consider a database schema consisting of two tables, Employee (<u>ID</u>, Name, Address), Project (<u>PID</u>, Name, Deadline), Assign(<u>EID</u>, <u>PID</u>, Date). Assign.EID is a foreign key referencing employee's ID and Assign.PID is a foreign key referencing the project.

Write SQL queries for:

i. Find projects that are not assigned to any employees (Name and Deadline of the project).

```
SELECT Name, Deadline
FROM Project
WHERE NOT Exists
  (
   SELECT *
   FROM Assign
   WHERE Assign.PID = Project.PID
   )
```

ii. For each date, find how many assignments were made that day.

```
SELECT Date, Count(PID)AS Assignments
FROM Assign
Group by Date
```

iii. Find all projects that have fewer than 2 employees assigned to them (note that the answer should include 0 or 1 employees to be correct).

```
SELECT Project.Name, Project.PID, Count(Assign.PID)
FROM Project
INNER JOIN Assign
    ON Assign.PID = Project.PID
GROUP BY Project.PID, Project.Name
HAVING COUNT(Assign.PID) <= 2
```

e) Mining of Massive Datasets, Exercise 1.3.3 :
Suppose hash-keys are drawn from the population of all nonnegative integers that are multiples of some constant c, and hash function h(x) is x mod 15. For what values of c will h be a suitable hash function, i.e., a large random choice of hash-keys will be divided roughly equally into buckets?
Justify your answer (an example only would be worth partial credit)

The c values can be 1, 2, 4 or 7. The c values cannot be any number that is a multiple of 3 or 5. This is justified because there can only be 15 buckets, from 0 to 14. Since c cannot be a multiple of 3 or 5, then the following values can not be used; 0, 3, 5 6, 9, 10, 12 and/or 15.

f) Hadoop Distributed Filesystem.

i. What are the guarantees offered by a replication factor of 3 (3 copies of each block)?

The replication factor of 3 reduces the risk of data loss (fault tolerance), in addition to increasing the availability of the data. Also, the data is processed locally as Hadoop copies the executable code of a job to the node where the data is.

ii. What action does NameNode have to take when a machine in the Hadoop cluster fails/crashes?

NameNode manages the DataNodes. Should a DataNode go down or become unresponsive after a set amount of time, then NameNode considers that DataNode dead or failed. Due to the replication process, the data is transferred directly from that DataNode to another DataNode without passing through the NameNode.

iii. What is the overall storage cost for a file of size 950 MBs, when the HDFS replication factor is set to 3?

The block size is 128MB. This means that 950MB will be in 8 total blocks, because 950/128 = 7.4. The first 7 blocks will consist of 128MB each with the 8^{th} block containing the remaining 54MB. With a replication factor of 3, this means that there will be 24 total blocks, with 21 blocks containing 128MB and 3 blocks containing 54MB.

Part 2

Please be sure to submit all python code with your answers (you can either include it in the same document or as a separate .py file).

a) Write python code that is going to read a text file and compute a total word count using a dictionary (e.g., {'Hadoop':3, 'cloud.': 2, 'MapReduce':4}. For our purposes, a word is anything split by space (.split(' ')), even if it includes things like punctuation.
Test the code on HadoopBlurb.txt (attached to the homework, from Apache Hadoop Wikipedia entry).
How many keys does your dictionary have?

```
The number of dictionary keys is  136
```

b) Write python code that is going to create two different word count dictionaries instead, assigning the words at random. Each time you process a word, choose at random which count dictionary to add it to (that means some words will appear in both dictionaries simultaneously).
How many keys does each dictionary have?

```
The number of dictionary keys in 1st dict is  85
The number of dictionary keys in 2nd dict is  75
```

c) Write python code to merge the two dictionaries into one (adding the counts) and verify that it matches the dictionary from Part 2-a.

```
The number of dictionary keys in the combined dict is  136
```

d) Write python code that is going to randomly but deterministically assign each word to one of the two dictionaries instead. For example, you can make that assignment using the remainder (YourNumber % 2 will always return 0 or 1 depending on the number). You can convert a word string into a numeric value using hash (e.g., hash('Hadoop.')). We will talk about hashing in more detail later in the quarter.
How many keys does each dictionary have?

```
The number of dictionary keys in 1st dict is  70
The number of dictionary keys in 2nd dict is  66
```

Part 3

Write (and test) python code that is going to measure the speed of reading from the web (using urllib or similar), reading from a file and writing to a file on your computer. That means your code will read or write some amount of data, time the operation, and compute the read or write rate (in MBytes/sec). The files have to be sufficiently large so that each of the measuring operations has to execute for at least 4 seconds or more (we'll check why in Part-a)

a) Compute the speed of reading from disk. We will do that in two different ways

1) Use the HadoopBlurb file as the file you read and time and compute the MB/sec speed (this one will be less than 4 seconds).

```
The filesize is  0.00138  MB.
The time is  0.0009965896606445312  seconds.
The rate is  1384722.3732057416  mbps
```

2) Use a large file (at least 4 seconds of reading from disk) and compute the MB/sec speed.

```
The filesize is  0.118034  MB.
The time is  0.001994609832763672  seconds.
The rate is  59176485.576858714  mbps
```

How do they compare? Which one do you think is more accurate?
To compare the file, the file sizes had to be compared along with the rate. I would gamble that the smaller file is a bit faster as the rate for the larger file is a bit more than 4x that of the smaller file. With the smaller file being must less that 4x the size of the larger file.

b) Compute the speed of reading from the web (you can use http://dbgroup.cdm.depaul.edu/DSC450/OneDayOfTweets.txt if you need a large file, but remember that you don't need to read the whole thing).

The file read was, http://www.textfiles.com/sf/aliensfaq.txt

```
The time is  0.3924527168273926  seconds.
```

c) Compute the speed of writing to disk

```
The filesize is  0.22007  MB.
The time is  0.23761630058288574  seconds.
The rate is  926156.9995835988  mbps
```

d) Finally, add a print statement in part 3-a (i.e., print everything you read from the file) and measure the new throughput in MBytes/sec.

```
The filesize is  0.00138  MB.
The time is  0.001995563507080078  seconds.
The rate is  691533.9928315412  mbps
```

Submit a single document containing your written answers. Be sure that this document contains your name and "CSC 555 Assignment 1" at the top.