**TO CALCULATE FITNESS SCORES OF INDIVIDUALS (DATASETS FOR NEUROIDAL NETWORKS)**

* We have implemented the AND gate for the below 2 inputs using the below code, after the AND gate output C, we have compared the Y6 values obtained from the earlier code.
* The comparison is expected to be either Correlation or multiplication, we have chosen to multiply C with Y6 to obtain a value.
* The resultant array from this comparison values are labelled as Output\_comparison.
* The output comparison is further used to find the average of the same. It is calculated by dividing sum of the Output\_comparison with that of the length of the same, which in our case is of length 100.
* The average of the expected /output C is calculated by dividing the sum of Output C and the length of C, this is labelled as avg\_c in our case.
* The total average of the individual is calculated by dividing both the above values, which are the comparison output with that of the average value of C.
* Since the matrices are bound to have a range of 1-10, we have just multiplied this resultant value with 10, this resultant value is called Fitness Score.
* This fitness score is for one individual or one among the entire population, similarly when we simulate multiple values for Y6, we get different values for Y6 and there by different fitness scores subsequently for each individual.
* The number of fitness scores can be calculated based on the population that can be defined during simulation.

We have implemented python code in Jupiter to fulfill the above requirements on top of the earlier code, please find the below snippet for reference in this documentation:

#to calculate fitness\_score using y6

C=[]

for i in range(0,len(A)):

for j in range(0,len(B)):

temp=A[i] & B[j]

C.append(temp)

break

output\_comparison=[]

for i in range(0,len(C)):

for j in range(0,len(y6)):

temp=C[i].astype(int) & math.ceil(y6[j])

output\_comparison.append(temp)

break

avg\_output\_comparison=sum(output\_comparison)/len(output\_comparison)

avg\_c=sum(C)/len(C)

avg\_score=avg\_output\_comparison/avg\_c

fitness\_score=avg\_score\*10

Please find the below reference for the inputs taken to show the values, this is not really needed but there can be another way to calculate the Fitness scores using excel formulae. However, we have used Jupiter notebooks so that we can leverage the code from the earlier group and also pass on our code to the next group all at one place.

The below is a Testing Dataset:

The truth table for 100 input pairs, and the resultant output is as below:

Output C = Input A. Input B

|  |  |  |
| --- | --- | --- |
| **Input A** | **Input B** | **Output C** |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |

Please find the below AND gate based on the inputs provided, and the matrix obtained is as below:

**Input A** 111101001010101000111001000000011111110000011011111010010111100000110101101100111111001001011101111

**Input B**

010010010101011111111111111011100101111011100111011100111011100110100101101100111011001110110011100

C= A. B

Output C = Input A. Input B

**Output C**

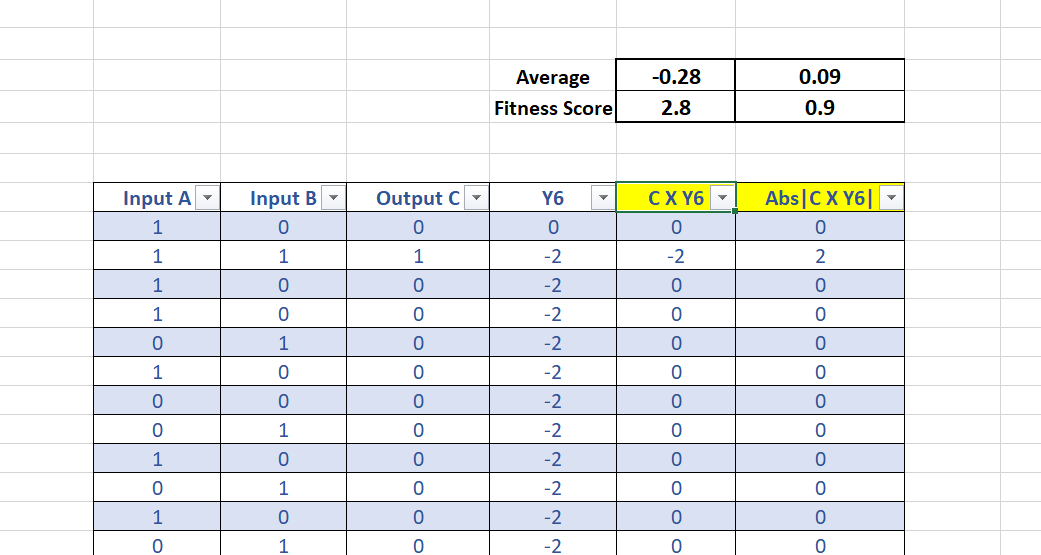
010000000000001000111001000000000101110000000011011000010011100000100001101100111011001000010001100

Since the vectors are large, we have just used labels for the diagram below:

Input A

Output C

Input B



2. Please find the below AND gate implementation for Training dataset:

|  |  |  |
| --- | --- | --- |
| **Input P** | **Input Q** | **Output Q** |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 0 | 0 |

Please find the below AND gate based on the inputs provided, and the matrix obtained is as below:

**Input P** 100101010100100101010100110011101110011111100101010100111001000110101110010001011001010101001010011

**Input Q**

100101011100000100001101100101010100111001110111001111001000101001010101001110010001110010001010010

R= P. Q

Output R = Input P. Input Q

**Output R**

100101010100000100000100100001000100011001100101000100001000000000000100000000010001010000001010010

Since the vectors are large, we have just used labels for the diagram below:

Input P

Output R

Input Q