Python & GIT Basics Assignment - Solutions

1. Create two vectors y and $\hat{y}$ having **the same** dimensions, where $\hat{y}$ should consist of random numbers between [0,1) and y should contain 0s and 1s, for example, y=[0,1,1,0,1,0,0,1,...,1]. Compute the given expression:

$$O = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log_2(\hat{y}_i) + (1 - y_i)\log_2(1 - \hat{y}_i)]$$

   Where n = 100, is the total number of elements in y and $\hat{y}$

Note: The expression *O*, which you have computed is actually a **Cross-Entropy** loss function used in machine learning for classification tasks which tells us how bad or good the model is performing, if *O* is large then the model is performing worst and vice versa.

Solution:

```
#Given n = 100
#Create a 1D array y, of size 100 with randomly selected 0s and
1s
#Create a 1D array y_hat, of size 100 with numbers randomly
(uniformly) selected from [0,1)
#Find the cross entropy loss

n = 100
y_hat = np.random.rand(n)
y = np.random.randint(low = 0, high = 2 , size = (n))
O = -np.sum(y * np.log2(y_hat) + (1 - y) * np.log2(1 - y_hat))/n
print(O)
```

2. Write a Python class to find a pair of elements (indices of the two numbers) from a given array whose sum equals a specific target number.
   Note: There will be multiple solutions, so create a dictionary where the keys represent just S.No(1,2,3,4…..) and the value corresponding to the key represents the indices of the two numbers
   For example: Input: numbers= [10,20,10,40,50,60,70], target=50
   Output: {1: [0, 3], 2: [2, 3], 3: [3, 0], 4: [3, 2]}

Solution:

```
class py_solution:
    def __init__(self, nums, target):
        self.lookup = {}
        c = 1
        for i in range(len(nums)):
            for j in range(len(nums)):
                if nums[i] + nums[j] == target:
                    self.lookup[c] = [i, j]
                    c = c + 1
    def show(self):
        print(self.lookup)

list_ = [10,20,10,40,50,60,70]
target = 50

answer = py_solution(list_, target)
answer.show()
```