IDFT_test2

April 1, 2019

```
In [1]: import numpy as np
```

```
In [2]: def IDFT_fn(x):
            sze_x = np.size(x)
            X_val = np.zeros((sze_x,),dtype=np.complex128)
            for m in range(0,sze_x):
                for n in range(0,sze_x):
                    X_val[m] += x[n]*np.exp(np.pi*2j*m*n/sze_x)
            return X_val/sze_x
```

```
In [3]: X = np.random.rand(1024,)
```

0.1 Now Lets Run our IDFT Function

```
In [4]: IDFT_fn(X)
```

```
Out[4]: array([ 0.51558626+0.j        ,  0.0045763 +0.00061027j,
               -0.00433047+0.00274487j, ...,  0.00783235-0.0040975j ,
               -0.00433047-0.00274487j,  0.0045763 -0.00061027j])
```

0.2 Now lets run the numpy's IFFT Function for comparision

```
In [5]: np.fft.ifft(X)
```

```
Out[5]: array([ 0.51558626-0.j        ,  0.0045763 +0.00061027j,
               -0.00433047+0.00274487j, ...,  0.00783235-0.0040975j ,
               -0.00433047-0.00274487j,  0.0045763 -0.00061027j])
```

```
In [6]: np.allclose(IDFT_fn(X), np.fft.ifft(X))
```

```
Out[6]: True
```

0.3 Hurray !! Our IDFT is equal to Numpy's IDFT implimentation