IFFT_test2

April 1, 2019

```
In [1]: from cmath import exp, pi

        def fft_fn(x):
            N = len(x)
            if N <= 1: return x
            even_part = fft_fn(x[0::2])
            odd_part =  fft_fn(x[1::2])
            T= [exp(-2j*pi*p/N)*odd_part[p] for p in range(N//2)]
            return [even_part[p] + T[p] for p in range(N//2)] + \
                   [even_part[p] - T[p] for p in range(N//2)]
```

```
In [2]: from numpy import array
```

```
In [3]: import numpy as np
```

```
In [4]: a = array(np.random.random(1024))
```

0.1 From here we are Implimenting IFFT

$IFFT(X) = 1/N\,conj(FFT(conj(X)))$

```
In [6]: N = len(a)
        ifft_output = (1/N)*np.conj(fft_fn(np.conj(a)))
```

0.1.1 Here we are comparing our result with Numpy's ifft

```
In [7]: np.allclose(ifft_output, np.fft.ifft(a))
```

```
Out[7]: True
```

0.1.2 Yes, As we can see that two results are element-wise equal within a tolerance.