密碼工程 Quiz6

Problem 1

(a)

```
def recursive_kronecker(H, h2, depth, max_depth):
    if depth == max_depth:
        return H
    elif depth == 0:
        return recursive_kronecker(h2, h2, depth + 1, max_depth)
    else:
        return recursive_kronecker(np.kron(H, h2), h2, depth + 1, max_depth)

def WHT(x):
    x = np.array(x)
    if (len(x.shape) < 2): # make sure x is 1D array
        if (len(x) > 3): # accept x of min length of 4 elements (M=2)
            # check length of signal, adjust to 2**m
            n = len(x)
            M = math.trunc(math.log(n, 2))
            x = x[0:2 ** M]
            h2 = np.array([[1, 1], [1,-1]])
            H = recursive_kronecker(None, h2, 0, M)
            return (np.dot(H, x) / 2. ** M, x, M)
```

→ The preceding part of the WHT function remains the same. (First of all, make sure x is 1D array and the length of x should be greater than 3. Secondly, adjust the length of x and initialize the matrix h2.) I only replace the for loop in the pseudocode with the recursive function to calculate H. I utilize **recursive_kronecker** function to calculate H. If the depth of the function is equal to 0, I call **recursive_kronecker** again and put h2 in the first parameter and make depth increase 1. Otherwise, I just call **recursive_kronecker** again and put np.kron(H, h2), which performs the Kronecker product of H and h2, in the first parameter and make depth increase 1. Finally, When the depth reaches the max depth, it will just return H. After calling the **recursive_kronecker** function, we will return the standardized inner product of Hadamard matrix H , vecotr x, and M.

(b)

1. Quantumn computing: The WHT has applications in some quantum computing algorithm design. The reason is that the properties of the WHT, such as its ability to generate superposition states and its fast computation, make it a valuable tool in quantum computing research and development.

2. Image compression: The WHT can be utilized to compress images, such as JPEG. The reason that people often use it is that the WHT's simpler basis functions and faster computation make it a viable alternative to the Discrete Cosine Transform (DCT), especially for applications where computational efficiency is critical.
3. Data analysis: The WHT can capture important patterns and structures in the data. Moreover, The WHT's ability to transform data into a different domain, where certain features may be more easily distinguishable, can be useful. Therefore, we usually apply it in data analysis.

Problem 2

(a)

In a traditional way, if we want to determine whether n is the prime or not, we have to discover all the prime which is smaller than $\sqrt{n}$. Since p is large, it is hard to find and also takes lots of time. However, if we use Miller-Rabin test. Since pq-1 is an even number but not 2, we can find s and k, such that $pq-1 = 2^s k$. Miller-Rabin test is a probabilistic primality test. Therefore, after we perform this test multiple times, the error rate will decrease exponentially. Finally, we will find that pq is a composite number. In comparison to traditional way, it can take less time and get the answer we want.

(b)

No, the Miller-Rabin test cannot break RSA. The reason is that the Miller-Rabin test is a probabilistic primality test, not a factorization algorithm. It can only check whether a given number is likely to be prime, but it does not reveal the prime factors of a composite number. Therefore, it is not possible to use Miller-Rabin test to break RSA.