**Experimental Requirements:**

Read the RC4 pseudorandom generation algorithm to reinforce understanding of the RC4 algorithm's process.

Implement the RC4 algorithm, generate a random number sequence of 20,000 consecutive random numbers to a file named "random.txt" using a given English string key.

Count the occurrence frequency of each integer from 0 to 255 in "random.txt" and output the results to a file named "frequency.txt".

Experimental Content:

**Algorithm Implementation Steps**

a) Initialize S[256].

b) Fill T[256] with the key.

c) Perform initial permutation on S[256] using T[256].

d) Generate a stream of random integers using the initially permuted S[256] and output it to the file "random.txt" (generate 20,000 random numbers).

Count the distribution of the 20,000 random numbers and output the results to the file "frequency.txt". The format of the statistics should include the random number and its frequency (use a reader-friendly format).

**Experimental Report Requirements:**

The sizeof() function in C gives the size of an array including the null terminator '\0'.

Experimental Summary

a) Write the RC4 algorithm process in your own words (you can refer to books, but avoid copying directly). Specify the encryption type to which it belongs (block cipher/stream cipher). Explain which factors can influence the generated random numbers.

RC4 Algorithm Process:

The RC4 algorithm initializes an array T using a key string and an S-box with elements initialized to 0-255.

Then, each element in the S-box is processed to shuffle it.

Finally, the shuffled S-box is used to generate a pseudorandom keystream. This keystream is XORed with the plaintext to obtain the ciphertext.