John Luke Denny

DES

There are five categories that groups the code: main function, encryption / decryption function, tables, conversion functions, and helper functions. Starting with conversion functions, these are conversion functions between hex and characters, hex and binary, and binary and decimal. For helper functions, there are 5. The first blocks text so larger messages can be encrypted/decrypted by the code. The second permutes a key by shifting it by a number of bits determined by one of the tables. The third and fourth functions are mathemaitcal: XOR and bitwise shift left. The final function is a key generation for DES. It takes a 16 character hex string and converts it to a 56-bit binary key by shifting the halves of the original 64-bits and then combining and permuting it to a 48-bit compressed key. Regarding the tables, they serve to determine the bit mangling done by DES. One table is to determine how many shifts are done in a round for the key. Within the DES encryption and decryption, it divides the block into halves and permutes, xors and shifts keys with the text to complete a round. A total of 16 rounds are performed in the encryption and decryption. In order to save code, the key used in the decryption is the encryption key reversed. To run the function, use “python3 des.py”. Note, there is a bug that results in singular words greater than 10 characters in length from working.

RSA

1. To check for prime numbers, a large boolean array is generated with the number being prime if the index of the number minus one is true: below is sample pseudocode

```
is 7 prime?
Index = 7 - 1
primes[index] == 1
        Return True
```

   This array is used later during GCD to make iterations occur quicker.
2. The 10th and 19th prime number between 1000 and 10000 are 1061 and 1117
3. Both encryption and decryption work. To use, run command “python3 rsa.py”.
4. The process to find the private key, given a public key, starts with finding p and q to derive d. Assuming space allows, generate an array of booleans to say whether a number is prime or not. To generate this array, the size only needs to be upto half of N because no number greater than N/2 can be a factor of N. This method, Sieve of Eratosthenes Algorithm, although space intensive, is of time O(nlog(logn)). Once this is generated, one can generate through all pairs of primes less than N/2 since N is the product of two primes. For testing the pairs, the second part of the pair is only tested if the first number is a factor of N. While this cuts down on the number of tests done, the time complexity is of at least O(n^2). Once p and q are found, Euclid’s algorithm is used to find d. The time complexity of this algorithm is O(logk). With these time complexities being additive, the largest represents the overall complexity being O(n^2).