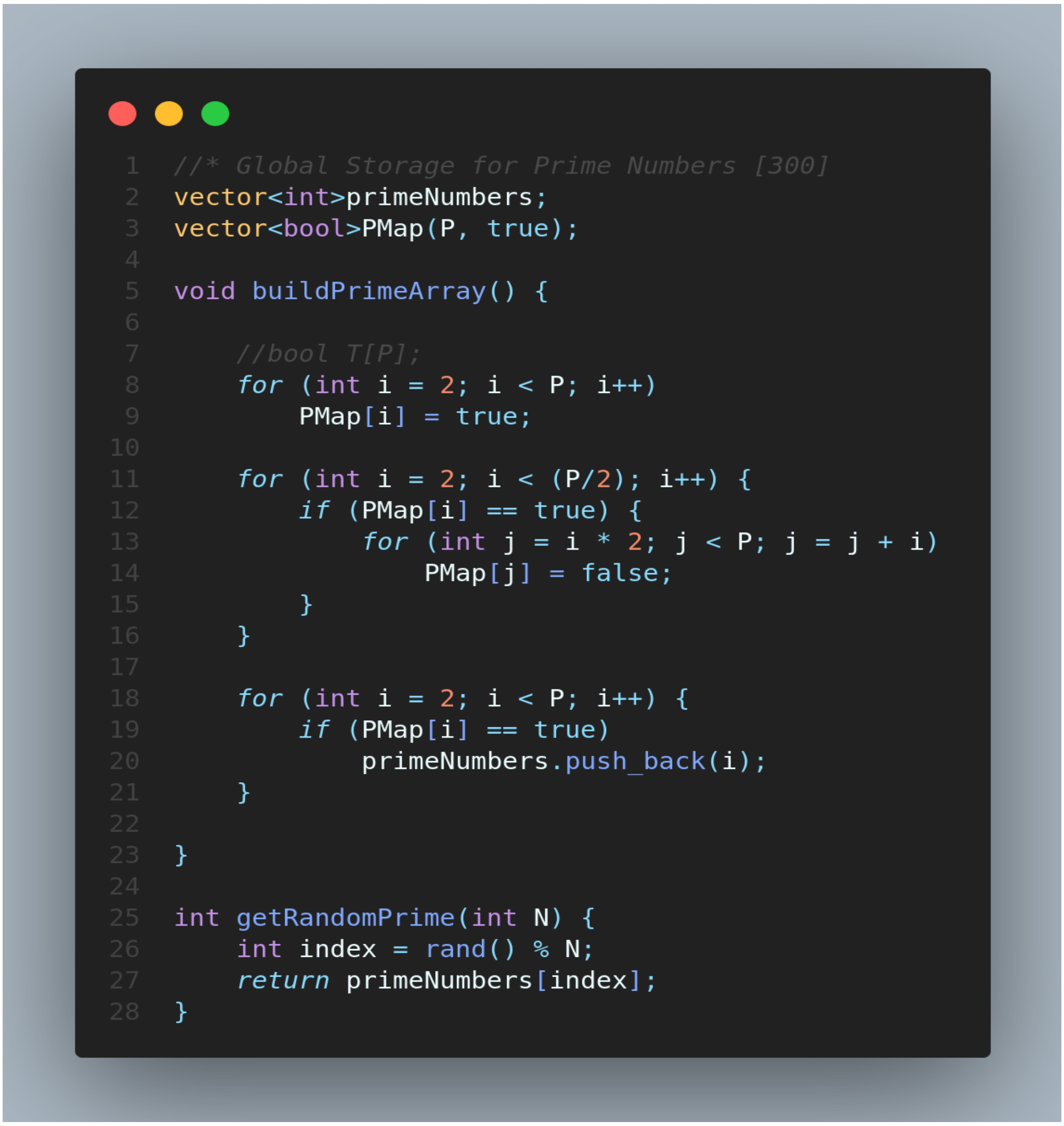
122022010 Shrayank Mistry
Implementation of RSA algorithm:

Design an experiment to estimate the amount of time required to

1. Generate key pair (RSA)
2. Encrypt n-bit message (RSA)
3. Decrypt n-bit message (RSA)

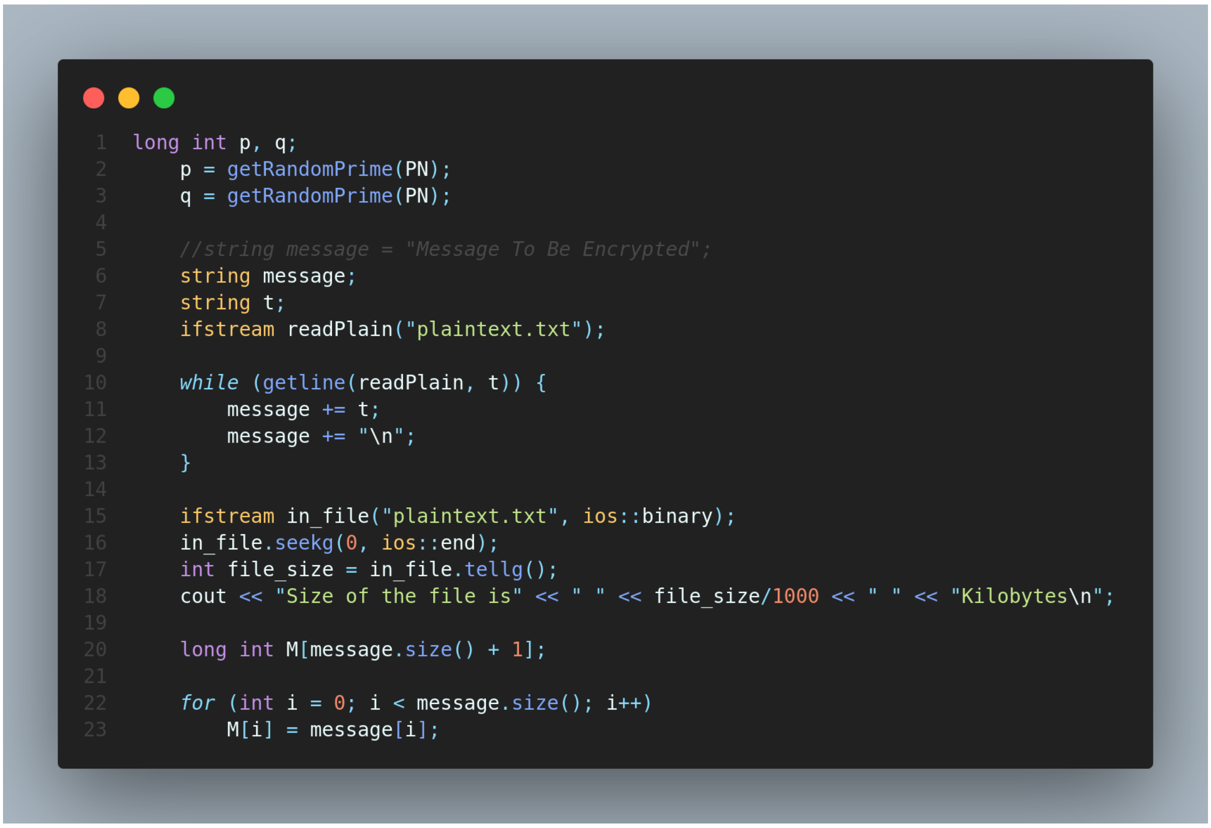
As a function of key size, experiment with different n-bit messages. Summarize your conclusion.

Step 1: Generate Random Prime Numbers to build Public and Private Keys



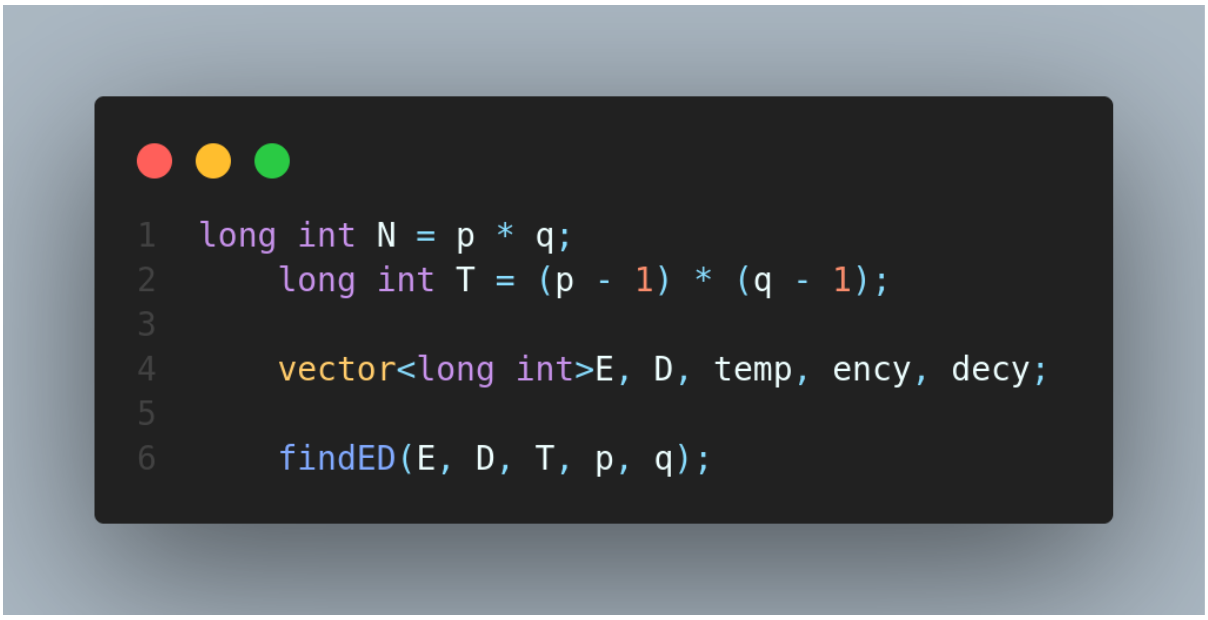
```
//* Global Storage for Prime Numbers [300]
vector<int>primeNumbers;
vector<bool>PMap(P, true);

void buildPrimeArray() {

    //bool T[P];
    for (int i = 2; i < P; i++)
        PMap[i] = true;

    for (int i = 2; i < (P/2); i++) {
        if (PMap[i] == true) {
            for (int j = i * 2; j < P; j = j + i)
                PMap[j] = false;
        }
    }

    for (int i = 2; i < P; i++) {
        if (PMap[i] == true)
            primeNumbers.push_back(i);
    }

}

int getRandomPrime(int N) {
    int index = rand() % N;
    return primeNumbers[index];
}
```

Step 2: Reading Input file and randomly generating p, q prime numbers



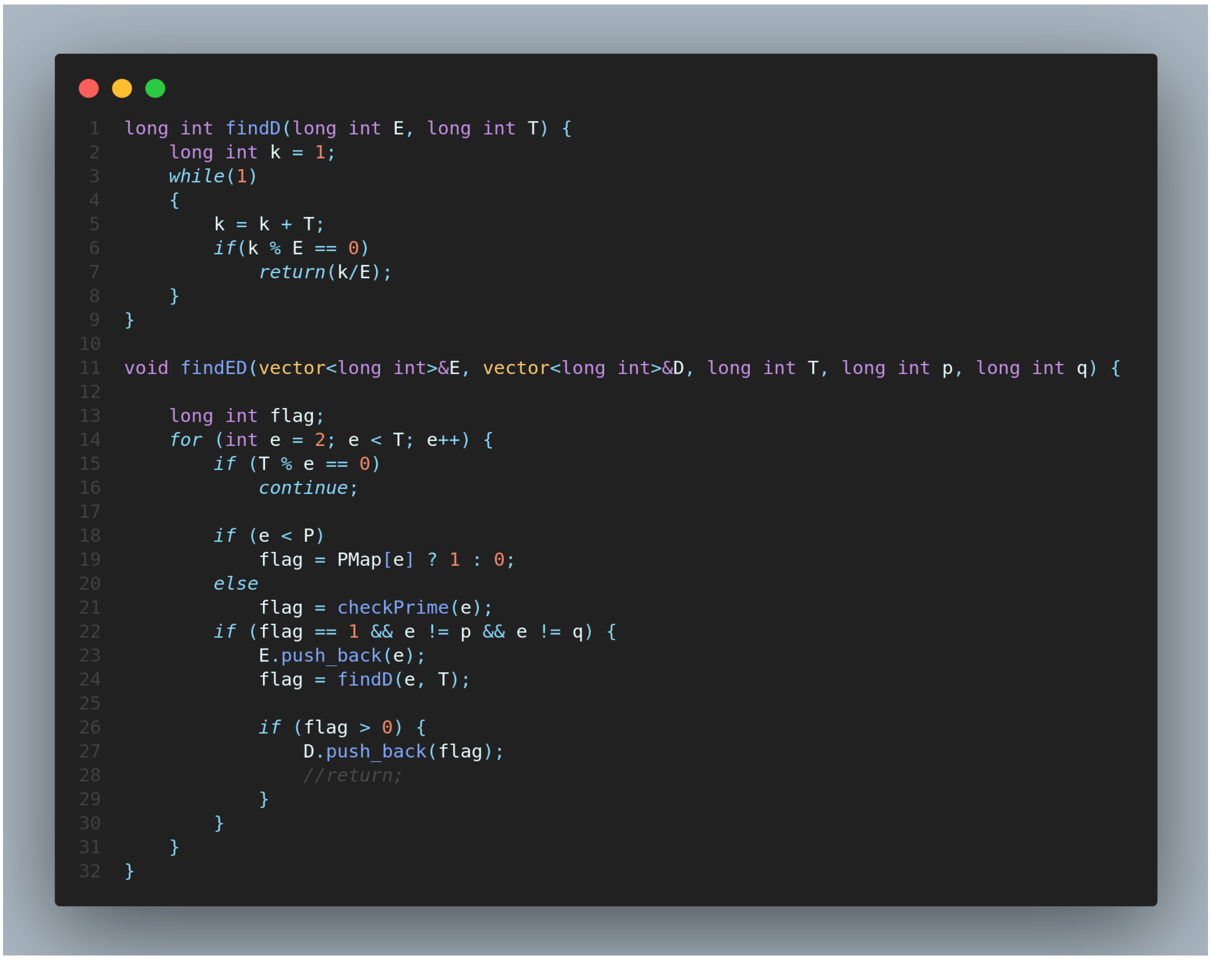
```
long int p, q;
    p = getRandomPrime(PN);
    q = getRandomPrime(PN);

    //string message = "Message To Be Encrypted";
    string message;
    string t;
    ifstream readPlain("plaintext.txt");

    while (getline(readPlain, t)) {
        message += t;
        message += "\n";
    }

    ifstream in_file("plaintext.txt", ios::binary);
    in_file.seekg(0, ios::end);
    int file_size = in_file.tellg();
    cout << "Size of the file is" << " " << file_size/1000 << " " << "Kilobytes\n";

    long int M[message.size() + 1];

    for (int i = 0; i < message.size(); i++)
        M[i] = message[i];
```

Step 3: Generating N and euler quotient of N



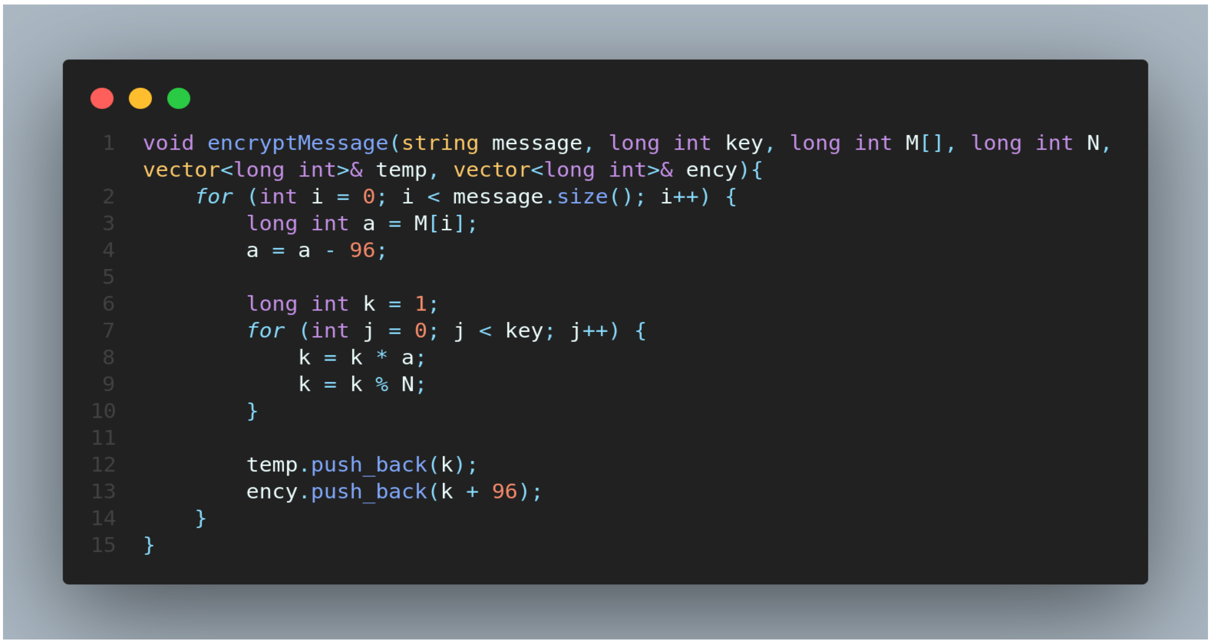
```
long int N = p * q;
    long int T = (p - 1) * (q - 1);

    vector<long int>E, D, temp, ency, decy;

    findED(E, D, T, p, q);
```

Step 4: Generating the Public and Private Keys



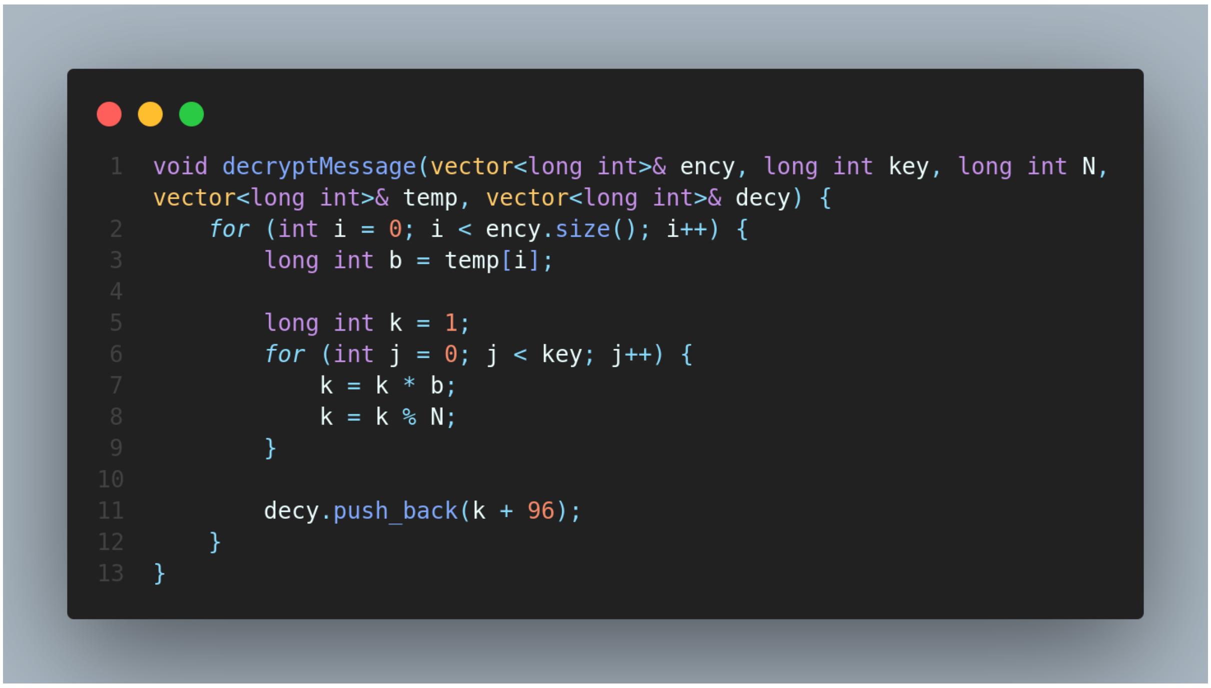
```
long int findD(long int E, long int T) {
    long int k = 1;
    while(1)
    {
        k = k + T;
        if(k % E == 0)
            return(k/E);
    }
}

void findED(vector<long int>&E, vector<long int>&D, long int T, long int p, long int q) {

    long int flag;
    for (int e = 2; e < T; e++) {
        if (T % e == 0)
            continue;

        if (e < P)
            flag = PMap[e] ? 1 : 0;
        else
            flag = checkPrime(e);
        if (flag == 1 && e != p && e != q) {
            E.push_back(e);
            flag = findD(e, T);

            if (flag > 0) {
                D.push_back(flag);
                //return;
            }
        }
    }
}
```

Step 5: Encrypting the message



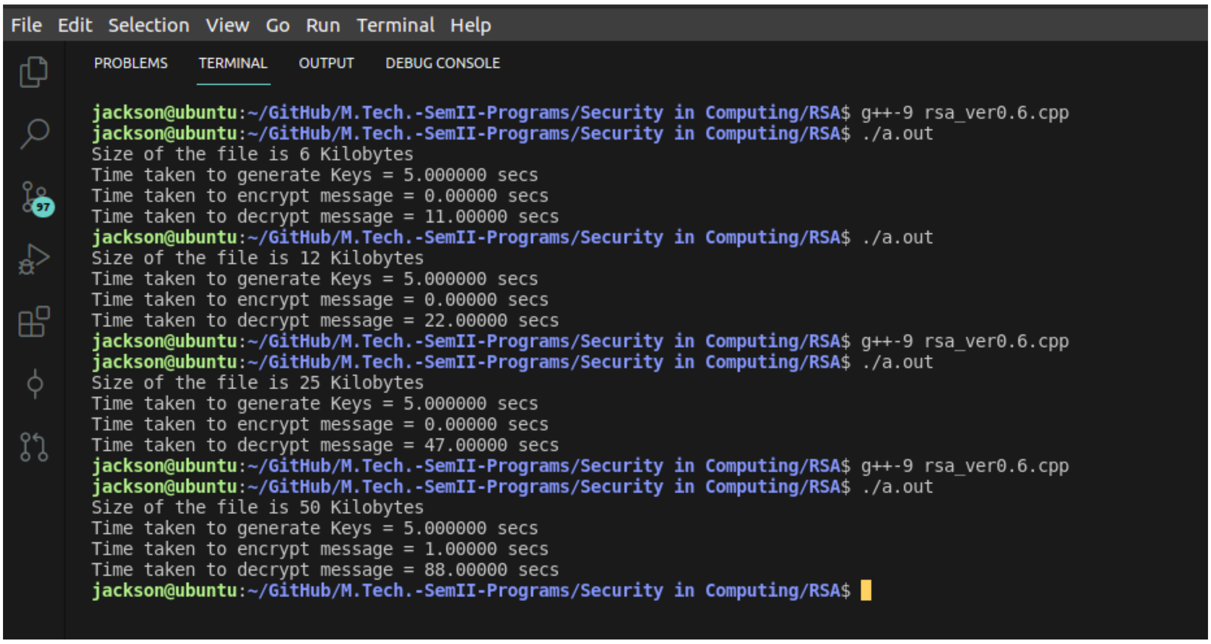
```
void encryptMessage(string message, long int key, long int M[], long int N, vector<long int>& temp, vector<long int>& ency){
    for (int i = 0; i < message.size(); i++) {
        long int a = M[i];
        a = a - 96;

        long int k = 1;
        for (int j = 0; j < key; j++) {
            k = k * a;
            k = k % N;
        }

        temp.push_back(k);
        ency.push_back(k + 96);
    }
}
```

Step 6: Decrypting the message

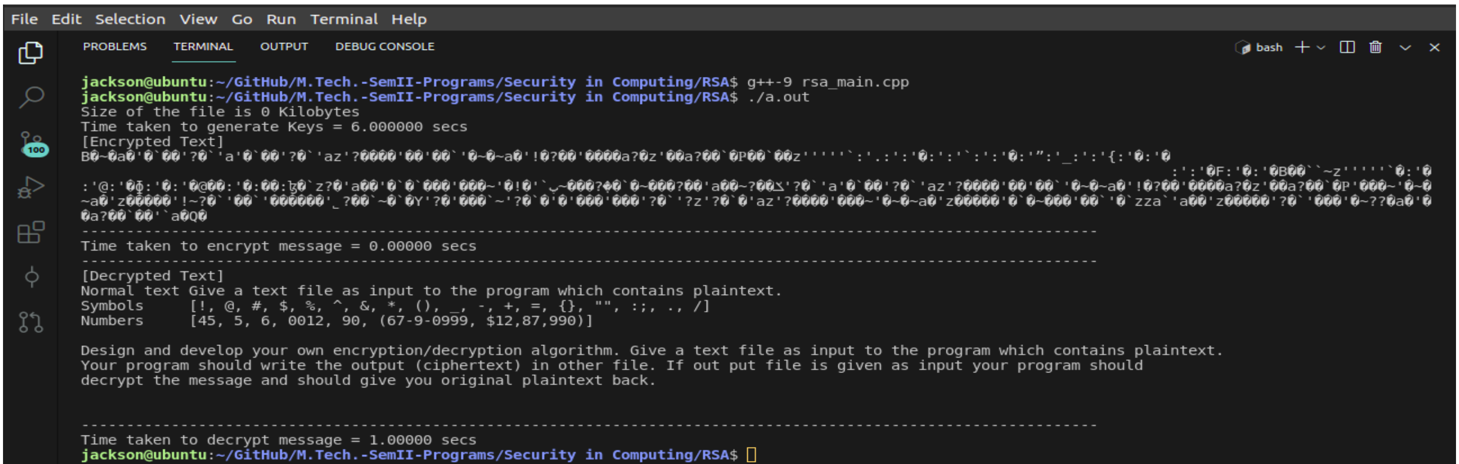


```cpp
void decryptMessage(vector<long int>& ency, long int key, long int N,
vector<long int>& temp, vector<long int>& decy) {
    for (int i = 0; i < ency.size(); i++) {
        long int b = temp[i];

        long int k = 1;
        for (int j = 0; j < key; j++) {
            k = k * b;
            k = k % N;
        }

        decy.push_back(k + 96);
    }
}
```

Results:



```
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ g++-9 rsa_ver0.6.cpp
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ./a.out
Size of the file is 6 Kilobytes
Time taken to generate Keys = 5.000000 secs
Time taken to encrypt message = 0.00000 secs
Time taken to decrypt message = 11.00000 secs
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ./a.out
Size of the file is 12 Kilobytes
Time taken to generate Keys = 5.000000 secs
Time taken to encrypt message = 0.00000 secs
Time taken to decrypt message = 22.00000 secs
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ g++-9 rsa_ver0.6.cpp
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ./a.out
Size of the file is 25 Kilobytes
Time taken to generate Keys = 5.000000 secs
Time taken to encrypt message = 0.00000 secs
Time taken to decrypt message = 47.00000 secs
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ g++-9 rsa_ver0.6.cpp
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ./a.out
Size of the file is 50 Kilobytes
Time taken to generate Keys = 5.000000 secs
Time taken to encrypt message = 1.00000 secs
Time taken to decrypt message = 88.00000 secs
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ █
```



```
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ g++-9 rsa_main.cpp
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ./a.out
Size of the file is 0 Kilobytes
Time taken to generate Keys = 6.000000 secs
[Encrypted Text]
B�~�a�'�`��'?�`'a'�`��'?�`'az'?����'��'��`'�~�~a�'!�?��'����a?�z'��a?��`�P��`��z'''''`:'.:':'�:':'`:':'�:'":'_:':'{:'�:'�
                                                                                                                        :':'�F:'�:'�B��``~z'''''`�:'�
:'@:'��:'�:'�@��:'�:��:ǥ�`z?�'a��'�`�`���'���~'�!�'`ب~���?�`�~���?��'a��~?��ﮊ'?�`'a'�`��'?�`'az'?����'��'��`'�~�~a�'!�?��'����a?�z'��a?��`�P'���~'�~�
~a�'z�����'!~?�`'��`'������'_?��`~�`�Y'?�'���`~'?�`�'�'���'���'?�`'?z'?�`�'az'?����'���~'�~�~a�'z�����'�`�~���'��`'�`zza`'a��'z�����'?�`'���'�~??�a�'�
�a?��`��'`a�Q�
----------------------------------------------------------------------------------------------------------------------
Time taken to encrypt message = 0.00000 secs
----------------------------------------------------------------------------------------------------------------------
[Decrypted Text]
Normal text Give a text file as input to the program which contains plaintext.
Symbols     [!, @, #, $, %, ^, &, *, (), _, -, +, =, {}, "", :;, ., /]
Numbers     [45, 5, 6, 0012, 90, (67-9-0999, $12,87,990)]

Design and develop your own encryption/decryption algorithm. Give a text file as input to the program which contains plaintext.
Your program should write the output (ciphertext) in other file. If out put file is given as input your program should
decrypt the message and should give you original plaintext back.


----------------------------------------------------------------------------------------------------------------------
Time taken to decrypt message = 1.00000 secs
jackson@ubuntu:~/GitHub/M.Tech.-SemII-Programs/Security in Computing/RSA$ ▯
```