**Name: Shangirne Kharbanda**

**Registration Number: 20BAI1154**

**ISM LAB – 1**

**CEASAR CIPHER AND PLAYFAIR CIPHER**

**Caesar Cipher:**

**caesar\_encrypt.cpp**

#include <iostream>

#include<string>

#include<fstream>

using namespace std;

string encrypt(string str, int n){

    string en = "";

    for(int i=0;i<str.length();i++){

        en += int(str[i]) + n;

    }

    return en;

}

int main() {

    string line;

    string something;

    ifstream file("text.txt");

    ofstream myfile;

    myfile.open("text\_encrypted.txt");

    int shift = 3;

    while(getline(file,line)){

        myfile << encrypt(line,shift);

    }

    file.close();

    myfile.close();

    return 0;

}

**caesar\_decrypt.cpp**

#include <iostream>

#include<string>

#include<fstream>

using namespace std;

string encrypt(string str, int n){

    string en = "";

    for(int i=0;i<str.length();i++){

        en += int(str[i]) - n;

    }

    return en;

}

int main() {

    string line;

    string something;

    ifstream file("text\_encrypted.txt");

    ofstream myfile;

    myfile.open("text\_decrypted.txt");

    int shift = 3;

    while(getline(file,line)){

        myfile << encrypt(line,shift);

    }

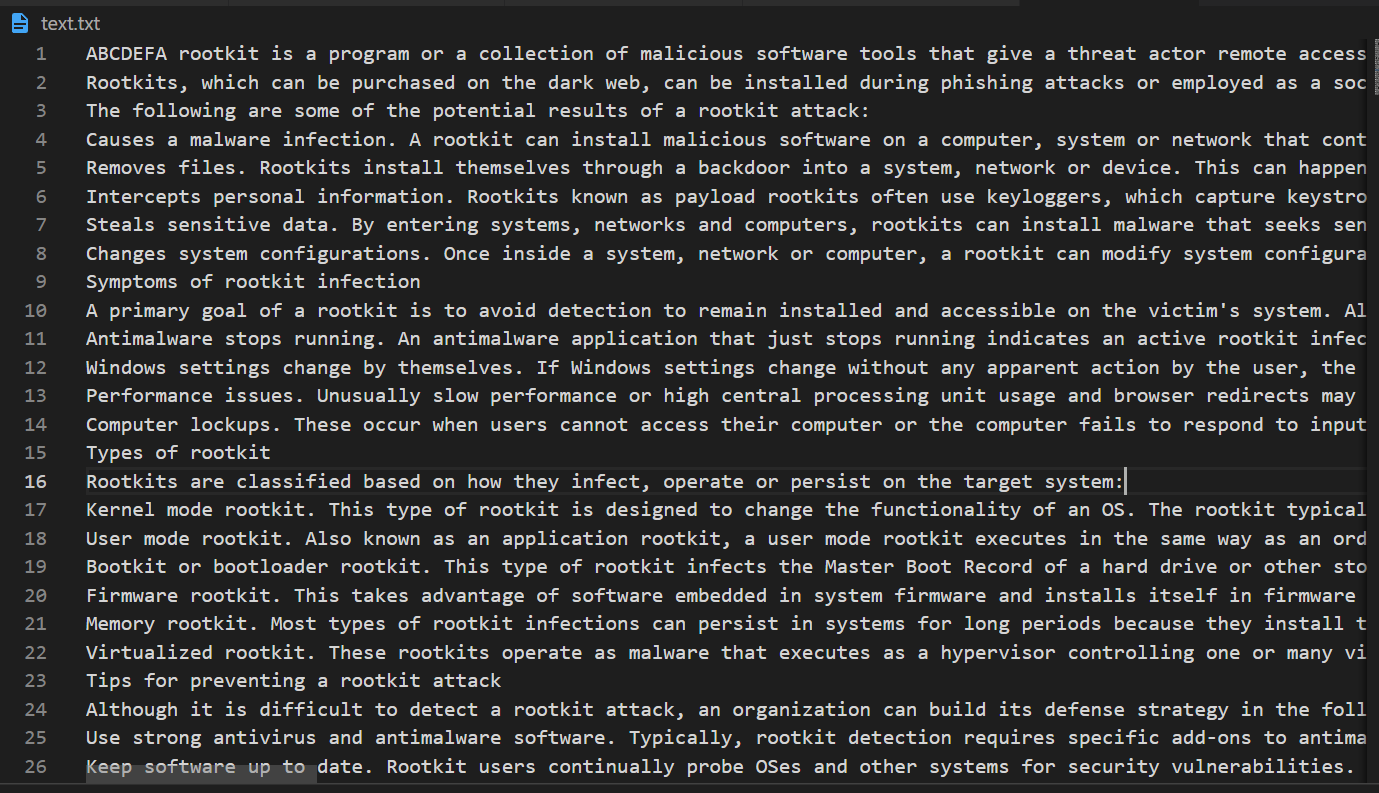
    file.close();

    myfile.close();

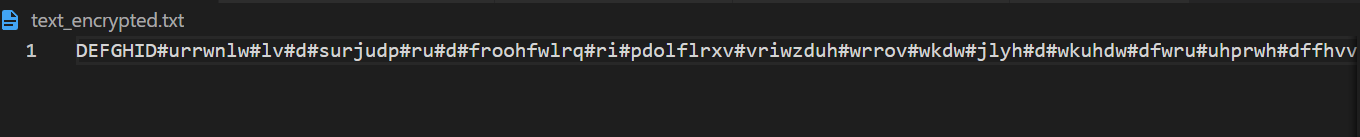
    return 0;

}

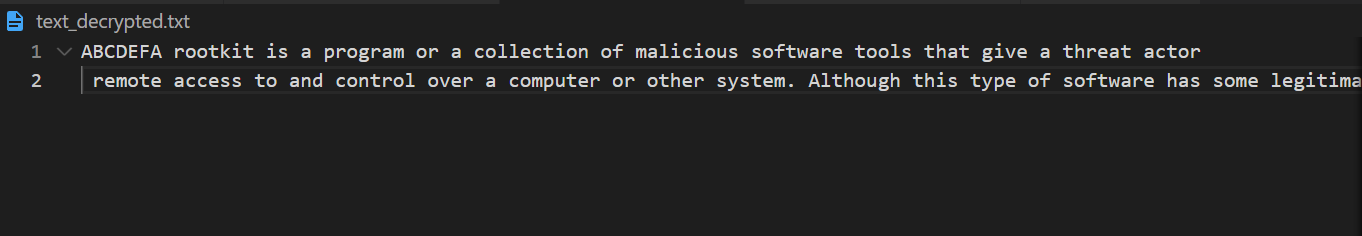
The text file **text.txt**:



We will run caesar\_encrypt.cpp on this text file. Output is encrypted text as shown below in **text\_encrypted.txt**.



Now we will decrypt this encrypted file using caesar\_decrypt.cpp to generate decrypted text. Output is decrypted text as shown below in **text\_decrypted.txt**.



**Playfair cipher:**

**playfair.cpp**

#include <iostream>

#include <string>

#include<fstream>

using namespace std;

class playfair {

   public:

      string msg;

       char n[5][5];

   void play( string k, string t, bool m, bool e , string l) {

      createEncoder( k, m );

      getText( t, m, e );

      if( e )

         play( 1 );

      else

         play( -1 );

      if(l[0] == 'e' || l[0] == 'E'){

        print();

      }else{

        print1();

      }

   }

   private:

      void play( int dir ) {

      int j,k,p,q;

      string nmsg;

      for( string::const\_iterator it = msg.begin(); it != msg.end(); it++ ) {

         if( getPos( \*it++, j, k ) )

         if( getPos( \*it, p, q)) {

            if( j == p ) {

               nmsg+= getChar( j, k + dir );

               nmsg += getChar( p, q + dir );

            }

            else if( k == q ) {

               nmsg += getChar( j + dir, k );

               nmsg += getChar( p + dir, q );

            } else {

               nmsg += getChar( p, k );

               nmsg += getChar( j, q );

            }

         }

      }

      msg = nmsg;

   }

   void print() {

      ofstream myfile;

      myfile.open("text\_encrypted.txt");

      string::iterator it = msg.begin();

      int count = 0;

      while( it != msg.end() ) {

         myfile << \*it;

         it++;

        myfile << \*it << "";

         it++;

         if( ++count >= 26 )

         myfile << endl;

         count = 0;

      }

      myfile.close();

   }

   void print1() {

      ofstream myfile;

      myfile.open("text\_decrypted.txt");

      string::iterator it = msg.begin();

      int count = 0;

      while( it != msg.end() ) {

         myfile << \*it;

         it++;

        myfile << \*it << "";

         it++;

         if( ++count >= 26 )

         myfile << endl;

         count = 0;

      }

      myfile.close();

   }

   char getChar( int a, int b ) {

      return n[ (b + 5) % 5 ][ (a + 5) % 5 ];

   }

   bool getPos( char l, int &c, int &d ) {

      for( int y = 0; y < 5; y++ )

      for( int x = 0; x < 5; x++ )

      if( n[y][x] == l ) {

         c = x;

         d= y;

      return true;

      }

      return false;

   }

   void getText( string t, bool m, bool e ) {

      for( string::iterator it = t.begin(); it != t.end(); it++ ) {

         \*it = toupper( \*it );

         if( \*it < 65 || \*it > 90 )

            continue;

         if( \*it == 'J' && m )

            \*it = 'I';

         else if( \*it == 'Q' && !m )

            continue;

         msg += \*it;

      }

      if( e ) {

         string nmsg = ""; size\_t len = msg.length();

         for( size\_t x = 0; x < len; x += 2 ) {

            nmsg += msg[x];

            if( x + 1 < len ) {

               if( msg[x] == msg[x + 1] ) nmsg += 'X';

               nmsg += msg[x + 1];

            }

         }

         msg = nmsg;

      }

      if( msg.length() & 1 )

      msg += 'X';

   }

   void createEncoder( string key, bool m ) {

      if( key.length() < 1 )

      key= "KEYWORD";

      key += "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

      string s= "";

      for( string::iterator it = key.begin(); it != key.end(); it++ ) {

         \*it = toupper( \*it );

         if( \*it < 65 || \*it > 90 )

            continue;

         if( ( \*it == 'J' && m ) || ( \*it == 'Q' && !m ) )

            continue;

         if( s.find( \*it ) == -1 )

            s += \*it;

      }

      copy( s.begin(), s.end(), &n[0][0] );

   }

};

int main(){

   string k, i, msg;

   playfair pf;

   bool m, c;

   cout << "Encrypt or Decrypt? ";

   getline( cin, i );

   c = ( i[0] == 'e' || i[0] == 'E' );

   cout << "Enter a key: ";

   getline( cin, k);

   cout << "I <-> J (Y/N): ";

   getline( cin, i );

   m = ( i[0] == 'y' || i[0] == 'Y' );

   ifstream file;

   if(i[0] == 'e' || i[0] == 'E' ){

    file.open("text.txt");

   }else{

    file.open("text\_encrypted.txt");

   }

    while(getline(file,msg)){

         pf.play( k, msg,m, c, i);

    }

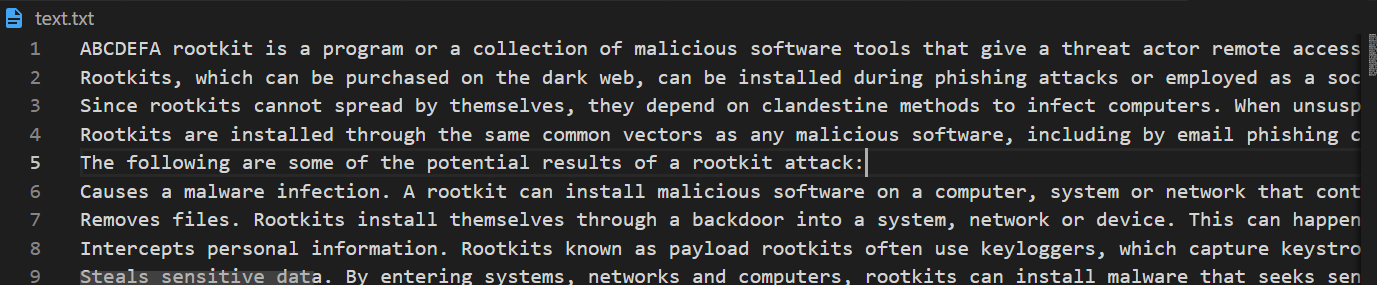
    file.close();

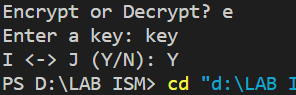
   return 0;

}

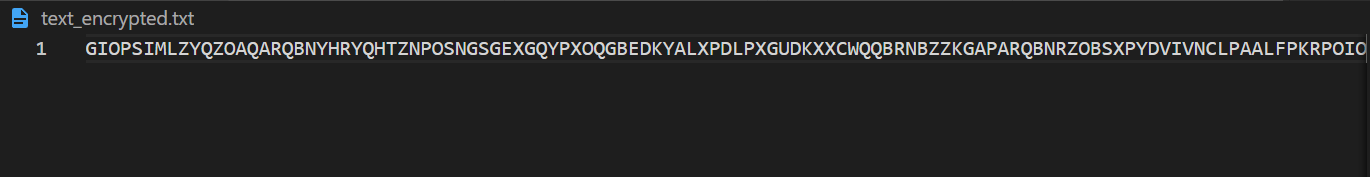
First we encrypt the file using playfair.cpp:

The text file to be encrypted **text.txt**:

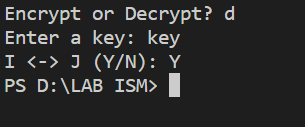




Output is Encrypted text as shown below in **text\_encrypted.txt**.



Then we decrypt the file using playfair.cpp again:



Output is Encrypted text as shown below in **text\_decrypted.txt**.

