```c
/**************************************************************
 *                                                            *
 * HW04 Q2                                                    *
 * Student Name: HASAN MEN                                    *
 * Student ID  : 131044009                                    *
 * Date        : 15.3.15                                      *
 * Points      : This program takes an file which has crypted*
 *  messages and turn this codes encoded file after that      *
 *  solves cypher and writes plain message in plain text file*
 *                                                            *
 **************************************************************/
#include <stdio.h>


#define PLAINTEXTFILE "Files/Q2/ReceivedMessage.txt"
#define ENCODEDFILE "Files/Q2/EncodedInput.txt"
#define CRYPTEDINPUT "Files/Q2/CryptedInput.txt"

/*function prototypes*/

/**************************************************************
 * Gets FILE* to write file and character to decode           *
 * uses encoding table to convert encoded message to          *
 * plain text message                                         *
 **************************************************************/
void
decode_and_write_to_file(FILE *f_out_ptr, int number_of_ones);

/**************************************************************
 * Gets FILE* f_in_ptr to read from encoded text file and     *
 * FILE* f_out_ptr to write message to plain text file        *
 * return number of characters read from encoded text         *
 **************************************************************/
int
decode_message(FILE *f_in_ptr, FILE *f_out_ptr);

/**************************************************************
 * Gets FILE* f_in_ptr to read from encrypted text file and  *
 * FILE* f_out_ptr to write message to encoded file           *
 * return encrypted character number                          *
 **************************************************************/
int
decrypt_message(FILE *f_in_ptr, FILE *f_out_ptr);

/**************************************************************
 * Reads encrypted text and creates encoded and               *
 * plain text files                                           *
 **************************************************************/
int
main(int argc, char* argv[])
{
    FILE *f_plane_ptr, *f_encoded_ptr, *f_crypted_ptr;

    /* exit progtam and print error if files couldn't be opened*/
    if((f_crypted_ptr= fopen(CRYPTEDINPUT,"r"))==NULL ||
       (f_encoded_ptr= fopen(ENCODEDFILE,"w"))==NULL )
    {
        if((f_crypted_ptr= fopen(CRYPTEDINPUT,"r"))==NULL)
            printf("Can't open the CRYPTEDINPUT to read.\n");
        else
            printf("Can't open the ENCODEDFILE to write.\n");
    }

    /*files opened , countinued */
    else
    {

        /*call decrypt_message() func. */
        decrypt_message(f_crypted_ptr, f_encoded_ptr);

        /*close files*/
        fclose(f_crypted_ptr);
        fclose(f_encoded_ptr);

        /* exit progtam and print error if files couldn't be opened*/
        if((f_plane_ptr= fopen(PLAINTEXTFILE,"w"))==NULL ||
            (f_encoded_ptr= fopen(ENCODEDFILE,"r"))==NULL )
        {
            if((f_plane_ptr= fopen(PLAINTEXTFILE,"w"))==NULL)
                    printf("Can't open the PLAINTEXTFILE to write\n");
            else
                printf("Can't open the ENCODEDFILE to read.\n");
        }

        /*files opened , countinued */
        else
        {

        /*call decode_message() */
        decode_message(f_encoded_ptr, f_plane_ptr);
        printf("\n****  ENCRYPTED   ****\n");

        /*close_files*/
        fclose(f_crypted_ptr);
        fclose(f_encoded_ptr);
        }
    }
    return 0;
    /*end of main */
}

/***********************************************************
 * Gets FILE* to write file and character to decode          *
 * uses encoding table to convert encoded message to         *
 * plain text message                                        *
 ***********************************************************/
void
decode_and_write_to_file(FILE *f_out_ptr, int number_of_ones)
{

    /* write character according to number of ones */
    switch(number_of_ones)
    {
        case 0: fprintf(f_out_ptr,"E"); break;
        case 1: fprintf(f_out_ptr,"I"); break;
        case 2: fprintf(f_out_ptr," "); break;
        case 3: fprintf(f_out_ptr,"T"); break;
        case 4: fprintf(f_out_ptr,"C"); break;
        case 5: fprintf(f_out_ptr,"N"); break;
        case 6: fprintf(f_out_ptr,"A"); break;
        case 7: fprintf(f_out_ptr,"G"); break;
        case 8: fprintf(f_out_ptr,"B"); break;
        case 9: fprintf(f_out_ptr,"Z"); break;
        case 10:fprintf(f_out_ptr,"H"); break;
        case 11:fprintf(f_out_ptr,"L"); break;
        case 12:fprintf(f_out_ptr,"U"); break;
        case 13:fprintf(f_out_ptr,"V"); break;
        case 14:fprintf(f_out_ptr,"R"); break;
        case 15:fprintf(f_out_ptr,"S"); break;
        case 16:fprintf(f_out_ptr,"Y"); break;

    }

}

/***********************************************************
 * Gets FILE* f_in_ptr to read from encoded text file and    *
 * FILE* f_out_ptr to write message to plain text file       *
 * return number of characters read from encoded text        *
 ***********************************************************/
int
decode_message(FILE *f_in_ptr, FILE *f_out_ptr)
{
    /* Hint: While reading from encoded text file keep reading
     *       character by character. Count the number of ones and
     *       Call decode_and_write_to_file function when you detect 0.
     */

    int counter = 0;    /* number of 1-0 readed from file */
    char character;     /* keeps 1-0 */
    int number_of_ones=0;   /* number of ones*/
    /*end of local variables */

    while((fscanf(f_in_ptr,"%c",&character))!=EOF)/*read until end of file*/
    {
                counter++;
                if(character=='1')
                {
                    number_of_ones++;
                }
                else if(character=='0')
                {
                    decode_and_write_to_file(f_out_ptr,number_of_ones);
                    number_of_ones=0; /* reset number of ones*/
                }
    }
    return counter;
}

/**************************************************************
 * Gets FILE* f_in_ptr to read from encrypted text file and  *
 * FILE* f_out_ptr to write message to encoded file          *
 * return encrypted character number                         *
 **************************************************************/
int
decrypt_message(FILE *f_in_ptr, FILE *f_out_ptr)
{
    /* Hint:While reading from encrypted text file check if character
     *       equals to '*' or '_' and write to file 1 or 0
     *
     */
    char number;    /* character, read from file */
    int counter = 0; /*encrypted character number */
    /*end of local variables */

    while((fscanf(f_in_ptr,"%c",&number))!=EOF)/* read,end of file*/
    {
        if(number!='\n')
        {
            if(number=='*')
            {
                fprintf(f_out_ptr,"1");
                counter++;
            }
            else if(number=='_')
            {
                fprintf(f_out_ptr,"0");
                counter++;
            }
        }
    }
    return counter;
}

/* end of HW04_HASAN_MEN_131044009_part2.c */
```