7.  
Write a High level code  for  
monoalphabetic cipher is that both sender and receiver must commit the permuted  
cipher sequence to memory. A common technique for avoiding this is to use a  
keyword from which the cipher sequence can be generated.For example, using the  
keyword CIPHER, write out the keyword followed by unused letters in normal order  
and match this against the plaintext letters:  
  
plain:  
a b c d e f g h i j k l m n o p q r s t u v w x y z  
  
cipher:  
C I P H E R A B D F G J K L M N O  
Q S T U V W X Y Z

PROGRAM:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

void generateMonoalphabeticKey(char keyword[], char key[]) {

int keywordLength = strlen(keyword);

int keyIndex = 0;

strcpy(key, keyword);

keyIndex = keywordLength;

char letter;

int i;

for (letter = 'A'; letter <= 'Z'; letter++) {

int found = 0;

for (i = 0; i < keywordLength; i++) {

if (toupper(keyword[i]) == letter) {

found = 1;

break;

}

}

if (!found) {

key[keyIndex++] = letter;

}

}

key[keyIndex] = '\0';

}

void monoalphabeticEncrypt(char message[], char key[]) {

int messageLength = strlen(message);

int i; for (i = 0; i < messageLength; i++) {

if (isalpha(message[i])) {

char base = islower(message[i]) ? 'a' : 'A';

int index = toupper(message[i]) - 'A';

message[i] = islower(message[i]) ? tolower(key[index]) : key[index];

}

}

}

int main() {

char keyword[] = "CIPHER";

char plaintext[100];

char cipherKey[27];

char ciphertext[100];

generateMonoalphabeticKey(keyword, cipherKey);

printf("Enter the plaintext: ");

scanf(" %[^\n]s", plaintext);

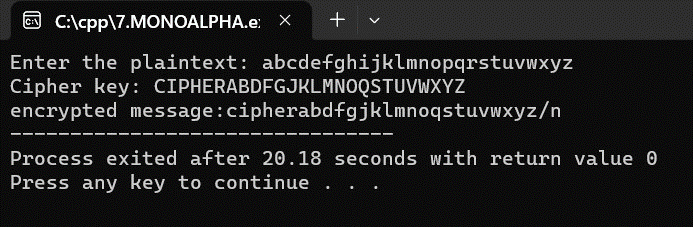
monoalphabeticEncrypt(plaintext, cipherKey);

printf("Cipher key: %s\n", cipherKey);

printf("encrypted message:%s/n",plaintext);

return 0;

}



8. 8. Write a High level codefor PT-109  
American patrol boat, under the command of Lieutenant John F.  
Kennedy, was sunk by a Japanese destroyer, a message was received at an Australian  
wireless station in  
Playfair  
code:  
KXJEY  
UREBE ZWEHE WRYTU HEYFS  
KREHE  
GOYFI WTTTU OLKSY CAJPO  
BOTEI  
ZONTX BYBNT GONEY CUZWR  
GDSON  
SXBOU YWRHE BAAHY USEDQ  
  
program:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

void findCharPosition(char matrix[5][5], char ch, int \*row, int \*col) {

if (ch == 'J') ch = 'I';

int i;

int j;

for (i = 0; i < 5; i++) {

for (j = 0; j < 5; j++) {

if (matrix[i][j] == ch) {

\*row = i;

\*col = j;

return;

}

}

}

}

void decryptPlayfair(char message[], char matrix[5][5]) {

int len = strlen(message);

int i = 0;

char plaintext[len];

while (i < len) {

if (!isalpha(message[i])) {

plaintext[i] = message[i]; i++;

continue;

}

char first = message[i++];

char second = message[i++];

if (first == 'J') first = 'I';

if (second == 'J') second = 'I';

int row1, col1, row2, col2;

findCharPosition(matrix, first, &row1, &col1);

findCharPosition(matrix, second, &row2, &col2);

if (row1 == row2) {

plaintext[i - 2] = matrix[row1][(col1 - 1 + 5) % 5];

plaintext[i - 1] = matrix[row2][(col2 - 1 + 5) % 5];

} else if (col1 == col2) {

plaintext[i - 2] = matrix[(row1 - 1 + 5) % 5][col1];

plaintext[i - 1] = matrix[(row2 - 1 + 5) % 5][col2];

} else {

plaintext[i - 2] = matrix[row1][col2];

plaintext[i - 1] = matrix[row2][col1];

}

}

plaintext[len] = '\0';strcpy(message, plaintext);

}

int main() {

char message[] = "KXJEY UREBE ZWEHE WRYTU HEYFS KREHE GOYFI WTTTU OLKSY CAJPO BOTEI ZONTX BYBNT GONEY CUZWR GDSON SXBOU YWRHE BAAHY USEDQ";

char matrix[5][5] = {

{'K', 'X', 'J', 'E', 'Y'},

{'U', 'R', 'E', 'B', 'Z'},

{'W', 'H', 'F', 'S', 'T'},

{'G', 'O', 'Y', 'I', 'L'},

{'C', 'A', 'P', 'D', 'Q'}

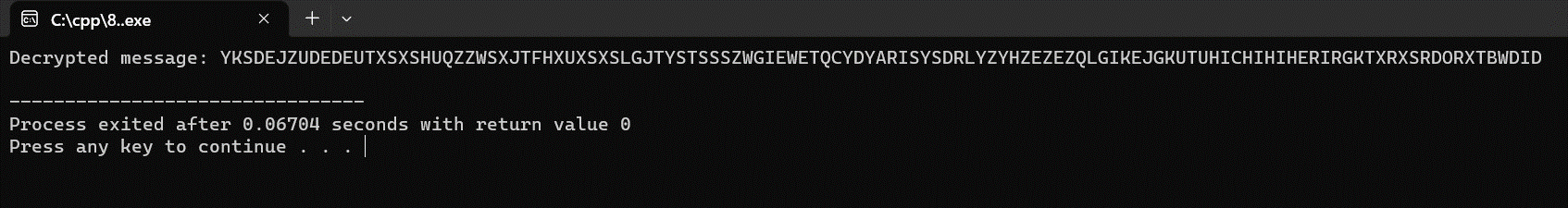
};

decryptPlayfair(message, matrix);

printf("Decrypted message: %s\n", message);

return 0;

}



9. Write a High level code for Playfair  
matrix:  
M  
F H I/J K  
U  
N O P Q  
Z  
V W X Y  
E  
L A R G  
D  
S T B C  
Encrypt  
this message: Must see you over Cadogan West. Coming at once.  
program:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

void findCharPosition(char matrix[5][5], char ch, int \*row, int \*col) {

if (ch == 'J') ch = 'I';

int i;

int j;

for (i = 0; i < 5; i++) {

for (j = 0; j < 5; j++) {

if (matrix[i][j] == ch) {

\*row = i;

\*col = j;

return;

}

}

}

}

void encryptPlayfair(char message[], char matrix[5][5]) {

int len = strlen(message);

int i = 0;

char ciphertext[len \* 2];

while (i < len) {

if (!isalpha(message[i])) {

ciphertext[i \* 2] = message[i]; i++;

continue;

}

char first = message[i++];

char second = (i < len && isalpha(message[i])) ? message[i++] : 'X';

if (first == 'J') first = 'I';

if (second == 'J') second = 'I';

int row1, col1, row2, col2;

findCharPosition(matrix, first, &row1, &col1);

findCharPosition(matrix, second, &row2, &col2);

if (row1 == row2) {

ciphertext[(i - 2) \* 2] = matrix[row1][(col1 + 1) % 5];

ciphertext[(i - 2) \* 2 + 1] = matrix[row2][(col2 + 1) % 5];

} else if (col1 == col2) {

ciphertext[(i - 2) \* 2] = matrix[(row1 + 1) % 5][col1];

ciphertext[(i - 2) \* 2 + 1] = matrix[(row2 + 1) % 5][col2];

} else {

ciphertext[(i - 2) \* 2] = matrix[row1][col2];

ciphertext[(i - 2) \* 2 + 1] = matrix[row2][col1];

}

}

ciphertext[len \* 2] = '\0';strcpy(message, ciphertext);

}

int main() {

char message[] = "Must see you over Cadogan West. Coming at once.";

char matrix[5][5] = {

{'M', 'F', 'H', 'I', 'K'},

{'U', 'N', 'O', 'P', 'Q'},

{'Z', 'V', 'W', 'X', 'Y'},

{'E', 'L', 'A', 'R', 'G'},

{'D', 'S', 'T', 'B', 'C'}

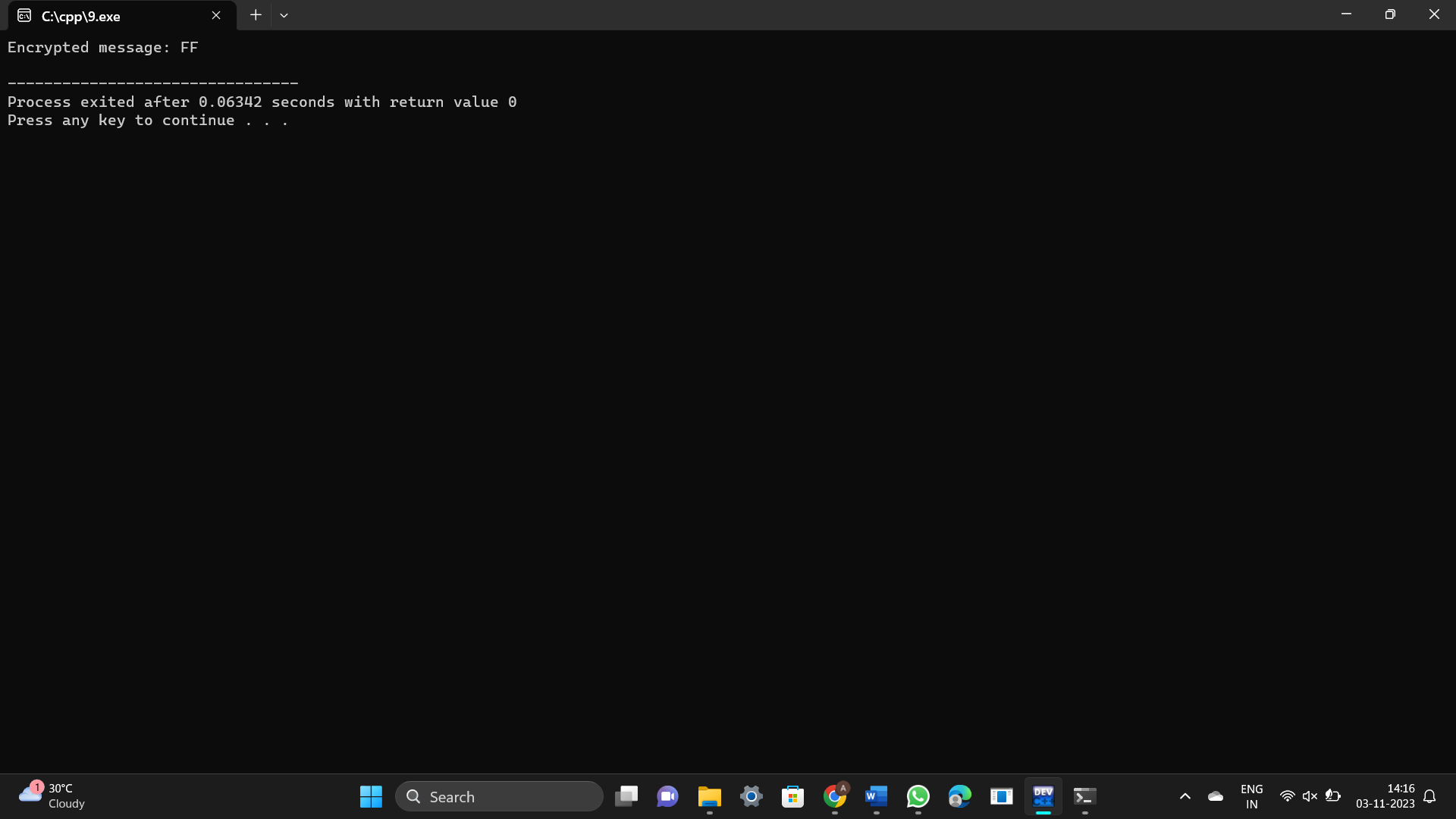
};

encryptPlayfair(message, matrix);

printf("Encrypted message: %s\n", message);

return 0;

}



10. Write a High level code for possible  
keys does the Playfair cipher have? Ignore the fact that some keys might  
produce identical encryption results. Express your answer as an approximate  
power of 2.  
a.  
Now take into account the fact that some Playfair keys produce the same  
encryption results. How many effectively unique keys does the Playfair cipher  
have

Program:

#include <stdio.h>

#include <math.h>

unsigned long long factorial(int n) {

if (n <= 1) {

return 1;

}

return n \* factorial(n - 1);

}

int main() {

int n = 25; // 25 unique characters for the 5x5 Playfair key matrix

unsigned long long total\_keys = factorial(n);

printf("Total number of keys: %llu\n", total\_keys);

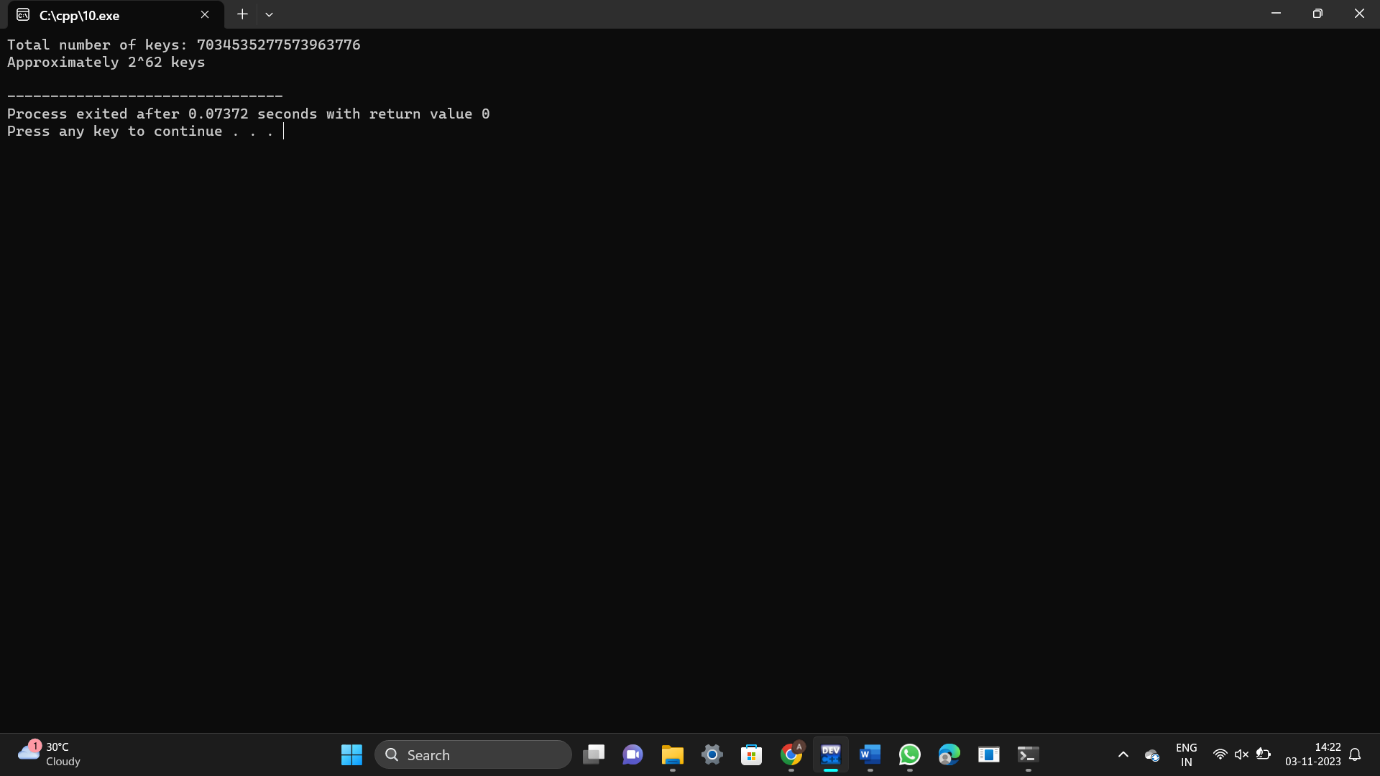
// Express as an approximate power of 2

int power\_of\_2 = (int)(log2(total\_keys));

printf("Approximately 2^%d keys\n", power\_of\_2);

return 0;

}



11. Write a High level code to Encrypt  
the message “meet me at the usual place at ten rather than eight oclock” using  
the Hill cipher with the key.  
9  
4  
5  
7  
a.  
Show your calculations and the result.  
b.  
Show the calculations for the corresponding decryption of the ciphertext to  
recover the original plaintext.

Program:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int gcd(int a, int b) {

if (b == 0)

return a;

return gcd(b, a % b);

}

int modInverse(int a, int m) {

a = a % m;

int x;

for (x = 1; x < m; x++) {

if ((a \* x) % m == 1) {

return x;

}

}

return -1;

}

int main() {

int keyMatrix[2][2] = {{9, 4}, {5, 7}};

char plaintext[] = "meet me at the usual place at ten rather than eight oclock";

char ciphertext[1000];

char decryptedText[1000];int determinant = keyMatrix[0][0] \* keyMatrix[1][1] - keyMatrix[0][1] \* keyMatrix[1][0];

if (gcd(determinant, 26) != 1) {

printf("The provided key matrix is not invertible.\n");

return 1;

}

int modInverseDet = modInverse(determinant, 26);

int len = strlen(plaintext);

int col, row, sum;

int i;

for (i = 0; i < len; i++) {

if (isalpha(plaintext[i])) {

if (islower(plaintext[i])) {

plaintext[i] = toupper(plaintext[i]) - 'A';

} else {

plaintext[i] = plaintext[i] - 'A';

}

} else if (plaintext[i] == ' ') {

plaintext[i] = 26;

}

col = i % 2;

if (col == 0) {

row = i;

} else {sum = 0;

int j;

for (j = 0; j < 2; j++) {

sum += keyMatrix[row][j] \* plaintext[i - j];

}

ciphertext[i - 1] = (sum % 26);

}

}

if (len % 2 != 0) {

sum = 0;

int j;

for (j = 0; j < 2; j++) {

sum += keyMatrix[row][j] \* plaintext[len - 1 - j];

}

ciphertext[len - 1] = (sum % 26);

}

for (i = 0; i < len; i++) {

if (ciphertext[i] == 26) {

decryptedText[i] = ' ';

} else {

decryptedText[i] = ciphertext[i] + 'A';

}

}

decryptedText[len] = '\0';printf("Plaintext: %s\n", plaintext);

printf("Ciphertext: %s\n", decryptedText);

return 0;

}

