**1)caesar cipher**

#include<stdio.h>

#include<string.h>

int main()

{

char str[50];

int k,ch;

printf("1.Encryption\n2.Decryption\nEnter your operarion:");

scanf("%d",&ch);

printf("Enter the string : ");

scanf("%s",str);

printf("Enter the key : ");

scanf("%d",&k);

if(ch==1){

for(int i=0;i<strlen(str);i++)

{

if(str[i]>='a' && str[i]<='z'){

int d=int(str[i])-96;

int n=(d+k)%26;

char c=char(n+96);

printf("%c",c);}

else if(str[i]>='0' && str[i]<='9')

{

printf("%c",str[i]+k);

}

else if(str[i]>='A' && str[i]<='Z')

{

int d=int(str[i])-64;

int n=(d+k)%26;

char c=char(n+64);

printf("%c",c);

}

}}

if(ch==2){

for(int i=0;i<strlen(str);i++)

{

if(str[i]>='a' && str[i]<='z'){

int d=int(str[i])-96;

int n=(d-k)%26;

char c=char(n+96);

printf("%c",c);}

else if(str[i]>='0' && str[i]<='9')

{

printf("%c",str[i]-k);

}

else if(str[i]>='A' && str[i]<='Z')

{

int d=int(str[i])-64;

int n=(d-k)%26;

char c=char(n+64);

printf("%c",c);

}

}}

}

1. **Playfair**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#define SIZE 5

// Function to prepare the key matrix

void prepareKeyMatrix(char key[], char keyMatrix[SIZE][SIZE]) {

int row, col, k = 0;

int isPresent[26] = {0};

for (row = 0; row < SIZE; row++) {

for (col = 0; col < SIZE; col++) {

if (k < strlen(key)) {

if (!isPresent[toupper(key[k]) - 'A']) {

keyMatrix[row][col] = toupper(key[k]);

isPresent[key[k] - 'A'] = 1;

k++;

} else {

col--;

}

} else {

for (int l = 0; l < 26; l++) {

if (!isPresent[l]) {

keyMatrix[row][col] = l + 'A';

isPresent[l] = 1;

break;

}

}

}

}

}

}

// Function to find the coordinates of a character in the matrix

void findPosition(char keyMatrix[SIZE][SIZE], char ch, int \*row, int \*col) {

if (ch == 'J') // Treat 'J' as 'I'

ch = 'I';

for (\*row = 0; \*row < SIZE; (\*row)++) {

for (\*col = 0; \*col < SIZE; (\*col)++) {

if (keyMatrix[\*row][\*col] == ch) {

return;

}

}

}

}

// Function to perform Playfair encryption

void playfairEncrypt(char keyMatrix[SIZE][SIZE], char message[], char encryptedMessage[]) {

int i, row1, col1, row2, col2;

for (i = 0; i < strlen(message); i += 2) {

findPosition(keyMatrix, message[i], &row1, &col1);

findPosition(keyMatrix, message[i + 1], &row2, &col2);

if (row1 == row2) {

encryptedMessage[i] = keyMatrix[row1][(col1 + 1) % SIZE];

encryptedMessage[i + 1] = keyMatrix[row2][(col2 + 1) % SIZE];

} else if (col1 == col2) {

encryptedMessage[i] = keyMatrix[(row1 + 1) % SIZE][col1];

encryptedMessage[i + 1] = keyMatrix[(row2 + 1) % SIZE][col2];

} else {

encryptedMessage[i] = keyMatrix[row1][col2];

encryptedMessage[i + 1] = keyMatrix[row2][col1];

}

}

encryptedMessage[i] = '\0';

}

int main() {

char key[26], message[100], encryptedMessage[100];

char keyMatrix[SIZE][SIZE];

printf("Enter the key (uppercase, no repeated letters, treat 'J' as 'I'): ");

scanf("%s", key);

printf("Enter the message to encrypt (uppercase, no spaces): ");

scanf("%s", message);

prepareKeyMatrix(key, keyMatrix);

playfairEncrypt(keyMatrix, message, encryptedMessage);

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

return 0;

}

1. **Hill cipher**

#include<stdio.h>

#include<string.h>

void matmul(int a[2][2],int b[2][1]){

int mul[2][1];

for(int i=0;i<2;i++){

for(int j=0;j<1;j++){

mul[i][j]=0;

for(int k=0;k<2;k++){

mul[i][j]+=a[i][k]\*b[k][j];

}

}

}

for(int i=0;i<2;i++){

for(int j=0;j<1;j++){

char c=char((mul[i][j]%26)+96);

printf("%c",c);

}

}

}

int main()

{

int k[2][2],a[2][1];

printf("Enter the key : ");

for(int i=0;i<2;i++){

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

scanf("%d",&k[i][j]);

}

}

char s[1000];

int l=0;

printf("Enter the plaintext : ");

scanf("%s",s);

int x=strlen(s);

if(x%2==1){

s[x]=='x';

}

for(int i=0;i<strlen(s);i=i+2){

if(s[i]!='\0'){

int x=0;

a[x++][0]=int(s[i])-96;

a[x++][0]=int(s[i+1])-96;

matmul(k,a);}

}

}

1. **Monoalphabetic**

#include<stdio.h>

#include<string.h>

int main()

{

char pt[50],k[50];

printf("Enter the key : ");

scanf("%s",k);

printf("Enter the plaintext : ");

scanf("%s",pt);

int x=strlen(k);

for(int i=0;i<strlen(pt);i++){

int a=int(pt[i])-96;

int b=int(k[i%x])-96;

int c=(a+b)%26;

char d= char(c+96);

printf("%c",d);

}

}

1. **Polyalphabetic**

#include<stdio.h>

#include<string.h>

int main()

{

char pt[50],k[50];

printf("Enter the key : ");

scanf("%s",k);

printf("Enter the plaintext : ");

scanf("%s",pt);

int x=strlen(k);

for(int i=0;i<strlen(pt);i++){

int a=int(pt[i])-96;

int b=int(k[i%x])-96;

int c=(a+b)%26;

char d= char(c+96);

printf("%c",d);

}

}

1. **rail fence**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

main()

{

int i,j,k,c,code[50][50];

char str[50];

printf("Enter string:\n");

scanf("%s",str);

int l=strlen(str);

printf("Enter key:\n");

scanf("%d",&k);

for(i=0;i<k;i++)

{

for(j=0;j<l;j++)

{

code[i][j]=0;

}}

c=0;

j=0;

while(j<l)

{

if(c%2==0)

{

for(i=0;i<k;i++)

{

code[i][j]=(int)str[j];

j++;

}}

else

{

for(i=k-2;i>0;i--)

{

code[i][j]=(int)str[j];

j++;

} }

c++;

}

for(i=0;i<k;i++)

{

for(j=0;j<l;j++)

{

if(code[i][j]!=0)

printf("%c",code[i][j]);

} }

}

**7)Diffie-Hellman**

#include<stdio.h>

#include<math.h>

int main(){

int p,g,a,b,x,y,ka,kb;

printf("Enter public number1 : \n");

scanf("%d",&p);

printf("Enter public number2 : \n");

scanf("%d",&g);

printf("Enter private number1 : \n");

scanf("%d",&a);

printf("Enter private number2 : \n");

scanf("%d",&b);

x=fmod(pow(g,a),p);

y=fmod(pow(g,b),p);

ka=fmod(pow(y,a),p);

kb=fmod(pow(x,b),p);

printf("secret key1 : %d\nsecret key2 : %d",ka,kb);

}

**8)RSA**

#include<stdio.h>

#include<math.h>

int finde(int n){

int a[]={2,3,5,7,11,13,17,21,23};

for(int i=0;i<9;i++){

if(n%a[i]!=0){

return a[i];

}

}

}

int findd(int e,int n){

for(int i=1;i<=n;i++){

int x=(e\*i)%n;

if(x==1){

return i;

}

}

}

int main(){

int p,q,m,n,n1,e,d,c,d1,ch;

printf("1.Encryption \t2.Decryption\nENter your choice : ");

scanf("%d",&ch);

if(ch==1){

printf("Enter the plain text : ");

scanf("%d",&m);

printf("Enter two prime numbers : ");

scanf("%d",&p);

scanf("%d",&q);

n=p\*q;

n1=(p-1)\*(q-1);

e=finde(n1);

d=findd(e,n1);

printf("public key : %d %d\nPrivate key : %d %d",e,n,d,n);

c=fmod(pow(m,e),n);

printf("\nCipher text : %d",c);

}

if(ch==2){

printf("Enter private key pair : ");

scanf("%d,%d",&d,&n);

printf("ENter the cipher text : ");

scanf("%d",&c);

m=fmod(pow(c,d),n);

printf("\nPlain text : %d",m);

}

}

1. **SHA**

#include <stdio.h>

#include <stdint.h>

#include <string.h>

#define LEFTROTATE(x, c) (((x) << (c)) | ((x) >> (32 - (c))))

void sha1(uint8\_t \*message, uint32\_t message\_length, uint32\_t hash[5]) {

uint32\_t h0, h1, h2, h3, h4;

h0 = 0x67452301;

h1 = 0xEFCDAB89;

h2 = 0x98BADCFE;

h3 = 0x10325476;

h4 = 0xC3D2E1F0;

hash[0] = h0;

hash[1] = h1;

hash[2] = h2;

hash[3] = h3;

hash[4] = h4;

}

int main() {

char message[50];

printf("\nEnter the message : ");

scanf("%s",message);

uint32\_t hash[5];

sha1((uint8\_t \*)message, strlen(message), hash);

printf("SHA-1 Hash: %08x%08x%08x%08x%08x\n", hash[0], hash[1], hash[2], hash[3], hash[4]);

return 0;

}

1. **MD5**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <stdint.h>

#define LEFTROTATE(x, c) (((x) << (c)) | ((x) >> (32 - (c))))

void md5(uint8\_t \*initial\_msg, size\_t initial\_len) {

uint32\_t h0, h1, h2, h3;

uint8\_t \*msg = NULL;

h0 = 0x67452301;

h1 = 0xEFCDAB89;

h2 = 0x98BADCFE;

h3 = 0x10325476;

size\_t new\_len = ((((initial\_len + 8) / 64) + 1) \* 64);

msg = (uint8\_t \*)calloc(new\_len, 1);

memcpy(msg, initial\_msg, initial\_len);

msg[initial\_len] = 128;

uint32\_t bits\_len = initial\_len \* 8;

memcpy(msg + new\_len - 8, &bits\_len, 4);

for (size\_t offset = 0; offset < new\_len; offset += 64) {

uint32\_t \*w = (uint32\_t \*)(msg + offset);

uint32\_t a = h0;

uint32\_t b = h1;

uint32\_t c = h2;

uint32\_t d = h3;

for (int i = 0; i < 64; i++) {

uint32\_t f, g;

if (i < 16) {

f = (b & c) | ((~b) & d);

g = i;

} else if (i < 32) {

f = (d & b) | ((~d) & c);

g = (5 \* i + 1) % 16;

} else if (i < 48) {

f = b ^ c ^ d;

g = (3 \* i + 5) % 16;

} else {

f = c ^ (b | (~d));

g = (7 \* i) % 16;

}

uint32\_t temp = d;

d = c;

c = b;

b = b + LEFTROTATE((a + f + 0x5A827999 + w[g]), 7);

a = temp;

}

h0 += a;

h1 += b;

h2 += c;

h3 += d;

}

free(msg);

printf("MD5 Hash: %08x%08x%08x%08x\n", h0, h1, h2, h3);

}

int main() {

char message[50];

printf("\nEnter the message : ");

scanf("%s",message);

size\_t message\_len = strlen(message);

md5((uint8\_t \*)message, message\_len);

return 0;

}