**Experiment -1**

1. Ceaser Cipher-
2. Ceaser Cipher Encryption

**Code-**

#include <bits/stdc++.h>

using namespace std;

int main()

{

cout << "Caesar Cipher program for encryption : " << endl;

int key;

cout << "Enter the message : ";

getline(cin, s);

cout << "Enter the key : ";

cin >> key;

for (int i = 0; i < s.size(); i++)

{

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

char originalChar = s[i];

t += (((originalChar + key) + 128) % 128);

}

else

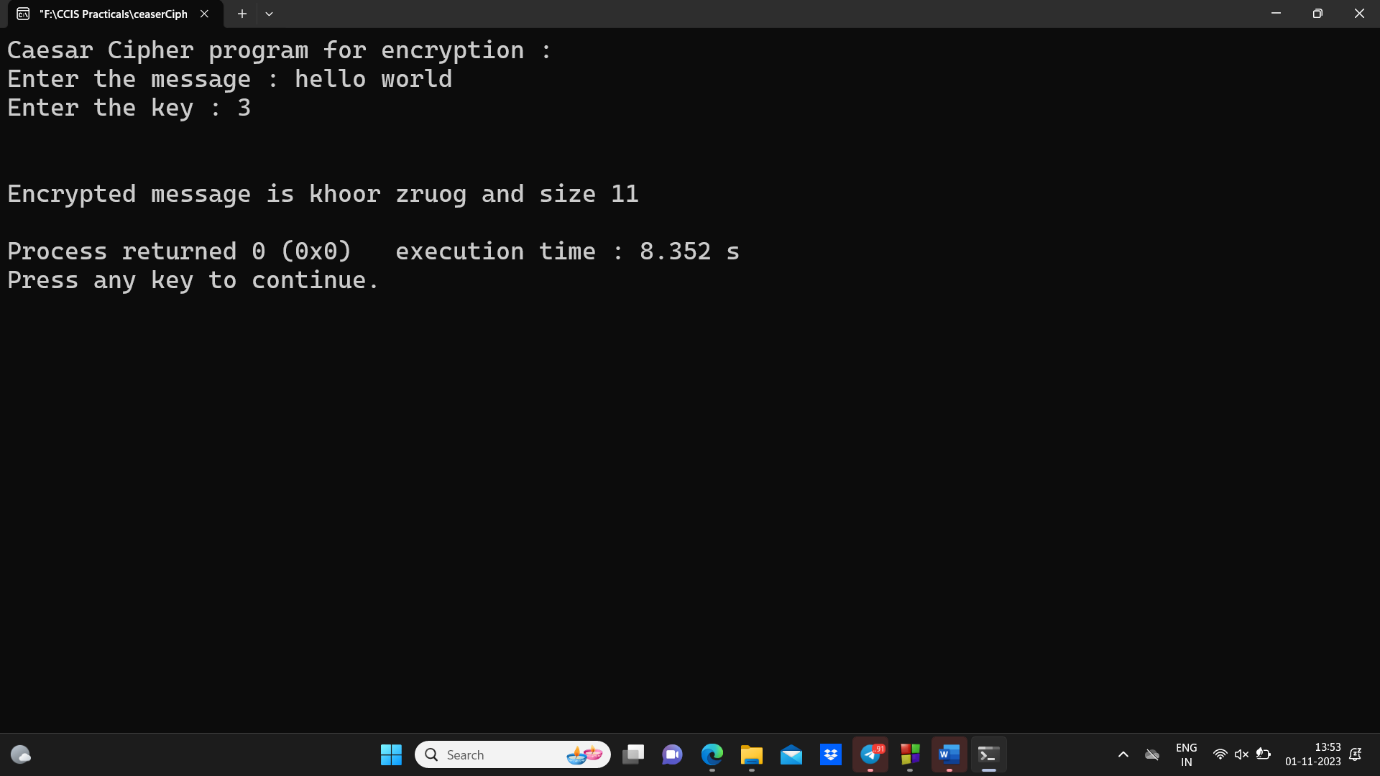
t += " ";

}

cout << "\n\nEncrypted message is " << t <<" and size "<<t.size() <<'\n';

return 0;

}

**Output-**

1. Ceaser Cipher Decryption-

**Code-**

#include <bits/stdc++.h>

using namespace std;

int main()

{

cout << "Caesar Cipher program for encryption : " << endl;

string s, t;

int key;

cout << "Enter the message : ";

getline(cin, s);

cout << "Enter the key : ";

cin >> key;

for (int i = 0; i < s.size(); i++)

{

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

char originalChar = s[i];

t += (((originalChar + key) + 128) % 128);

}

else

t += " ";

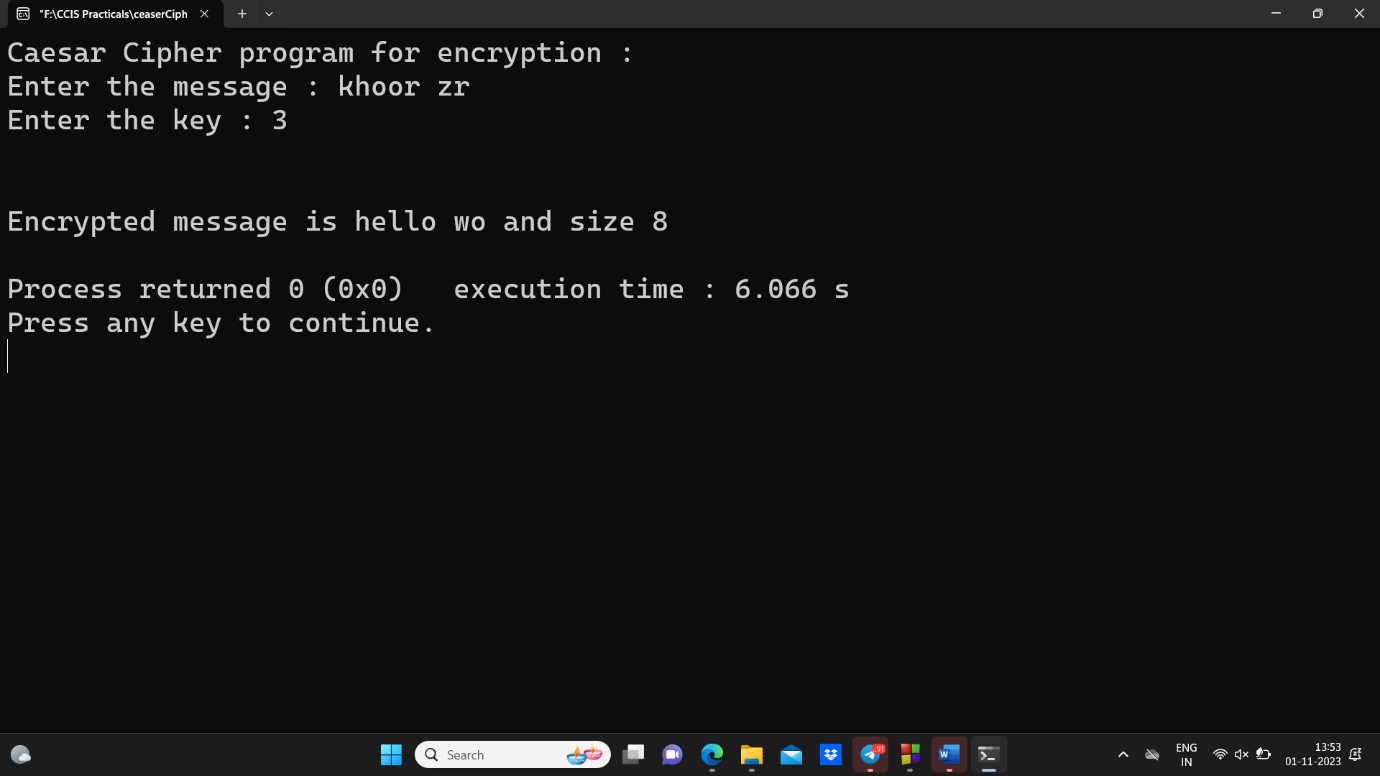
}

cout << "\n\nEncrypted message is " << t <<" and size "<<t.size() <<'\n';

return 0;

}

**Output-**



1. PlayFair-
2. Encryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int main(){

int i,j,k,n;

cout<<"Enter the message"<<endl;

string s,origin;

getline(cin,origin);

cout<<"Enter the key"<<endl;

string key;

cin>>key;

for(i=0;i<origin.size();i++){

if(origin[i]!=' ')

s+= origin[i];

}

vector<vector<char> > a(5,vector<char>(5,' '));

n=5;

map<char,int> mp;

k=0;

int pi,pj;

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

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

while(mp[key[k]]>0&&k<key.size()){

k++;

}

if(k<key.size()){

a[i][j]=key[k];

mp[key[k]]++;

pi=i;

pj=j;

}

if(k==key.size())

break;

}

if(k==key.size())

break;

}

k=0;

for(;i<n;i++){

for(;j<n;j++){

while(mp[char(k+'a')]>0&&k<26){

k++;

}

if(char(k+'a')=='j'){

j--;

k++;

continue;

}

if(k<26){

a[i][j]=char(k+'a');

mp[char(k+'a')]++;

}

}

j=0;

}

string ans;

if(s.size()%2==1)

s+="x";

for(i=0;i<s.size()-1;i++){

if(s[i]==s[i+1])

s[i+1]='x';

}

map<char,pair<int,int> > mp2;

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

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

mp2[a[i][j]] = make\_pair(i,j);

}

}

for(i=0;i<s.size()-1;i+=2){

int y1 = mp2[s[i]].first;

int x1 = mp2[s[i]].second;

int y2 = mp2[s[i+1]].first;

int x2 = mp2[s[i+1]].second;

if(y1==y2){

ans+=a[y1][(x1+1)%5];

ans+=a[y1][(x2+1)%5];

}

else if(x1==x2){

ans+=a[(y1+1)%5][x1];

ans+=a[(y2+1)%5][x2];

} else {

ans+=a[y1][x2];

ans+=a[y2][x1];

} }

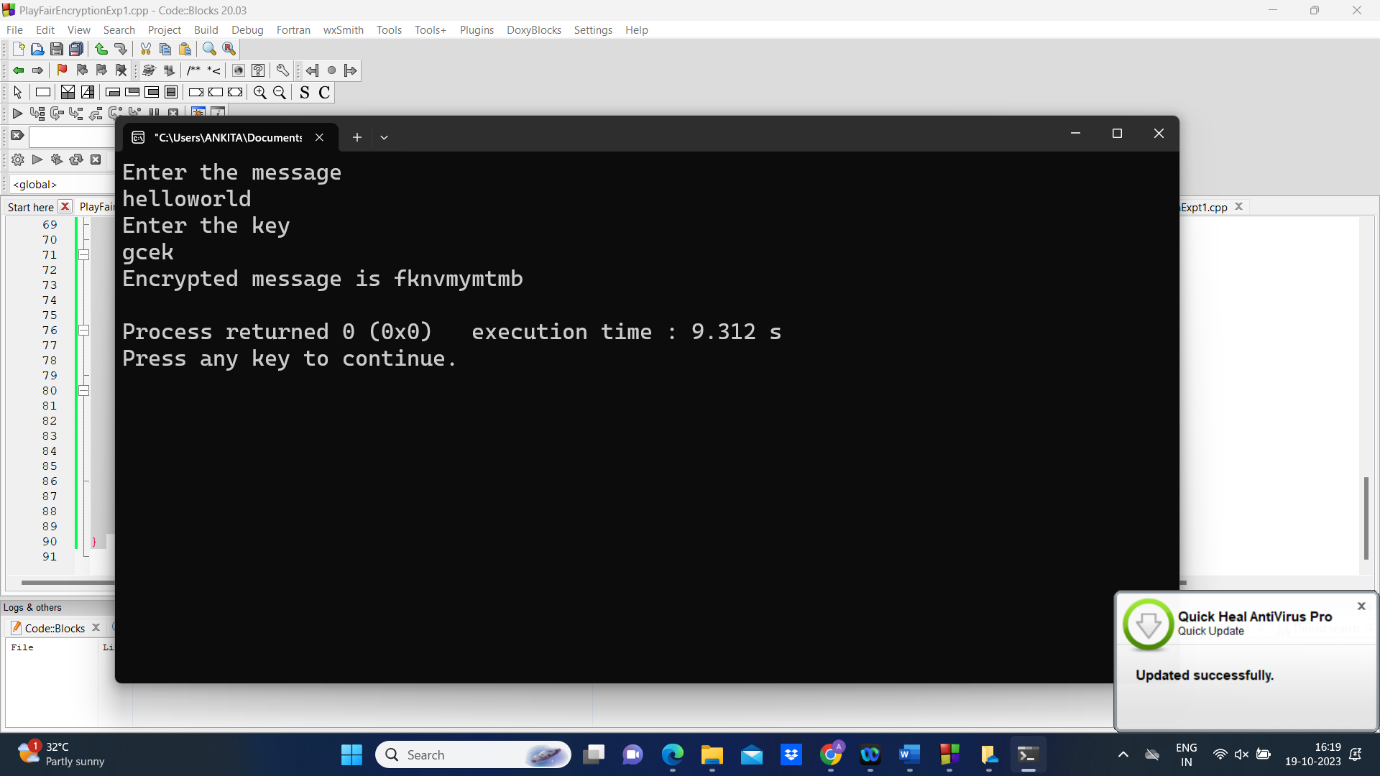
cout<<"Encrypted message is ";

cout<<ans<<'\n';

return 0;

}

**Output-**



1. Decryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int main(){

cout<<"Enter the encrypted message\n";

string s;

cin>>s;

int i,j,k,n;

cout<<"Enter the key\n";

string key;

cin>>key;

vector<vector<char> > a(5,vector<char>(5,' '));

n=5;

map<char,int> mp;

k=0;

int pi,pj;

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

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

while(mp[key[k]]>0&&k<key.size()){

k++;

}

if(k<key.size()){

a[i][j]=key[k];

mp[key[k]]++;

pi=i;

pj=j;

}

if(k==key.size())

break;

}

k=0;

for(;i<n;i++){

for(;j<n;j++){

while(mp[char(k+'a')]>0&&k<26){

k++;

}

if(char(k+'a')=='j'){

j--;

k++;

continue;

}

if(k<26){

a[i][j]=char(k+'a');

mp[char(k+'a')]++;

}

}

j=0;}

string ans;

map<char,pair<int,int> > mp2;

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

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

mp2[a[i][j]] = make\_pair(i,j);

}

}

for(i=0;i<s.size()-1;i+=2){

int y1 = mp2[s[i]].first;

int x1 = mp2[s[i]].second;

int y2 = mp2[s[i+1]].first;

int x2 = mp2[s[i+1]].second;

if(y1==y2){

ans+=a[y1][(x1-1)%5];

ans+=a[y1][(x2-1)%5];

}

else if(x1==x2){

ans+=a[(y1-1)%5][x1];

ans+=a[(y2-1)%5][x2];

}

else {

ans+=a[y1][x2];

ans+=a[y2][x1];

}

}

if(ans[ans.size()-1]=='x')

{

ans[ans.size()-1]='\0';

}

for(i=1;i<ans.size();i++){

if(ans[i]=='x') {

ans[i]=ans[i-1];

}

}

cout<<"Decrypted message is ";

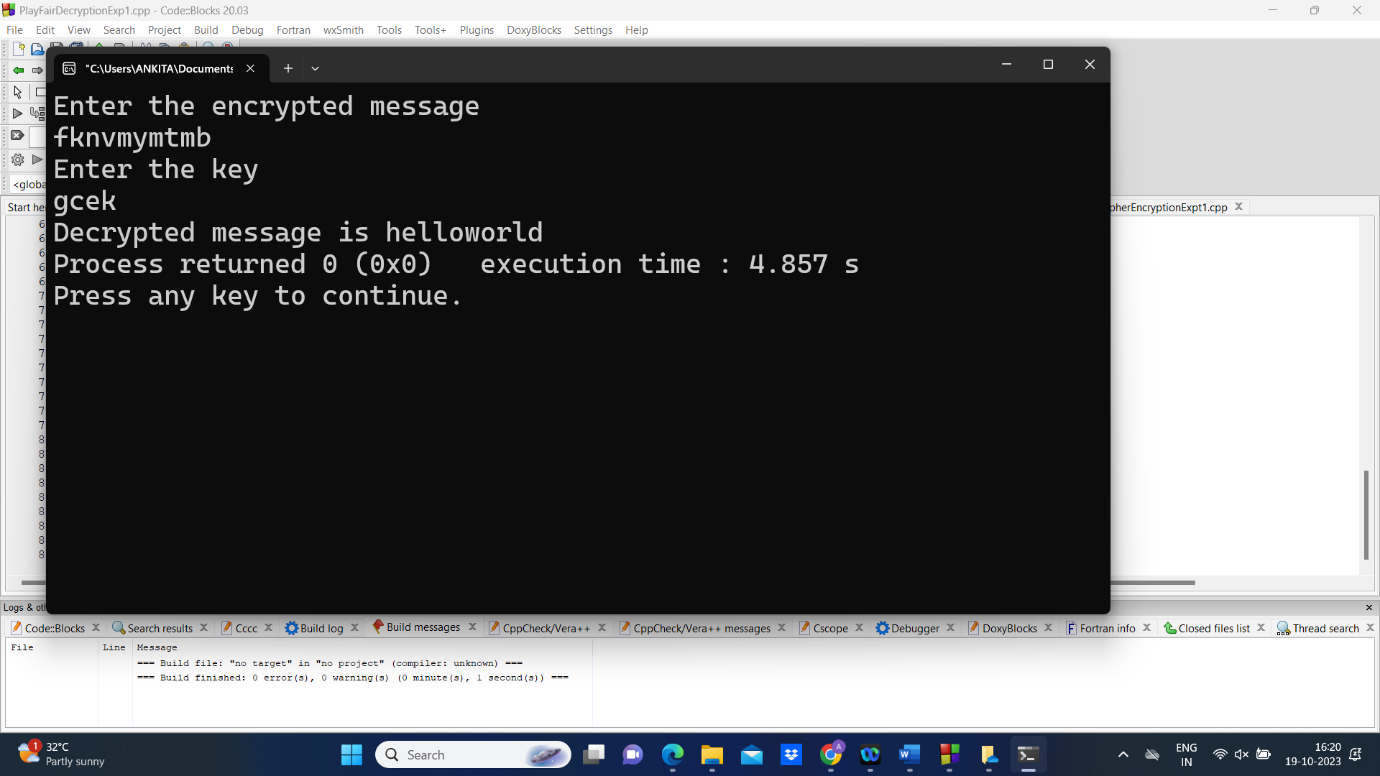
cout<<ans<<" ";

return 0;

}

}

**Output-**



1. Hill Cipher-
2. Encryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int main(){

int x,y,i,j,k,n;

cout<<"Enter the size of key matrix\n";

cin>>n;

cout<<"Enter the key matrix\n";

int a[n][n];

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

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

cin>>a[i][j];

} }

cout<<"Enter the message to encrypt\n";

string s;

cin>>s;

int temp = (n-s.size()%n)%n;

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

s+='x';

}

k=0;

string ans="";

while(k<s.size()){

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

int sum = 0, temp = k;

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

sum += (a[i][j]%26\*(s[temp++]-'a')%26)%26;

sum = sum%26;

}

ans+=(sum+'a');

}

k+=n;

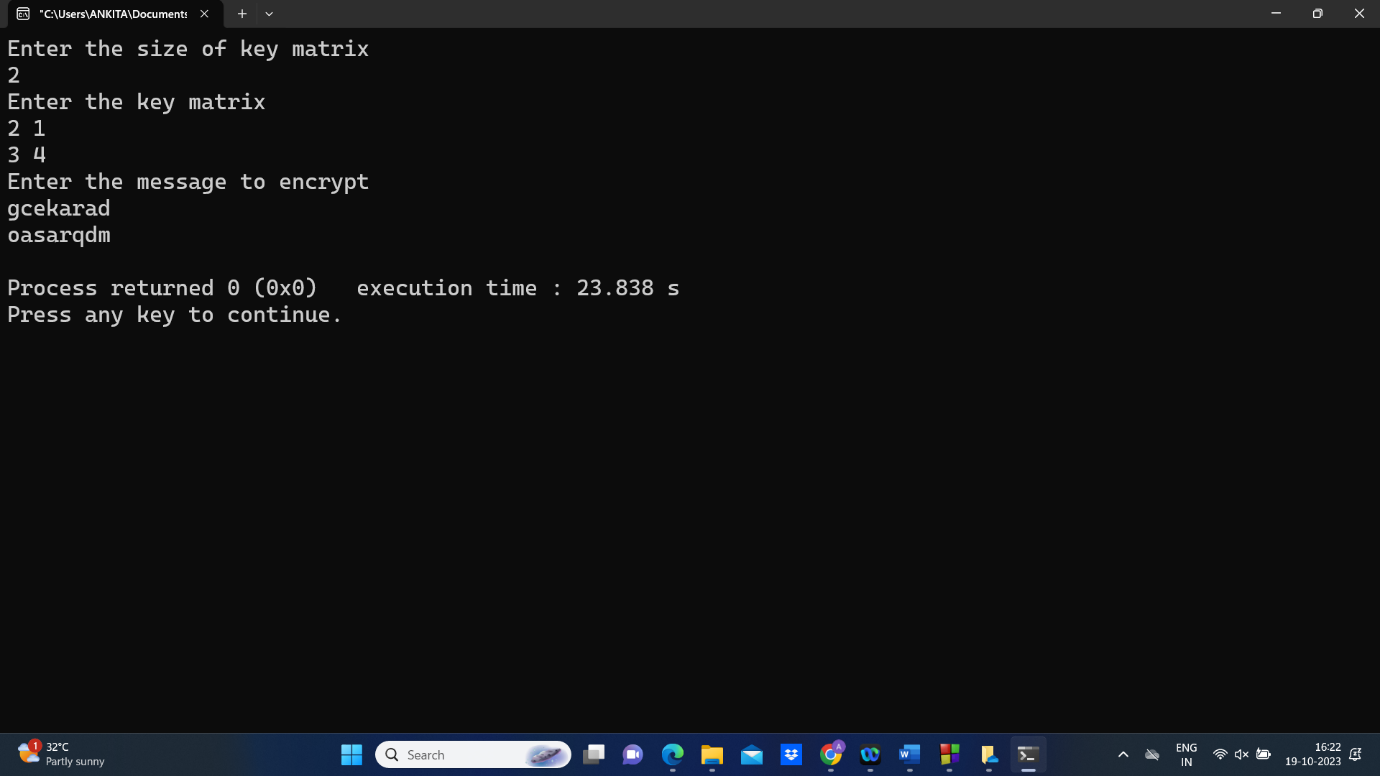
}

cout<<ans<<'\n';

return 0;

}

**Output-**



1. Decryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int modInverse(int a, int m){

a=a%m;

for(int x=-m;x<m;x++)

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

return x;

}

void getCofactor(vector<vector<int> > &a, vector<vector<int> > &temp, int p, int q, int n){

int i=0,j=0;

for(int row=0;row<n;row++){

for(int col=0;col<n;col++){

if(row!=p&&col!=q){

temp[i][j++] = a[row][col];

if (j==n-1){

j=0;

i++;

} } } } }

int determinant(vector<vector<int> > &a, int n, int N){

int D = 0;

if(n==1)

return a[0][0];

vector<vector<int> > temp(N, vector<int>(N));

int sign = 1;

for(int f=0;f<n;f++){

getCofactor(a, temp, 0, f, n);

D += sign \* a[0][f] \* determinant(temp, n - 1, N);

sign = -sign;

} return D; }

void adjoint(vector<vector<int> > &a,vector<vector<int> > &adj,int N){

if(N == 1){

adj[0][0] = 1;

return;

}

int sign = 1;

vector<vector<int> > temp(N, vector<int>(N));

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

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

getCofactor(a, temp, i, j, N);

sign = ((i+j)%2==0)? 1: -1;

adj[j][i] = (sign)\*(determinant(temp, N-1 , N));

} }

}

bool inverse(vector<vector<int> > &a, vector<vector<int> > &inv, int N){

int det = determinant(a, N, N);

if(det == 0){

cout << "Inverse does not exist";

return false; }

int invDet = modInverse(det,26);

cout<<det%26<<' '<<invDet<<'\n';

vector<vector<int> > adj(N, vector<int>(N));

adjoint(a, adj, N);

for(int i=0;i<N;i++)

for(int j=0;j<N;j++)

inv[i][j] = (adj[i][j]\*invDet)%26;

return true;

}

int main(){

cout<<"Hill Cipher Decryption \n";

int x,y,i,j,k,n;

cout<<"Enter the size of key matrix\n";

cin>>n;

cout<<"Enter the key matrix\n";

vector<vector<int> > a(n, vector<int>(n));

vector<vector<int> > adj(n, vector<int>(n));

vector<vector<int> > inv(n, vector<int>(n));

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

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

cin>>a[i][j];

}

}

if(inverse(a,inv,n))

cout<<"Inverse exist\n";

cout<<"Enter the message to decrypt\n";

string s;

cin>>s;

k=0;

string ans;

while(k<s.size()){

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

int sum = 0;

int temp = k;

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

sum += ((inv[i][j] + 26)%26\*(s[temp++]-'a')%26)%26;

sum = sum%26;

}

ans+=(sum+'a');

} k+=n;

}

int f=ans.size()-1;

while(ans[f]=='x')

f--;

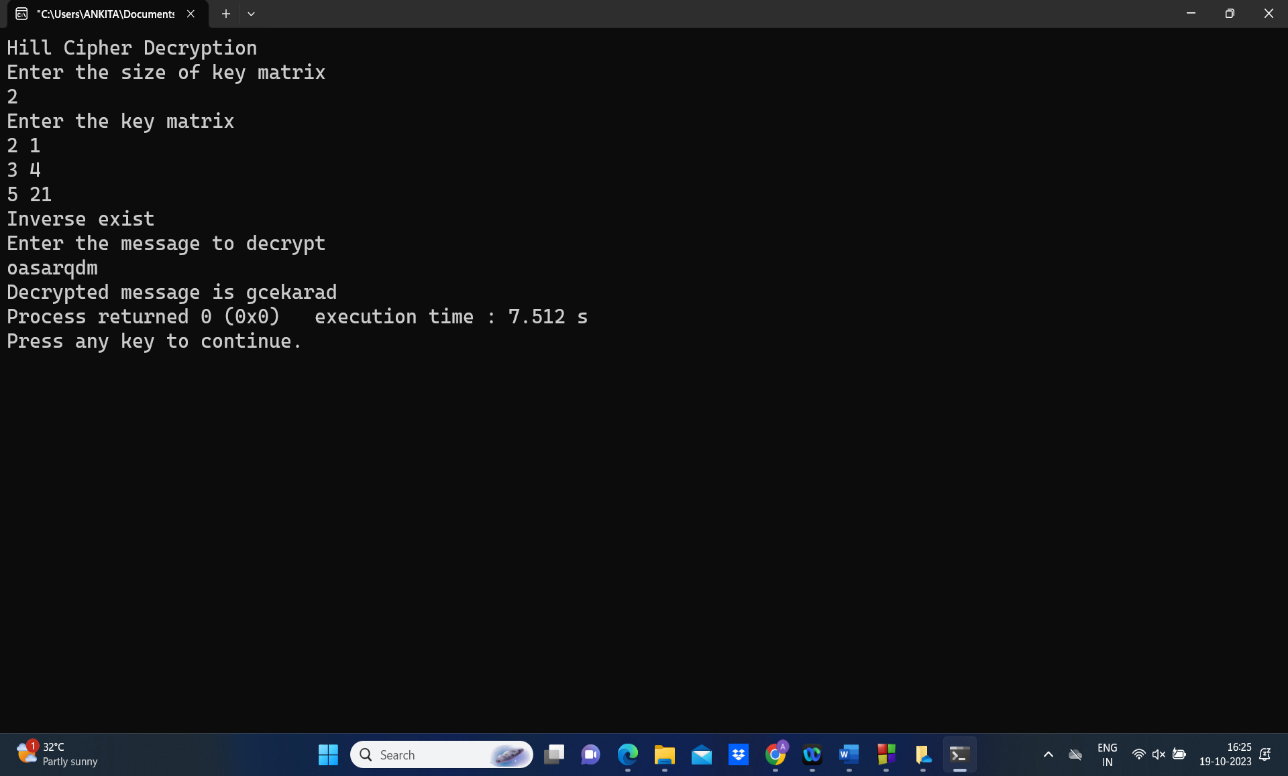
cout<<"Decrypted message is ";

for(i=0;i<=f;i++){ cout<<ans[i]; }

return 0;

}

**Output-**



1. Vigenere –
2. Encryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int main(){

cout<<"Vigenere Cipher Encryption\n";

int i,j,k,n;

vector<vector<char> > a(26,vector<char>(26));

k=0;

n=26;

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

k=i;

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

a[i][j]='A'+k;

k++;

if(k==26)

k=0;

}

}

cout<<"Enter the message\n";

string s;

cin>>s;

cout<<"Enter the key\n";

string key;

cin>>key;

k=0;

int mod = key.size();

for(i=key.size();i<s.size();i++){

key+=key[k%mod];

k++;

}

string encrypt;

for(i=0;i<s.size();i++){

encrypt+= a[s[i]-'A'][key[i]-'A'];

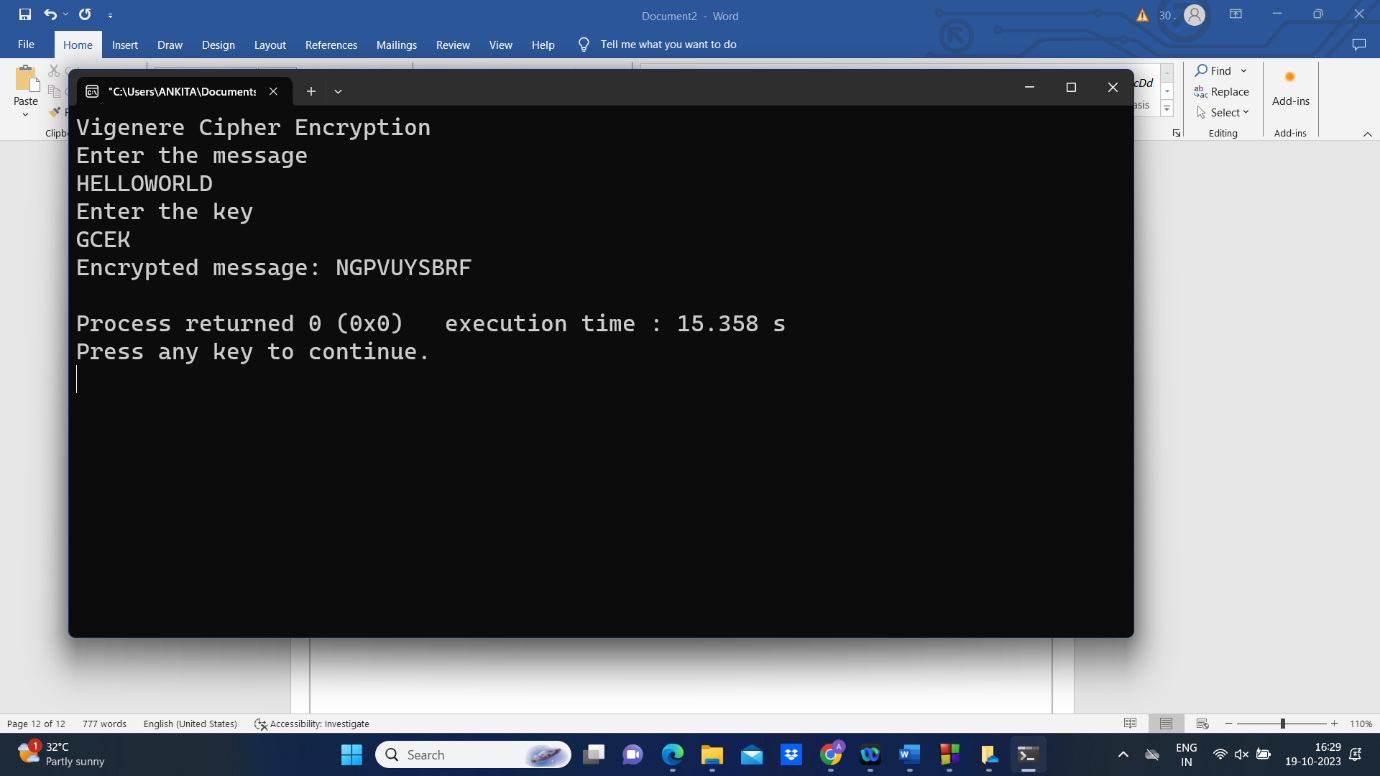
}

cout<<"Encrypted message: "<<encrypt<<'\n';

return 0;

}

**Output-**



1. Decryption-

**Code-**

#include<bits/stdc++.h>

using namespace std;

int main(){

cout<<"Vigenere Cipher Decryption\n";

int i,j,k,n;

vector<vector<char> > a(26,vector<char>(26));

k=0; n=26;

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

k=i;

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

a[i][j]='A'+k;

k++;

if(k==26) k=0;

} }

cout<<"Enter the encrypted message\n";

string s;

cin>>s;

cout<<"Enter the key\n";

string key;

cin>>key;

k=0;

for(i=key.size();i<s.size();i++){

key+=key[k]; k++;

}

string decrypt;

for(i=0;i<s.size();i++){

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

if(a[j][key[i]-'A']==s[i]){

decrypt += 'A'+j;

break;

} } }

cout<<"Decrypted message: "<<decrypt<<'\n';

return 0;

}

for(i=0;i<s.size();i++){

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

if(a[j][key[i]-'A']==s[i]){

decrypt += 'A'+j;

break;

} } }

cout<<"Decrypted message: "<<decrypt<<'\n';

return 0;

}

**Output-**

