

Final Year B. Tech. (CSE) – I: 2022-23

4CS451: Cryptography and Network Security Lab

Assignment No. 1

PRN: 2019BTECS00077

Batch: B7

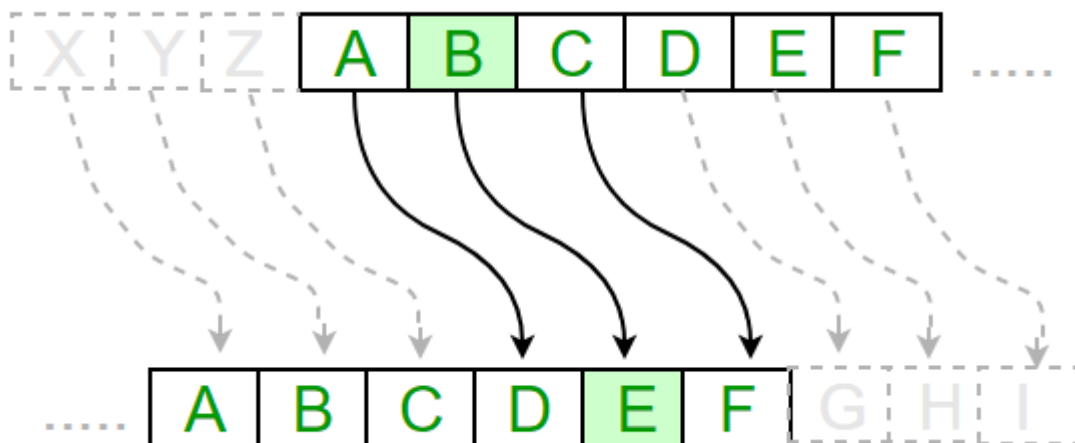
Full name: Biradar Avinash Vishnu

Title: Implementation of Caser Cipher algorithm.

Objective: write a program to encrypt the plain text and decrypt the cipher text using Caser cipher algorithm.

Introduction & Theory:

- The Caesar Cipher technique is one of the earliest and simplest methods of encryption technique.
- It's simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter with a fixed number of positions down the alphabet. For example, with a shift of 1, A would be replaced by B, B would become C, and so on..



Code:

```
// Implementation of Ceasor Cypher
#include<iostream>
using namespace std;
```

```

string encrypt(string plainText,int pos){

    int n=plainText.size();
    for(int i=0;i<n;i++){
        if(plainText[i]==32)
            continue;
        if(plainText[i]>96 && plainText[i]<=122)
            plainText[i]-=32;
        plainText[i]=(plainText[i]-'A'+pos)%26+'A';
    }
    cout<<"\nCypher Text :"<<plainText<<endl;
    return plainText;
}

void decrypt(string cypherText,int pos){
    int n=cypherText.size();
    for(int i=0;i<n;i++){
        if(cypherText[i]==32)
            continue;
        cypherText[i]=(cypherText[i]-'A'-pos+26)%26+'A';
    }
    cout<<"\nPlain Text :"<<cypherText<<endl;
}

int main(){
    int pos;
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);

    string plain_text;
    getline(cin,plain_text);

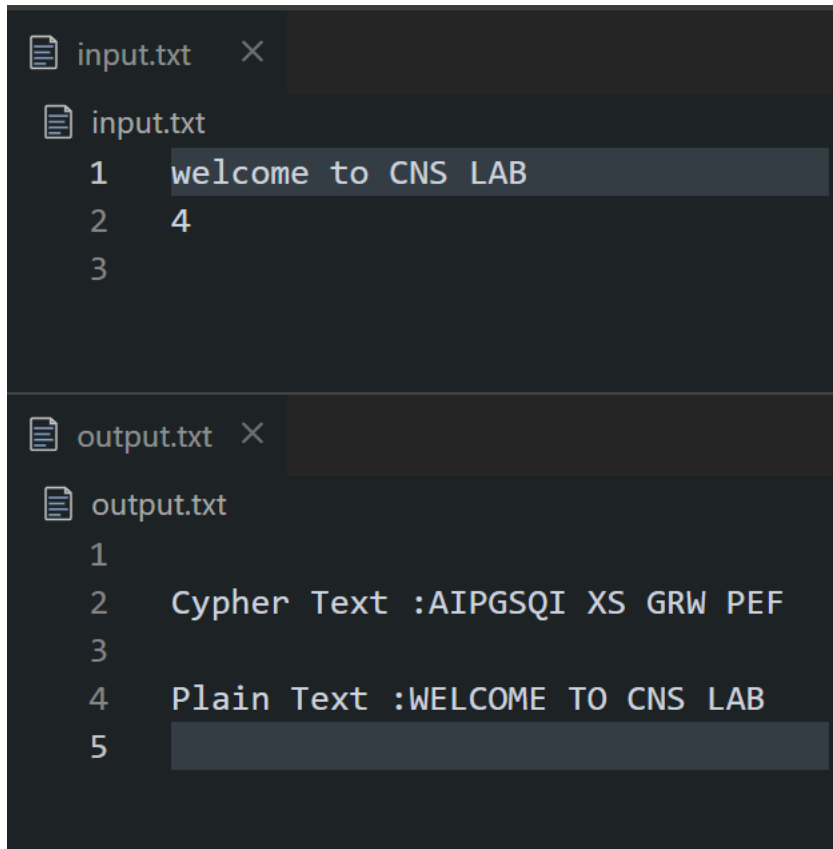
    // cout<<"Plain text:"<<plain_text<<endl;
    cin>>pos;

    string cypher_text=encrypt(plain_text,pos);
    decrypt(cypher_text,pos);
}

```

Result:

File input output:



The screenshot shows a text editor with two files open. The first file, 'input.txt', contains three lines: '1 welcome to CNS LAB', '2 4', and '3'. The second file, 'output.txt', contains five lines: '1', '2 Cypher Text :AIPGSQI XS GRW PEF', '3', '4 Plain Text :WELCOME TO CNS LAB', and '5' followed by a blank line.

```
input.txt  X
1 welcome to CNS LAB
2 4
3

output.txt  X
1
2 Cypher Text :AIPGSQI XS GRW PEF
3
4 Plain Text :WELCOME TO CNS LAB
5
```

Console input output:

```
PS E:\College\Final year\C&NS\practical> cd "e:\College\Final year\C&N
.\caesar }
plain text is encrypting using ceasor cipher algorithm
27

Cypher Text :QMBJO UFYU JT FODSZQUJOH VTJOH DFBTPS DJQIFS BMHPSJWIN

Plain Text :PLAIN TEXT IS ENCRYPTING USING CEASOR CIPHER ALGORITHM
```

Final Year B. Tech. (CSE) – I: 2022-23

4CS451: Cryptography and Network Security Lab

Assignment No. 2

PRN: 2019BTECS00077

Batch: B7

Full name: Biradar Avinash Vishnu

Title: Cryptanalysis of Caesar cipher algorithm.

Objective: Crack the given code and output the corresponding plain text.

Introduction & Theory:

In this we have write a code to crack the given Cipher Text where key is not provided and produce Output as plain Text.

Steps I have done:

- 1) I have taken a text file of 10000 English common words.
- 2) Produces the 25 possibilities and calculated the associated score of each possibility.
- 3) The max score possibility plain text is our expected ans.

Code:

```
#include<bits/stdc++.h>
using namespace std;

unordered_set<string> uset;

// Capitalize the string
void capitalize(string &str){
    for(char &c:str){
        if(c>=97 && c<=122)
            c-=32;
    }
}

int score(string text){
```

```

string word="";
int score=0;
for(char &c:text){
    if(c==32){
        if(uset.find(word)!=uset.end())
            score++;
        word.clear();
        continue;
    }
    word.push_back(c);
}
if(uset.find(word)!=uset.end())
    score++;
return score;
}

void decryptCrackit(string &cypherText){

    string ans="";
    int maxScore=0;
    for(int i=0;i<26;i++){
        string tmp=cypherText;
        for(char &c:tmp){
            if(c==32)
                continue;
            c=((c-'A'-i+26)%26+'A');
        }
        if(score(tmp) > maxScore){
            maxScore=score(tmp);
            ans=tmp;
        }
        cout<<i<<": "<<tmp<<" "<<score(tmp)<<"\n";
    }

    cout<<"\n\nFinal output of crack the code:\n"<<ans<<endl;
}

string encrypt(string plainText,int pos){

    int n=plainText.size();
    for(int i=0;i<n;i++){
        if(plainText[i]==32)
            continue;
        if(plainText[i]>96 && plainText[i]<=122)
            plainText[i]-=32;
        plainText[i]+=pos;
    }
}

```

```

        if(plainText[i]>90)
            plainText[i]=(plainText[i]%90+64);
    }
    cout<<"\nCypher Text :\n"<<plainText<<"\n\n";
    return plainText;
}

int main(){

    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);

    string plainText;
    getline(cin,plainText);
    capitalize(plainText);
    int pos;
    cin>>pos;

    string cypherText=encrypt(plainText,pos);

    ifstream file;
    file.open ("words.txt");

    string word;
    while (file >> word){
        capitalize(word);
        uset.insert(word);
    }
    decryptCrackit(cypherText);

    return 0;
}

```

Result:

File input output:

```
input.txt ×
input.txt
1  is this algorithm is correct
2  28

output.txt ×
output.txt
1
2  Cypher Text :
3  KU VJKU CNIQTKVJO KU EQTTGEV
4
5  0: KU VJKU CNIQTKVJO KU EQTTGEV 0
6  1: JT UIJT BMHPSJUIN JT DPSSFUDU 0
7  2: IS THIS ALGORITHM IS CORRECT 5
8  3: HR SGHR ZKFNQHSGL HR BNQQDBS 2
9  4: GQ RFGQ YJEMPGRFK GQ AMPPCAR 0
10 5: FP QEFP XIDLOFQEJ FP ZLOOBZQ 2
11 6: EO PDEO WHCKNEPDI EO YKNNAYP 0
12 7: DN OCDN VGBJMDOCH DN XJMMZXO 0

13 8: CM NBCM UFAILCNBG CM WILLYWN 2
14 9: BL MABL TEZHKBMAF BL VHKKXVM 2
15 10: AK LZAK SDYGJALZE AK UGJJWUL 2
16 11: ZJ KYZJ RCXFIZKYD ZJ TFIIIVTK 0
17 12: YI JXYI QBWEHYJXC YI SEHHUSJ 0
18 13: XH IWXH PAVDGXIWB XH RDGGTRI 0
19 14: WG HVWG OZUCFWHVA WG QCFFSQH 0
20 15: VF GUVF NYTBEVGUZ VF PBEERPG 0
21 16: UE FTUE MXSADUFTY UE OADDQOF 0
22 17: TD ESTD LWRZCTESX TD NZCCPNE 2
23 18: SC DRSC KVQYBSDRW SC MYBBOMD 2
24 19: RB CQRB JUPXARCQV RB LXAANLC 2
25 20: QA BPQA ITOWZQBPU QA KWZZMKB 0
26 21: PZ AOPZ HSNVYPAOT PZ JVVYLJA 0
```

```

27 22: OY ZNOY GRMUXOZNS OY IUXXKIZ 0
28 23: NX YMNX FQLTWNMYR NX HTWWJHY 0
29 24: MW XLMW EPKSVMXLQ MW GSVVIGX 2
30 25: LV WKLV DOJRULWKP LV FRUUHFW 0
31
32
33 Final output of crack the code:
34 IS THIS ALGORITHM IS CORRECT
35

```

Console Input Output:

```

PS E:\College\Final year\C&NS\practical> cd "e:\College\Final year\C&NS\practical"
} ; if ($?) { .\Cryptanalysis }
Department of CSE walchand college of engineering sangli
7

Cypher Text :
KLWHYATLUA VM JZL DHSJOHUK JVSSLNL VM LUNPULLYPUN ZHUNSP

0: KLWHYATLUA VM JZL DHSJOHUK JVSSLNL VM LUNPULLYPUN ZHUNSP 0
1: JKVGXZSKTZ UL IYK CGRINGTJ IURRKMK UL KTMOTKKXOTM YGTMRO 2
2: IJUFWYRJSY TK HXJ BFQHMFSI HTQQJLJ TK JSLNSJJWNSL XFSLQN 0
3: HITEVXQIRX SJ GWI AEPGLERH GSPPIKI SJ IRKMRIIVMRK WERKPM 0
4: GHSDUWPHQW RI FVH ZDOFKDQG FROOHJH RI HQJLQHHULQJ VDQJOL 2
5: FGRCTVOGPV QH EUG YCNEJCPF EQNNGIG QH GPIKPGGTKPI UCPINK 0
6: EFQBSUNFOU PG DTF XBMDIBOE DPMMFHF PG FOHJOFFSJOH TBOHMJ 2
7: DEPARTMENT OF CSE WALCHAND COLLEGE OF ENGINEERING SANGLI 5
8: CDOZQSLDMS NE BRD VZKBGZMC BNKKDFD NE DMFHMDDQHMF RZMFKH 2
9: BCNYPRKCLR MD AQC UYJAFYLB AMJJCEC MD CLEGLCCPGLE QYLEJG 2
10: ABMXOQJBKQ LC ZPB TXIZEXKA ZLIIBDB LC BKDFKBBOFKD PXKDIF 2

```



```
11: ZALWNPIAJP KB YOA SWHYDWJZ YKHHACA KB AJCEJAANEJC OWJCHE 2
12: YZKVMOHZIO JA XNZ RVGXCVIY XJGGZBZ JA ZIBDIZZMDIB NVIBGD 2
13: XYJULNGYHN IZ WMY QUFWBUHX WIFFYAY IZ YHACHYYLCHA MUHAFC 0
14: WXITKMFxGM HY VLX PTEVATGW VHEEXZX HY XGZBGXXKBGZ LTGZEB 0
15: VWHSJLEWFL GX UKW OSDUZSFV UGDDWYW GX WFYAFWWJAFY KSFYDA 0
16: UVGRIKDVEK FW TJV NRCTYREU TFCCVXV FW VEXZEVVIZEX JREXCZ 2
17: TUFQHJCUDJ EV SIU MQBSXQDT SEBBUWU EV UDWYDUUHYDW IQDWBY 2
18: STEPGIBTCI DU RHT LPARWPCS RDAATVT DU TCVXCTTGXCV HPCVAX 2
19: RSDOFHASBH CT QGS KOZQVOBR QCZZSUS CT SBUWBSSFWBU GOBUZW 2
20: QRCNEGZRAG BS PFR JNYPUNAQ PBYYRTR BS RATVARREVAT FNATYV 2
21: PQBMDFYQZF AR OEQ IMXOTMZP OAXXQSQ AR QZSUZQQDUZS EMZSXU 2
22: OPALCEXPYE ZQ NDP HLWNSLYO NZWWPRP ZQ PYRTYPPCTYR DLYRWT 0
23: NOZKBDWOXD YP MCO GKVMRKXN MYVVOQO YP OXQX00BSXQ CKXQVS 0
24: MNYJACVNWC XO LBN FJULQJWM LXUUNPN XO NWPRWNNARWP BJWPUR 0
25: LMXIZBUMVB WN KAM EITKPIVL KWTTMOM WN MVOQVMMZQVO AIVOTQ 2
```

Final output of crack the code:

DEPARTMENT OF CSE WALCHAND COLLEGE OF ENGINEERING SANGLI

Final Year B. Tech. (CSE) – I: 2022-23

4CS451: Cryptography and Network Security Lab

Assignment No. 3

PRN: 2019BTECS00077

Batch: B7

Full name: Biradar Avinash Vishnu

Title: Implementation of Playfair cipher algorithm.

Objective: write a program to encrypt the plain text and decrypt the cipher text using Playfair cipher algorithm.

Introduction & Theory:

The Playfair Cipher Encryption Algorithm:

The Algorithm consists of 2 steps:

Generate the key matrix Square (5×5):

- The key square is a 5×5 grid of alphabets that acts as the key for encrypting the plaintext. Each of the 25 alphabets must be unique and one letter of the alphabet (usually J) is omitted from the table (as the table can hold only 25 alphabets). If the plaintext contains J, then it is replaced by I.
- The initial alphabets in the key square are the unique alphabets of the key in the order in which they appear followed by the remaining letters of the alphabet in order.
-

Encrypt the plain text:

The plaintext is split into pairs of two letters (digraphs). If there is an odd number of letters, a Z is added to the last letter.

Rules for Encryption:

- **If both the letters are in the same column:** Take the letter below each one (going back to the top if at the bottom).
- **If both the letters are in the same row:** Take the letter to the right of each one (going back to the leftmost if at the rightmost position).

- **If neither of the above rules is true:** Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.

Code:

```
#include<bits/stdc++.h>
using namespace std;
map<char,pair<int,int>> charPos;

// Capitalize the character
void capitalize(string &str){

    for(char &c:str){
        if(c>=97 && c<=122)
            c-=32;
    }
}

// add char pairs to the set
void addToSet(vector<pair<char,char>> &v,char a,char b){
    v.push_back({a,b});
}

// build matrix
void buildMatrix(vector<vector<char>> &mat,string key){

    vector<bool>vis(26,false);

    int i=0,j=0;
    for(int k=0;k<key.length();k++){

        if(!vis[key[k]-65]){
            mat[i][j]=key[k];
            if(key[k]=='I' || key[k]=='J'){
                vis['I'-65]=true;
                vis['J'-65]=true;
            }
            else vis[key[k]-65]=true;
            j++;
            if(j==5){
                j%=5;
                i++;
            }
        }
    }
}
```

```

    }
    for(int k=0;k<26;k++){

        if(!vis[k]){
            if(k+'A'=='I' || k+'A'=='J'){
                vis['I'-65]=true;
                vis['J'-65]=true;
            }
            mat[i][j]=k+'A';
            j++;
            if(j==5){
                j%=5;
                i++;
            }
        }

    }

    // cout<<"\n \nMatrix:\n";
    for(int i=0;i<5;i++){
        for(int j=0;j<5;j++){
            cout<<mat[i][j]<<" ";
        }
        cout<<"\n";
    }
}

void encrypt(vector<vector<char>> &mat,vector<pair<char,char>>
&pairSet){

    // storing position of chars in the matrix
    for(int i=0;i<5;i++){
        for(int j=0;j<5;j++){
            if(mat[i][j]=='I' || mat[i][j]=='J')
                charPos['J']=charPos['I']=make_pair(i,j);
            else
                charPos[mat[i][j]]=make_pair(i,j);
        }

    }

    // print
    cout<<"\n";
    // cout<<"Plain Text:";
    for(auto it:pairSet)
        cout<<it.first<<it.second<<" ";
    cout<<endl;

    // cout<<"Cypher Text: ";

```

```

for(auto &it:pairSet){
    int i1=charPos[it.first].first;
    int j1=charPos[it.first].second;
    int i2=charPos[it.second].first;
    int j2=charPos[it.second].second;

    if(i1==i2){
        it.first=mat[i1][(j1+1)%5];
        it.second=mat[i1][(j2+1)%5];
        cout<<mat[i1][(j1+1)%5]<<mat[i1][(j2+1)%5]<<" ";
    }
    else if(j1==j2){
        it.first=mat[(i1+1)%5][j1];
        it.second=mat[(i2+1)%5][j2];
        cout<<mat[(i1+1)%5][j1]<<mat[(i2+1)%5][j2]<<" ";
    }
    else{
        it.first=mat[i1][j2];
        it.second=mat[i2][j1];
        cout<<mat[i1][j2]<<mat[i2][j1]<<" ";
    }

}
cout<<"\n\n";
}

void decrypt(vector<vector<char>> &mat,vector<pair<char,char>>
&pairSet){

    // print
    // for(auto it:charPos)
    //     cout<<it.first<<" "<<it.second.first<<"
"<<it.second.second<<endl;
    // cout<<endl;

    // cout<<"Plain Text: ";
    for(auto it:pairSet){
        int i1=charPos[it.first].first;
        int j1=charPos[it.first].second;
        int i2=charPos[it.second].first;
        int j2=charPos[it.second].second;

        //     cout<<i1<<j1<<" "<<i2<<j2<<endl;

        if(i1==i2){
            cout<<mat[i1][((j1-1)+5)%5]<<mat[i1][((j2-1)+5)%5]<<" ";

```

```

    }
    else if(j1==j2){
        cout<<mat[((i1-1)+5)%5][j1]<<mat[((i2-1)+5)%5][j2]<<" ";
    }
    else{
        cout<<mat[i1][j2]<<mat[i2][j1]<<" ";
    }

}
}

int main(){

    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);

    string tmp,key,plainText;
    vector<pair<char,char>> pairSet;

    getline(cin,tmp);

    for(char c:tmp){
        if(c!=32)
            plainText.push_back(c);
    }

    // cout<<plainText<<endl;
    capitalize(plainText);

    getline(cin,key);

    capitalize(key);

    // cout<<key<<endl;

    int n=plainText.size();
    for(int i=0;i<n;){
        char a=plainText[i];
        if(i==n-1 || plainText[i+1]==a){
            addToSet(pairSet,a,'X'); // added if x as dummy for same
chars
            i++;
        }
        else{
            addToSet(pairSet,a,plainText[i+1]);

```

```

        i+=2;
    }
}
// breaking the plain text into 2chars
// for(auto it:pairSet)
//     cout<<it.first<<it.second<<" ";

// build matrix
vector<vector<char>> mat(5,vector<char>(5));
buildMatrix(mat,key);

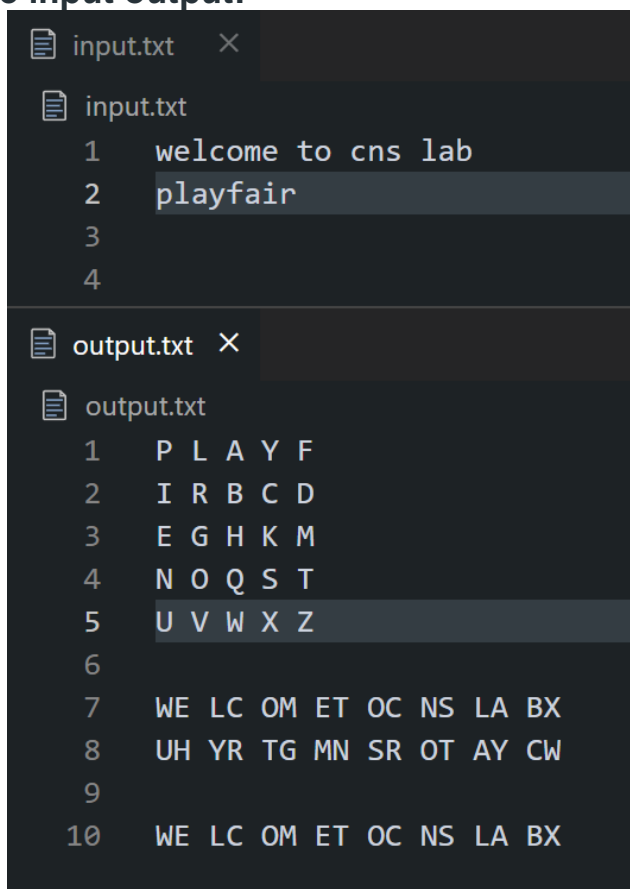
// Encrypt the text
// cout<<"Encryption\n";
encrypt(mat,pairSet);

// Decrypt the Cypher Text
// cout<<"Decryption\n";
decrypt(mat,pairSet);
}

```

Result:

File input output:



The screenshot shows a code editor with two files: `input.txt` and `output.txt`.

input.txt

```

1 welcome to cns lab
2 playfair
3
4

```

output.txt

```

1 P L A Y F
2 I R B C D
3 E G H K M
4 N O Q S T
5 U V W X Z
6
7 WE LC OM ET OC NS LA BX
8 UH YR TG MN SR OT AY CW
9
10 WE LC OM ET OC NS LA BX

```

Console Input Output:

```
PS E:\College\Final year\C&NS\practical> cd "e:\College\Final year\  
) { .\playfair }  
CnS lab playFair algorithm implementation  
walchand  
W A L C H  
N D B E F  
G I K M O  
P Q R S T  
U V X Y Z  
  
CN SL AB PL AY FA IR AL GO RI TH MI MP LE ME NT AT IO NX  
WE RC LD RW CV DH KQ LC IG QK ZF OK GS CB SM FP HQ KG BU  
  
CN SL AB PL AY FA IR AL GO RI TH MI MP LE ME NT AT IO NX
```


Final Year B. Tech. (CSE) – I: 2022-23

4CS451: Cryptography and Network Security Lab

Assignment No. 4

PRN: 2019BTECS00077

Batch: B7

Full name: Biradar Avinash Vishnu

Title: Implementation of Vigenère cipher algorithm.

Objective: write a program to encrypt the plain text and decrypt the cipher text using Vigenère cipher algorithm.

Introduction & Theory:

- Vigenère Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets
- To generate a new key, the given key is repeated in a circular manner, as long as the length of the plain text does not equal to the new key.

J	A	V	A	T	P	O	I	N	T
B	E	S	T	B	E	S	T	B	E

Encryption

- The first letter of the plaintext is combined with the first letter of the key. The column of plain text "J" and row of key "B" intersects the alphabet of "K" in the Vigenère table, so the first letter of ciphertext is "K".
- Similarly, the second letter of the plaintext is combined with the second letter of the key. The column of plain text "A" and row of key "E" intersects the alphabet of "E" in the Vigenère table, so the second letter of ciphertext is "E".
- This process continues continuously until the plaintext is finished.

Ciphertext = KENTUTGBOX

Code:

```
#include<bits/stdc++.h>
using namespace std;

// Capitalize the character
void capitalize(string &str){
    for(char &c:str){
        if(c>=97 && c<=122)
            c-=32;
    }
}

string encrypt(string &plainText,string &key){
    int n=key.size();
    int i=0;
    for(char &c:plainText){
        if(c>=65 && c<=90){
            int a=c-65;
            int b=key[i%n]-65;
            c=((a+b)%26+65);
            i++;
        }
    }
    return plainText;
}

string decrypt(string &cypherText,string &key){

    int n=key.size();
    int i=0;
    for(char &c:cypherText){
        if(c>=65 && c<=90){
            int a=c-65;
            int b=key[i%n]-65;
            c=(a-b+26)%26+65;
            i++;
        }
    }
    return cypherText;
}

int main(){

    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
```

```

string key,plainText;
getline(cin,plainText);

// cout<<plainText<<endl;
capitalize(plainText);
getline(cin,key);
capitalize(key);

string CypherText=encrypt(plainText,key);
cout<<"Cypher Text:"<<CypherText<<"\n\n";

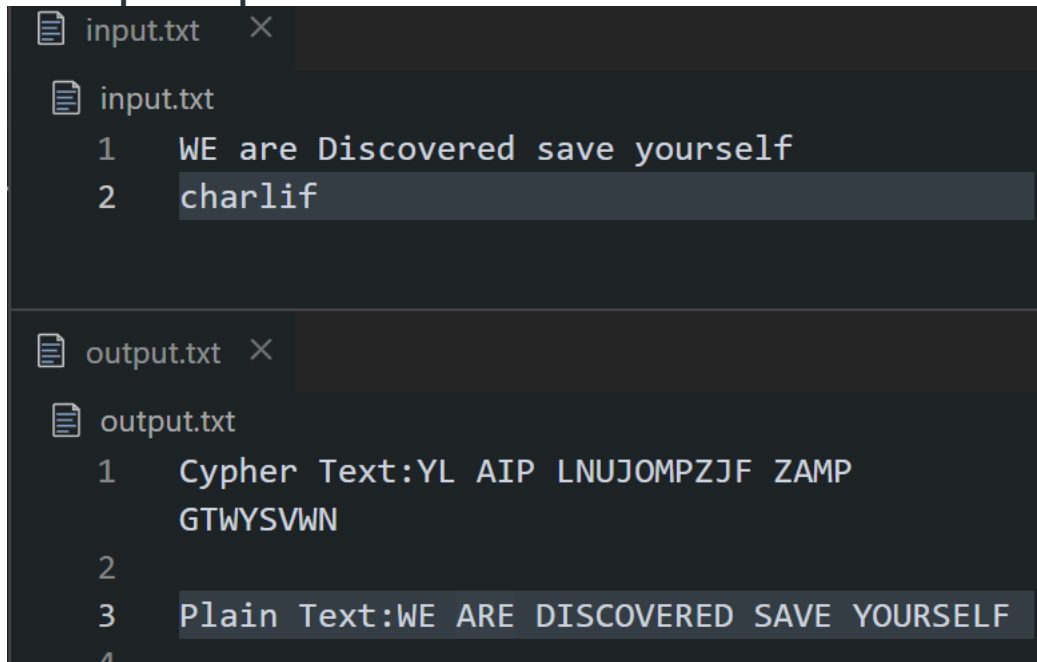
plainText=decrypt(CypherText,key);

cout<<"Plain Text:"<<plainText<<endl;
return 0;
}

```

Result:

File input output:



```

input.txt
1 WE are Discovered save yourself
2 charlif

output.txt
1 Cypher Text:YL AIP LNUJOMPZJF ZAMP
  GTWYSVWN
2
3 Plain Text:WE ARE DISCOVERED SAVE YOURSELF
4

```

Console Input Output:

```
PS E:\College\Final year\C&NS\practical>  
) { .\vigenere }  
she is Listening  
pascal  
Cypher Text:HHW KS WXSLGNTCG  
  
Plain Text:SHE IS LISTENING
```

Final Year B. Tech. (CSE) – I: 2022-23

4CS451: Cryptography and Network Security Lab

Assignment No. 5

PRN: 2019BTECS00077

Batch: B7

Full name: Biradar Avinash Vishnu

Title: Implementation of Rail Fence and Columnar Transposition cipher algorithms.

Objective: write a program to encrypt the plain text and decrypt the cipher text using Rail Fence and Columnar Transposition cipher algorithms.

Introduction & Theory:

In a transposition cipher, the order of the alphabets is re-arranged to obtain the cipher-text.

Rail Fence Cipher:

- the plain-text is written downwards and diagonally on successive rails of an imaginary fence.
- When we reach the bottom rail, we traverse upwards moving diagonally, after reaching the top rail, the direction is changed again. Thus, the alphabets of the message are written in a zig-zag manner.
- After each alphabet has been written, the individual rows are combined to obtain the cipher-text.

For example, if the message is “GeeksforGeeks” and the number of rails = 3 then cipher is prepared as:

G			S			G			S	
	E		K		F		R		E	K
		E				O			E	

Columnar Transposition:

Columnar Transposition involves writing the plaintext out in rows, and then reading the ciphertext off in columns one by one

RailFence Code:

```
#include<bits/stdc++.h>
using namespace std;

// Capitalize the string
void capitalize(string &str){
    for(char &c:str){
        if(c>=97 && c<=122)
            c-=32;
    }
}

string encrypt(string &plaintext,int depth){

    int n=plaintext.size();
    vector<vector<char>> mat(depth,vector<char>(n,'*'));

    bool dirDown= false;
    int row = 0, col = 0;

    for (int i=0; i<n; i++)
    {
        if (row == 0 || row == depth-1)
            dirDown = !dirDown;
        mat[row][col++] = plaintext[i];
        dirDown?row++ : row--;
    }

    string ans;
    for (int i=0; i < depth; i++)
        for (int j=0; j <n; j++)
            if (mat[i][j]!='*')
                ans.push_back(mat[i][j]);
    return ans;
}

string decrypt(string &cypherText,int depth){

    int n=cypherText.size();
    vector<vector<char>> mat(depth,vector<char>(n,'#'));
```

```

bool dirDown=false;

int row=0, col=0;
for (int i=0;i<n; i++)
{
    if (row == 0 || row==depth-1)
        dirDown=!dirDown;
    mat[row][col++] = '*';
    dirDown?row++ : row--;
}

int index = 0;
for (int i=0; i<depth; i++){
    for (int j=0; j<n; j++)
        if (mat[i][j] == '*' && index<n)
            mat[i][j] =cypherText[index++];
}

string ans;
row = 0, col = 0;
for (int i=0; i<n; i++)
{
    if (row == 0)
        dirDown = true;
    if (row == depth-1)
        dirDown= false;

    if (mat[row][col] != '*')
        ans.push_back(mat[row][col++]);

    dirDown?row++: row--;
}
return ans;
}

int main(){

    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);

    string plainText;
    getline(cin,plainText);
    // cout<<plainText<<endl;
    capitalize(plainText);

```

```

int depth;
cin>>depth;

string cypherText=encrypt(plainText,depth);

cout<<"Cypher Text:\n"<<cypherText<<"\n\n";

plainText=decrypt(cypherText,depth);

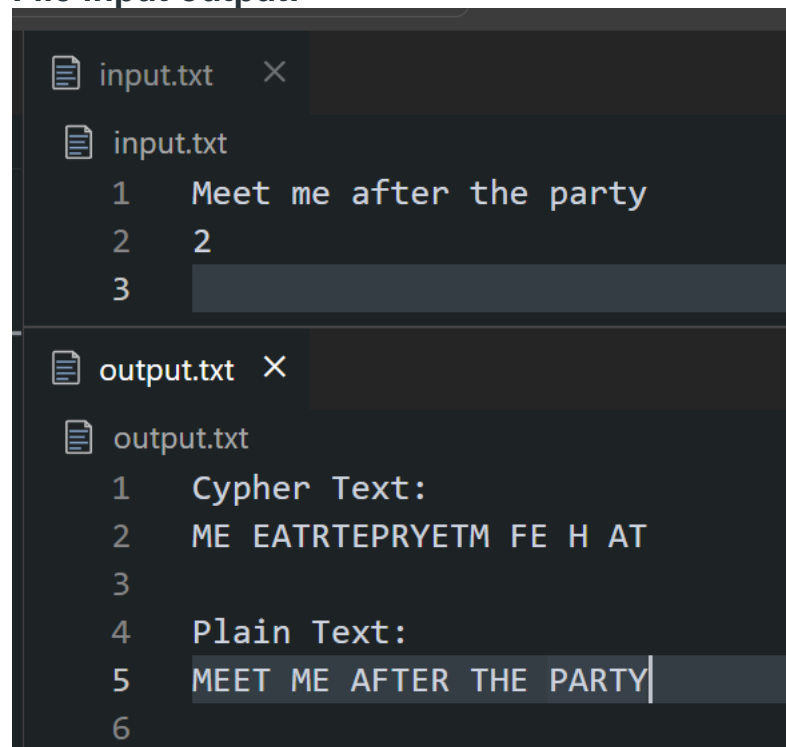
cout<<"Plain Text:\n"<<plainText<<endl;

return 0;
}

```

RailFence Result:

File input output:



The screenshot shows a file input/output window with two files: input.txt and output.txt. The input file contains three lines of text, and the output file contains the encrypted and decrypted results.

File	Line	Content
input.txt	1	Meet me after the party
	2	2
	3	
output.txt	1	Cypher Text:
	2	ME EATRTEPRYETM FE H AT
	3	
	4	Plain Text:
	5	MEET ME AFTER THE PARTY
	6	

Console Input Output:

```
PS E:\College\Final year\C&NS\practico
ilfence } ; if ($?) { .\Transpo_Railf
this is rail fence cipher
5
Cypher Text:
TRCRH ANEE ISIE HSILFCP I

Plain Text:
THIS IS RAIL FENCE CIPHER
```

Columnar Code:

```
#include<bits/stdc++.h>
using namespace std;

// Capitalize the string
void capitalize(string &str){
    for(char &c:str){
        if(c>=97 && c<=122)
            c-=32;
    }
}

string encrypt(int n,int m,string &plaintext,vector<int> &key){

    vector<vector<char>> mat(n,vector<char>(m));

    int k=0;
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(k>=plaintext.size())
                mat[i][j]=32;
            else
                mat[i][j]=plaintext[k++];
        }
    }

    // Matrix
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
```

```

        if(mat[i][j]!=32)
            cout<<mat[i][j]<<" ";
    }
    cout<<endl;
}

string cypherText="";
for(int i=0;i<m;i++){
    for(int j=0;j<n;j++){
        if(mat[j][key[i]-1]!=32)
            cypherText.push_back(mat[j][key[i]-1]);
    }
}
return cypherText;
}

string decrypt(int m,string &cypherText,vector<int> key){

    int n=(cypherText.size()/m)+1;

    vector<vector<char>> mat(n,vector<char>(m));

    int k=0;
    string plainText="";
    for(int i=0;i<m;i++){
        for(int j=0;j<n;j++){
            if(j==n-1 && key[i]-1>=(cypherText.size()%m))
                mat[j][key[i]-1]=32;
            else
                mat[j][key[i]-1]=cypherText[k++];
        }
    }

    // Matrix
    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(mat[i][j]!=32)
                cout<<mat[i][j]<<" ";
        }
        cout<<endl;
    }

    for(int i=0;i<n;i++){
        for(int j=0;j<m;j++){
            if(mat[i][j]!=32)
                plainText.push_back(mat[i][j]);
        }
    }
}

```

```

    }
}

return plainText;
}

int main(){

    // freopen("input.txt", "r", stdin);
    // freopen("output.txt", "w", stdout);

    string plainText;
    getline(cin,plainText);

    int m;
    cin>>m;

    vector<int> key(m);

    for(int i=0;i<m;i++)
        cin>>key[i];

    capitalize(plainText);
    string tmp="";

    for(char c:plainText){
        if(c!=32)
            tmp.push_back(c);
    }

    int n=(tmp.size()/m)+1;

    string Cyphertext=encrypt(n,m,tmp,key);

    cout<<"CypherText:"<<Cyphertext<<"\n\n";

    plainText=decrypt(m,Cyphertext,key);
    cout<<"plainText:"<<plainText<<"\n";

    return 0;
}

```

Columnar Result:

File input output:

```
input.txt X
input.txt
1 Meet Me after the party
2 6
3 2 5 3 1 6 4
4
output.txt X
output.txt
1 M E E T M E
2 A F T E R T
3 H E P A R T
4 Y
5 CypherText:EFEMRRETPMAHYETTTEA
6
7 M E E T M E
8 A F T E R T
9 H E P A R T
10 Y
11 plainText:MEETMEAFTERTHEPARTY
12
```

Console Input Output:

```
PS E:\College\Final year\C&NS\practical> cd "e:\College\
nar } ; if ($?) { .\Transo_columnar }
this is columnar encryption and decryption
9
6 4 1 9 8 3 7 5 2
T H I S I S C O L
U M N A R E N C R
Y P T I O N A N D
D E C R Y P T I O
N
CypherText:SENPSAIRTUYNLRDOOCNIINTCCNATIROYHMPE

T H I S I S C O L
U M N A R E N C R
Y P T I O N A N D
D E C R Y P T I O
N
plainText:THISISCOLUMNARENCRIPTIONANDDECRYPTION
```