**Cryptography And System Security**

**Experiment No. : 02**

**Aim : Design and Implementation of Play Fair Cipher.**

**Program :**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define SIZE 30

// Function to convert the string to lowercase

*void* toLowerCase(*char* *plain*[], *int* *ps*)

{

*int* i;

    for (i = 0; i < *ps*; i++) {

        if (*plain*[i] > 64 && *plain*[i] < 91)

*plain*[i] += 32;

    }

}

// Function to remove all spaces in a string

*int* removeSpaces(*char*\* *plain*, *int* *ps*)

{

*int* i, count = 0;

    for (i = 0; i < *ps*; i++)

        if (*plain*[i] != ' ')

*plain*[count++] = *plain*[i];

*plain*[count] = '\0';

    return count;

}

// Function to generate the 5x5 key square

*void* generateKeyTable(*char* *key*[], *int* *ks*, *char* *keyT*[5][5])

{

*int* i, j, k, flag = 0, \*dicty;

    // a 26 character hashmap

    // to store count of the alphabet

    dicty = (*int*\*)calloc(26, sizeof(*int*));

    for (i = 0; i < *ks*; i++) {

        if (*key*[i] != 'j')

            dicty[*key*[i] - 97] = 2;

    }

    dicty['j' - 97] = 1;

    i = 0;

    j = 0;

    for (k = 0; k < *ks*; k++) {

        if (dicty[*key*[k] - 97] == 2) {

            dicty[*key*[k] - 97] -= 1;

*keyT*[i][j] = *key*[k];

            j++;

            if (j == 5) {

                i++;

                j = 0;

            }

        }

    }

    for (k = 0; k < 26; k++) {

        if (dicty[k] == 0) {

*keyT*[i][j] = (*char*)(k + 97);

            j++;

            if (j == 5) {

                i++;

                j = 0;

            }

        }

    }

}

// Function to search for the characters of a digraph

// in the key square and return their position

*void* search(*char* *keyT*[5][5], *char* *a*, *char* *b*, *int* *arr*[])

{

*int* i, j;

    if (*a* == 'j')

*a* = 'i';

    else if (*b* == 'j')

*b* = 'i';

    for (i = 0; i < 5; i++) {

        for (j = 0; j < 5; j++) {

            if (*keyT*[i][j] == *a*) {

*arr*[0] = i;

*arr*[1] = j;

            }

            else if (*keyT*[i][j] == *b*) {

*arr*[2] = i;

*arr*[3] = j;

            }

        }

    }

}

// Function to find the modulus with 5

*int* mod5(*int* *a*) { return (*a* % 5); }

// Function to make the plain text length to be even

*int* prepare(*char* *str*[], *int* *ptrs*)

{

    if (*ptrs* % 2 != 0) {

*str*[*ptrs*++] = 'z';

*str*[*ptrs*] = '\0';

    }

    return *ptrs*;

}

// Function for performing the encryption

*void* encrypt(*char* *str*[], *char* *keyT*[5][5], *int* *ps*)

{

*int* i, a[4];

    for (i = 0; i < *ps*; i += 2) {

        search(*keyT*, *str*[i], *str*[i + 1], a);

        if (a[0] == a[2]) {

*str*[i] = *keyT*[a[0]][mod5(a[1] + 1)];

*str*[i + 1] = *keyT*[a[0]][mod5(a[3] + 1)];

        }

        else if (a[1] == a[3]) {

*str*[i] = *keyT*[mod5(a[0] + 1)][a[1]];

*str*[i + 1] = *keyT*[mod5(a[2] + 1)][a[1]];

        }

        else {

*str*[i] = *keyT*[a[0]][a[3]];

*str*[i + 1] = *keyT*[a[2]][a[1]];

        }

    }

}

// Function to encrypt using Playfair Cipher

*void* encryptByPlayfairCipher(*char* *str*[], *char* *key*[])

{

*char* ps, ks, keyT[5][5];

    // Key

    ks = strlen(*key*);

    ks = removeSpaces(*key*, ks);

    toLowerCase(*key*, ks);

    // Plaintext

    ps = strlen(*str*);

    toLowerCase(*str*, ps);

    ps = removeSpaces(*str*, ps);

    ps = prepare(*str*, ps);

    generateKeyTable(*key*, ks, keyT);

    encrypt(*str*, keyT, ps);

}

// Driver code

*int* main()

{

*char* str[SIZE], key[SIZE];

*int* i, j;

    // Plaintext to be encrypted

    printf("Enter a string : ");

    gets(str);

    for(i = 0,j = 0; i<strlen(str); i++){

        if(str[i]!=" "){

            str[j]=toupper(str[i]);

            j++;

        }

    }

    str[j] = '\0';

    printf("Entered string is : %s\n", str);

    // Key to be encrypted

    printf("Enter the key(Non repeated elements if possioble) : ");

    gets(key);

    for(i = 0,j = 0; i<strlen(str); i++){

        if(str[i]!=" "){

            str[j]=toupper(str[i]);

            j++;

        }

    }

    key[j]='\0';

    // encrypt using Playfair Cipher

    encryptByPlayfairCipher(str, key);

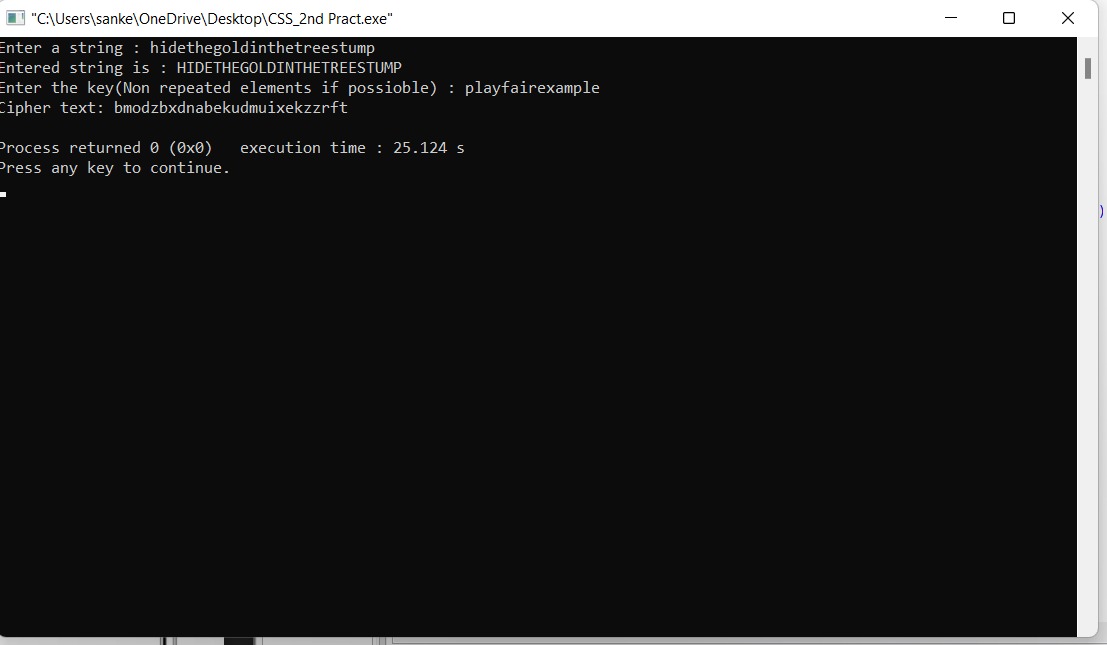
    printf("Cipher text: %s\n", toupper(str));

    return 0;

}

// This code is contributed by AbhayBhat

**Output :**

****

**Decryption :**

#include<stdio.h>

#include<string.h>

#include<ctype.h>

*int* removerepeated(*int* *size*,*int* *a*[]);

*int* insertelementat(*int* *position*,*int* *a*[],*int* *size*);

main()

{

*int* i,j,k,numstr[100],numcipher[100],numkey[100],lenkey,templen,tempkey[100],flag=-1,size,cipherkey[5][5],lennumstr,row1,row2,col1,col2;

*char* str[100],key[100];

 printf("Enter a string\n");

 gets(str);

 //converting entered string to Capital letters

 for(i=0,j=0;i<strlen(str);i++)

 {

  if(str[i]!=' ')

  {

   str[j]=toupper(str[i]);

   j++;

  }

 }

 str[j]='\0';

 printf("Entered String is %s\n",str);

 //Storing string in terms of ascii and to restore spaces I used -20

 size=strlen(str);

 for(i=0;i<size;i++)

 {

  if(str[i]!=' ')

  numstr[i]=str[i]-'A';

 }

 lennumstr=i;

 //Key processing

 printf("Enter the key (Non repeated elements if possible)\n");

 gets(key);

 //converting entered key to Capital letters

 for(i=0,j=0;i<strlen(key);i++)

 {

  if(key[i]!=' ')

  {

   key[j]=toupper(key[i]);

   j++;

  }

 }

 key[j]='\0';

 printf("%s\n",key);

 //Storing key in terms of ascii

 k=0;

 for(i=0;i<strlen(key)+26;i++)

 {

  if(i<strlen(key))

  {

   if(key[i]=='J')

   {

    flag=8;

    printf("%d",flag);

   }

       numkey[i]=key[i]-'A';

  }

  else

  {

    if(k!=9 && k!=flag)//Considering I=J and taking I in place of J except when J is there in key ignoring I

    {

            numkey[i]=k;

      }

      k++;

  }

 }

 templen=i;

 lenkey=removerepeated(templen,numkey);

 printf("Entered key converted according to Play Fair Cipher rule\n");

 for(i=0;i<lenkey;i++)

 {

     printf("%c",numkey[i]+'A');

 }

 printf("\n");

 //Arranging the key in 5x5 grid

 k=0;

 for(i=0;i<5;i++)

 {

  for(j=0;j<5;j++)

  {

   cipherkey[i][j]=numkey[k];

   k++;

  }

 }

 printf("Arranged key\n");

 for(i=0;i<5;i++)

 {

  for(j=0;j<5;j++)

  {

   printf("%c ",cipherkey[i][j]+'A');

  }

  printf("\n");

 }

   //Message Processing

   for(i=0;i<lennumstr;i+=2)

   {

      if(numstr[i]==numstr[i+1])

      {

       insertelementat(i+1,numstr,lennumstr);

       lennumstr++;

      }

   }

   if(lennumstr%2!=0)

   {

    insertelementat(lennumstr,numstr,lennumstr);

    lennumstr++;

   }

   printf("Entered String/Message After Processing according to Play fair cipher rule\n");

   for(i=0;i<lennumstr;i++)

   {

    printf("%c",numstr[i]+'A');

   }

   for(k=0;k<lennumstr;k+=2)

   {

    for(i=0;i<5;i++)

    {

     for(j=0;j<5;j++)

     {

      if(numstr[k]==cipherkey[i][j])

      {

         row1=i;

         col1=j;

      }

      if(numstr[k+1]==cipherkey[i][j])

      {

         row2=i;

         col2=j;

      }

     }

    }

    //Only change between Ecryption to decryption is changing + to -

    //If negative add 5 to that row or column

    if(row1==row2)

    {

     col1=(col1-1)%5;

     col2=(col2-1)%5;

     if(col1<0)

     {

      col1=5+col1;

     }

     if(col2<0)

     {

      col2=5+col2;

     }

     numcipher[k]=cipherkey[row1][col1];

     numcipher[k+1]=cipherkey[row2][col2];

    }

    if(col1==col2)

    {

     row1=(row1-1)%5;

     row2=(row2-1)%5;

      if(row1<0)

     {

      row1=5+row1;

     }

     if(row2<0)

     {

      row2=5+row2;

     }

     numcipher[k]=cipherkey[row1][col1];

     numcipher[k+1]=cipherkey[row2][col2];

    }

    if(row1!=row2&&col1!=col2)

    {

     numcipher[k]=cipherkey[row1][col2];

     numcipher[k+1]=cipherkey[row2][col1];

    }

   }

   printf("\nCipher Text is\n");

   for(i=0;i<lennumstr;i++)

   {

    if((numcipher[i]+'A')!='X')//Should remove extra 'X' which were created during Encryption

      printf("%c",numcipher[i]+'A');

   }

   printf("\n");

}

*int* removerepeated(*int* *size*,*int* *a*[])

{

*int* i,j,k;

 for(i=0;i<*size*;i++)

  {

 for(j=i+1;j<*size*;)

 {

    if(*a*[i]==*a*[j])

    {

     for(k=j;k<*size*;k++)

     {

*a*[k]=*a*[k+1];

     }

*size*--;

        }

    else

    {

      j++;

     }

 }

 }

return(*size*);

}

*int* insertelementat(*int* *position*,*int* *a*[],*int* *size*)

{

*int* i,insitem=23,temp[*size*+1];

    for(i=0;i<=*size*;i++)

        {

        if(i<*position*)

        {

            temp[i]=*a*[i];

        }

        if(i>*position*)

        {

         temp[i]=*a*[i-1];

        }

        if(i==*position*)

        {

            temp[i]=insitem;

        }

        }

        for(i=0;i<=*size*;i++)

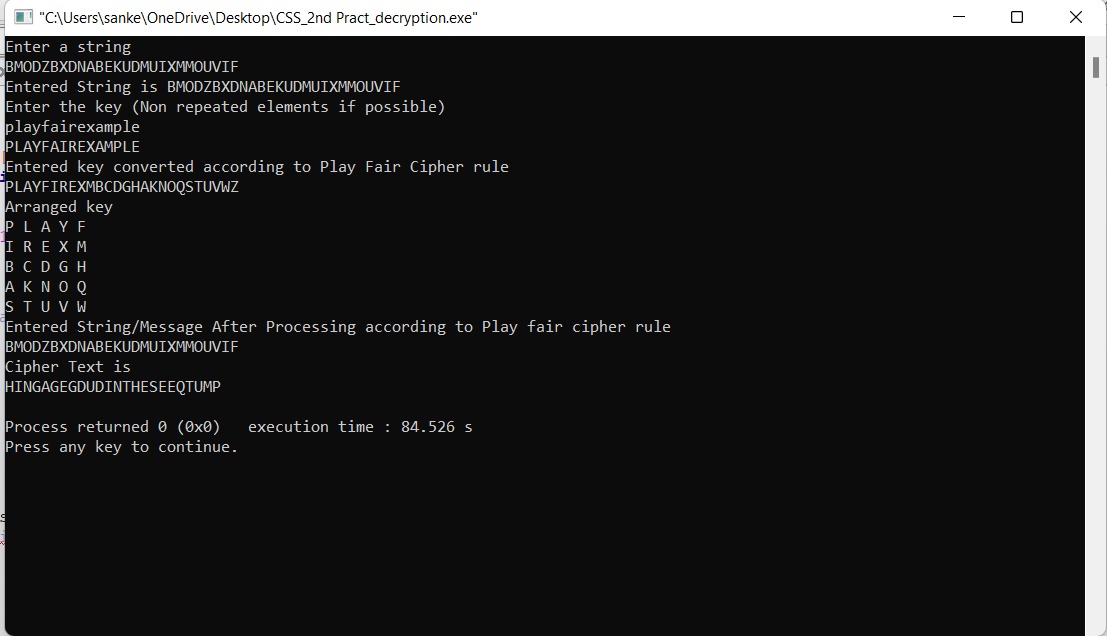
        {

*a*[i]=temp[i];

        }

}

**Output :**

****