Experiment No: 6

Aim: To study and implement 6*6 Playfair Cipher.

Introduction:

Playfair Cipher is well known multiple letter encryption cipher. In this we make 6*6 matrix and put a key merging and with “abcdefghijklmnopqrstuvwxyz0123456789” delete duplication.
For example, the key is “ict4ever”, then matrix will be like,

i	c	t	4	e	v
r	a	b	d	f	g
h	j	k	l	m	n
o	p	q	s	u	w
x	y	z	0	1	2
3	5	6	7	8	9

In this cipher, we make groups of two letters and replace them with the encrypted particular letters. Here we have taken if, the letters are in the same row, then left particular letters will be printed and if the letters are in the same column, then below particular letters will be printed. And vice versa for decryption.

Program (Source Code):

```
#include<iostream>
using namespace std;

int encry(string key, string text)
{
	char pf[6][6];
	string temp = "abcdefghijklmnopqrstuvwxyz0123456789";
	key += temp;
	int size = key.length();

	for(int i=0; i<size; i++)
	{
		for(int j=i+1; j<size; j++)
		{
			if(key[i] == key[j])
			{
				for(int k=j; k<size; k++)
				{
					key[k] = key[k+1];
				}
				size--;
			}
		}
	}
```

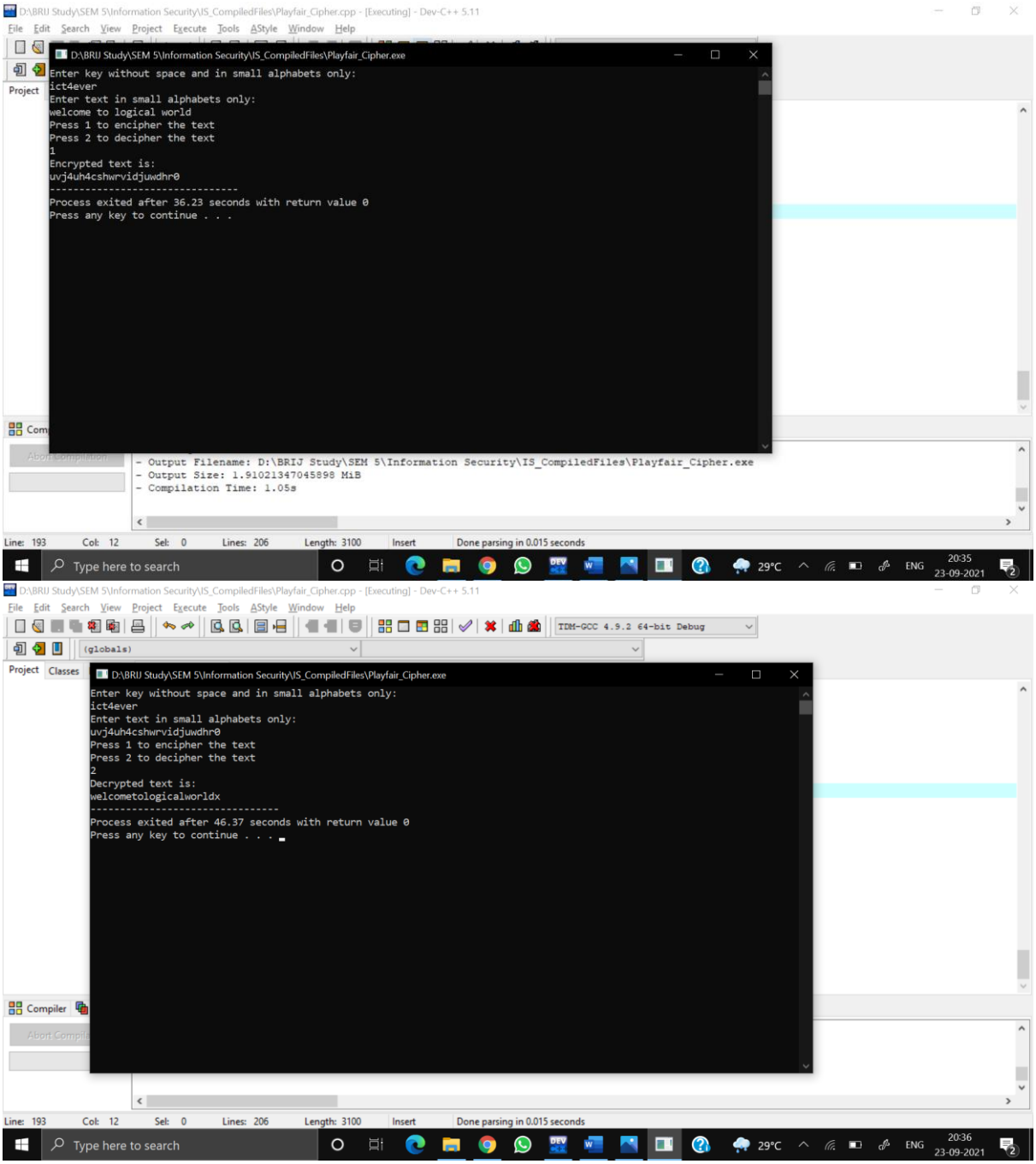
```
for(int i=0,k=0; i<6; i++)
{
	for(int j=0; j<6; j++)
	{
		pf[i][j] = key[k++];
	}
}

for(int i=0; i<text.length()-1; i++)
{
	if(text[i] == ' ')
	{
		text.erase(text.begin()+i);
	}
}

for(int i=0; i<text.length()-1; i+=2)
{
	if(text[i] == text[i+1])
	{
		string s1 = "";
		if(text[i] == 'x')
		{
			s1 += "q";
		}
		else
		{
			s1 += "x";
		}

		text.insert(i+1, s1);
	}
}

if(text.length() % 2 != 0)
{
	if(text[text.length()-1] == 'x')
	{
		text += "q";
	}
	else
	{
		text += "x";
	}
}

cout<<"Encrypted text is: "<<endl;

for(int i=0; i<text.length(); i+=2)
{
```

```
                int j1,j2,k1,k2;
                for(int j=0; j<6; j++)
                {
                        for(int k=0; k<6; k++)
                        {
                                if(text[i] == pf[j][k])
                                {
                                        j1=j;
                                        k1=k;
                                }
                                if(text[i+1] == pf[j][k])
                                {
                                        j2=j;
                                        k2=k;
                                }
                        }
                }
                if(j1==j2)
                {
                        cout<<pf[j1][(k1+5) % 6]<<pf[j2][(k2+5) % 6];
                }
                else if(k1==k2)
                {
                        cout<<pf[(j1+7) % 6][k2]<<pf[(j2+7) % 6][k1];
                }
                else
                {
                        cout<<pf[j1][k2]<<pf[j2][k1];
                }
        }

        return 0;
}

int decry(string key, string text)
{
        char pf[6][6];
        string temp = "abcdefghijklmnopqrstuvwxyz0123456789";
        key += temp;
        int size = key.length();

        for(int i=0; i<size; i++)
        {
                for(int j=i+1; j<size; j++)
                {
                        if(key[i] == key[j])
                        {
                                for(int k=j; k<size; k++)
                                {
                                        key[k] = key[k+1];
```

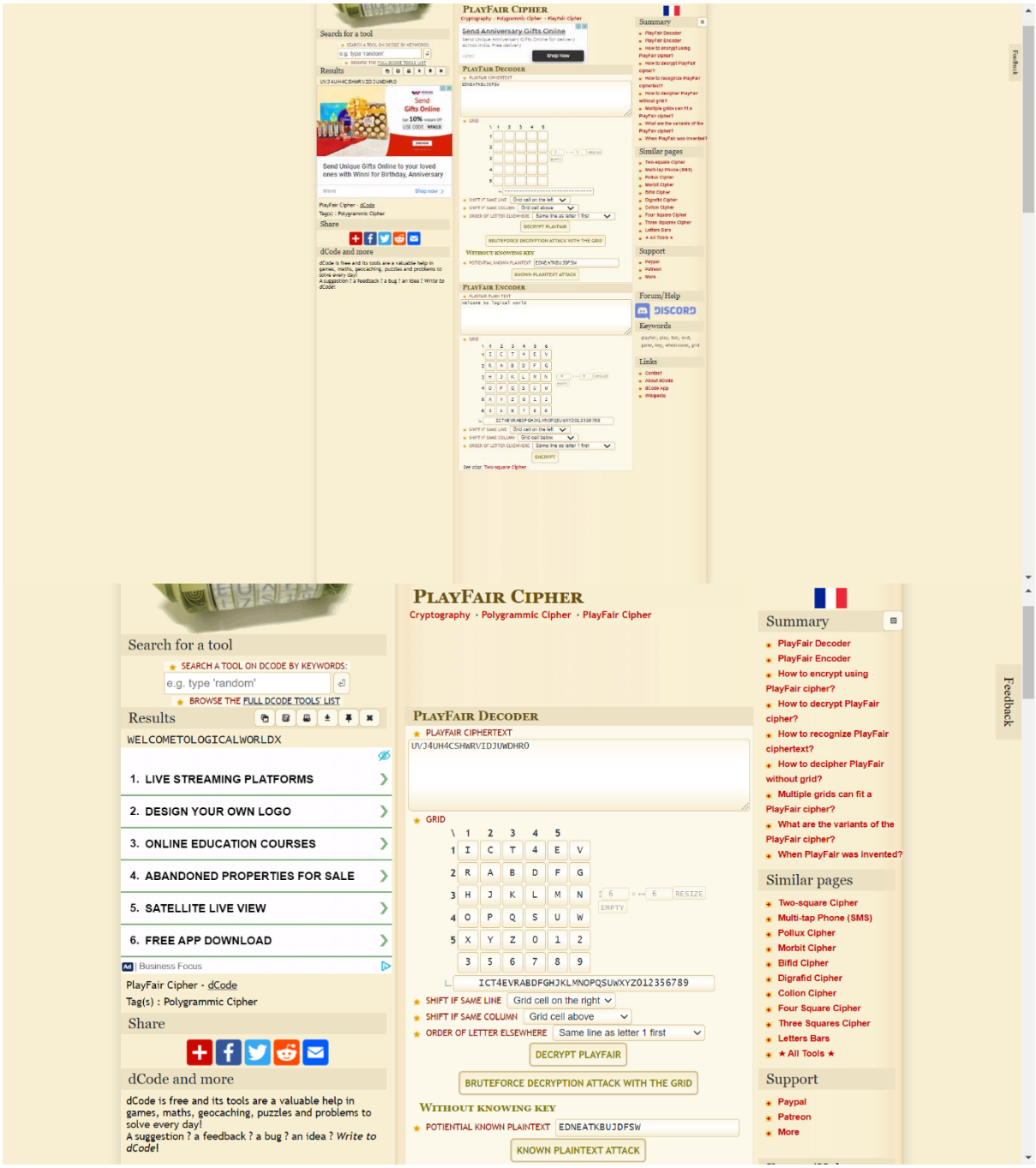
```
                        }
                        size--;
                }
        }
}

for(int i=0,k=0; i<6; i++)
{
        for(int j=0; j<6; j++)
        {
                pf[i][j] = key[k++];
        }
}

cout<<"Decrypted text is: "<<endl;

for(int i=0; i<text.length(); i+=2)
{
        int j1,j2,k1,k2;
        for(int j=0; j<6; j++)
        {
                for(int k=0; k<6; k++)
                {
                        if(text[i] == pf[j][k])
                        {
                                j1=j;
                                k1=k;
                        }
                        if(text[i+1] == pf[j][k])
                        {
                                j2=j;
                                k2=k;
                        }
                }
        }
        if(j1==j2)
        {
                cout<<pf[j1][(k1+7) % 6]<<pf[j2][(k2+7) % 6];
        }
        else if(k1==k2)
        {
                cout<<pf[(j1+5) % 6][k2]<<pf[(j2+5) % 6][k1];
        }
        else
        {
                cout<<pf[j1][k2]<<pf[j2][k1];
        }
}

return 0;
```

```
}

int main()
{
        string key,text;
        cout<<"Enter key without space and in small alphabets only: "<<endl;
        getline(cin>>ws,key);
        cout<<"Enter text in small alphabets only: "<<endl;
        getline(cin>>ws,text);

        int choice;
   cout<<"Press 1 to encipher the text"<<endl;
   cout<<"Press 2 to decipher the text"<<endl;
   cin>>choice;

   switch(choice)
   {
   case 1:
      encry(key, text);
      break;

   case 2:
      decry(key, text);
      break;

   default:
      cout<<"Try again next time"<<endl;
   }

        return 0;
}
```

Output (Program):



Output (dCode):



Cryptanalysis:

- We can crack the Playfair cipher without knowing the key by hill- climbing algorithm.
- Hill-climbing algo:-

  1. Generate a random key, called a 'parent', and click the text link using this key. Rate the validity of the specified text, save the result.
  2. Change the key slightly (change the two characters to the key at random), and measure the accuracy of the target text using the new key.
  3. If firmness is high with a fixed key, keep it as our current key and discard

the parent.

4. Go back to 2, except where no solid improvement has occurred in the last 1000 times.

- Sometimes Hill-climbing Algo gives stuck. So, we can use the simulated annealing Algo. It is very similar to Hill-climbing and it is the modification of Hill-climbing algo.

Applications:

- The Playfair cipher was used for strategic purpose by British forces in the second Boer War.
- It was used in World War I by New Zealand's for sending the confidential message.
- Also, it was used in World War II by Australians.
- It was used in many wars because it is fast to use and only requires the pen and paper.

References:

1. https://www.geeksforgeeks.org/playfair-cipher-with-examples/
2. https://www.dcode.fr/playfair-cipher