Name:-Harshita Garg

Roll No.:-19103046

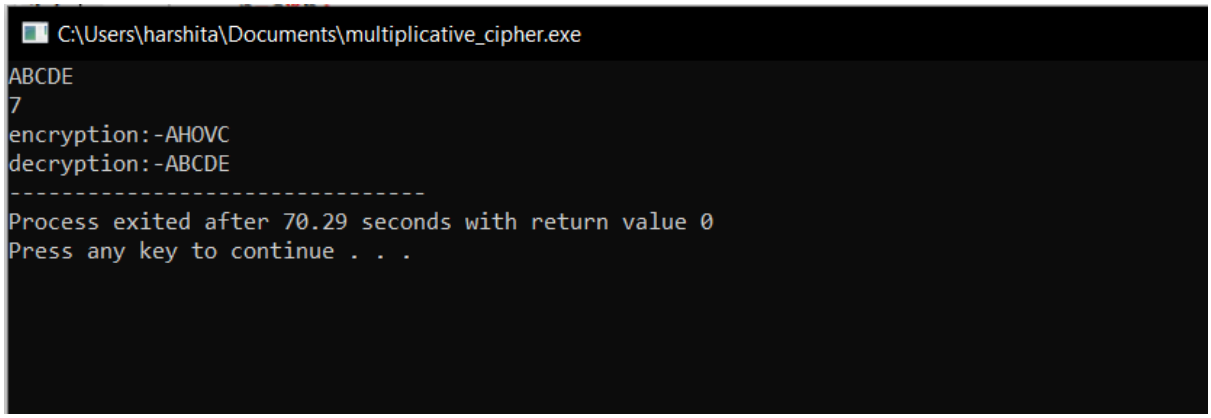
ISS lab-2

1)MULTIPLICATIVE CIPHER

```
#include<iostream>
using namespace std;
int inverse_key(int key)
{
	int n=26,a=n,b=key;
	int q=a/b,r=a%b;
	int t1=0,t2=1,t3=t1-q*t2;
	while(r!=0)
	{
		a=b;
		b=r;
		q=a/b;
		r=a%b;
		t1=t2;
		t2=t3;
		t3=t1-q*t2;
	}
	return t2<0?t2+26:t2;
}
int main()
{
	string str;
	cin>>str;
	int key;
	cin>>key;
	string encrypt="",decrypt="";
	int i=0;
```

```
        while(str[i]!='\0')
        {
                encrypt=encrypt+char(((str[i]-65)*key)%26+65);
                i++;
        }
        cout<<"encryption:-"<<encrypt;
        key=inverse_key(key);
        i=0;
        while(encrypt[i]!='\0')
        {
                decrypt=decrypt+(char) (( (encrypt[i]-'A')*key)%26+65);
                i++;
        }
        cout<<"\ndecryption:-"<<decrypt;

}
```



2) AUTOKEY CIPHER

```
#include<iostream>
#include<bits/stdc++.h>
#include<string>
using namespace std;
int main()
{
        string str;
```

```
        cin>>str;
        int k;
        cout<<"enter key";
        cin>>k;
        char key=k+65;
        string autokey="";
        autokey=key;
        for(int j=0;j<str.size()-1;j++)
        autokey=autokey+str[j];
        string encrypt="",decrypt="";
        int i=0;
        while(str[i]!='\0')
        {
                int a=str[i]-'A';
                int b=autokey[i]-'A';
                encrypt=encrypt+(char)((a+b)%26+65);
                i++;
        }
        cout<<"encrypted msg:-"<<encrypt;
        i=0;
        while(encrypt[i]!='\0')
        {
                int a=encrypt[i]-'A';
                int b=autokey[i]-'A';
                decrypt=decrypt+(char)((a-b)%26+65);
                i++;
        }
        cout<<"\ndecryption :-"<<decrypt;

}
```

```
C:\Users\harshita\Documents\autokey_cipher.exe
HELLO
enter key13
encrypted msg:-ULPWZ
decryption :-HELLO
--------------------------------
Process exited after 4.61 seconds with return value 0
Press any key to continue . . .
```

3) AFFINE CIPHER

```
#include<iostream>

using namespace std;

int inverse_key(int key)

{

        int n=26,a=n,b=key;

        int q=a/b,r=a%b;

        int t1=0,t2=1,t3=t1-q*t2;

        while(r!=0)

        {

                a=b;

                b=r;

                q=a/b;

                r=a%b;

                t1=t2;

                t2=t3;

                t3=t1-q*t2;

        }

        return t2<0?t2+26:t2;

}

int main()
```

```
{
        string str;
        cin>>str;
        int key1,key2;
        cin>>key1>>key2;
        string encrypt="";
        int i=0;
        while(str[i]!='\0')
        {
                encrypt=encrypt+(char) ((((key1 * (str[i]-'A') ) + key2) % 26) + 'A');;
                i++;
        }
        cout<<"encryption:-"<<encrypt;
        string decrypt;
        key1=inverse_key(key1);
 i=0;
        while(encrypt[i]!='\0')
        {
                decrypt=decrypt+(char) (((key1 * ((encrypt[i]+'A' - key2)) % 26)) + 'A');
                i++;
        }
        cout<<"\ndecryption:-"<<decrypt;
}
```

```
C:\Users\harshita\Documents\affine.exe
AFFINE
17
20
encryption:-UBBAHK
decryption:-AFFINE
--------------------------------
Process exited after 5.986 seconds with return value 0
Press any key to continue . . .
```

4)PLAYFAIR CIPHER

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

typedef struct
{
        int row;
        int col;
}
position;

char mat[5][5];

void generateMatrix(string key)
{
   int flag[26] = {0};
   int x = 0, y = 0;
   for(int i=0; i<key.length(); i++)
   {
      if(key[i] == 'j') key[i] = 'i'; // replace j with i

      if(flag[key[i]-'a'] == 0)
      {
         mat[x][y++] = key[i];
         flag[key[i]-'a'] = 1;
      }
      if(y==5) x++, y=0;
   }
   for(char ch = 'a'; ch <= 'z'; ch++)
   {
      if(ch == 'j') continue;
```

```
        if(flag[ch - 'a'] == 0)
        {
            mat[x][y++] = ch;
            flag[ch - 'a'] = 1 ;
        }
        if(y==5) x++, y=0;
    }
}
string formatMessage(string msg)
{
    for(int i=0; i<msg.length(); i++)
    {
        if(msg[i] == 'j')  msg[i] = 'i';
    }

    for(int i=1; i<msg.length(); i+=2)
    {
        if(msg[i-1] == msg[i])  msg.insert(i, "x");
    }

    if(msg.length()%2 != 0)  msg += "x";
    return msg;
}
position getPosition(char c)
{
    for(int i=0; i<5; i++)
        for(int j=0; j<5; j++)
            if(c == mat[i][j])
            {
                position p = {i, j};
                return p;
```

```
        }
}

string encrypt(string message)
{
   string ctext = "";
   for(int i=0; i<message.length(); i+=2)
   {
                    position p1 = getPosition(message[i]);
                    position p2 = getPosition(message[i+1]);
      int x1 = p1.row; int y1 = p1.col;
      int x2 = p2.row; int y2 = p2.col;

      if( x1 == x2 ) // same row
      {
         ctext +=  mat[x1][(y1+1)%5];
         ctext +=  mat[x2][(y2+1)%5];
      }
      else if( y1 == y2 ) // same column
      {
         ctext += mat[ (x1+1)%5 ][ y1 ];
         ctext += mat[ (x2+1)%5 ][ y2 ];
      }
      else
      {
         ctext += mat[ x1 ][ y2 ];
         ctext += mat[ x2 ][ y1 ];
      }
   }
   return ctext;
}
```

```
string Decrypt(string message)
{
   string ptext = "";
   for(int i=0; i<message.length(); i+=2)
   {
      position p1 = getPosition(message[i]);
                        position p2 = getPosition(message[i+1]);
      int x1 = p1.row; int y1 = p1.col;
      int x2 = p2.row; int y2 = p2.col;

      if( x1 == x2 ) // same row
      {
         ptext += mat[x1][ --y1<0 ? 4: y1 ];
         ptext += mat[x2][ --y2<0 ? 4: y2 ];
      }
      else if( y1 == y2 ) // same column
      {
         ptext += mat[ --x1<0 ? 4: x1 ][y1];
         ptext += mat[ --x2<0 ? 4: x2 ][y2];
      }
      else
      {
         ptext += mat[ x1 ][ y2 ];
         ptext += mat[ x2 ][ y1 ];
      }
   }
   return ptext;
}

int main()
```

```
{
    cout<<endl;
    string plaintext;
    cout <<"Enter text  : ";
    cin >> plaintext;

    string key;
    cout <<"Enter key   : ";
    cin >> key;
    cout<<endl;
    generateMatrix(key);

    cout << "Matrix:" << endl;
    for(int k=0;k<5;k++)
    {
        for(int j=0;j<5;j++)
        {
            cout << mat[k][j] << "  ";
        }
        cout << endl;
    }
    cout<<endl;
    cout << "Actual Message \t\t: " << plaintext << endl;

    string fmsg = formatMessage(plaintext);
    cout << "Formatted Message \t: " << fmsg << endl;

    string ciphertext = encrypt(fmsg);
    cout << "Encrypted Message \t: " << ciphertext << endl;

    string decryptmsg = Decrypt(ciphertext);
```

```
    cout<< "Decrypted Message \t: " << decryptmsg << endl<<endl;


    return 0;
}
```

```
C:\Users\harshita\Documents\playfair_cipher.exe

Enter text  : instruments
Enter key   : monarchy

Matrix:
m  o  n  a  r
c  h  y  b  d
e  f  g  i  k
l  p  q  s  t
u  v  w  x  z

Actual Message          : instruments
Formatted Message       : instrumentsx
Encrypted Message       : gatlmzclrqxa
Decrypted Message       : instrumentsx


--------------------------------
Process exited after 8.155 seconds with return value 0
Press any key to continue . . .
```