Experiment 2

Aim: Write a program to implement Monoalphabetic encryption and decryption.

Theory: Monoalphabetic cipher is a substitution cipher in which for a given key, the cipher alphabet for each plain alphabet is fixed throughout the encryption process. For example, if 'A' is encrypted as 'D', for any number of occurrences in that plaintext, 'A' will always get encrypted to 'D'. The possible number of keys is large (26!) and so Monoalphabetic ciphers are highly susceptible to cryptanalysis.

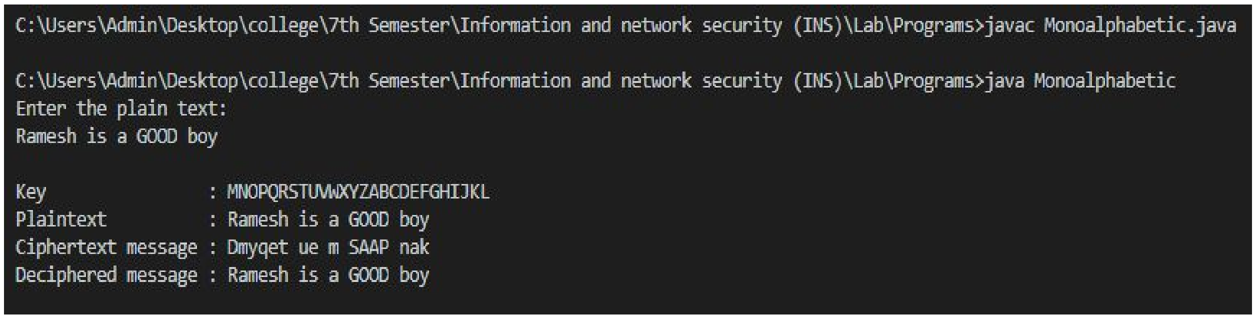
Source Code:

```
import java.util.Scanner;
public class Monoalphabetic {
   private static final String alphabet = "abcdefghijklmnopqrstuvwxyz";
   private static final String Al = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
   public static String encrypt(String plaintext, String key) {
      String ciphertext = "";

      for (char chr : plaintext.toCharArray()) {
         if(Character.isUpperCase(chr))
         {
         byte position = (byte) Al.indexOf(chr);
         if (chr == ' ') {
               ciphertext+=" ";
            }
         else {
               ciphertext+=Character.toUpperCase(key.charAt(position));
               }
         }
         else
         {
            byte position = (byte) alphabet.indexOf(chr);
            if (chr == ' ') {
                  ciphertext+=" ";
               }
            else {
                  ciphertext+=key.charAt(position);
               }
            }
        }
        return ciphertext;
    }

    public static String decrypt(String ciphertext, String key) {
        String plaintext ="";
        for (char chr : ciphertext.toCharArray()) {
            if(Character.isUpperCase(chr))
            {
            byte position = (byte) key.indexOf(Character.toLowerCase(chr));
            if (chr == ' ')
                    {       plaintext+=" ";         }
            else
                    {       plaintext+=Al.charAt(position);       }
            }
            else
            {
                byte position = (byte) key.indexOf(chr);
                if (chr == ' ')
                    {       plaintext+=" ";         }
                else
                    {   plaintext+=alphabet.charAt(position);     }
            }
        }
        return plaintext;
    }
    public static void main(String[] args) {
        Scanner myObj = new Scanner(System.in);
        System.out.println("Enter the plain text:");
        String text = myObj.nextLine();
        String key="mnopqrstuvwxyzabcdefghijkl";
        System.out.println("\nKey              : "+key.toUpperCase()+"\nPlaintext         : "+text );
        String ciphertext=encrypt(text, key);
        System.out.println("Ciphertext message : "+ciphertext);
        System.out.println("Deciphered message : " +decrypt(ciphertext,key ));
    }
}
```

Output:



```
C:\Users\Admin\Desktop\college\7th Semester\Information and network security (INS)\Lab\Programs>javac Monoalphabetic.java

C:\Users\Admin\Desktop\college\7th Semester\Information and network security (INS)\Lab\Programs>java Monoalphabetic
Enter the plain text:
Ramesh is a GOOD boy

Key               : MNOPQRSTUVWXYZABCDEFGHIJKL
Plaintext         : Ramesh is a GOOD boy
Ciphertext message : Dmyqet ue m SAAP nak
Deciphered message : Ramesh is a GOOD boy
```

Learning Outcomes:

The modern computing systems are not yet powerful enough to comfortably launch a brute force attack to break the system having these ciphers. However, the Monoalphabetic Cipher has a simple design and it is prone to design flaws, say choosing obvious permutation, this cryptosystem can be easily broken.