Ex No: 1	**One Time Pad Cryptography**
Date: /12/24	

Objective: To implement One time pad cryptography using .

Description:

1. Generate Key:
The generateRandomKey method generates a random binary string of the same length as the plaintext using the Random class.
2. XOR Operation:
The XOR Operation method performs a bitwise XOR operation between the plaintext and key, iterating through their characters.
3. Encryption:
The plaintext is XORed with the key to generate the ciphertext.
4. Decryption:
The ciphertext is XORed with the same key to retrieve the original plaintext.
5. Verification:
After decryption, the program verifies if the decrypted text matches the original plaintext.package onetime;

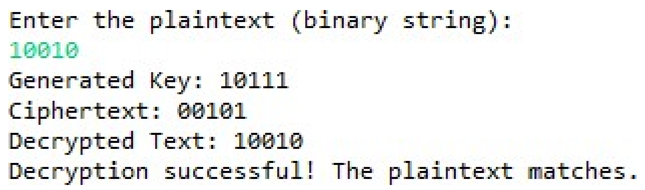
Algorithm:

Step 1: Input the plaintext as a binary string.
Step 2:Generate a random key of the same length as the plaintext.
Step 3:Use a random number generator to create a binary string.
Step 4:Encrypt the plaintext using XOR operation:
Step 5:For each bit in the plaintext, XOR it
Step 6:Store the result as the ciphertext.
Step 7:Decrypt the ciphertext using the same XOR operation:
Step 8:XOR each bit of the ciphertext with the corresponding bit in the key.

Programs:

```
package ex1;
import java.util.Random;
import  java.util.Scanner;
public class ex1 {
 private static String generateRandomKey(int length) {
  Random random = new Random();
  StringBuilder key = new StringBuilder();
  for (int i = 0; i < length; i++) {
   key.append(random.nextInt(2)); // Append random 0 or 1
  }
  return key.toString();
 }
 private static String xorOperation(String text, String key) {
  stringBuilder result = new StringBuilder();
  for (int i = 0; i < text.length(); i++) {
   result.append(text.charAt(i) ^ key.charAt(i)); // XOR each bit
  }
  return result.toString();
 }
 public static void main(String[] args) {
  Scanner scanner = new Scanner(System.in);
  System.out.println("Enter the plaintext (binary string): ");
  String plaintext = scanner.nextLine().trim();
  if (!plaintext.matches("[01]+")) {
   System.out.println("Error: Plaintext must be a binary string containing only 0s and 1s. ");
   return;
  }
  String key = generateRandomKey(plaintext.length());
  System.out.println("Generated Key: " + key);
  String ciphertext = xorOperation(plaintext, key);
  System.out.println("Ciphertext: " + ciphertext);
  String decryptedText = xorOperation(ciphertext, key);
  System.out.println("Decrypted Text: " + decryptedText);
  if (plaintext.equals(decryptedText)) {
   System.out.println("Decryption successful! The plaintext matches.");
  } else {
   System.out.println("Decryption failed! The plaintext does not match.");
  }
  scanner.close();
 }
}
```

Output:



```
Enter the plaintext (binary string):
10010
Generated Key: 10111
Ciphertext: 00101
Decrypted Text: 10010
Decryption successful! The plaintext matches.
```

Result:

The result has been obtained and the output has been verified.