```java
/**
 * Standalone program to encrypt messages using a key.
 *
 * @author  Terry Sergeant
 * @date    11 Jan 2005
 * @course  Discrete Structures
 *
 * Uses a primitive key-based encryption to encrypt a given file with
 * a specified key.  The key comes from the command line.  The input
 * comes from stdin and the output goes to stdout.  The end of line
 * characters have been kept so that each "line" of the file remains
 * in tact.
*/

import java.util.Scanner;

public class Encrypt
{
  /**
   * Obtains key and accepts input to be encrypted.
   */
  public static void main(String [] args)
  {
    Scanner cin= new Scanner(System.in);  // input comes from console
    String msg;  // message to be encrypted
    String key;  // key used to perform encryption

    if (args.length < 1) {  // command-line argument is missing
      System.out.print("Enter Encryption Key: ");   // we prompt for it
      key= cin.next();
      cin.nextLine();   // toss out left-over \n
    }
    else         // use the provided argument as the key
      key= args[0];

    while (cin.hasNextLine()) {
      msg= cin.nextLine();
      msg= encrypt(msg,key);
      System.out.println(msg);
    }
  }


  /**
   * Given a plaintext message and a key, produces an encrypted message.
   *
   * <pre>
   * Suppose the message was: "This is my message."
   *     and the key     was: "MYKEY"
   *
   * The encrypted message is found by adding aligning chars to produce
   * a new ASCII value.  If the value is bigger than 128 then it is
   * wrapped around starting from 0 ... like this:
   *
   *     This is my message.
   *   + MYKEYMYKEYMYKEYMYK
   *   ---------------------
   *     somegobbledygookxx
   * </pre>
   *
   * @param msg  The message to be encrypted
   * @param key  The key to be used to encrypt the message
   * @return The encrypted message
  */
  public static String encrypt(String msg, String key)
  {
    int i;          // counter
    int mlen;       // length of message
    int klen;       // length of key
    char [] cipher;// holds encrypted characters

    mlen= msg.length();
    klen= key.length();
```



```java
    cipher= new char[mlen];

    for (i=0; i<mlen; i++)
      cipher[i]= (char) ((int)(msg.charAt(i) + key.charAt(i%klen)) % 128);

    return new String(cipher);
  }
}
```