```
import java.util.*;
import java.io.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
/**
 * Time to screw around.
 *
 * Certain segments borrowed from CardLayoutDemo by Oracle
 *
https://docs.oracle.com/javase/tutorial/uiswing/examples/layout/CardLayoutDemoProject/src/layout/CardLayoutDemo.j
ava
 */
public class Menu implements ItemListener {
    private Encryptor enc = new Encryptor();
    private PasswordManager manager = new PasswordManager();
    public static final ErrorLogger el = new ErrorLogger();

    private JTextField website = new JTextField("Website", 10);
    private JTextField username = new JTextField("Username", 10);
    private JTextField password = new JTextField("Password",10);

    private JPanel cards;
    private final String BUTTONPANEL = "Encryption";
    private final String TEXTPANEL = "PasswordManager";

    private JButton encrypt = new JButton("Encrypt");
    private JTextField encinput = new JTextField("message...", 25);
    private JButton decrypt = new JButton("Decrypt");
    private JButton b = new JButton("Generate");

    private JButton add = new JButton("Add account");
    private JTextField pwinput = new JTextField("",20);
    private JButton search = new JButton("Search");

    public static void main() {
        JFrame program = new JFrame("CardLayoutDemo");
        Menu menu = new Menu();
        program.addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent e) {
                menu.el.log();
                menu.manager.writeToFile();
            }
        });
        program.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        menu.addComponents(program.getContentPane());
        program.pack();
        program.setVisible(true);
    }

    private void addComponents(Container pane) {
        JPanel comboBoxPane = new JPanel();
        String comboBoxItems[] = { BUTTONPANEL, TEXTPANEL };
```

```
        JComboBox box = new JComboBox(comboBoxItems);
        box.setEditable(false);
        box.addItemListener(this);
        comboBoxPane.add(box);

        JPanel card1 = new JPanel();
        card1.add(encrypt);
        card1.add(encinput);
        card1.add(decrypt);
        card1.add(b);

        JPanel card2 = new JPanel();
        card2.add(pwinput);
        card2.add(add);
        card2.add(search);

        addActionListeners();

        cards = new JPanel(new CardLayout());
        cards.add(card1, BUTTONPANEL);
        cards.add(card2, TEXTPANEL);

        pane.add(comboBoxPane, BorderLayout.PAGE_START);
        pane.add(cards, BorderLayout.CENTER);
    }

    /**Buttons in both cards*/
    private void addActionListeners() {
        encrypt.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                String in = encinput.getText();
                if(in.equals("message...") || in.equals(""))
                    JOptionPane.showMessageDialog(new JFrame(),"Enter something first!");
                else {
                    if(!enc.empty()) {
                        JTextArea wat = new JTextArea(enc.encrypt(in));
                        JOptionPane.showMessageDialog(new JFrame(), wat);
                    } else
                        JOptionPane.showMessageDialog(new JFrame(), "No conversion file found. If you are supposed to
receive a file from a friend,\nclose the program, add the file to the folder with the program and\nrestart.\n\nIf you are
not receiving a file for this from a friend, click generate and proceed.");
                }
            }
        });
        decrypt.addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent e) {
                String in = encinput.getText();
                if(in.equals("message...") || in.equals(""))
                    JOptionPane.showMessageDialog(new JFrame(),"Enter something first!");
                else {
                    if(!enc.empty()) {
                        JTextArea wat = new JTextArea(enc.decrypt(in));
                        JOptionPane.showMessageDialog(new JFrame(), wat);
                    } else
```

```
                JOptionPane.showMessageDialog(new JFrame(), "No conversion file found. If you are supposed to receive a file from a friend,\nclose the program, add the file to the folder with the program and\nrestart.\n\nIf you are not receiving a file for this from a friend, click generate and proceed.");
            }
        }
    });
    b.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            enc.generateNewValues();
            JOptionPane.showMessageDialog(new JFrame(), "Values generated.");
        }
    });
    add.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            createAndShow2ndGUI();
        }
    });
    search.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            String input = pwinput.getText();
            JTextArea wat = new JTextArea(manager.getCredentials(input).toString());
            JOptionPane.showMessageDialog(new JFrame(), wat);
        }
    });
}

private void createAndShow2ndGUI() {
    JFrame frame = new JFrame("Account Creation");
    JPanel panel = new JPanel();
    JButton generate = new JButton("Generate password");
    generate.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            int s = JOptionPane.showConfirmDialog(new JFrame(),"Special characters?","",JOptionPane.YES_NO_OPTION);
            boolean special = false;
            if(s == 0)
                special = true;
            password.setText(manager.generatePassword(special));
        }
    });
    JButton save = new JButton("Save");
    save.addActionListener(new ActionListener(){
        public void actionPerformed(ActionEvent e) {
            manager.addAccount(website.getText(), username.getText(), password.getText());
            JOptionPane.showMessageDialog(new JFrame(), "Saved!");
        }
    });
    panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
    website.setAlignmentX(Component.CENTER_ALIGNMENT);
    username.setAlignmentX(Component.CENTER_ALIGNMENT);
    password.setAlignmentX(Component.CENTER_ALIGNMENT);
    generate.setAlignmentX(Component.CENTER_ALIGNMENT);
    save.setAlignmentX(Component.CENTER_ALIGNMENT);
    JPanel comboBoxPane = new JPanel();
```

```
        String comboBoxItems[] = { TEXTPANEL };
        JComboBox box = new JComboBox(comboBoxItems);
        box.setEditable(false);
        box.addItemListener(this);
        comboBoxPane.add(box);

        panel.add(website);
        panel.add(username);
        panel.add(password);
        panel.add(generate);
        panel.add(save);

        frame.setSize(200,200);
        frame.getContentPane().add(comboBoxPane, BorderLayout.PAGE_START);
        frame.getContentPane().add(panel, BorderLayout.CENTER);
        frame.pack();
        frame.setVisible(true);
    }

    public void itemStateChanged(ItemEvent evt) {
        CardLayout cl = (CardLayout)(cards.getLayout());
        cl.show(cards, (String)evt.getItem());
    }
}
import java.util.*;
import java.io.*;
/**
 * Encryption
 * @Author: Danylo Mirin
 * @Copyright 2018
 * @Date: 27 February 2018
 * @Version: 5
 * @Error Codes:
 * 1 = no writing or reading access
 * 2 = IOException - writing fail
 * 3 = File Not Found - file(s) were removed midway through operation
 * 4 = IOException - reading fail
 * 5 = when message was copied, it was not copied in full. Decryption
 * impossible.
 *
 * @Notes:
 * If you and your partner have different keys, one of you needs to send his to the other.
 * The conversion file wouldn't normally be safe - intercept it and you're done. Therefore,
 * a second, common algorithm encrypts the conversion file so that it is unreadable to humans
 * and by the time a machine finds a decryption for it, your message will already go through
 * and encryption can be changed again
 * If the conversion file is empty/nonexistent or after initialisation it is found that not all
 * characters are present, it will be rewritten. This WILL screw with decryption on the other
 * end! You will have to pass on the key to your partner or decryption will fail and give you
 * gibberish answers. (encryption stored in l2n.enc)
 */
public class Encryptor {
    private HashMap<String, String> alphabet = new HashMap<>();
    private BufferedReader reader;
```

```
private BufferedWriter writer;
private File convFile = new File("l2n.enc");
private File log = new File("log.txt");
public final ErrorLogger el = new ErrorLogger();
public Encryptor() {
   if(!convFile.exists()) {
      try {
         if(!log.exists())
            log.createNewFile();
         convFile.createNewFile();
      } catch(IOException ex) {
         el.add(ex,1);
      }
   } else {
      try {
         if(!log.exists())
            log.createNewFile();
         readFromFile();
      } catch(FileNotFoundException ex) {
         //this should already be taken care of by the "if" part of this statement but just in case...
         el.add(ex, 3);
      } catch(IOException ex) {
         el.add(ex, 2);
      }
   }
}

//This constructor simply saves the computer the trouble of making up new values, everything else remains the same
public Encryptor(HashMap<String, String> alphabet) {
   this.alphabet.putAll(alphabet);
   if(!convFile.exists()) {
      convFile.mkdirs();
      try {
         if(!log.exists())
            log.createNewFile();
         convFile.createNewFile();
         writeToFile();
      } catch(IOException ex) {
         el.add(ex, 1);
         System.exit(2);
      }
   } else {
      try {
         if(!log.exists())
            log.createNewFile();
         readFromFile();
      } catch(FileNotFoundException ex) {
         el.add(ex, 3);
         System.exit(3);
      } catch(IOException ex) {
         el.add(ex, 2);
         System.exit(4);
      }
   }
```

```
    }

    public static void main() {
        System.out.println(Encryptor.ln("740403184938595",false));
    }

    /*#Encryption*/
    public String encrypt(String message) {
        String scrambled = "";
        for(int i = 0; i < message.length(); i++)
            scrambled += alphabet.get(message.substring(i,i+1));
        return scrambled;
    }
    public void generateNewValues() {
        String[] chars =
"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890~!@#$%^&*()-_=+{}[]|;:'\",.
<>/?".split("");
        for(String s : chars) {
            String value = keyGen();//generates a random 10-character key for encryption
            System.out.println(s+", "+value);
            alphabet.put(s, value);
        }
        try{
            writeToFile();
        } catch(IOException ex) {
            el.add(ex, 1);
        }
    }
    public String keyGen() {//In the EXTREMELY improbable event that keys are the same, just rerun the program. If
that doesn't fix it...
        String[] chars = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890".split("");
        String key = "";
        for(int i = 0; i < 10; i++) {
            int rand = (int)(Math.random()*chars.length);
            key += chars[rand];
        }
        return key;
    }

    public String decrypt(String scrambled) {
        String message = "";
        //should always be false because every encryption value is 10 chars long
        if(scrambled.length() % 10 != 0)
            System.exit(5);
        for(int i = 0; i < scrambled.length(); i+= 10) {
            String expression = scrambled.substring(i,i+10);
            for(String s : alphabet.keySet())
                if(alphabet.get(s).equals(expression))
                    message += s;
        }
        return message;
    }

    private void readFromFile() throws FileNotFoundException, IOException {
```

```
    reader = new BufferedReader(new FileReader(convFile));
    String line;
    while( (line = reader.readLine()) != null) {
        //line = ln(line,false);
        alphabet.put(line.substring(0,1), line.substring(2));
    }
    reader.close();
}

private void writeToFile() throws IOException {
    writer = new BufferedWriter(new FileWriter(convFile));
    for(String k : alphabet.keySet()) {
        String s = k+"|"+alphabet.get(k);
        System.out.println(s);
        writer.write(s+"\n");
    }
    writer.flush();
    writer.close();
}
/**This method is gonna be ugly because I do not want to rely on external files*/
public static String ln(String line, boolean encrypt) {
    String answer = "";
    HashMap<String, Integer> map = new HashMap<>();
    map.put("A",116);
    map.put("B",404);
    map.put("C",754);
    map.put("D",891);
    map.put("E",534);
    map.put("F",713);
    map.put("G",482);
    map.put("H",996);
    map.put("I",309);
    map.put("J",414);
    map.put("K",745);
    map.put("L",296);
    map.put("M",824);
    map.put("N",678);
    map.put("O",732);
    map.put("P",541);
    map.put("Q",563);
    map.put("R",527);
    map.put("S",926);
    map.put("T",830);
    map.put("U",755);
    map.put("V",130);
    map.put("W",508);
    map.put("X",153);
    map.put("Y",627);
    map.put("Z",235);
    map.put("a",538);
    map.put("b",322);
    map.put("c",184);
    map.put("d",969);
    map.put("e",276);
```

```
map.put("f",740);
map.put("g",120);
map.put("h",436);
map.put("i",136);
map.put("j",394);
map.put("k",938);
map.put("l",277);
map.put("m",483);
map.put("n",593);
map.put("o",942);
map.put("p",388);
map.put("q",627);
map.put("r",989);
map.put("s",604);
map.put("t",773);
map.put("u",403);
map.put("v",357);
map.put("w",195);
map.put("x",226);
map.put("y",164);
map.put("z",271);
map.put("1",562);
map.put("2",520);
map.put("3",905);
map.put("4",258);
map.put("5",650);
map.put("6",349);
map.put("7",444);
map.put("8",471);
map.put("9",962);
map.put("0",550);
map.put("~",812);
map.put("!",595);
map.put("@",346);
map.put("#",895);
map.put("$",925);
map.put("%",690);
map.put("^",585);
map.put("&",875);
map.put("*",984);
map.put("(",606);
map.put(")",502);
map.put("-",258);
map.put("_",984);
map.put("=",183);
map.put("+",835);
map.put("{",624);
map.put("}",838);
map.put("[",862);
map.put("]",962);
map.put("|",813);
map.put(";",474);
map.put(":",778);
map.put("'",762);
```

```
        map.put("\"",215);
        map.put(",",113);
        map.put(".",232);
        map.put("<",341);
        map.put(">",560);
        map.put("/",951);
        map.put("?",922);

        if(encrypt)
            for(int i = 0; i < line.length(); i++)
                answer += map.get(line.substring(i,i+1));
        else
            for(int i = 0; i < line.length(); i+=3)
                for(String key : map.keySet())
                    if(map.get(key).equals(Integer.parseInt(line.substring(i,i+3))))
                        answer += key;

        return answer;
    }

    public boolean empty() {
        return alphabet.isEmpty();
    }
}
import java.io.*;
import java.util.*;
/**
 * Password Manager by Danylo Mirin
 * @Author Danylo Mirin
 * @Copyright 2018
 * @Date 4/3/18 (d/m/y)
 * @Version 1
 */
public class PasswordManager {
    //There are 3 bits to hold: website/company, username, password.
    //Cannot do that conveniently with just a hashmap
    private HashMap<String, Account> map;
    private File list = new File("list.pw");
    private ErrorLogger el = new ErrorLogger();
    public PasswordManager() {
        map = new HashMap<>();
        try {
            if(!list.exists())
                list.createNewFile();//may have issues b/c of writing access
            else {
                readFromFile();
            }
        } catch(FileNotFoundException ex) {
            el.add(ex, 3);
        } catch(IOException ex) {
            el.add(ex, 1);
        }
    }
```

```
    public static void main() {
        PasswordManager pw = new PasswordManager();
        pw.run();
    }

    public void run() {
        System.out.println(generatePassword(true));
    }

    public boolean addAccount(String site, String name, String password) {
        return map.putIfAbsent(site, new Account(name,password)) == null;
    }

    public Account getCredentials(String website) {
        return map.get(website);
    }

    public String generatePassword(boolean special) {
        String[] chars =
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890~!@#$%^&*()_+-=[]{}|;:,.
<>?/'".split("");
        int upperBound; String pw = "";
        if(special)
            upperBound = chars.length;
        else
            upperBound = 26+26+10;
        for(int i = 0; i < 12; i++)
            pw += chars[(int)(Math.random()*upperBound)];
        return pw;
    }

    public void readFromFile() {
        try {
            BufferedReader reader = new BufferedReader(new FileReader(list));
            String line;
            while( (line = reader.readLine()) != null) {
                //line = Encryptor.ln(line,false);
                int arrow = line.indexOf("-->");
                int semcol = line.indexOf("; ");
                map.put(line.substring(0, arrow-1), new Account(line.substring(arrow+4, semcol), line.substring(semcol+2)));
            }
        } catch(FileNotFoundException ex) {
            el.add(ex, 3);
        } catch(IOException x) {
            el.add(x, 2);
        }
    }

    public boolean writeToFile() {
        try {
            BufferedWriter writer = new BufferedWriter(new FileWriter(list));
            for(String s : map.keySet()) {
                String line = s+" --> "+map.get(s);
                System.out.println(line);
```

```
            writer.write(line+"\n");
          }
          writer.flush();
          writer.close();
          System.out.println("Success, terminated");
          return true;
       } catch(FileNotFoundException ex) {
          el.add(ex, 3);
       } catch(IOException ex) {
          el.add(ex, 1);
       }
       return false;
    }
    public class Account {
       public String name, password;
       public Account(String name, String password) {
          this.name = name;
          this.password = password;
       }

       public String toString() {
          return name+"; "+password;
       }
    }
}
import java.io.*;
import java.util.HashMap;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import javax.swing.JFrame;
/**
 * @Author Danylo Mirin
 * @Copyright 2018
 * @Date 4/3/18 (d/m/y)
 * @Version 1
 * @Purpose: to escape the need to write the same error codes everywhere
 */

/*# OPERATIONAL! */
public class ErrorLogger {
    private HashMap<Integer, String> codes;
    private ArrayList<String> text;
    private ArrayList<Exception> exceptions;
    public ErrorLogger() {
       text = new ArrayList<>();
       codes = new HashMap<>();
       codes.put(1,"Writing access denied");
       codes.put(2,"Reading access denied");
       codes.put(3,"File not present");
       codes.put(4,"What the hell? (unknown error)");
       for(Integer i : codes.keySet())
          text.add(i+" - "+codes.get(i)+"\n");
       text.add("\n\n");
    }
```

```
    public static void main() {
        ErrorLogger el = new ErrorLogger();
        try {
            Integer.parseInt("13a");
        } catch(NumberFormatException ex) {
            el.add(ex, 4);
        }
        el.log();
    }

    public void add(Exception e, int code) {
        text.add("code "+code+"; "+e.getClass());
        StackTraceElement[] elements = e.getStackTrace();
        for(StackTraceElement el : elements)
            text.add(el.toString());
    }

    public void log() {
        File log = new File("log.txt");
        try {
            log.createNewFile();
            BufferedWriter writer = new BufferedWriter(new FileWriter(log));
            while(!text.isEmpty())
                writer.write(text.remove(0));
            writer.flush();
            writer.close();
        } catch(FileNotFoundException ex) {
            //Should never run
            JOptionPane.showMessageDialog(new JFrame(), "log file not found, cannot write the log. Potential cause: read
& write access denied.");
        } catch(IOException ex) {
            //Implies lack of reading/writing access
            JOptionPane.showMessageDialog(new JFrame(), "Read & write access denied, logs cannot be recorded.");
        }
        //Since this is a log and not a crucial step of the application, it is not necessary to autoexit
    }

    public void printLines() {
        for(String s : text)
            System.out.println(s);
    }
}
```