```
 1  // joi/8/juno/Password.java//
 2  //
 3  //
 4  //  Copyright 2003 Bill Campbell and Ethan Bolker
 5
 6  /**
 7   * Model a good password.
 8   *
 9   * <p>
10   * A password is a String satisfying the following conditions
11   * (close to those required of Unix passwords, according to
12   * the <code> man passwd </code> command in Unix):
13   * <br>
14   * <ul>
15   * <li> A password must have at least PASSLENGTH characters, where
16   *      PASSLENGTH defaults to 6. Only the first eight characters
17   *      are significant.
18   *
19   * <li> A password must contain at least two alphabetic characters
20   *      and at least one numeric or special character. In this  case,
21   *      "alphabetic" refers to all upper or lower case letters.
22   *
23   * <li> A password must not contain a specified string as a substring
24   *      For comparison purposes, an upper case letter and its
25   *      corresponding lower case letter are equivalent.
26   *
27   * <li> A password must not be a substring of a specified string.
28   *      For comparison purposes, an upper case letter and its
29   *      corresponding lower case letter are equivalent.
30   *
31   * </ul>
32   * <br>
33   * A password string may be stored in a Password object only in
34   * encrypted form.
35   */
36
37  public class Password
38  {
39      private String password;
40
41      /**
42       * Construct a new Password.
43       *
44       * @param password the new password.
45       * @param notSubstringOf a String that may not contain the password.
46       * @param doesNotContain a String the password may not contain.
47       *
48       * @exception BadPasswordException when password is unacceptable.
49       */
50
51      public Password(String password, String notSubstringOf,
52                      String doesNotContain)
53           throws BadPasswordException
54      {
55          // if password is not acceptable
56          // throw new BadPasswordException( reason )
```



```
57          this.password = encrypt(password);
58      }
59
60      // Rewrite s in a form that makes it hard to guess s.
61
62      private String encrypt( String s )
63      {
64          return Integer.toHexString(s.hashCode());
65      }
66
67      /**
68       * See whether a supplied guess matches this password.
69       *
70       * @param guess the trial password.
71       *
72       * @exception BadPasswordException when match fails.
73       */
74
75      public void match(String guess)
76           throws BadPasswordException
77      {
78      }
79
80      /**
81       * Unit test for Password objects.
82       */
83
84      public static void main( String[] args )
85      {
86      }
87  }
```