**PasswordVerifyApp.Java**

//Chandler Perry, COP-3330C, 9/20/24  
// Throwing and Catching a Basic Exception, Designing Your Own Custom Exceptions  
//For the division application, User will give two intergers and the programm will do the division and output the number.  
//For the Password application, the programm will ask for a username and password both have their own special characters  
// that need to be implemented to get an account created, once the username and password requirements have been meet,  
//the program will them allow the account to be created.  
  
package com.example.passwordvalidation;  
  
import java.util.Scanner;  
  
public class PasswordVerifyApp {// This class compiles the passwordVerify.Java and passwordverifyexception.java to produce the output that the user will use to create there own account  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 System.*out*.print("How many accounts would you like to create? ");  
 int accountLimit = scanner.nextInt();  
 scanner.nextLine(); // Clear the newline  
  
 PasswordVerify[] validAccounts = new PasswordVerify[accountLimit];  
 int validCount = 0;  
  
 while (validCount < accountLimit) {  
 System.*out*.println("Username must be at least 8 characters and contain a special character (!, @, #, $).");  
 System.*out*.println("Password must contain exactly 3 vowels and must not contain digits 4, 6, or 9.");  
  
 String username = null;  
 String password = null;  
  
 boolean isValidUsername = false;  
 boolean isValidPassword = false;  
  
 // Get a valid username  
 while (!isValidUsername) {  
 try {  
 System.*out*.print("Enter a username: ");  
 username = scanner.nextLine();  
  
 if (username.length() < 8) {  
 throw new PasswordVerifyException("Username must be at least 8 characters long.");  
 } else if (!username.matches(".\*[!@#$].\*")) {  
 throw new PasswordVerifyException("Username must contain at least one special character (!, @, #, $).");  
 }  
  
 isValidUsername = true; // Username is valid if no exception is thrown  
 } catch (PasswordVerifyException e) {  
 System.*out*.println(e.getMessage());  
 System.*out*.println("Please try again.");  
 }  
 }  
  
 // Get a valid password  
 while (!isValidPassword) {  
 try {  
 System.*out*.print("Enter a password: ");  
 password = scanner.nextLine();  
  
 int vowelCount = 0;  
 for (char c : password.toCharArray()) {  
 if ("AEIOUaeiou".indexOf(c) != -1) vowelCount++;  
 if (c == '4' || c == '6' || c == '9') {  
 throw new PasswordVerifyException("Password must not contain digits 4, 6, or 9.");  
 }  
 }  
  
 if (vowelCount != 3) {  
 throw new PasswordVerifyException("Password must contain exactly 3 vowels.");  
 }  
  
 isValidPassword = true; // Password is valid if no exception is thrown  
 } catch (PasswordVerifyException e) {  
 System.*out*.println(e.getMessage());  
 System.*out*.println("Please try again.");  
 }  
 }  
  
 // If both username and password are valid, create a PasswordVerify object and add it to the array  
 try {  
 PasswordVerify account = new PasswordVerify(username, password);  
 validAccounts[validCount++] = account;  
 System.*out*.println("Account created successfully!");  
 } catch (PasswordVerifyException e) {  
 System.*out*.println(e.getMessage());  
 }  
 }  
  
 // Display all valid accounts  
 System.*out*.println("All valid accounts created:");  
 for (PasswordVerify account : validAccounts) {  
 System.*out*.println(account);  
 }  
  
 scanner.close();  
 }  
}

**PasswordVerify.java**

package com.example.passwordvalidation;  
  
public class PasswordVerify {  
 private String username;  
 private String password;  
 // Constructor that takes in a username and password  
 public PasswordVerify(String username, String password) throws PasswordVerifyException {  
 this.username = username;  
 this.password = password;  
 // Perform the validation checks during object creation  
 if (!isValidUsername()) {  
 if (username.length() < 8) {  
 throw new PasswordVerifyException("Username must be at least 8 characters long.");  
 } else if (!username.matches(".\*[!@#$].\*")) {  
 throw new PasswordVerifyException("Username must contain at least one special character (!, @, #, $).");  
 }  
 }  
  
 if (!isValidPassword()) {  
 throw new PasswordVerifyException("Password must contain exactly 3 vowels and must not contain digits 4, 6, or 9.");  
 }  
 }  
 // Validate the username based on the custom rules  
 private boolean isValidUsername() {  
 return username.length() >= 8 && username.matches(".\*[!@#$].\*");  
 }  
  
 private boolean isValidPassword() {  
 int vowelCount = 0;  
 for (char c : password.toCharArray()) {  
 if ("AEIOUaeiou".indexOf(c) != -1) vowelCount++;  
 if (c == '4' || c == '6' || c == '9') return false;  
 }  
 return vowelCount == 3;  
 }  
  
 @Override  
 public String toString() {  
 return "Username: " + username + ", Password: " + password;  
 }  
}

**PasswordVerifyExcpetion.java**

package com.example.passwordvalidation;  
// Constructor that accepts a message and passes it to the parent Exception class  
public class PasswordVerifyException extends Exception {  
 public PasswordVerifyException(String message) {  
 super(message);// Call the parent class (Exception) constructor with a custom message  
  
 }  
}

**DivisionApplication.java**

package com.example.passwordvalidation;  
  
import java.util.Scanner;  
  
public class DivisionApplication {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 try {  
 System.*out*.print("Enter the first number (integer): ");  
 int num1 = scanner.nextInt();  
  
 System.*out*.print("Enter the second number (integer): ");  
 int num2 = scanner.nextInt();  
  
 double result = (double) num1 / num2;  
 System.*out*.printf("%d / %d = %.2f%n", num1, num2, result);  
  
 } catch (ArithmeticException e) {  
 System.*out*.println("Error: Division by zero is not allowed.");  
 } finally {  
 scanner.close();  
 }  
 }  
}