import java.security.SecureRandom;

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

import java.util.Scanner;

public class PasswordGenerator {

private static final String LOWERCASE = "abcdefghijklmnopqrstuvwxyz";

private static final String UPPERCASE = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

private static final String DIGITS = "0123456789";

private static final String SPECIALS = "!@#$%^&\*()-\_=+[]{}|;:,.<>?";

private static final SecureRandom random = new SecureRandom();

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter desired password length (4-32): ");

int length = scanner.nextInt();

if (length < 4 || length > 32) {

System.out.println("Password length must be between 4 and 32.");

return;

}

System.out.print("Include uppercase letters? (y/n): ");

boolean includeUppercase = scanner.next().equalsIgnoreCase("y");

System.out.print("Include lowercase letters? (y/n): ");

boolean includeLowercase = scanner.next().equalsIgnoreCase("y");

System.out.print("Include digits? (y/n): ");

boolean includeDigits = scanner.next().equalsIgnoreCase("y");

System.out.print("Include special characters? (y/n): ");

boolean includeSpecials = scanner.next().equalsIgnoreCase("y");

String password = generatePassword(length, includeUppercase, includeLowercase, includeDigits, includeSpecials);

System.out.println("Generated Password: " + password);

String strength = calculateStrength(password);

System.out.println("Password Strength: " + strength);

}

public static String generatePassword(int length, boolean includeUppercase, boolean includeLowercase,

boolean includeDigits, boolean includeSpecials) {

StringBuilder password = new StringBuilder();

List<Character> characterPool = new ArrayList<>();

if (includeUppercase) {

characterPool.addAll(UPPERCASE.chars().mapToObj(c -> (char) c).toList());

password.append(randomChar(UPPERCASE));

}

if (includeLowercase) {

characterPool.addAll(LOWERCASE.chars().mapToObj(c -> (char) c).toList());

password.append(randomChar(LOWERCASE));

}

if (includeDigits) {

characterPool.addAll(DIGITS.chars().mapToObj(c -> (char) c).toList());

password.append(randomChar(DIGITS));

}

if (includeSpecials) {

characterPool.addAll(SPECIALS.chars().mapToObj(c -> (char) c).toList());

password.append(randomChar(SPECIALS));

}

// Fill the rest of the password length with random characters from the pool

for (int i = password.length(); i < length; i++) {

password.append(randomChar(characterPool));

}

// Shuffle the password to ensure randomness

List<Character> passwordList = new ArrayList<>();

for (char c : password.toString().toCharArray()) {

passwordList.add(c);

}

Collections.shuffle(passwordList);

StringBuilder finalPassword = new StringBuilder();

for (char c : passwordList) {

finalPassword.append(c);

}

return finalPassword.toString();

}

private static char randomChar(String chars) {

return chars.charAt(random.nextInt(chars.length()));

}

private static char randomChar(List<Character> chars) {

return chars.get(random.nextInt(chars.size()));

}

public static String calculateStrength(String password) {

int score = 0;

if (password.length() >= 12) score += 2;

else if (password.length() >= 8) score += 1;

if (password.chars().anyMatch(Character::isUpperCase)) score += 1;

if (password.chars().anyMatch(Character::isLowerCase)) score += 1;

if (password.chars().anyMatch(Character::isDigit)) score += 1;

if (password.chars().anyMatch(c -> SPECIALS.indexOf(c) >= 0)) score += 1;

switch (score) {

case 5:

return "Very Strong";

case 4:

return "Strong";

case 3:

return "Good";

case 2:

return "Weak";

default:

return "Very Weak";

}

}

}