CSCI 2916 Lab 10 – Week 10

Lab: Password Validation

A **password** is a word or string of characters used for user authentication to prove identity or access approval to gain access to a resource (example: an access code is a type of password), which is to be kept secret from those not allowed access.

The use of passwords is known to be ancient. Sentries would challenge those wishing to enter an area or approaching it to supply a password or *watchword*, and would only allow a person or group to pass if they knew the password. In modern times, user names and passwords are commonly used by people during a log in process that controls access to protected computer operating systems, mobile phones, cable TV decoders, automated teller machines (ATMs), etc. A typical computer user has passwords for many purposes: logging into accounts, retrieving e-mail, accessing applications, databases, networks, web sites, and even reading the morning newspaper online.

Despite the name, there is no need for passwords to be actual words; indeed passwords which are not actual words may be harder to guess, a desirable property. Most organizations specify a password policy that sets requirements for the composition and usage of passwords, typically dictating minimum length, required categories (e.g. upper and lower case, numbers, and special characters), prohibited elements (e.g. own name, date of birth, address, telephone number). (https://en.wikipedia.org/wiki/Password)

So, today we will be writing a program to check to see if a password entered is valid given a set of rules
The program should ask the user to enter a password and pass it to a **method** that checks to see if it meets the rules (below) it will output valid password and if it doesn't meet the rules it should output invalid password.
Rules for password:
A password must have at least eight characters
A password can contain letters, digits and special symbols(!#$%&-_)
A password must contain at least two digits
A password must contain at least one special symbol(!#$%&-_)

Sample Output example (computer prompts are in bold, user input is in italic):
Enter a string for password: *r3t5th*
Invalid Password

Enter a string for password: *abadabado12*
Invalid Password

Enter a string for password: *zeb$78345*
Valid Password

Extra Credit: include 2 more rules in your password validator program. For example, there must be one uppercase letter and the password can't be over 20 characters.
BEFORE YOU START WRITING CODE . . . On the back of this paper, sketch out:

- How will you capture the password?
- How will you pass the password to the method?
- What will your method contain?
- In the method how will you go through the characters to determine whether they are letters or numbers?
- How will you return the responses (valid password or invalid password)?

Guidelines for a good program:

- The program works, following the dialog and rules above.
- The code is clear and understandable:
    - Properly indented
    - Representative variable names
    - Blank lines separate logical sections of code
    - Appropriate comments included
    - Preamble documentation is included
    - Review program assignment rubric