EECS 2110
Professor
Brent Nowlin

Computer Architecture and Organization

THE UNIVERSITY OF TOLEDO 1872

Fun With Passwords (if they could ever be fun)

In this project you are going to write an assembly language program to check a user-supplied password for strength. Most modern definitions of a strong password requires that the password has at least one character from each of the following sets: a lower case character a-z; an uppercase character A-Z; a number 0-9; and a character from the set of special characters: ?@*()+{} &'_ ^ [ ] \ |

In addition to the requirement that the password contain at least two characters from each of the four groups, the password must be 10 to 22 characters in length. You will also check to ensure that a newly-accepted password isn't identical to the last 3 passwords that were acceptable. If the user attempts to enter an identical password to one of the 3 previous passwords, the password fails the check and the user has to enter a new password that isn't identical to one of the last 3 passwords, is 10-22 characters in length, and contains at least one character from the aforementioned types.

Your program has the following requirements:
1. Obtain a password from the user. As the user is typing the password, place a "*" on the screen for each character typed.
2. Check the password to verify it meets the criteria as described above.
3. If the password meets the criteria, And is not identical to one of the three previously accepted passwords, output the message "password meets requirements" on the screen. If not, state the requirement that is not met: no lowercase character, no uppercase character, no lowercase character, no numeric character, no special character, too short, too long, or is identical to one of the three previous passwords. Note that it is possible for more than one criteria to not be met.
4. Ask the user if they wish to run the program again (accept either a "Y" or "y" for a possitive response, "N" or "n" for a negative response). If yes, then loop back to step 1. If not, exit. If the user response is anything except "Y", "y", "N", or "n", ask the user again. Lastly, if the user chooses to exit the program, output the last password entered to the screen, (whether it was valid or not) with appropriate prompts. Note: you would never do this in real life!!!

Example interaction would be:

Please enter a password: *********

Your password is lacking a special character
Your password is lacking an uppercase character

Do you wish to try again (y/n)? Y
Please enter a password: *********

EECS 2110

Professor
Brent Nowlin

Computer Architecture and Organization

Your password is successful!

Do you wish to try again (y/n)? n
Your last password was < output the last password entered>
<exit>

Deliverables: submit your source code via email by April 16, including screen shots showing successful assembly, linking, and execution of your program.