PASSWORD STRENGTH CHECKER
DEVELOPER/TECHNICAL GUIDE

Created By

- Vinit Kumar
- Aman Raj
- Lavish Gulati

CONTENTS

1 INTRODUCTION

The Password Strength Checker application allows users a simple interface to check the strength of their passwords. The users can also save their passwords and view their last saved password. The application also suggests strong passwords which the user can use.

This document will provide instructions for using the application and its various features.

2 SYSTEM REQUIREMENTS

- Windows 7 or higher
- Visual Studio 2013
  - 1.6 GHz or faster processor
  - 1 GB of RAM (1.5 GB if running on a virtual machine)
  - 20 GB of available hard disk space
  - 5400 RPM hard disk drive
  - DirectX 9-capable video card that runs at 1024 x 768 or higher display resolution
  - KB2883200 (available through Windows Update) is required

3 COMMON WORDS CHECKER

- We have created a dictionary.txt to store some of the common words used. We have placed this file in a newly created “Media” folder.
- We are reading that file line by line and storing it in a dictionary “wordDict” with the read word as key and ‘1’ as value.
- When the user inputs a password, we generate all possible substrings of the password and check for its presence as key in the dictionary “wordDict”.
- If the password exactly matches with any word, we gave score “zero” with warning to the user not to use common words.
- If any substring of password is matched with any word in dictionary, we halve the score of the password with a warning to the user not to use common words in password.

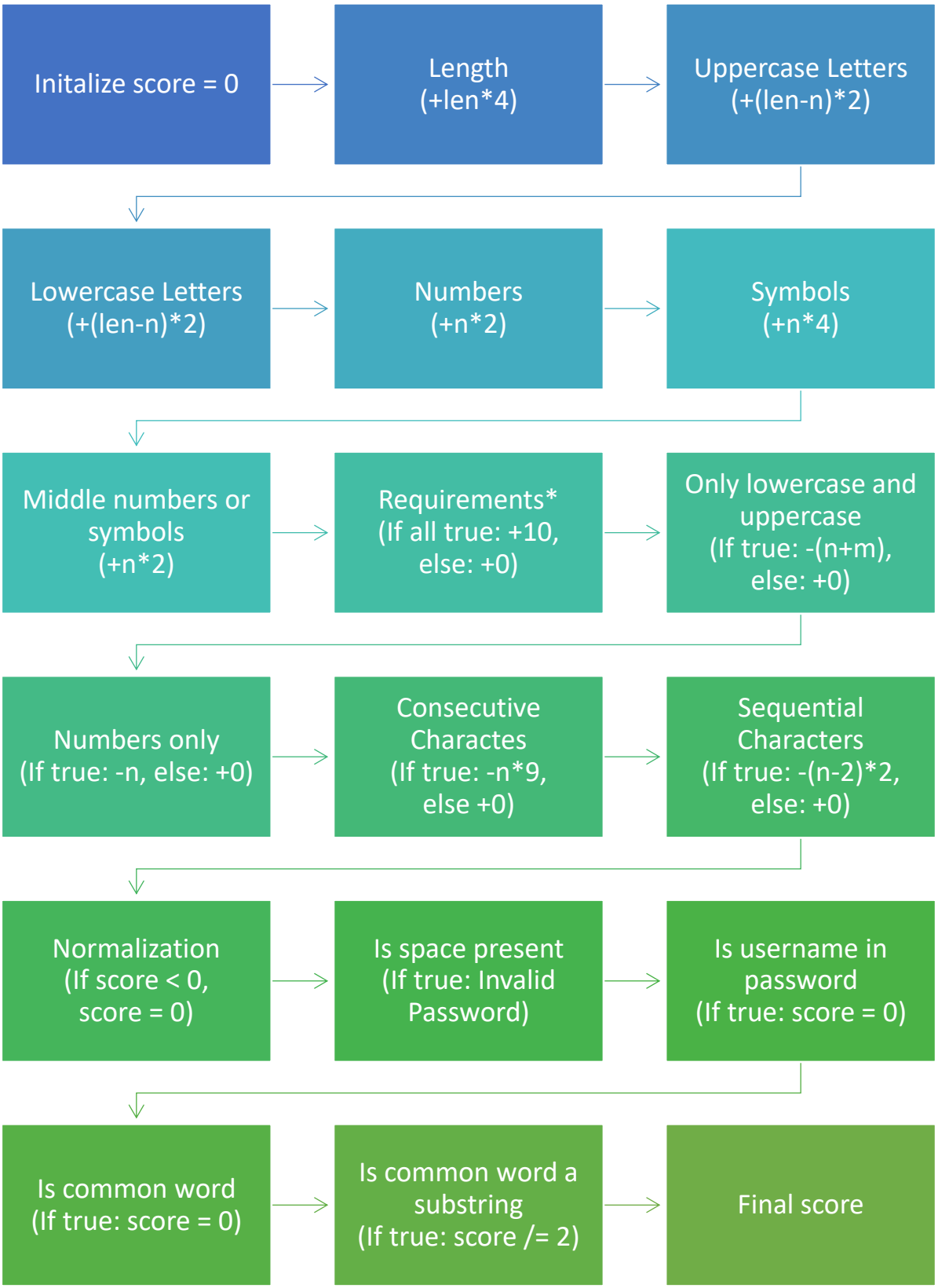
4 SHOW LAST SAVED PASSWORD

- We have used a dictionary “dictSavedPasswords” to store the last saved password of each user.
- Passwords are stored in the dictionary with username as key and password as value.
- Whenever user saves a password, we look if there is already any value with that username as key. If there exists, we simply replace the password with the new one, else we store the password as a new one.
- We have displayed the last saved password in “Form2” by passing the present dictionary “dictSavedPasswords” with the username to “Form2” by Public declaration of variables in Form2.

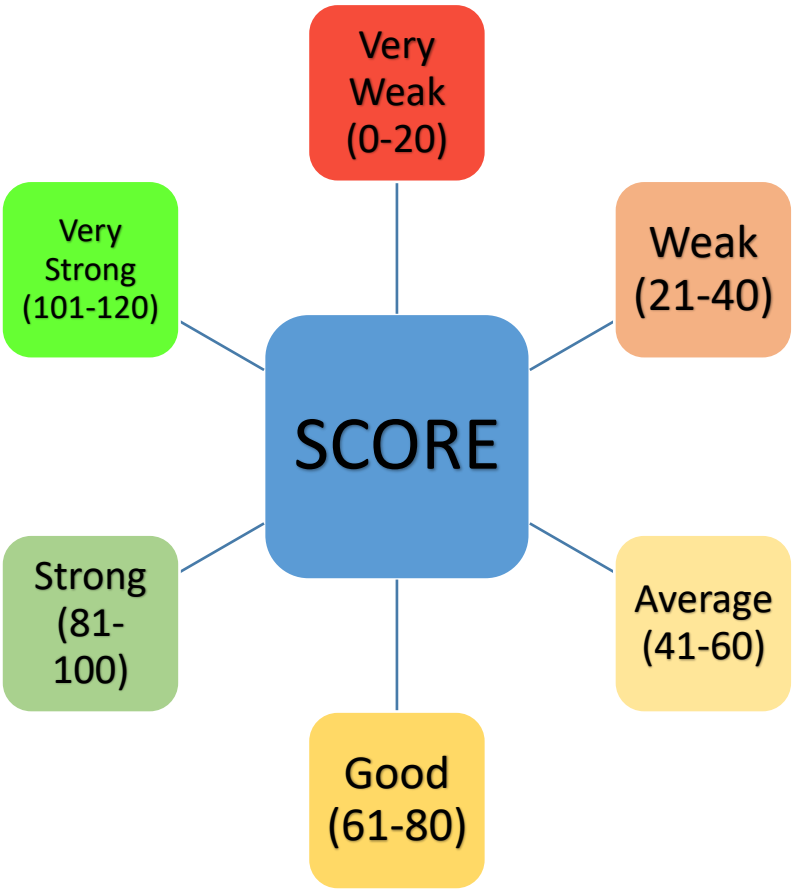
5 SUGGEST PASSWORD

- We have generated a random 12 characters long string to suggest a strong password to the user.
- Randomize() is used to initialize the random-number generator. The value returned by the system timer is used as the new seed value.
- We have used “Rnd() * n” and typecasting it to integer gives a random integer between 0 to n-1.
- Asc() function returns the ASCII value of the character.
- We ran a loop to do the following tasks 3 times:
    1. Get an Uppercase letter randomly from “A” to “Z”.
    2. Get a lowercase letter randomly from “a” to “z”.
    3. Get a number randomly from “0” to “9”.
    4. Get a symbol randomly from all available and valid symbols.
- User can simply click on “Use This” button to get that suggested password copied to input password textbox “textPassword”.

6 SCORE CALCULATION ALGORITHM



7 SCORE CLASSIFICATION CHART



8 REFERENCES

8.1 PASSWORD METER

We took reference from this website to implement our algorithm of score calculation to check strength of the input password.

8.2 CODE PROJECT

We took reference from this to implement the section in we resize the controls present in a form when the size of form is changed.

9 FUTURE ADDITIONS

9.1 TIME ESTIMATION

In future versions, a feature to estimate the time needed to crack the input password can be included. For example – an eight characters long password takes few days to be cracked, a ten characters long password takes weeks and a twelve characters long password takes months to crack by brute force. Due to lack of proper algorithm and time, we were unable to include this feature this time.

9.2 A BETTER RANDOMIZED PASSWORD

This version of Password Strength Checker suggests a strong password, but it has some limitations like it always suggests a password that is twelve characters long. Although it always selects a random character, the order of appearance of uppercase, lowercase, symbols and numbers in password is fixed.

10 CONTRIBUTIONS

10.1 AMAN RAJ

- Show last saved password feature
- Suggest a password feature

10.2 LAVISH GULATI

- Algorithm for score calculation and classification
- Interface Improvement

10.3 VINIT KUMAR

- Use of dictionary for checking common words
- Interface Improvement