IB Computer Science

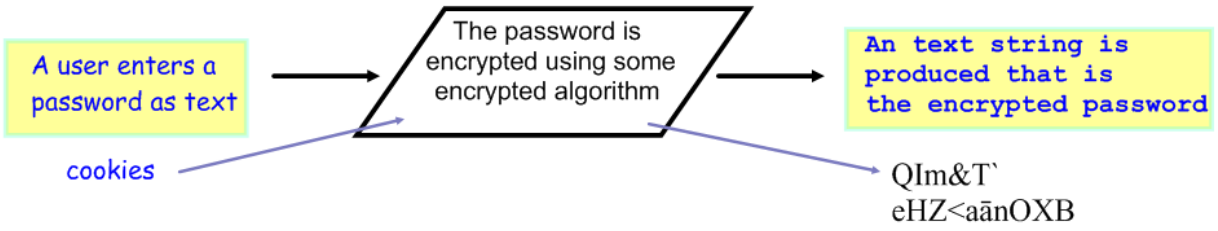
Student_____

Web Sciences

Date: _____

Background

A brute-force password attack (or password cracking) is a method for generating and testing all possible combinations of a password. Sometimes this may imply an effort to gain unauthorized access to a system, and sometimes it could in a password recovery process.
Consider the following process that shows how a password can be stored:



When a password is chosen using 6 lower case letters, there are 266 possible passwords that can be generated (because there can be 26 letters chosen as the first character, 26 letters can chosen as the second character…and for 6 character positions that can be 26*26*26*26*26*26.

Problem statement

Bill and Ted are building a system and they want it to be secure, so they are creating a password policy. Bill says people hate passwords, so having a minimum password length of 4 should be good enough so that people can remember their password. Ted thinks it should be longer so people don't pick easy passwords to crack, like 1234 or dogs.

Calculate the number of different passwords that can be generated using each of the following requirements:

1. A password length of 4 characters, where a character can be any upper or lower case letters

2. A password length of 4 characters, where a character can be any upper or lower case letter or a number

3. A password of length 6 characters, where a character can be any upper or lower case letter or a number

4. A password of length 6 characters, where a character can be any upper or lower case letter or a number, or one of the 10 special characters `#@$.,&!~?*`

5. A password of length 8 characters, where a character can be any upper or lower case letter or a number, or one of the 10 special characters `#@$.,&!~?*`

6. A password whose length is between 4 and 8 characters (including 4 and 8) where a character can be any upper or lower case letter or a number, or one of the 10 special characters `#@$.,&!~?*`

IB Computer Science

Student_____

Web Sciences

Date: _____

Writing Algorithms

1. Write an algorithm, using pseudo code or a flow chart, that can generate all possible password combinations A password of length 6 characters, where a character can be any upper or lower case letter or a number [do this on separate paper]

2. Bill has forgotten his password. He remembers that it was 6 characters long, that it had upper and lower case letters, and it had exactly one number from 0 to 9. When the password was encrypted the clear-text password was passed to a method named HidePassword, which accepted a String variable, and returned a String variable. Although he can not find his original password he was clever enough to find his encrypted password in a file, and his encrypted password is L*8upKM*mm*
   Write an algorithm using pseudo code that will find Bill's password. Your algorithm may call the method HidePassword that accepts a String parameter and returns a String value, and you do not need to worry about how HidePassword works internally.

3. Extension: Add a counter to your algorithm from problem 2 to see how many calls to HidePassword must be made before the password is recovered. Output this count along with Bill's password. Can you think of any changes to your algorithm that could reduce the number of times that HidePassword might have to be called?