Statement and explanation of the problem

* **Statement**

A strong password is based on three conditions: length (between 6 and 20 characters), at least one lowercase letter/uppercase letter/digit, NOT containing 3 identical characters in a row.

* **Analysis and assumptions**

The main goal is to turn the input string *s* into a strong password. In order to do that, we must think of the MINIMUM changes that would preferably fix as many problems as possible.

To begin with, I’ve noticed 3 problems that we have to deal with:

1. Length: if length is <6 => INSERT, else if length is >20 => DELETE, both of them applied on any position.
2. One lowercase/uppercase letter, digit missing: either the length problem is fixed or not, we should do INSERT or REPLACE with the missing ones, applied on any position.
3. Identical characters: this one implies all the possible changes that could be made, but at this point the position matters.

The third problem consists of different combinations of changes, considering also the combinations of the other 2 problems. Having in mind that if we fix 1 or 2 problems might also fix the other one, I considered the following problem combinations:

* **1st & 2nd problem** depends on the length of the string

If the length is 5 or 19 and we have a missing character => INSERT.

If the length is checked and we have a missing character => REPLACE.

I supposed that fixing this one should be a priority, since it can also repair the 3rd one.

* **1st & 3rd problem**

If the length is 5 or 19 => REPLACE (the repeating character) and INSERT (a new one to complete).

If the length is checked => REPLACE

* **2nd & 3rd problem**

REPLACE the repeating character with the missing one.

After this analysis, I’ve noticed that REPLACE is the most powerful change and DELETE is the weakest.

*Assumptions*: Since I didn’t know if the MINIMUM change means the number of changes necessary or to specify each change needed (+number of them) or just the changes involved, I presumed that we have to return the number of changes.

for(i=0;i<strlen(s);i++)

{

if(islower(s[i]))

lower\_ver=0;

if(isupper(s[i]))

upper\_ver=0;

if(isdigit(s[i]))

digit\_ver=0;

}

changes=lower\_ver+upper\_ver+digit\_verl;

if(strlen(s)<6)

{

if(6-(strlen(s)+changes)>0)

result=6-strlen(s);

}

else if(strlen(s)-20>0)

result=strlen(s)-20;

for(i=0;i<strlen(s)-1;i++)

if(s[i]==s[i+1])

rep[i]=1;

else rep[i]=0;