Lab 1 – Option B

The given text file will be opened and converted into a list that is used to extract every salt value and hashed password. To get the hashed passwords and salt values, the *get\_password* and *get\_salt* method goes through each line and creates a new list containing the hash passwords or salt values – depending on which method is currently being used. These lists containing the passwords and salt values are stored for further use after generating a series of numbers.

A brute force calls for generating permutations with and only numbers. These numbers from 0 to 9 are in a list that is given to the *gen* method as a parameter. The *gen* method takes an empty list that will hold all permutations, the list of numbers we’re using, an integer that serves a minimum length to the permutations, and an integer similar to the former but is for a maximum length.

The first base case checks when we’ve reached the maximum length which adds the current permutation to the list of all permutation and returns it. The second base case increments the minimum limitation by recursively calling the *gen* method with the minimum integer incremented and adding the current permutation list to keep track of it. A for loop goes through each element in the list of numbers we can use and recursively makes a permutation for every element. The final list of permutations is returned (which can take a while) and is processed to turn every element into a string in the *to\_string* method.

The *find\_password* method takes the list of all permutations of numbers, the list of all salt values, and the list of all hashed passwords. To keep track of the current user an index is created with the value of zero and will be used to access our hashed passwords from its list. The nested for loop goes through every element in the list of number permutation and combines them with every salt value into a string (this process may take a while). We run every concatenate string through the *hash\_with\_sha256* method and compare the results with the current hashed password. We compare each concatenate string until we find a match or have compared every possible string. If a match, we print the password for the user.