For tasks one until three I implemented simple algorithms that cracked the passwords in a very short amount of time. For task 1, used the “product” method of “itertools” to generate all possible strings out of the possible password characters, which in this case were a-z and 1-9. The time it takes to generate these strings increases exponentially with the length of the strings, so I programmed the function to stop when the number of cracked passwords matches the number of input passwords, so no strings are generated unnecessarily. This greatly decreased run time. For the second task, I imported the dictionary using urlopen, split the dictionary into lines (each line was a different password), hashed the passwords, and compared them with the input passwords. For task three, I implemented the same method, but I added the salts to the end of the passwords before hashing them, and then removed them after by removing the last 8 characters of the string.