PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100 Ft. Road, BSK III Stage, Bengaluru – 560 085

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Course Title: Problem Solving with C Laboratory		
Course code: UE19CS152		
Semester : II Sem	**Section:** G	**Team Id:** 4
SRN: PES1UG19CS563	**Name:** Suhas Thalanki	
SRN: PES1UG19CS432	**Name:** Sanjiv Sridhar	
SRN: PES1UG19CS003	**Name:** Aaditya Shah	
SRN: PES1UG19CS127	**Name:** Chinmay Kulkarni	

PROJECT REPORT

Problem Statement: Encrypting the contents of a file using a user-provided keyword and password protecting the file after either generating a password or classifying a user-provided password

Description:

- **Encryption using Caesar mechanism**
  - Encryption is done by taking a keyword and plaintext (text to be encrypted).
  - Each letter of the keyword is converted to a number where A is 0, B is 1 and so on (**not** case-sensitive).
  - Each letter of plaintext is then encrypted using Caesar encryption by taking an individual letter from keyword as key and each character in the plaintext has a different key.
  - In case the length of the keyword is lesser than the length of the plain text, it goes back to the first letter of the keyword and continues encrypting.
  - Caesar encryption is shifting a particular letter by x places. For example, if the key is 4 and the letter is A, the final letter will be E.
  - In case the key is greater than 26 (total no. of letters), the program corrects this as moving a letter by 28 places (for example) is the same as moving a letter by 2 places.
  - This encryption is applied to a file containing text and the contents of the file are encrypted.
- **Password Generation**
  - Password generation is done by taking a randomly generated string containing upper-case letters, lower-case letters, numbers and special characters.
  - The length of the password is randomly generated.

    - Every fourth letter is of the same type, i.e., upper-case, lower-case, etc.
    - Each letter is randomly chosen by finding the modulus of a randomly-generated number and a number specific to the type of character (26 for letters, 10 for numbers, etc.).
    - At the end, this password is then finally shuffled randomly.
- **Password Classification**
    - Password classification is done on the number of upper-case letters, lower-case letters, numbers and special characters.
    - The more type of characters and the more random they are, the higher the strength of the password.
    - The strength is assessed by counting the number of characters of each type.
- **Decryption**
    - The decryption is done by taking the same keyword used in the encryption algorithm.
    - Instead of moving ahead by x places as in the encryption algorithm, it moves back x places giving back the original content of the text.

C-Concepts Used

- **Functions:**
    - Isalpha from ctype.h
    - Strlen from string.h
    - rand from stdlib.h
    - srand from stdlib.h
    - time() from time.h
    - RAND_MAX definition from stdlib.h
    - Fgetc and fputc from stdlib.h
    - Fflush() from stdio.h
- Usage of multiple files (.c and .h files)
- Pointers and arrays
- Dynamic memory allocation
- Random generation
- File handling

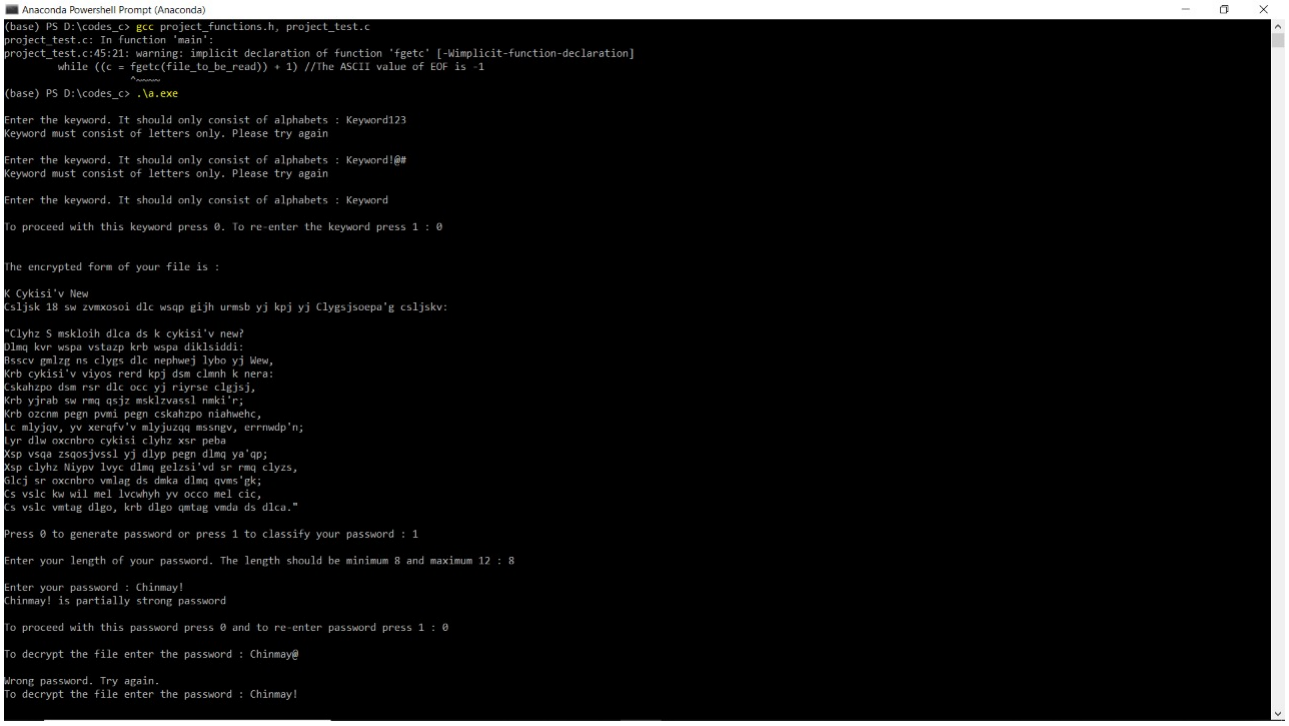
Learning Outcome

- Learning how encryption and decryption works.
- Learning the simple method of how passwords are classified based on the types of characters it has in it.
- Used the above knowledge to generate strong passwords.
- Learned how to read to and write from files.
- Learned how to use random generating functions to get desired values.
- Learning about new libraries like time.h and ctype.h.

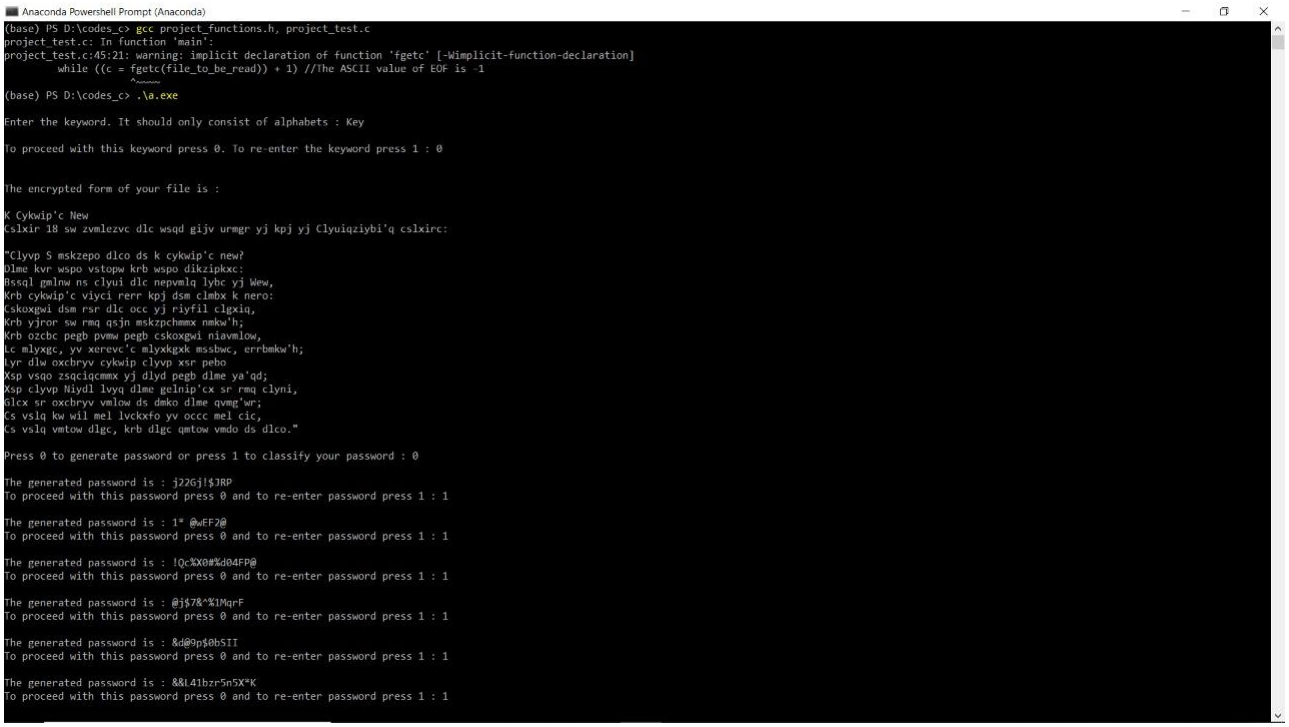
Output Screenshots:

This is the file which holds the content to be encrypted.

Now, running the program and allowing the user to enter their own password gives the following output:



As shown in the next set of output, the user chooses the option to generate a password:



When the user is allowed to choose their own password, the classifier comes into play as it checks the types of characters in the password and then based on that, it determines whether the password is a strong one or a weak one.
The password generator generates a strong password that holds all kinds of characters.

The next use-case shows how the user is blocked on typing in the wrong password four times:

Explanation:
There is one file at the beginning which holds the content that has to be encrypted. On running the program, a new file is created which holds the encrypted form of the data and then if the user wishes to decrypt the file by entering the right password, another file is created which holds the decrypted data.

Name and Signature of the Faculty