Firstly, I designed an easy menu and coded the menu in a switch case structure by the way that each operation can go to its own function.

SEARCH

Firstly, I asked the user to enter a value f, then I opened the 100000.txt file with the FILE * file command and connected the value in each line to the "pass" variable with the "scan" command in the "for" loop. I had it checked whether it is equal to the value entered by the user. I added a screen message stating that the value entered by the user is in the text file in case of equality. In case of not being equal, I added a message that the value entered by the user is not found in the text file.

SORT

Secondly, I opened the 10000.txt file with the command "FILE * Myfile" and transferred the array named "DATAARR". The reason I use arrays is that I wanted to sort the data in it faster and avoid it taking up space in the cache, then I used the "INSERTİONSORT" algorithm to sort each data in the array by size. After sorting the data, I used 2 separate loops for the first 10 and the last 10 data with the "For" loop and printed the sequences on the screen.

HASH

First, I opened the 10000.txt file and transferred the data in it to the array named "PassHashTableArr". Then I conditioned the directory whether it is empty to check the pull. I have transferred every element in the array to Hash Table. I used the "Linear Probing" method to avoid conflicts after the transfer was provided. This method ensures that the conflicting data is passed to the box next to the data it collides with to prevent each data collision. If the box next to it is not empty, it checks 1 side box until it finds an empty box. After all, data has been transferred, each data has a hash value. Then I asked the user to enter a value and I created a condition that ensures whether the entered value and the Hash value of the data in the Hash Table are the same.

LINK LIST

I transferred the data from 1000.txt to a single linked list and evaluated each character of the data I pulled. These are capitalization, use of numbers, and special characters. I added a separate scoring system for each condition; We have added 1 point for upper / lower case letters, 2 points for number usage, 3 points for special characters, and 10 bonus points for data that meet all conditions. After scoring, I used the "Insertion Shorts" sorting method for the data in the single linked list. Finally, I printed the data with the highest and lowest scores on the screen.