Всего существует 4 вида поиска данных:

1) прямой доступ - просто по объекту вычисляется адрес, где этот искомый объект находится - выч. сложность - О(1);

2) последовательный поиск сравнением с объектом - О(n);

3) двоичный поиск - О(log2(n));

4) поиск с использованием хеш-функций.

Последовательный поиск для больших файлов требует огромного времени, а прямой - обычно недоступен.

Суть проблемы, что прямой доступ требует, чтобы все данные располагались в пространстве в соответствии с ключами, но, как правило, данных мало, а ключей - много. И мы просто не можем хранить данные таким образом. Например в Вашем случае, если слова имеют длину до 10 символов, то для их хранения понадобится 33 в 10 степени ячеек.

Двоичный поиск для строк также слишком медленный...

Хеш-функции позволяют использовать приближение к прямому доступу. С помощью хеш-функции по ключу получаем примерное местоположение данных. Возможны коллизии, когда для разных ключей получаем одно и то же место...

Ваша задача построить файл, который содержит, например, двухбуквенные ключи и позицию соответствующих слов в файле F. Затем сортировать файл по возрастанию ключей. Файл затем используют для поиска слов. В нем можно искать слова, например, двоичным поиском, а среди группы слов - последовательно.

Можно использовать дополнительный массив на 33 буквы, который будем содержать адрес начала сочетаний с этой буквой в индексном файле... Это уже дерево...

Можно поискать другую функцию, например - сумма кодов букв по некоторому модулю...

Для всех ваших строк надо определить их место в файле. Затем строят индексный файл из записей, включающих посчитанный с помощью хеш-функции ключ и смещение. Записи в индексном файле сортируют по возрастанию ключей. Теперь, чтобы найти запись в первом файле, надо найти группу совпадающих с искомым ключей во втором (используя двоичный поиск), а затем уже, применяя смещение, найти нужные данные.

Еще здесь почитайте

<http://citforum.ru/programming/theory/sorting/sorting1.shtml>

И в Ин-те методы поиска поищите :-)...