

# Malnad College of Engineering

(An Autonomous Institution under Visvesvaraya Technological University, Belagavi)

Hassan, Karnataka, India – 573202



**Course Title: Data Structures**

**Course Code: 23AI304**

**Project Based Learning**

**Title: “Implementation of English dictionary using appropriate data structure.”**

**Submitted by:**

Pranathi R	4MC23CI038
Inchara	4MC23CI019
Varshini Mohan	4MC23CI058
Varsha Shashikumar	4MC23CI057

**Submitted to:**

**Dr. Balaji Prabhu B V**

Associate professor and HOD

Dept. of CSE (AI and ML)



**Department of Computer Science and Engineering  
(Artificial Intelligence and Machine Learning)**

**Malnad College of Engineering**

**PB# 21, Hassan, Karnataka, India – 573 202**

# CONTENT

## **1. Problem Statement**

## **2. Data Set**

## **3. Data Structure**

### 3.1 Array Of Structures

## **4. Steps Followed In The Program**

### 4.1 Initialization

### 4.2 User Menu

### 4.3 User Operations(Functions)

### 4.4 File Handling

## **5. Pseudo Code**

### 5.1 Search Word

### 5.2 Insert Word

### 5.3 Delete Word

### 5.4 Main Function

## **6. Input And Output**

### 6.1 Search Word Operation

### 6.2 Insert Word Operation:

### 6.3 Delete Word Operation

## **7. Final Analysis**

# 1. Problem Statement

The list of 2500 English words along with their meanings is given in a text file. Implement an English dictionary using appropriate data structure. The operations to be enabled are

1. Insert Words:
2. Allow the insertion of new words along with their meanings. Use an appropriate data structure (e.g., a dictionary or hash table) to store word-meaning pairs.
3. Delete Words:

Enable deletion of words from the dictionary. When a word is deleted, its meaning should also be removed.

4. Search Words:

Implement a search function to check if a word exists in the dictionary and display its meaning if found.

# 2. Data Set

```
1 apple,A round fruit with red or green skin and a whitish interior typically eaten raw
2 airplane,A powered flying vehicle with fixed wings and a weight greater than that of the air it displaces
3 banana,A long curved fruit with yellow skin and soft sweet white flesh inside
4 balloon,A flexible bag that can be inflated with air or gas typically made of rubber or a similar material
5 cherry,A small round fruit that is typically red or black with a pit in the center
6 cat,A small domesticated carnivorous mammal with a soft coat a short snout and retractile claws
7 dog,A domesticated carnivorous mammal with a bark typically kept as a pet or for work
8 doghouse,A small shelter or enclosure for a dog
9 elephant,A large herbivorous mammal with a trunk native to Africa and Asia
10 engine,A machine for converting energy into mechanical force or motion often used to power vehicles
11 flower,The reproductive structure in flowering plants often colorful and fragrant
12 flowerpot,A container used for growing plants typically made of clay plastic or ceramic
13 grape,A small round smooth-skinned fruit that typically grows in clusters and is used for making wine
14 guitar,A musical instrument with six strings played by plucking or strumming
15 house,A building for human habitation typically consisting of rooms and a roof
16 honey,A sweet sticky substance produced by bees from flower nectar and used as a sweetener
17 internet,A global computer network providing a variety of information and communication facilities
18 ice,Frozen water a solid form of H2O typically used to cool drinks or for various cooling applications
19 jacket,A piece of clothing worn on the upper body typically with sleeves for warmth
20 jungle,A dense tropical forest with a lot of vegetation and wildlife
21 kangaroo,A large marsupial from Australia known for its powerful hind legs and jumping ability
22 key,A small device used to open locks or start machines
23 laptop,A portable computer that can be used on one's lap or a surface
24 lemon,A sour yellow fruit that is used in cooking and as a flavoring or garnish
```

A dataset consisting of words and their corresponding meanings or definitions. The structure of the data appears to follow a word-definition pair format, where each line contains:

- Word: The term being defined (e.g., *apple*, *airplane*, *dog*).
- Definition: A brief explanation or description of the word's meaning.

## 3. Data Structures

### 2.1 Array of Structures

The program uses an **array of structures** to store words and their meanings in memory. Each structure contains two fields:

- word: A string representing the word.
- meaning: A string representing the meaning of the word.

It stores up to 2500 word-meaning pairs. Each element of the array represents a unique entry in the dictionary.

#### Defination Of Sturcture:

```
struct Dictionary
{
    char word[100]; // To store the word
    char meaning[250]; // To store the meaning
}dictionary[2500];
```

## 4. Steps Followed in the Program

### 4.1 Initialization:

The program begins by loading existing data from the file dictionary.csv using the files() function. Each line of the file contains a word and its meaning, separated by a comma. The data is read into the dictionary array, and countwords is updated.

### 4.2 User Menu:

 The program displays a menu with four options:

1. Search for a word.
2. Insert a new word.
3. Delete an existing word.
4. Exit the program.

### 4.3 User Operations(Functions):

#### i. Search Word:

Accepts a word from the user.

Performs a **linear search** in the dictionary array.

Displays word & its meaning if found or an appropriate message if not.

**ii. Insert Word:**

Allows user to input a new word and its meaning.

Ensures the word contains only alphabetic characters. Checks if the word already exists in the dictionary.

Appends the word to both file (dictionary.csv) & the dictionary array.

**iii. Delete Word:**

Allows user to input a word to delete.

Reads from the file line by line and writes all entries except the one to delete into a temporary file.

Replaces the original file with the temporary file.

Updates the in-memory array by reloading the data.

**iv. Exit:** Terminates the program.

## 4.4 File Handling

The program uses dictionary.csv as a persistent storage mechanism.

**File Operations:**

Opened in read mode to load data (files()).

Opened in append mode to add new words (insertword()).

Temporary files are created during deletion to rewrite the data without the deleted word.

## 5. Pseudo Code

### 5.1 Search Word:

```
function searchWord():  
    input word  
    for each entry in dictionary:  
        if strcasecmp(entry.word, word) == 0:  
            print "Word found: ", entry.word, entry.meaning  
    return
```

**Function Search word():** This defines the search Word function for searching a word in the dictionary.

**Input word:** the user to input the word they want to search for in the dictionary. The word is stored in a variable ( word).

**For each entry in dictionary:** This represents a loop that iterates through all entries in the dictionary array. Each entry corresponds to an element of the dictionary array ( Dictionary struct containing word and meaning).

**if strcmp(entry. Word, word) == 0:** Compares the input word with the current entry's word in the dictionary. **strcmp** is a string comparison function. It returns 0 if the two strings are equal, ignoring case differences.

**print "Word found: ", entry.word, entry.meaning:** If a match is found:The function prints the word and its corresponding meaning.

**Return: Exits the function immediately after finding the word and displaying its details.**

**print "Word not found":** If the loop completes without finding a match, this statement is executed. It indicates that the word is not present in the dictionary.

## 5.2 Insert Word:

```
function insertWord():  
input word  
validate word (alphabetic and non-duplicate)  
if word exists:  
print "Word already exists"  
return  
input meaning  
append word, meaning to file  
add word, meaning to dictionary array  
increment countwords
```

**function insertWord():**This defines the insertWord function, which is responsible for adding a new word and its meaning to the dictionary.

**input word:**Prompts the user to enter a new word to add to the dictionary.The word is stored in a variable ( word).

**validate word (alphabetic and non-duplicate):**Ensures the input word is valid before adding it to the dictionary. This involves two checks tha if input is alphabets and also ensures the word contains only alphabetic characters. Duplicate Check that ensures **the word** is not already in the dictionary.Loops through the dictionary to check if the word already exists.

**if word exists:** If the word is found during the duplicate check:Prints a message indicating the word already exists.Terminates the function early using return.

**input meaning:** If the word is valid and does not exist, prompts the user to input its meaning.

**append word, meaning to file:** Appends the new word and its meaning to the dictionary.csv file. This ensures the new entry is saved for future use.

**add word, meaning to dictionary array** Updates the in-memory dictionary array by adding the new word and its meaning. This allows the program to work with the new entry without needing to reload the file.

**increment countwords:** Increments the countwords variable to reflect the addition of the new word to the dictionary.

### 5.3 Delete Word:

```
function deleteWord():  
    input word  
    validate word  
    create temporary file  
    for each entry in file:  
        if entry.word != word:  
            write entry to temporary file  
    else:  
        mark as found  
    if word found:  
        replace original file with temporary file  
        reload dictionary array  
    else:  
        print "Word not found"
```

**function deleteWord():** This defines the deleteWord function, which handles removing a word and its meaning from the dictionary.

**input word:** Prompts the user to input the word they want to delete from the dictionary.

**validate word:** Ensures the input word is valid by checking for Alphabetic Validation that ensures the word contains only alphabetic characters. If validation fails, the function exits or prompts the user again.

**create temporary file:** Creates a temporary file to store all dictionary entries except the one to be deleted. This avoids directly modifying the original file.

**for each entry in file:** Iterates through each entry (line) in the original file. Reads the word and its meaning from the file.

**entry.word != word:** Compares the current word in the file to the input word. If the words are not equal, the entry is written to the temporary file. If the words are equal, the entry is skipped, effectively "deleting" it.

**mark as found:** A flag (found) is set when the word to be deleted is encountered. This flag is used later to determine whether the word was successfully found and deleted.

**if word found:** If the found flag is set. Replaces the original file with the temporary file. Reloads the in-memory dictionary array to reflect the updated file contents.

**else:** If the found flag is not set. The word was not found in the dictionary. The temporary file is deleted without replacing the original file. and prints as "word not found".

## 5.4 Main Function:

```
function main():  
    Initialize countwords to 0  
    call files() to load dictionary data from file  
    while true:  
        display menu:1. Search Word 2. Insert Word3. Delete Word4. Exit  
        input choice  
        switch choice: case 1:call searchWord()  
                        case 2:calls the insertWord()  
                        case 3:calls the deleteWord()  
                        case 4: print "Exiting program" . exit  
                        default: print "Invalid choice. Please try again."
```

**initialize countwords to 0:** Initializes the variable countwords, which keeps track of the total number of words in the dictionary.

**call files() to load dictionary data from file:** Loads the existing word-meaning pairs from the dictionary.csv file into the in-memory dictionary array.Updates the countwords variable.

**while true:**Starts an infinite loop to continuously prompt the user until they choose to exit.

**display menu:** Prints the menu options: Search Word, Insert Word, Delete Word, Exit

**input choice:** Reads the user's menu choice.



**switch choice:**Executes different actions based on the user's input.

**case 1:**Calls the searchWord() function to allow the user to search for a word in the dictionary.

**case 2:**Calls the insertWord() function to add a new word and its meaning to the dictionary.

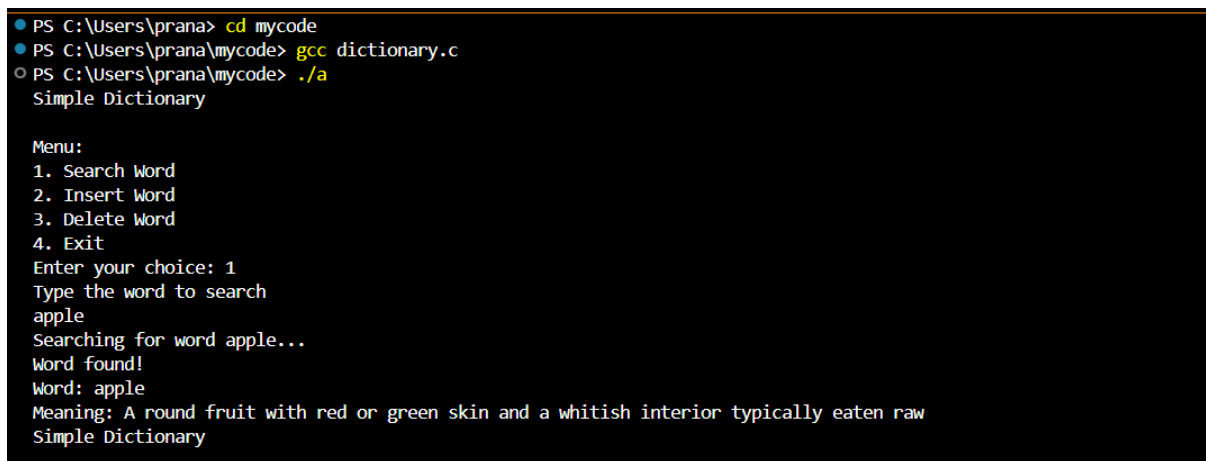
**case 3:**Calls the deleteWord() function to remove a word from the dictionary.

**case 4:**Prints a message indicating the program is exiting.Terminates the program using exit.

**default:**Handles invalid inputs by printing an error message and looping back to display the menu again.

## 6. Input And Output :

### 6.1 Search Word Operation :

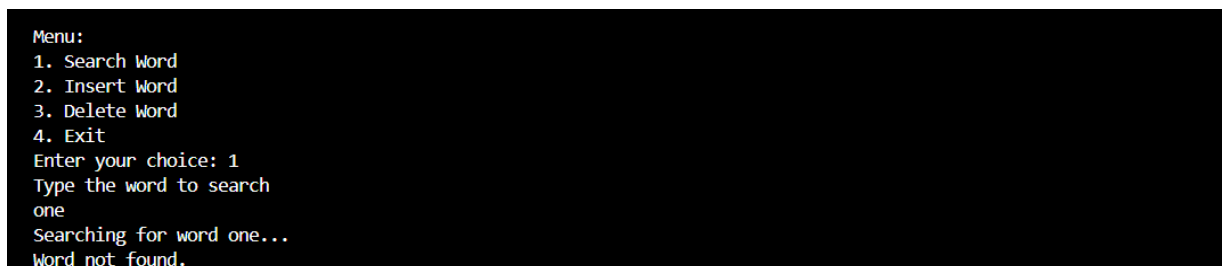


```
PS C:\Users\prana> cd mycode
PS C:\Users\prana\mycode> gcc dictionary.c
PS C:\Users\prana\mycode> ./a
Simple Dictionary

Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 1
Type the word to search
apple
Searching for word apple...
Word found!
Word: apple
Meaning: A round fruit with red or green skin and a whitish interior typically eaten raw
Simple Dictionary
```

Fig 1

First Query(fig 1): The user selects option 1 to search for a word. They input the word apple. The program searches for the word and successfully finds it in the dictionary.Meaning: A round fruit with red or green skin and a whitish interior typically eaten raw



```
Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 1
Type the word to search
one
Searching for word one...
Word not found.
```

Fig 2

Second Query(fig 2): The user repeats the Search Word operation and inputs the word one. The program fails to find the word in the dictionary.

## 6.2 Insert Word Operation:

```
Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 2
Type the word to insert
dictionary
Type the meaning.
collection word meaning
Word 'dictionary' has been added to the dictionary.
Simple Dictionary
```

Fig 3

First Query(fig 3):

Selection of Option: The user chooses option 2 from the menu to insert a new word into the dictionary. User Input: the user to input the word to add. "dictionary" It then prompts the user to enter the meaning of the word. Input: "collection word meaning" Confirmation: The program confirms the successful addition of the word and its meaning to the dictionary.

```
Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 2
Type the word to insert
tree
word already exists. Please type a different word.
Type the word to insert
```

Fig 4

Second query(fig 4):

Selection of Option: The user selects option 2 from the menu to insert a word into the dictionary. Duplicate Entry: The user attempts to insert the word tree. The program checks its existing entries and finds that the word tree already exists in the dictionary.

```
C: > Users > prana > mycode > dictionary.csv > data
1  apple,A round fruit with red or green skin and a whitish interior typically eaten raw
39  tree,A tall plant with a trunk branches and leaves typically growing to a considerable height
40  dictionary,collection word meaning
```

Fig 5

In fig 5 the file "dictionary.csv" contains a list of words along with their meanings, stored in a comma-separated format. Each row represents one dictionary entry.

Updates after insertion: The entries dictionary is added during the program's execution, as demonstrated. The existing entries are apple and tree were already present in the file.

### 6.3 Delete Word Operation:

```
Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 3
Type the word to delete
dictionary
The word 'dictionary' is deleted from the dictionary.
```

Fig 6

First query(fig 6): Selection of Option: The user selects option 3 from the menu to delete a word from the dictionary. User Input: the user to enter the word they want to delete. Input: dictionary Deletion Process: The program searches for the word in the dictionary file. Upon finding the word, it removes the entry from the dictionary. Confirmation: The program confirms the successful deletion of the word:

```
Menu:
1. Search Word
2. Insert Word
3. Delete Word
4. Exit
Enter your choice: 3
Type the word to delete
pen
The word 'pen' was not found in the dictionary.
```

Fig 7

Second query(fig 7): The user selects option 3 to delete a word from the dictionary. Input: pen The program searches for the word pen in its dictionary database. The word is not found in the dictionary. The program ensures no changes are made to the dictionary and allows the user to continue.

## 7.Final Analysis:

The Simple Dictionary Program is a commendable demonstration of essential programming concepts, offering a functional and user-friendly interface for basic dictionary operations. It integrates core features such as word search, insertion, and deletion, along with persistent storage using a CSV file. The program effectively handles input validation, ensuring that only alphabetic characters are accepted, and prevents duplicate entries during insertion. Additionally, the program dynamically updates an in-memory dictionary array, ensuring smooth operation and quick access.

The structured implementation and modular design of functions like `files()`, `searchword()`, `insertword()`, and `deleteword()` highlight a clear and organized approach to programming. These functions make it easy to understand, maintain, and extend the codebase. The use of file handling ensures that data is retained across program executions, enhancing its usability...

The Simple Dictionary Program offers several benefits that make it practical and effective for basic dictionary operations. Its user-friendly interface ensures easy navigation with clear options for searching, inserting, and deleting words. The use of persistent storage through a CSV file guarantees data retention across sessions, enhancing reliability. The program effectively demonstrates core programming concepts such as file handling, string manipulation, and input validation, making it an excellent learning tool for beginners. It allows dynamic updates, ensuring that changes are instantly reflected in both the file and in-memory data structure. Additionally, its lightweight design enables quick setup and use without complex dependencies, and its portability allows it to run on various platforms with minimal modifications. Overall, the program's simplicity and educational value make it an excellent tool for foundational programming and data management tasks.

Overall, this program stands out as a well-designed and functional solution for basic dictionary operations, providing an excellent learning tool for understanding file handling, input validation, and data persistence in programming. It successfully meets its intended purpose and showcases the developer's ability to implement practical and reliable software