#### 

**A MINI PROJECT REPORT**

#### ON

# [Mini Google]

## *Submitted in partial fulfillment for the award of the degree of Bachelor of Engineering*

#### *In*

**COMPUTER SCIENCE AND ENGINEERING**

***Submitted by***

#### [Dipesh Shrestha]

#### [1NH17CS039]

#### [A]

***Reviewed by***

#### [Mrs. Yogitha]

#### [Assistant Professor, CSE]

## New Horizon College of Engineering

Certificate

# This is to certify that the mini project work titled

[Mini Google]

##### *Submitted in partial fulfillment for the award of the degree of Bachelor of Engineering*

#### [Dipesh Shrestha]

#### [1NH17CS039]

**During the academic year**

***2018-2019***

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Signature of Reviewer** |  | **Signature of HOD** |

**Semester End Examination**

|  |  |
| --- | --- |
| *Name of the Examiner*  ***1. .…………………………….*** | *Signature with date* ***……………………………*** |

***2. ..…………………………… ……………………………***

|  |  |  |
| --- | --- | --- |
| **TABLE OF CONTENTS** | | |
| **CHAPTERS** | | |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **Acknowledgment** | i |
|  | **Abstract** | ii |
|  | **List of Figures** | iii |
| 1 | **Introduction** | 1 |
| 2 | **Analysis** | 2 |
| 2.1 | Objectives of the Project | 2 |
| 2.2 | Requirement Specification | 2 |
| 3 | **Implementation and Design** | 3 |
| 3.1 | Algorithm | 3 |
| 3.2 | Data Structures | 4-8 |
| 4 | **Functionality** | 8-11 |
| 5  6 | **Sample Output**  **Conclusion** | 11-14  15 |
| 7 | **Bibliography** | 15 |
|  |  |  |
|  |  |  |
|  |  |  |
|  | i. |  |

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be, but impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned my efforts with success.

I thank the management, **Dr. Mohan Manghnani**, Chairman of NEW HORIZON EDUCATIONAL INSTITUTIONS for providing necessary infrastructure and creating good environment.

I also record here the constant encouragement and facilities extended to me by

**Dr. Manjunatha**, Principal, NHCE, **Dr. Prashanth.C.S.R,** Dean Academics, **Dr. B. Rajalakshmi**, Head of the Department of Computer Science and Engineering. I extend my sincere gratitude to them.

I express my gratitude to **[Ms Yogitha]**, my project reviewer for constantly monitoring the development of the project and setting up precise deadlines. [His/Her] valuable suggestions were the motivating factors in completing the work.

Finally a note of thanks to all the teaching and non-teaching staff of Computer Science and Engineering Department for their cooperation extended to me and my friends, who helped me directly or indirectly in the course of the project work.

**[ Dipesh Shrestha (1NH17CS039) ]**

ii.

**ABSTRACT**

As we are well known what Google means and what actually it does. This project is also based on same logic but it’s a smaller version of Google where it’s working only with words but not with images, videos, gifs, etc. as Google does, so it is named as ‘Mini Google’. Realizing the truth I got this concept from Google itself, also while performing sorting, search techniques and also while playing ‘4 pictures 1 word game’.

The main **Objective** of this project is to filter out the words that includes the letters which are given as an input but its specialty is that although the word may be of ten letters or more (let) this code can access it even with any of its two letters. Let’s us take an example for that, if ‘i’ and ‘a’ is taken as input and you want meaningful words with the combination of these letters under car brands then output will be like; ‘Maserati’, ‘Ferrari’ and all possible words. Not only car brands but my project will give user access to choose across other options too. This fulfils the concept of filtering the contents. This makes our work a lot easier and has application in various organizations like: auto showrooms, library, hospitals, schools, corporate office and so on.

Also the concept of this project is that the codes understands the jumbled letters and delivers the clean word. For example, for the jumbled letters ‘orspceh’, this project delivers the output as ‘Porsche’.

Above mentioned examples gives the clear cut objectives of my current project.

iii.

**LIST OF FIGURES**

**Figure No Figure Name Page No**

Fig a. One-dimensional array 4

Fig b. Two-dimensional array 4

Fig c. Single linked list 6

Fig d. Double linked list 6

Fig e. Circular linked list 7

Fig f. Header linked list 7

**Introduction:**

‘Mini Google’ uses basic C statements like: loops, switch case, if condition, etc. and linked list from Data Structure. I got the concept of Mini Google from today’s AI, Google itself & and while performing search, filter, sort techniques.

This project search an element with almost least letters (i.e. minimum 2 letters) with different suggestion, option and filter out the matching ones to the console.

Mini Google makes the searching process less time consuming, interactive and suggestive. This project uses file concept, here file acts as a data storage or server which consists of meaningful words. The inputs from the user is matched with the words in those files and passes the matching word to the linked list and finally displays. Working out with an example, let’s assume a file named demo.txt has three people names (i.e. ‘dipesh’, ‘anusha’, ‘mills’) and suppose users gives input as either ‘sh’ or ‘hs’ then the codes will first try to match the letters ‘s’ and ‘h’ each at a time with each words in the file demo.txt and filters out the matching ones i.e ‘dipesh’ and ‘anusha’ which are finally displayed through linked list.

In order to make more user friendly environment Mini Google has some sub headings as Sports, Music, Car Brands, Perfume Brands etc. so that user can go through what actually they are looking for.

**Analysis:**

**Objectives of projects:**

The goal of this project is:

* To provide easy interface for the user to interact with the options available.
* To search the element that user want with least possible letters (min 2 letters).
* To display as many results that matches the user input or to display similar results.
* To make the search less time consuming.
* To rearrange the jumbled words.

**Requirements:**

**Software requirements:**

* Compiler: Netbeans
* Operating system: windows 7 or later

**Hardware Requirements:**

* Processor: Intel Pentium 4 or more
* Ram: 1 GB or more.
* Hard disk: 120 GB hard

**Implementation and Design:**

**Design:**

**ALGORITHM** (10 Marks)

Step1: Start

Step 2: Declare required gloabal variables and functions.

Step 3: Inside main function, read the value of option variable from user and use switch case then call that function present in specified case.

Step 4: Inside the specific function, ask user to input minimum of two letters to search their result and assign it to a variable let it be input.

Step 5: Open a specific file in read mode and use while loop to call another

Function let it be commonFun().

Step 6: Inside commonFun() Read a string from the same file and assign it to global variable buffer and check each letter of user input variable with each letter of that string from buffer and increment hit if the given input matches with the each words in file.

Step 8: If hit becomes greater than or equal to 2 (hit>=2) then pass that string to linked list.

Step 9: Repeat step 6 to 8 until end of file is encountered.

Step 10: Display whole linked list.

Step 11: Repeat step 4 to 10 until user choose Y or y.

Step 12: Repeat step 3 to 11 until user choose Y or y.

Step 13: End.

**Implementation:**

**Data Structure:**

**Arrays:**

Array is a non-primitive data structure which stores data of consisting of similar data type. There are two types of array,

1. **One-dimensional array:**

This type of array has linear structure. It looks like:

A

24 46 12 1 8

Fig a:

The above figure reflects a one dimensional array with name as A and has four field. It is represented as A[4]. We can iterate through all the values using loops. One dimensional array not only used to store numeric value but also strings and characters having similar data type but cannot store strings, and numeric value at a time.

1. **Two-dimensional array:**

Two dimensional array has table or matrix like structure. There is no limits on the number of dimensions in an array. This type of array is used when you work out with matrix related problems or needs to store data in the tabular form.

|  |  |  |
| --- | --- | --- |
| 1 | **0** | **0** |
| **1** | **0** | **1** |
| **0** | **1** | **0** |

Fig b:

The above figure reflects a two dimensional array with name as A and has 9 fields. It is represented as A[2][2].

This project utilizes only one dimensional array to store strings from files and user input. The strings which are stored in array will be ending with ‘\0’ which makes us easier to iterate and point the ending of loop.

**Files:**

A file is a non-primitive data structure which is used to store related information. File is created for permanent storage of the data. Normally when we close the program all the data will be erased but storing data into file will preserve your data. There are different modes of opening a file like read mode, write mode and append mode. In C a file is declared using pointer as FILE \*<name\_of\_the\_file>.

This project utilizes files as a data storage server which stores different contents under specific headings. It consist of many files storing different field contents. Files fulfils the data server requirement of this project. Even though program gets terminated but the contents still remains in the file so that its not required to re-enter the data again and again. The contents of the file is fed into the linked list one by one only after the condition satisfied.

**Linked list:**

Linked list is a linear data structure which is a collection of nodes where each nodes consist of data part and an address part which stores the address of other node. Each node in the linked list will be connected or linked to another node through addresses. Memory is allocated to the data elements using Dynamic Memory Allocation i.e. malloc(), calloc(), realloc(). There are mainly four types of linked list like single, double, circular and header linked list. There are normally 4 types of linked list:

1. Single linked list
2. Double linked list
3. Circular linked list
4. Header linked list

In C, linked list is represented using structures which is also called as self-referential structure.

* **Single linked list:**

In SLL, each node consist two parts, one is data part and the other one is address part which is clearly shown in fig c.

Head

10

10 20 30 40

a 20

b 30

c 40

d

Fig c:

The above figure reflects Single linked list having 4 nodes and a head node pointing to first node.

* **Double linked list:**

In DLL, each nodes will have one data part and address of previous node and address of next node.

Head

a 20

10

10

a 20

10 b 30

20 c

Head

10 20 30

Fig d:

The above figure reflects Double linked list having 3 nodes and a head node pointing to first node.

* **Circular linked list:**

The CLL may be SLL or DLL along with the last node connecting back to the first node. Below figure clarifies CLL.

Head

a 20

10

20 c 10

10 b 30

**10 20 30**

Fig e:

The above figure reflects Circular Double linked list having 3 nodes, the last node pointing to the first node and a head node pointing to first node.

* **Header linked list:**

A HLL is a SLL along with a extra header node called as header node whose data part will be null and address part will be pointing to next node. The head node will be pointing to the header node. The following figure gives the clear picture of HLL.

Head

10 20 30 40

Head

10

b 40

a 30

c

20

Header

Fig f:

The above figure reflects Header linked list having 3 nodes , a header node and a head node pointing to header node.

This project utilizes Single linked list in order to link the matched results between user and the file contents and display those results. It uses two operations of the single linked list which are;

1. **Insertion at beginning:**

During insertion at beginning, the final result is fed into the data part of the node one by one and the address part of the new node will be storing the address of previous node and the head node will be pointing to the new node.

1. **Display:**

During this operation starting from head node till the last node all the data part is displayed. This project consists of separate function for the display part which uses while loop to iterate through all the nodes and each node data part will be displayed.

Functionality:

When the code is executed, the output window will first ask for the user to enter the website which is “www.minigoogle.com” after this you will be allowed to the main page. If you type the website other than this or if you miss any letters then it will display an error message and allows you to re-enter the website. For this whole thing to happen mini Google uses strcmp() function. The website is already stored in a variable as string which is matched with the user input and if strcmp() returns 0 then the main page is displayed otherwise it will throw an error message.

In the main page of Mini Google various options are displayed for the user to choose and anchor their search. Those options can be easily decreased by just removing the files and the related function and can be easily increased with extra more options by including files with the related contents and linking that file with a new separate workable function. All of these options has separate functions of their own. So that the first thing user needs to do is choose among those options (fields) provided and it will redirect you to that particular field. Normal printf() function is used for this. And then user is asked to choose among the option which is being provided. Utilizing the option provided by the user into switch case, particular function is called. These functions inside the switch case are linked with the options that were displayed in the main page.

Switch case will call the function according to the option provided by the user.

Inside the called function, the main concept of the program highlights as code uses simple input function to read data from the user, here user needs to input minimum of two letters that s/he wants to search and store it into a variable let it be input. Program first checks whether the input given by the user is of length greater than 1 and proceeds only if the condition is true.

This was necessary inorder to make the search proactive. Since Mini Google uses file as data storage platform, all the words are related to specific topic are stored in files. Mini Google uses many files and each file has direct related with the options that were provided earlier in the main page. Show user should choose specific field properly to anchor search. Next code opens the file in read mode to get access to all the contents of the file. This project has a common function named as commonFun() which has the actual code of processing operations. As we know that all of the options in the main page has separate functions where all of these functions calls this common function i.e commonFun().

Here common function is iterated using while loop until the end of the file is reached. Inside common function the contents on each line is read using fgets() function. This function reads each line at a time and stores that value into a variable let it be buffer. Now the value read from the user present in input is matched with the value present in buffer. To perform this operation this project uses two while loops and an if condition. One while loop will iterate through the letters of value present in variable input and the other while loop is present inside the previous loop and will iterate through the letters of value present in buffer. Inside the second loop an if-statement is used to check the matching possibilities, if the first letter of both variables buffer and input matches then the first while loop will be incremented, second while loop will again start iterating from the first letter, a count is saved in variable hit and break statements breaks out of the inner loop but if it doesn’t match then only the inner while loop will be incremented to next letter and continues till the array encounters new line and first iteration is completed. Similarly, for next iteration same pattern is followed and the iteration continues till the array in the first while loop reaches the ending of the string. After the looping condition comes to an end, another if-condition comes to an action which ensures if the count which was stored in hit is greater than or equal to two and if condition is true the particular buffer value is inserted to the linked list, count is initiated and stored into variable check and continues the same for the next iterations until the main while loop condition is satisfied. The program is coded in such a way that if any of the two letters provided by the user matches with any of the contents of the file then the corresponding content of that file will be passed to the linked list.

Here linked list is used in order to display the result processed operations. This project uses single linked list which has one data part which is of character type and one address part which stores the address of another node. This project incorporates two operations of linked list i.e. insertion at beginning and displaying the list. Here single linked list is declared using self-referential structure which is present at the top of the code. Here head is declared as the linking node which will point to the first node of the linked list and is initially initialized to NULL. When hit is greater than or equal to two insert function of linked list is called along with the buffer value as argument, let the insert function be insertFun(). Inside the insertFun() of linked list the values are stored at the beginning of the list like the last value will be at the beginning or the head will be pointing to the last value and the value which was stored at beginning will be at the last node. Here memory is allocated for each node dynamically using malloc function and the value of buffer is copied to the data part of the linked list using strcpy() function. The address part of the node will store the head value and head will change its value to the address of the new node everytime. This process will continue till hit is greater than or equal to 2. After the insertion process is over the printFun() function comes to action. Here inside print function the data item present inside the linked list is printed.

To check whether the user input didn’t match any of the content of the file, count value was stored in check variable. Now if check has a zero value then not found statement will be displayed. This project gives user option to repeatedly perform their search on to the same field or different field. This was possible using do while loop.

Sample output:

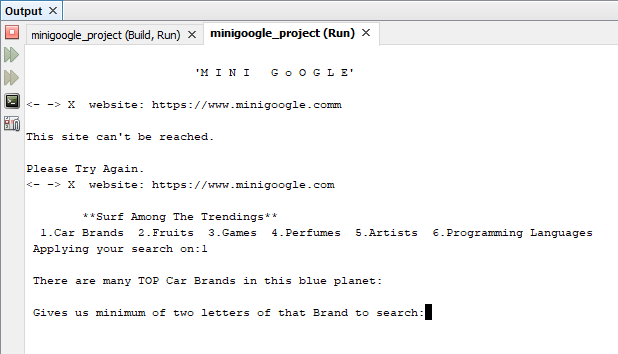
1. Starting Mini Google Interface;



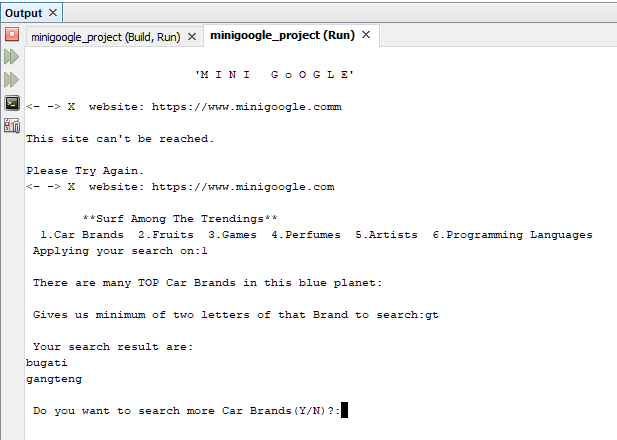
1. Applying correct website;



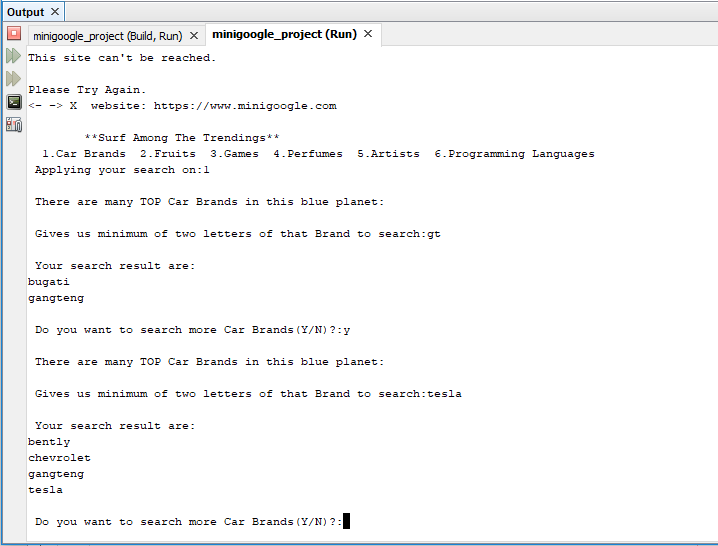
1. Getting into one field;



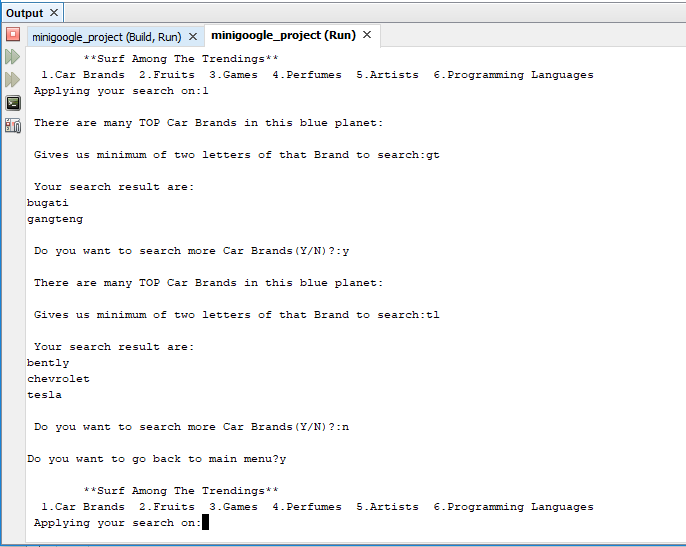
1. Reading minimum of two values;



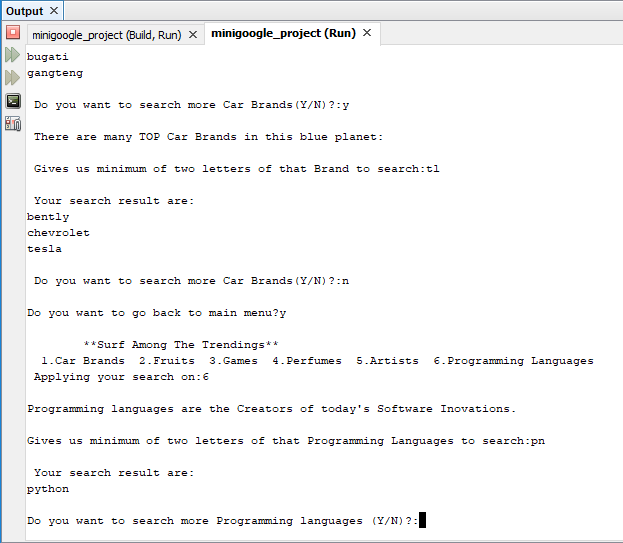
1. Complete word as input;



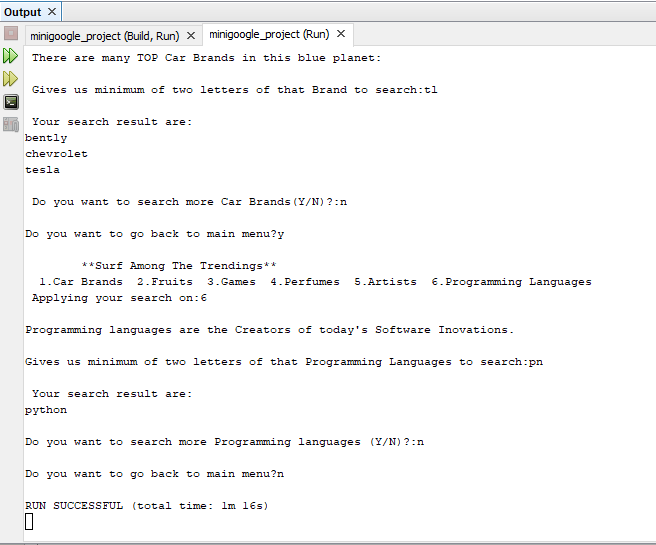
1. Returning to Main menu;



1. Applying search on next option;



1. Exit;



Conclusion:

The actual concept in making this project was to perform the searching technique as fast as possible. Here when u make a search u may not get only one output of your search but also as many suggestions as possible. Traditionally we almost type every letters of a word to search but this program is codded in such a way that you only need to give any two letters of that word. In this busy world everyone is in rush so this project help all and it actually saves a lot of time. We cannot find the time difference when we search one or two words but if we consider hundreds or thousands of words then actual time difference will be clarified.

Bibliography:

**Data Structures Using C** - Pearson publications, Aaron M. Tenenbaum, Yedidyah Langsam, Moshe J. Augenstein

<https://www.hackerearth.com/practice/data-structures/linked-list/singly-linked-list/tutorial/> - for linked list concept