NEW HORIZON COLLEGE OF ENGINEERING

Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC

Accredited by NAAC with 'A' Grade.

WORD BOOK REPORT

A MINI PROJECT

REPORT

Submitted by

RANJITHA A R

In partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade.

Certificate

This is to certify that the mini project work titled

WORD BOOK

Submitted in partial fulfillment of the degree of Bachelor of Engineering in Computer Science and Engineering

Submitted by

RANJITHA A R

1NH18CS157

DURING

ODD SEMESTER 2019-2020

For

19CSE39

Signature of Reviewer   Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner	*Signature with date*
1.______________________	______________________
2.______________________	______________________

ABSTRACT

Dictionary is windows software which can be useful for various types of people. Imagine that a vocabulary tutor is always available on your pc, laptop, and mobile. You can find meaning of all words in just few seconds. This dream comes true using this windows software. In this world we are mostly dependant on the electronic media.

Many times when we get some of the words which meaning is not known to us then the problem comes, but the problem is not that tuff now, meaning of these typical words is now in the gap of the few clicks. We open the dictionary then type the words and answer is provided offline. Yes thus dictionary doesn't need internet connectivity. So whenever you wish to know meaning of words just click on our software and the meaning is known. A good online dictionary can be fast, efficient, and more current than print editions.

Better still, consulting an online dictionary may be more convenient than unearthing the cumbersome desk dictionary from a growing mound of journals and manuscripts.
Whether print or online, dictionaries are helpful only if they include the words that users are seeking. Unfortunately, many dictionaries available on the Web are not so good.
After selecting 15 uncommon but not quite obscure words from various fields of science, editing, publishing, and emerging technologies, we entered them into the search field of each dictionary.

Some online dictionaries like WorldNet (Princeton University), the Free Online Dictionary of Computing, and Cancer Webs Online Medical Dictionary are devoted to single dictionaries and do not provide the variety of additional resources.
But our Online Dictionary providing three types of dictionaries, general English dictionary, Medical dictionary and Computer dictionary.
.

ACKNOWLEDGEMENT

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

I have great pleasure in expressing gratitude to **Dr. Mohan Mangham**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal NHCE, for his constant support and encouragement.

I am grateful to **Dr.Prashanth C.S.R**, Dean Academics, for his unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank Dr**. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Ms. Soja Rani S,** Sr. Assistant Professor, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

NAME-RANJITHA AR

USN – 1NH18CS157

CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 INFORMATION CONSISTENCY

One of the best ways for students to understand how dictionaries work is to make one themselves. This project is designed to get students to create a dictionary on a subject of their choice, collecting words that are unique to that subject. In doing so they will conduct their own original research.

Students will:

- collect and record words associated with their special subject
- decide which words merit an entry in their dictionary
- write definitions for those words
- Provide any other information that has been agreed upon for each entry, such as its pronunciation, part of speech, and etymology.
- Provide each dictionary entry with examples of its use, i.e. illustrative quotations.

1.2 OBJECTIVES

- develop an interest in words and language
- learn how dictionaries work
- become familiar with the format and components of a dictionary entry
- learn that words have a history
- learn that language is home grown and responsive to new areas of knowledge
- be encouraged to analyse their own use of language
- learn that language can help create a group or cultural identity
- recognise that we each use different kinds of language in different situations
- learn that language reflects the values of the user speaker
- Discover that language is constantly changing.

1.3 METHODOLOGY

This project is very use full for students. Even students can make their own word book depending on their choice of the subject by collecting the difficult words in that subject. By doing this u can learn easily and create an own research...
We can get definition, autonomy and of synonym the words

Where to get for words

- Newspapers
- TV
- Interviews
- Books

1.4 HISTORY OF DICTINOARY REPORT

1992-Annie Plummer of savannah, GA, started by donating 50 dictionaries to a school which is near to her house. Early on, bonnie beeferman of Hilton head, SC, learns of Annie’s effort and started to selling crafts to raise more money for providing dictionaries to the schools Bonnie has so many requests in 1995 from local schools that she contacts a Charleston, SC, newspaper explaining her project and asking for support.

Mary French reads the article and thinks about the project which was done by her. After realizing that covering schools effectively would take serious fund raising, she starts a 501 organization with her hubby, Arno in 1995 and the dictionary project was ready.
Since 1995, with the great support of countless sponsors and volunteers, more than 10 million dictionaries were been able to distribute.

REQUIREMENT ANALYSIS

FUNCTION

Function is set of inputs User askes to enter a word, and then asks to enter the meanings, autonomy; synonyms Check everything and give the output stored. When u want search a word and the if u want display everything give show dictionary

MAITAINABILITY

New words can be added based on our interest easily maintainable.

1.5 EXPECTED OUTCOMES

Example- u will get output as add a word, meanings, synonyms, antonyms.
Then u can search a word from the word book. Then u can delete and display all the words in that

1.6 HARDWARE REQUIREMENTS

Processor	-- processor above 400 MHz
RAM	-- 500MB
Hardware	-- 12 GB
Input device	-- mouse and keyboard
Output device	– monitor

SOFTWARE REQIREMENTS

Operating system	: windows xp
Front end	: ASP.NET 2.0
IDE	: Dev. c++
Data structure	: linked list
Sever	: Google

CHAPTER 2

DATA STRUCTURE

A data structure is the particular way of organizing data in a computer so that we can use whenever we want. The idea is to reduce the Time and Space complexities of different tasks. There are some popular linear data structures.

- Array
- Linked List
- Stack
- Queue
- **Bi**nary tree
- Binary search tree
- Binary heap
- Hashing

Array

Array is a data structure used to store homogeneous elements at the contiguous locations. Size of an array must be given before storing data.

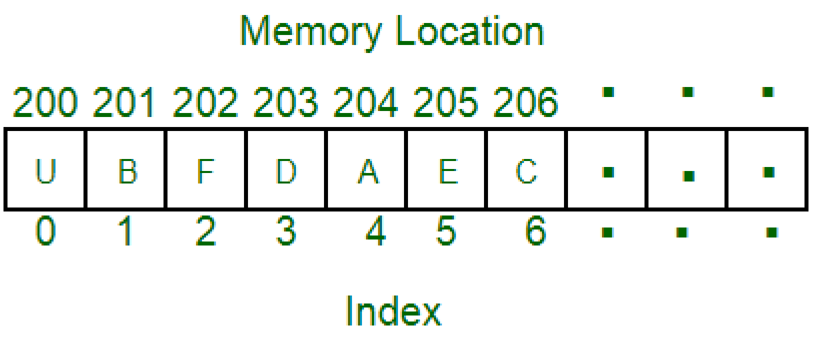


Fig 2.0

2.1 STACK

A stack is LIFO (last in, first out) it is an abstract data type that serves as a collection of elements, with two operations: push, which adds an element to the Box, and pop, which removes the last element that was added. In stack, both the operation of push and pop takes place at the same end that is at the top of the stack. It can be implemented by using the array and linked list.

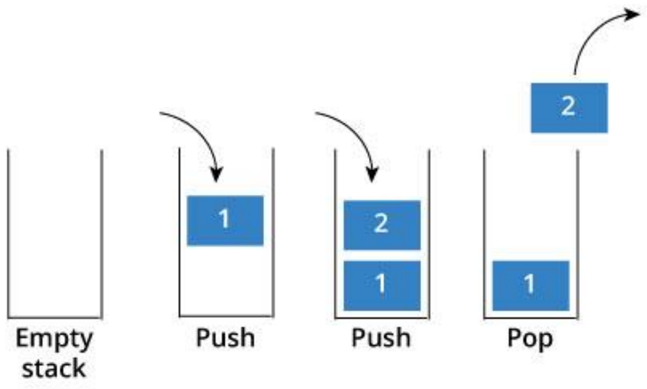


Fig 2.1

Example: Stacks are used for maintaining function calls (the last called function must finish execution first) we can Also always remove recursion with the help of the stacks.

Stacks are also used in cases where we have to reverse a word, check for balanced parenthesis and in the editors where the word you typed the last is the first to be removed when you use undo operations. Similarly, to implement back functionality in web browsers and stack.

Fig 2.2

Marriage plates are according to 1st in 1st out

2.2 QUEUES

A queue or FIFO (first in, first out) is an abstract data type that serves as the collection of elements, with two principal operations: enquire the process of adding an element to the collection.

(The element is added from the rear side) and dequeue, the process of removing first element that was added in that. (The element is removed from, the front side). It can be implemented by using both array and the linked list.

Circular Queue

The advantage of this data structure is that it reduces wastage of space in this case of array implementation, as the insertion of the (n+1)'th element, is done at the 0'Th index if it is empty.

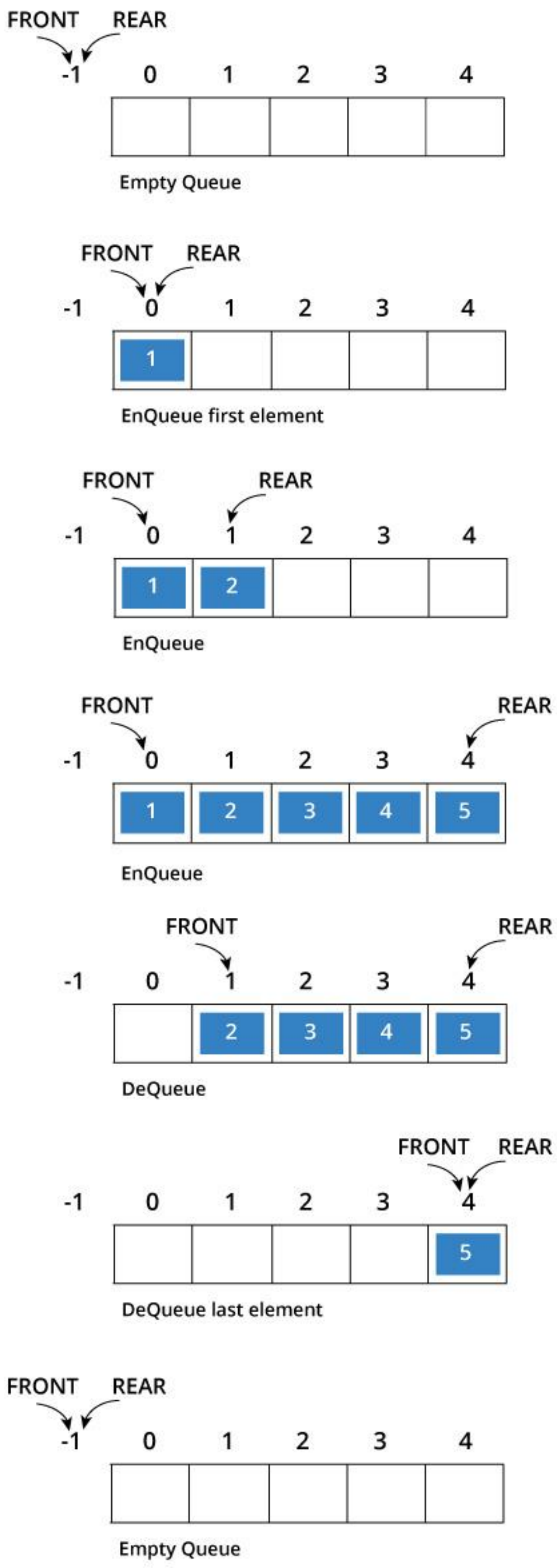

FRONT
REAR
-1 0 1 2 3 4
Empty Queue
FRONT
REAR
-1 0 1 2 3 4
1
EnQueue first element
FRONT
REAR
-1 0 1 2 3 4
1 2
EnQueue
FRONT
REAR
-1 0 1 2 3 4
1 2 3 4 5
EnQueue
FRONT
REAR
-1 0 1 2 3 4
2 3 4 5
DeQueue
FRONT
REAR
-1 0 1 2 3 4
5
DeQueue last element
FRONT
REAR
-1 0 1 2 3 4
Empty Queue

Fig 2.3

2.3 LINKEDLIST

It’s a linear data structures which consists set of nodes where each node has 2 parts

* Data

* Link

Data contain elements and link contains address .A linked list is a linear data structure; it is collection of zeros and nodes where each node has information.

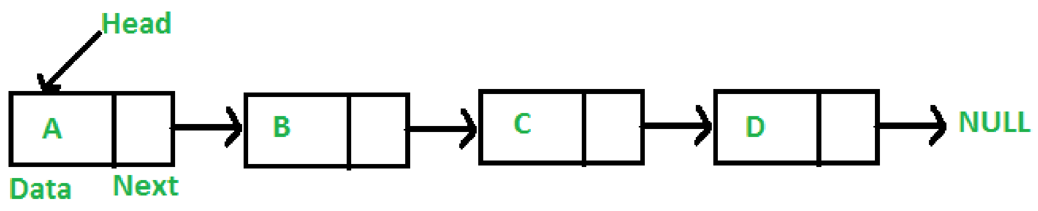


fig 2.4

The arrays size is fixed So we must know the upper limit on the number of elements in advance, the allocated memory is equal to the upper limit irrespective of the usage. Inserting a new element in an array of elements is expensive because the node has to be created for the new elements and to create room existing elements have to be shifted.

id [] = [105, 110, 120, 130, 140].

And if we want to insert a new ID 107, then to maintain the sorted order, we have to move all after 105 .Deletion is also expensive with arrays. For example, to delete 110 in id [], everything after 110 has to be move.

Memory is allocated to the nodes using dynamic memory allocation functions such as Malloc (), calloc (), realloc () and free ().

1. Malloc (): This function is used to allocate a complete single block of memory of the specified size. A pointer is used to the store the address returned my Malloc.
Syntax –
Data type*ptr= (data type*) Malloc (size)

2. Calloc (): It is function which allocates a specified size of memory in multiple blocks of same size. Each block should be assigned to null. A pointer is used to store the address.
Syntax –
Data type*ptr= (data type*) calloc (size, number of blocks)

3. Realloc (): For reallocating the allocated memory this function is used. A pointer is used to store the address returned.
Syntax –
Data type*ptr= (data type*) realloc (ptr, size)

4. free (): It is a function which is used to free the allocated memory.
Syntax –
free (pointer name)

MAIN ADVANTAGES OF LINKEDLIST OVER ARRAYS IS:
1. Size of array is fixed; we must know its upper limit in advance. But in linked list size is not fixed.
2. Insertion and deletion is easy compared to array.
3. No memory wastage will be there in linked list.

Self-referential code:

1. Single linked list
```
Structslist
{
Int data:
Structslist *ptr;
}
```

2.Double linked list
```
structslist
{
        int data;
        structslist *prev;
        structdlist *next;
}
```

TYPES OF LINKEDLIST

-> Single linked list

-> Double linked list

-> Circular linked list

-> Header linked list

SINGLE LINKEDLIST

Self-reflection structure or linked list, every node stores address or reference To 1 node in list and the last node has next address or reference is NULL. For Eg- 5->6->7->8->NULL

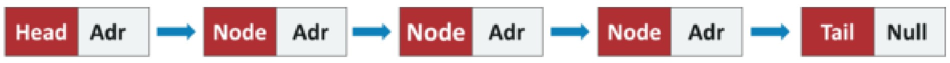


Fig 2.5

Doubly Linked List

In this type of Linked list there are two references associated with every node, one of the Testimonial points to the next node and one to the previous node

. Advantage of Dll data structure is that we can traverse in both the directions and for deletion we don't need to have explicit access to previous node. Eg. NULL<-4<->5<->6->NULL

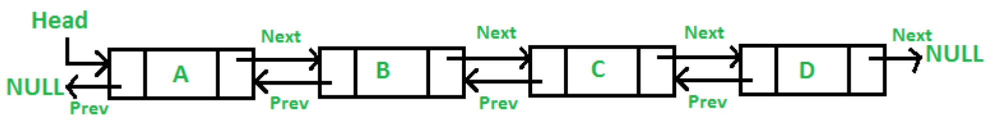


Fig 2.6

Circular Linked List

Circular linked list is a linked list where all nodes are connected In a Form of circle. There is no NULL In the ending. A circular linked list can be a singly circular linked list, or doubly circular linked list.

Advantage of the data structure is that any node can be made As starting node. This is useful in a implementation of circular queue in linked list. Eg. 1->2->3->1 [The next pointer of last node is pointing to the first Node]

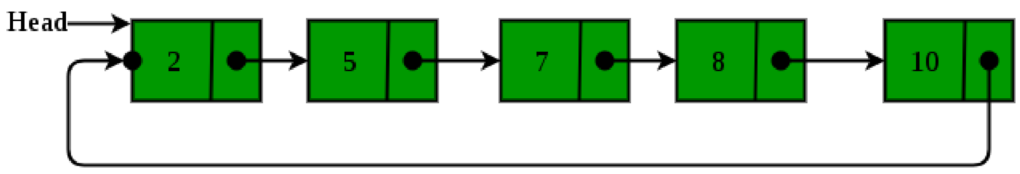


Fig 2.7

Example: Consider where we can make an array of marks of student. Now if a new sub is added in the course its marks also to be added in the array of marks. But a size of a array was fixed and it is already full so that it cannot add any new element. If we make an array of a size, lot more than a number of subjects it is Capable that most of the array will remain empty. We reduce the space wastage Linked List is formed which adds a node only when a new Number is introduced. Insertions and deletions also become easier with linked list. One big drawback of the linked list is random access is not allowed. With arrays, we can access element in time.

2.4 BINARY TREE

Arrays, Linked Lists, Stack and queues, which are linear data structures, trees are hierarchical in data structures. Binary tree is a tree which has at most 2 child nodes A binary tree is a tree data structure in which each node has at most of two children, which are referred to as the left child and the right child. It is implemented mainly using Links.

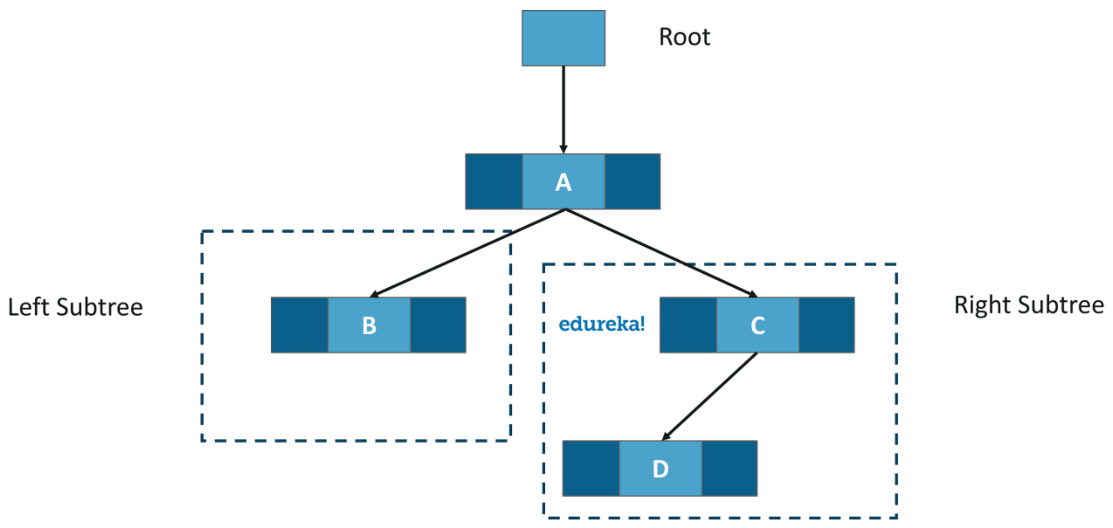


Fig 2.8

Properties-

Maximum number of nodes there = 2h + 1 – 1.

In this h is height of a tree. Height is considered

As the maximum number of edges on a path from root to the leaf.

Strictly binary tree-a tree which has 0 or 2 child node

Full binary tree-in which every node should have 2 child node expert the leaf node

Complete binary tree- it is a full BT till h=1 level but in the last level all the nodes are filled from lift to right

Binary search tree

In Binary Search Tree all the elements in the left sub tree of a node contains only nodes with keys less than the node is key.
The right sub tree, of a node contains only nodes with keys greater than the node's key.
The left and the right sub tree each must also be a binary search tree.

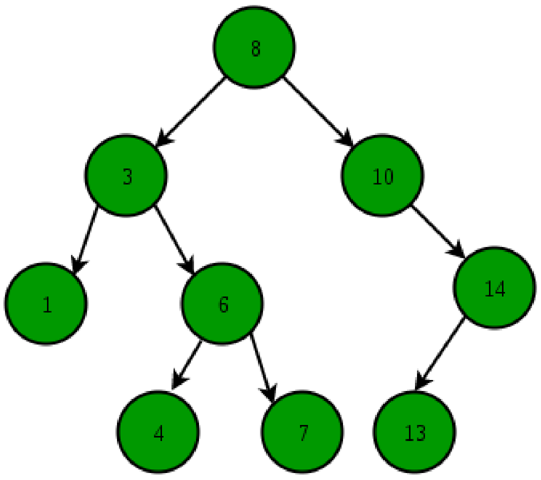


Fig 2.9

Binary Heap

It's a complete tree (All levels are completely filled except possibly the last level, and the last level has all keys as left as its possible). This property of Binary Heap makes them suitable to be stored in the array.
A Binary Heap is an either Min Heap or Max Heap. In a Min Binary Heap, the key at root must be minimum among all keys present in the Binary Heap. The same property must be recursively true for all nodes in the Binary Tree. Max Binary Heap is similar to the Min Heap. It is mainly implemented using an array.

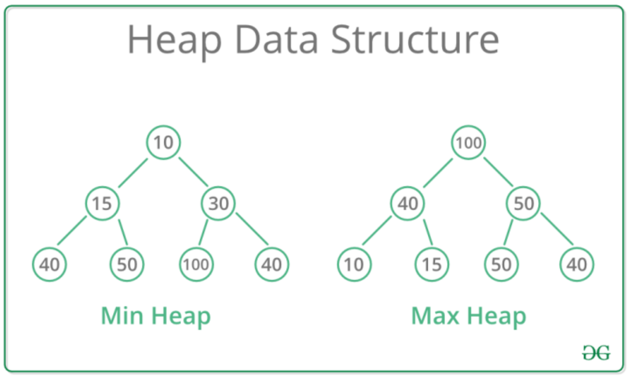
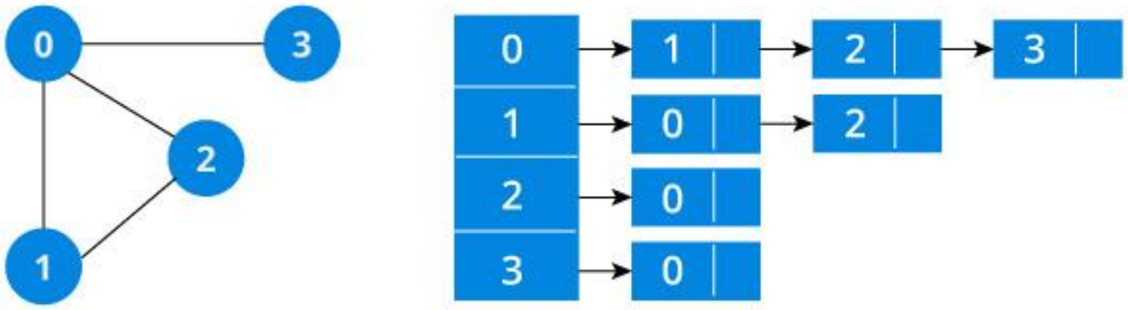
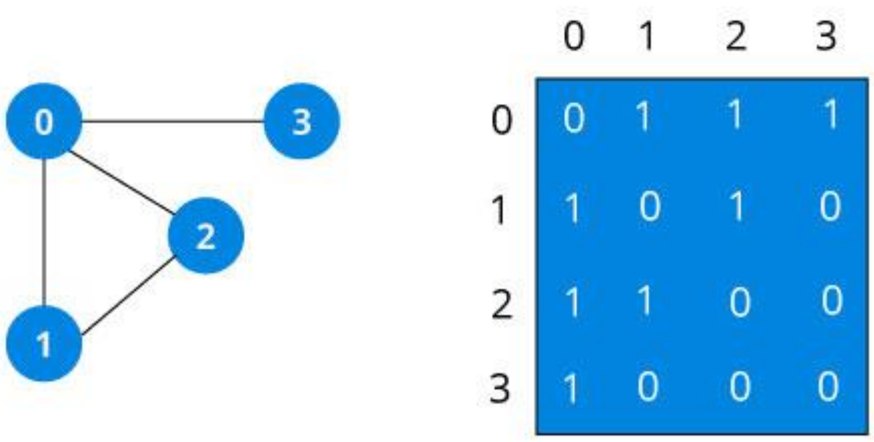


Fig 2.10

2.5 Graph

Graph is the non-linear data structures consisting of nodes and the edges. The nodes are sometimes also referred to as vertices and edges these lines or arcs that connect any two nodes in a graph. More formally Graphs can be defined like
The Graph consists of finite set of vertices (nodes) and set of the Edges which connects a pair of the nodes.





$$\begin{array}{c|cccc} & 0 & 1 & 2 & 3 \\ \hline 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 2 & 1 & 1 & 0 & 0 \\ 3 & 1 & 0 & 0 & 0 \end{array}$$

Fig 2.11

CHAPTER 3

ALGORITHM

Algorithm is a set of well-defined instructions in a sequence of solving the problem.

Qualities of a good algorithm

1. Input and output should be a well-defined precisely.
2. Each step in algorithm should be clear and complete.
3. Algorithm should be effective among many different ways of solving a problem.
4. An algorithm should never have computer code. Instead, the algorithm should be written in such a way so that, it can be used in similar programming languages.

. There are mainly 3 parts:

1. Search O(nlogn) and then insert O(1)
2. Search O(logn) and then insert O(logn)
3. Search and then insert both O(1)

First one is used for an example during when you insert at the final position of your linked list and search with the iterations after the sorting has been done. This is helpful when you have got to maintain your database with less search usages than insert usages. Second one is helpful for example when you sort insertion and search with the help of k-ary search method, taking logn time. This is helpful e.g. in telephone directory, where there are more searches in insertion. Now a days, hashing techniques are more used to deal with a such problems (O (1)). So, using hashing and indexing, we can achieve this functionality of O (1) time search and the insertion. Deletion time is same as search time since you can delete in 0 time just after you have found your element

INSERT

This function takes the starting node and data to be inserted as arguments. New node to be inserted at the end so, iterate through the list till we encounter the last node. Then, allocate memory in the new node and put data in it. Then store the address in the next field of the new node as NULL.

DELETE

This function takes the starting node (as pointer) and data to be deleted as the arguments. go to the node for which the node next to it have to be deleted, If that node points to NULL (i.e. pointer->next=NULL) then the element has to be deleted if not present in the list. Else, now pointer points to the node and the node next to it has to be removed, declare a temporary node which points to the node which has to be removed. Store the address of the node next to a temporary node in the next and the
field of the node pointer (pointer->next = temp->next). by breaking the link we removed the node which is next to the pointer (temp). Because we deleted the node, we no longer require the memory used for it, free() that will deallocate the memory.

SEARCH

This function takes the start node (as pointer) and data value of the node (key) to be found as the arguments. Iterate through entire linked list and search for the key. Until next field of the pointer is equal to NULL and check if pointer->data = key. If it is then the key is found else, move to the next node and then search (pointer = pointer -> next). If key is not found return 0, orelse return 1.

PRINT

function takes the starting node (as pointer) as an argument. If pointer = NULL, then there is no element in a list. Else, print the data value of the node (pointer->data) and move to the next node by recursively calling the print function with pointer->next sent as an argument.

Step 1 : insert and declare all header files which are i required in the project
Step 2 : and declare all variables which are required for project
Step 3 : create the structure for the link and create a node by using single linked list user can display add word
Step 4 : Next we use display function for displaying words how many to insert
Step 5 : after that user will create an main function which consists of switch case (switch ch)
For adding word ,searching word, displaying word
Step 6: then user use an default statement to display an wrong option

Step 7 : in the final it displays all words ,meanings,synonyms,and antonyms.

All the function called header files and global variable are declared in the beginning of the problem.
Then the main function is created and all the local variable in it are declared.
A structure is created and the structure members are declared in it. And the structure variable is declared.

A loop is created with the information of the function defined in it. Write the help of print statement.
A switch case is defined in the loop with the appropriate cases in respective to the functions defined above
In case 1: the user is provided with the opportunity to add words to the dictionary
Case 2: the user can search the meaning of the word which he desires to know.
Case 3: the user can see the words which he /she added to the dictionary

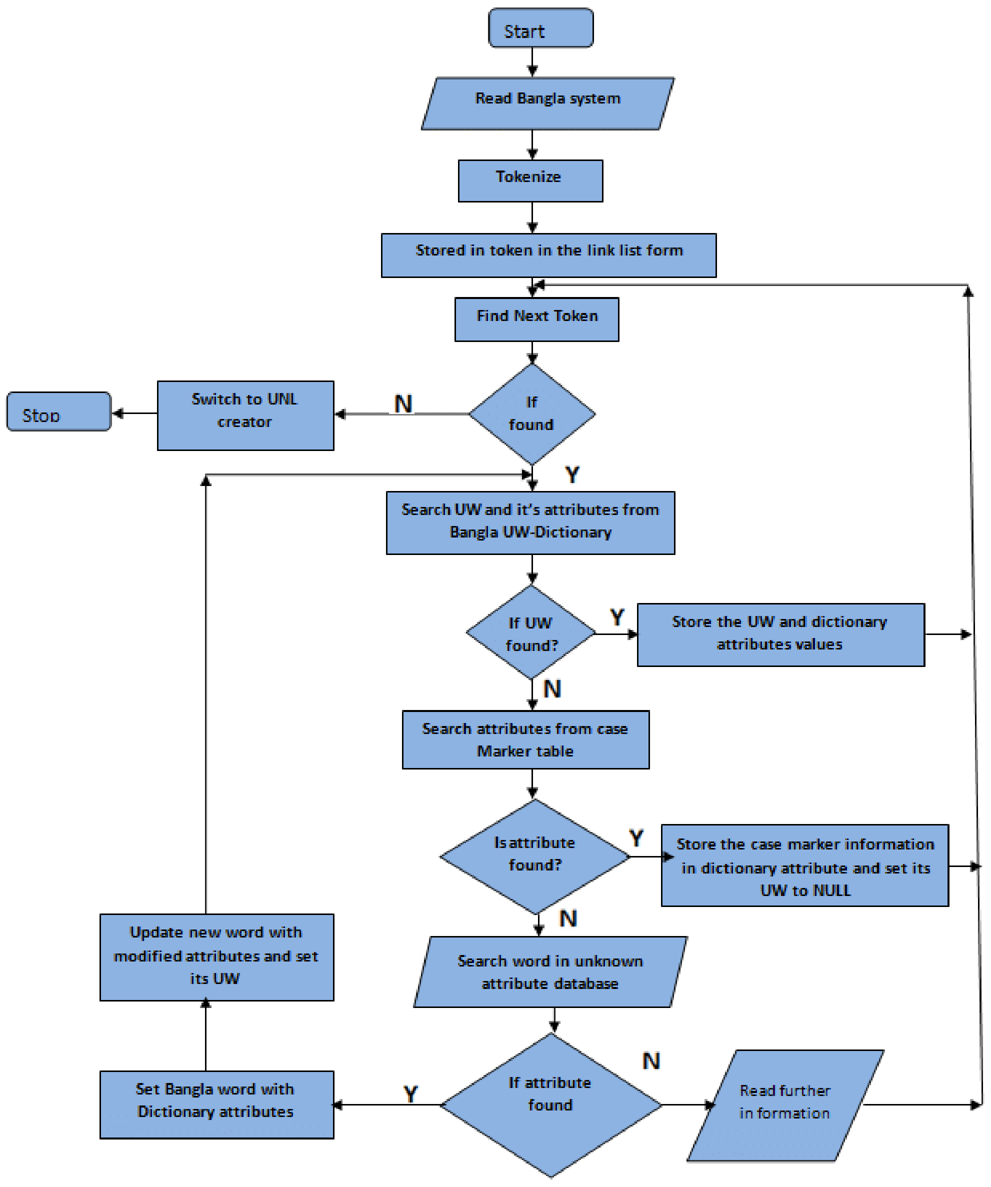

Start
Read Bangla system
Tokenize
Stored in token in the link list form
Find Next Token
If found
N
Y
Switch to UNL creator
Stop
Search UW and it's attributes from Bangla UW-Dictionary
If UW found?
Y
Store the UW and dictionary attributes values
N
Search attributes from case Marker table
Is attribute found?
Y
Store the case marker information in dictionary attribute and set its UW to NULL
N
Search word in unknown attribute database
If attribute found
Y
N
Set Bangla word with Dictionary attributes
Update new word with modified attributes and set its UW
Read further in formation

Fig 3.1

All the above cases work with a user. Defined function called add word, find, dilword.
In addword function all the local variables are declared and a code is written with the following if conditions and for loop. The find function all the local variable are declared and write the help of if conditions and for loop the word is searched through the file.

In display functions all the words. Are displayed with the help of the loop.
In delete word, function all the local variable are declared and with help of the for loop and if conditions the words are deleted.
In all the above function the string comparison function plays the major role.

CHAPTER 4

IMPLEMENTATION

Linked List Insertion

Imagine that Now we are inserting a node B (New Node), between A (Left Node) and C (Right Node). Then point B.next to C −

NewNode1.next −> Right Node;

Now, the next node at the left side should point to the new node.

LeftNode1.next −> New Node;

Deletion is also more than one step process in this

The left (previous) node of the target node now should point to the next node of a target node −

LeftNode1.next −> TargetNode1.next;

This will remove the link that was pointing to a target node. Now, using the following code, we will remove what the target node at pointing

TargetNode1.next −> NULL;

We need to use a deleted node. We can keep that in memory otherwise we can simply deal locate memory and then wipe off the target node completely.

```
Void add word (char * str)

{

int p, j = toupper (str [0]) - 65;

struct node * r, * tempword = dpc [ j ], * qra ;

 char mean [ 5 ] [ 20 ], ch1 = 'y' ;

 char h[5][20], ch2='y';

char a [5][20] , ch3='y';

p = notice ( str ) ;

 if ( p )
```

```
  {
printf ( "\nWord already exists" ) ;
getch( ) ; return ;
 }
  qra = ( struct node * ) Malloc ( size of ( struct node ) ) ;
   strcpy ( qra -> d, str ) ;
    qra -> link = NULL ;
for ( p = 0 ; tolower ( ch ) == 'y' && p < 5 ; p++ )
{
fflush ( stdin ) ;
 printf ( "  Enter any  word  " ) ;
 fflush ( stdin ) ;
  gets ( word ) ;  strcpy ( qra -> m [ p ] , mean [ p ] ) ;
   if ( p != 4 )
printf ( " Add a further meanings if it’s there (y/n) " ) ;
 else
 printf ( "enter no more more than five meanings " ) ;
 fflush ( stdin ) ; ch = getche( ) ; } qra -> mcount = p ;
for ( p = 0 ; tolower ( ch2 ) == 'y' && p < 5 ; p++ )
 {
fflush ( stdin ) ;
 printf ( " Enter the synonyms  " ) ;
 gets ( h[ p ] ) ;
  strcpy ( qra -> h [ p ] , h [ p ] ) ;
   if ( p != 4 )
printf ( " add further  (y/n) " ) ;
```

```
 else
  printf ( " enter no more than  five synonyms." ) ;
  fflush ( stdin ) ;
   ch2 = getche( ) ;

    }
    qra -> scount = p ;
      for ( p = 0 ; tolower ( ch3 ) == 'y' && p < 5 ; p++ )
      {
fflush ( stdin ) ;
printf ( " Enter the antonym " ) ;
gets ( a[ p ] ) ;
 strcpy ( qra -> a [ p ] , a [ p ] ) ;
 if ( p != 4 )
printf ( " Add further(y/n) " ) ;
else
 printf ( "enter no more than  5 antonyms " ) ;
  fflush ( stdin ) ; ch3 = getche( ) ; } qra -> acount = p ;
if ( dic [ j ] == NULL || strcmp ( dic [ j ] -> d, str ) > 0 )
{
r = dic [ j ] ;
 dic [ j ] = qra ;
  qra -> link = r ;
   return ;
    }
    else
```

```
     {
while ( temp != NULL )
{
if ( ( strcmp ( temp -> d, str ) < 0 ) && ( ( strcmp ( temp -> link -> d, str ) > 0 ) || temp ->
link == NULL ) )
{
qra -> link = temp -> link ;
 temp -> link = qra ; return ;
 }
 temp = temp -> link ;
  }
   }
   }
```

This is where the project is initiated with its primary step. With the help of this part of the code this project becomes a user friendly, where the user can add words and one or more meaning as well.

Here by selecting choice one that is to add words the user add the word of his interest and is asked to add meaning to that word and then it will confirm it from the user whether to add one or more meaning and with the help of this function we can add one or more synonyms and antonyms as well. With the help of this function you cannot enter more than five antonyms and synonyms.

int notice (char *str)

```
   {
struct node *n ; char temp1 [ 20 ] ; char temp2 [ 20 ] ; int p ;
n = dic [ toupper ( str [ 0 ] ) - 65 ] ; strcpy1 ( temp2, str ) ; strupr1 ( temp2 ) ;
while ( n != NULL )
 {
Strcpy1 ( temp1, n -> d ) ;
printf("Word :\t");
if ( strcmp ( strupr ( temp1 ), temp2 ) == 0 )
 {
printf ( "%s", n -> d) ;
 printf("\meanings:");
 for ( p = 0 ; p < n -> mcount ; p++ )
printf ( "\n\t%d\t%s", p,n -> m [ p ] ) ;
 printf("\nSynnonyms:");
 for ( p = 0 ; p < n -> scount ; p++ )
printf ( "\n\t  %d\t%s", p,n -> h [ p ] ) ;
 printf("\antonyms");
for ( p = 0 ; p < n -> acount ; p++ )
printf ( "\n\t %d\t%s", p,n -> a [ p ] ) ;
 return 1 ; } n = n -> link ; } return 0 ;
  }
```

This is the second most steps in the word dictionary which allows the user to search a word. With the help of this function the user will be able to search the word which is added and he will also be able to read the meaning of that particular word. This function is also used to display the count of synonyms and antonyms the user has added.

```
void display( )
  {
struct node *n ; int p, j ;
printf ( "Dictionary" ) ;
 for ( p = 0 ; p <= 30 ; p++ )
printf ( "-" ) ;
for ( j = 0 ;j <= 25 ; j++ )
```

```
 {
n = dic [ j ] ;
while ( n != NULL ) {
printf("\nWord:");
printf ( "%s", n ->d) ;
printf("\meanings:");
 for ( p = 0 ; p < n -> mcount ; p++ )
printf ( "\n\t(%d)\t%s", p+1,n -> m [ p ] ) ;
 printf("\nSynnonyms:");
  for ( p= 0 ; p < n -> scount ; p++ )
printf ( "\n\t(%d)\t%s", p+1,n -> h [ p ] ) ;
 printf("\antonyms");
 for ( p = 0 ; p < n -> acount ; p++ )
printf ( "\n\t (%d)\t%s", p+1,n -> a[ p ] ) ;
 f = f-> link;  } }
 }
```

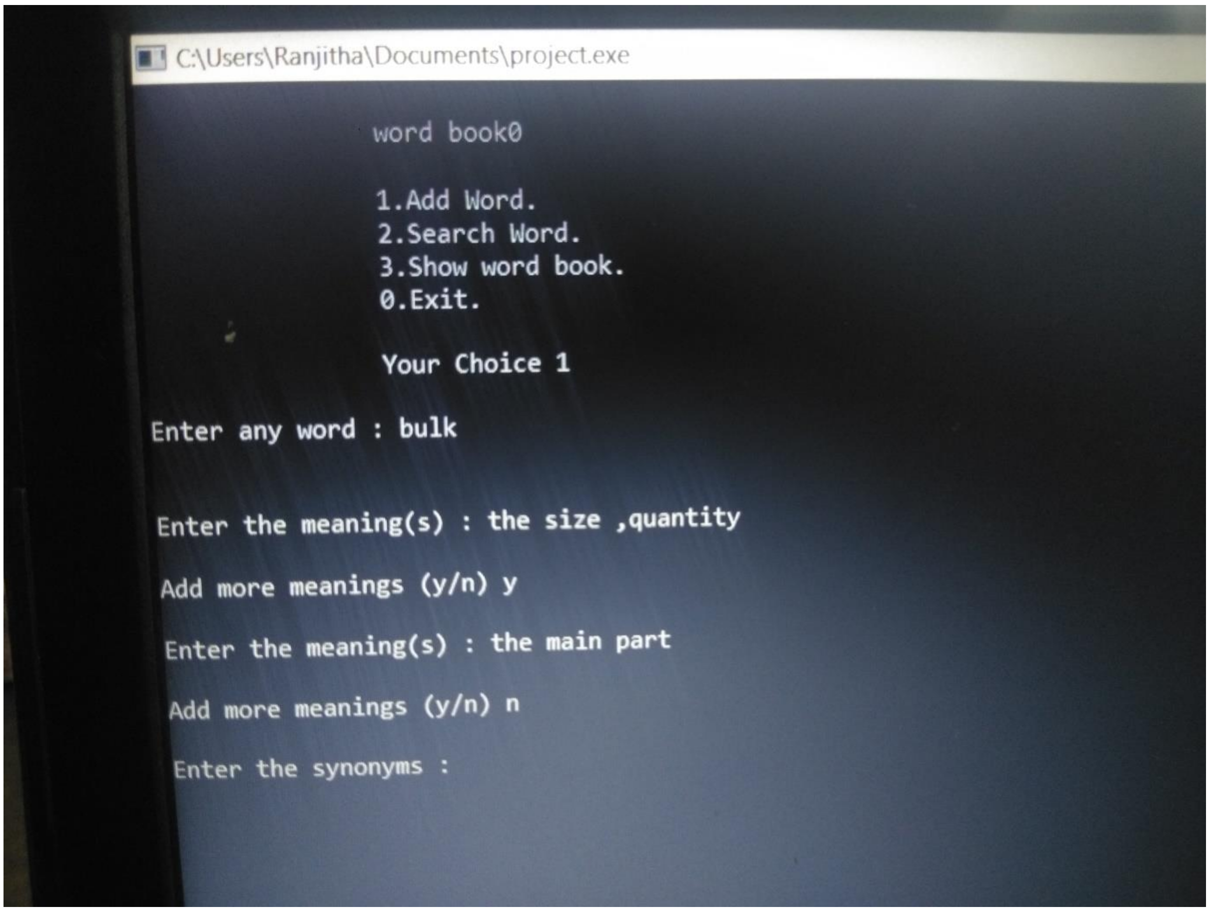
This is the third theme of this project which helps us to display the word dictionary. With the help of this function by giving its suitable choice the can display the dictionary with all the available words the user has added or that that exists in the dictionary. This is also used to check the meaning and will show the number count of synonyms and antonyms.

CHAPTER 5

RESULTS

- Now we are adding a word

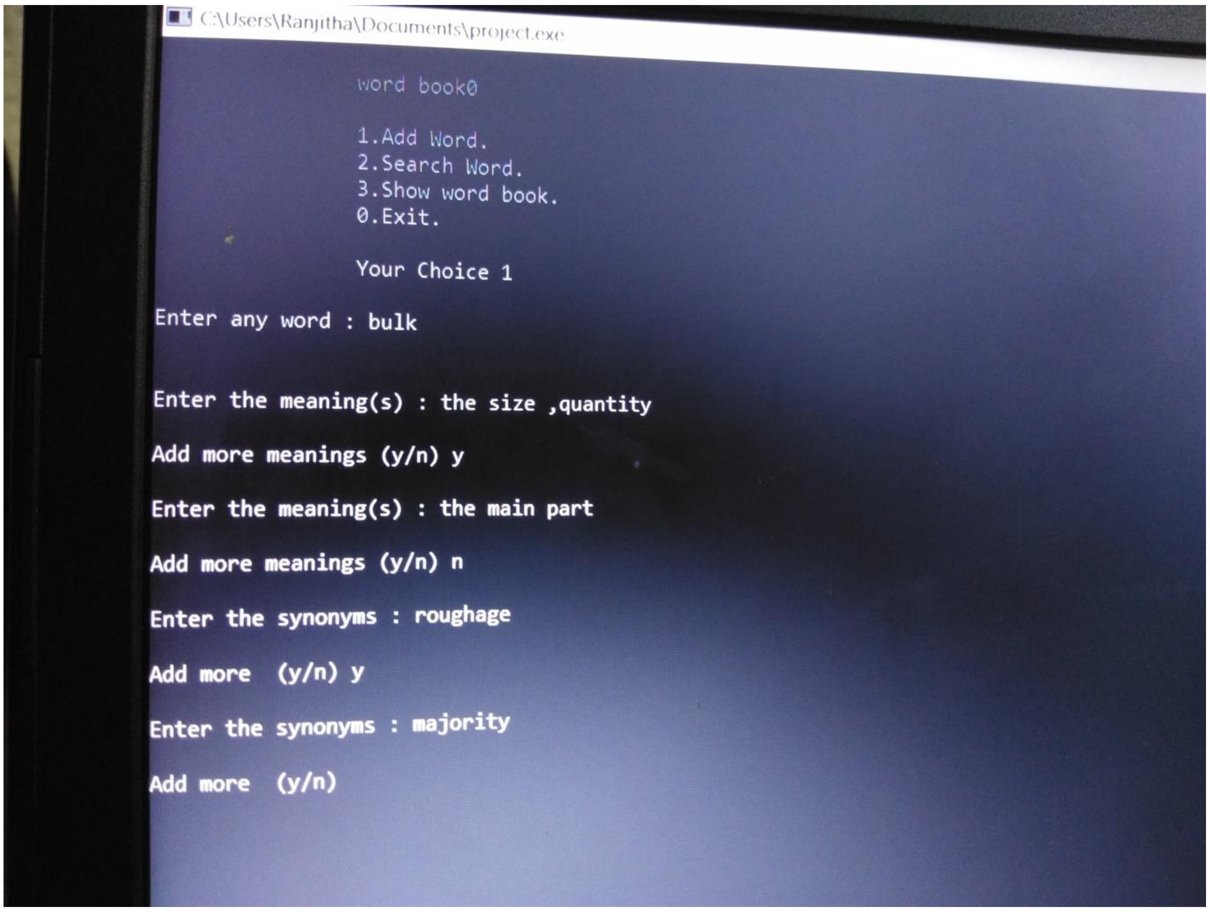
Fig 5.1



```
C:\Users\Ranjitha\Documents\project.exe

                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 1

Enter any word : bulk


Enter the meaning(s) : the size ,quantity

Add more meanings (y/n) y

Enter the meaning(s) : the main part

Add more meanings (y/n) n

Enter the synonyms :
```

- Now adding synonyms

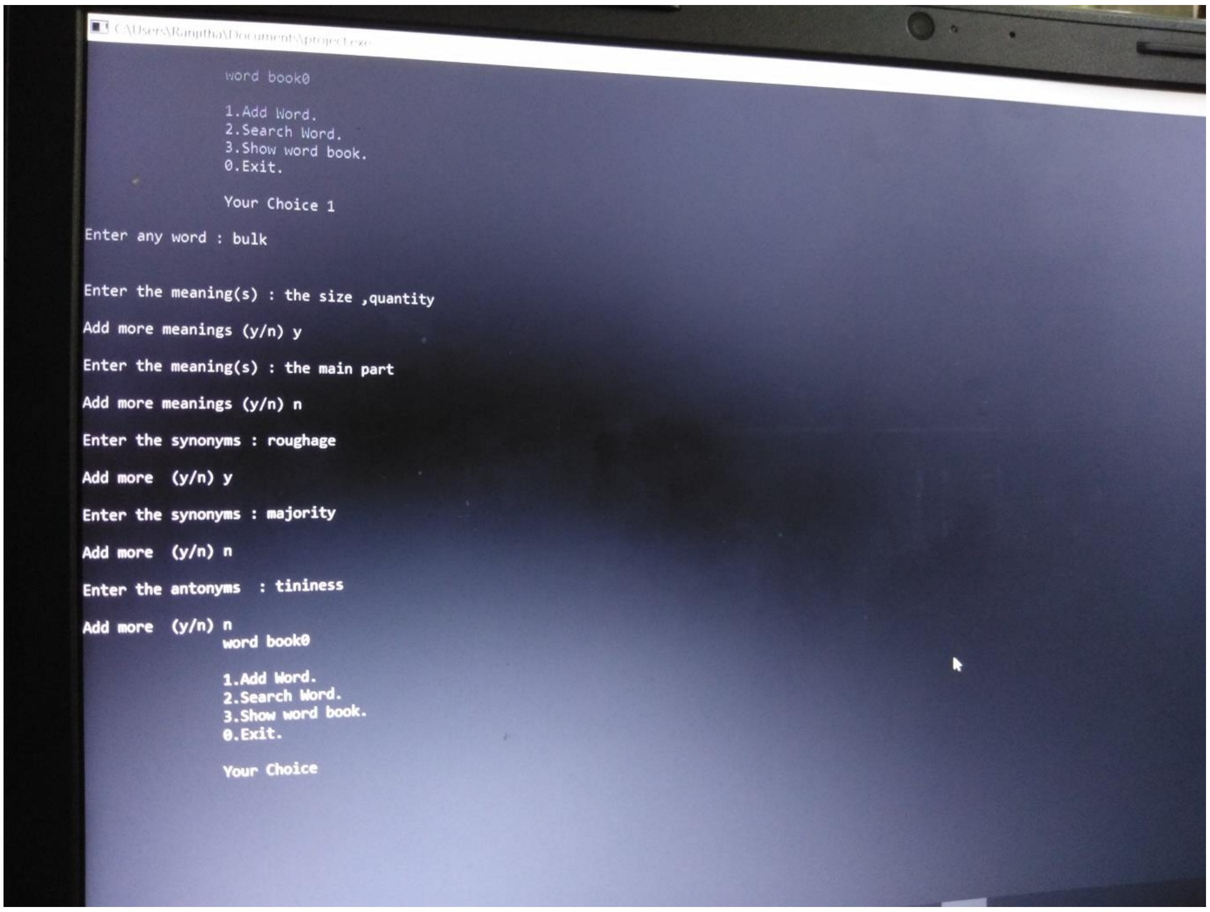
Fig 5.2



```
C:\Users\Ranjitha\Documents\project.exe
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 1
Enter any word : bulk


Enter the meaning(s) : the size ,quantity

Add more meanings (y/n) y

Enter the meaning(s) : the main part

Add more meanings (y/n) n

Enter the synonyms : roughage

Add more  (y/n) y

Enter the synonyms : majority

Add more  (y/n)
```

- Add antonyms and then after u enter that press enter u will get the options again

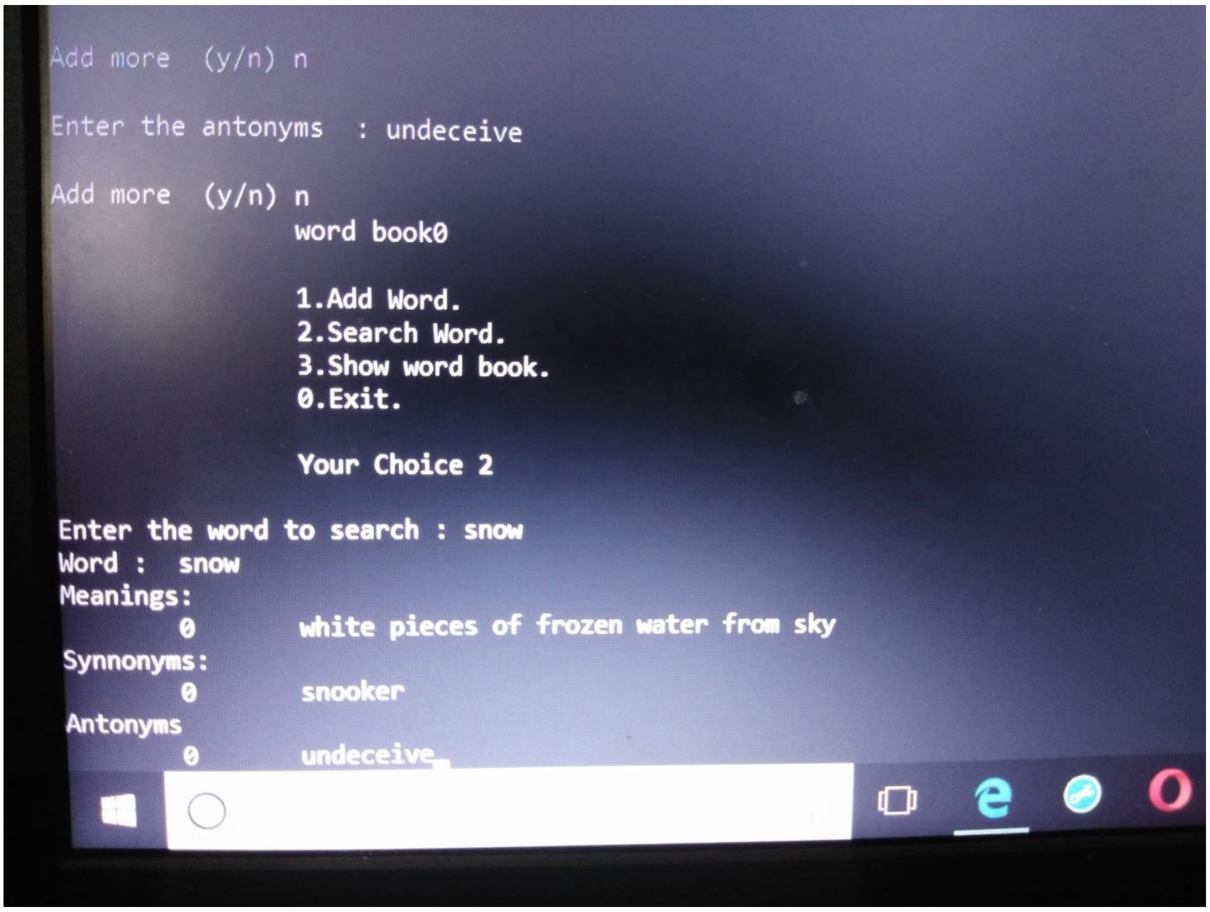
Fig 5.3



```
C:\Users\Ranjitha\Documents\project.exe
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 1
Enter any word : bulk


Enter the meaning(s) : the size ,quantity

Add more meanings (y/n) y

Enter the meaning(s) : the main part

Add more meanings (y/n) n

Enter the synonyms : roughage

Add more  (y/n) y

Enter the synonyms : majority

Add more  (y/n) n

Enter the antonyms  : tininess

Add more  (y/n) n
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice
```

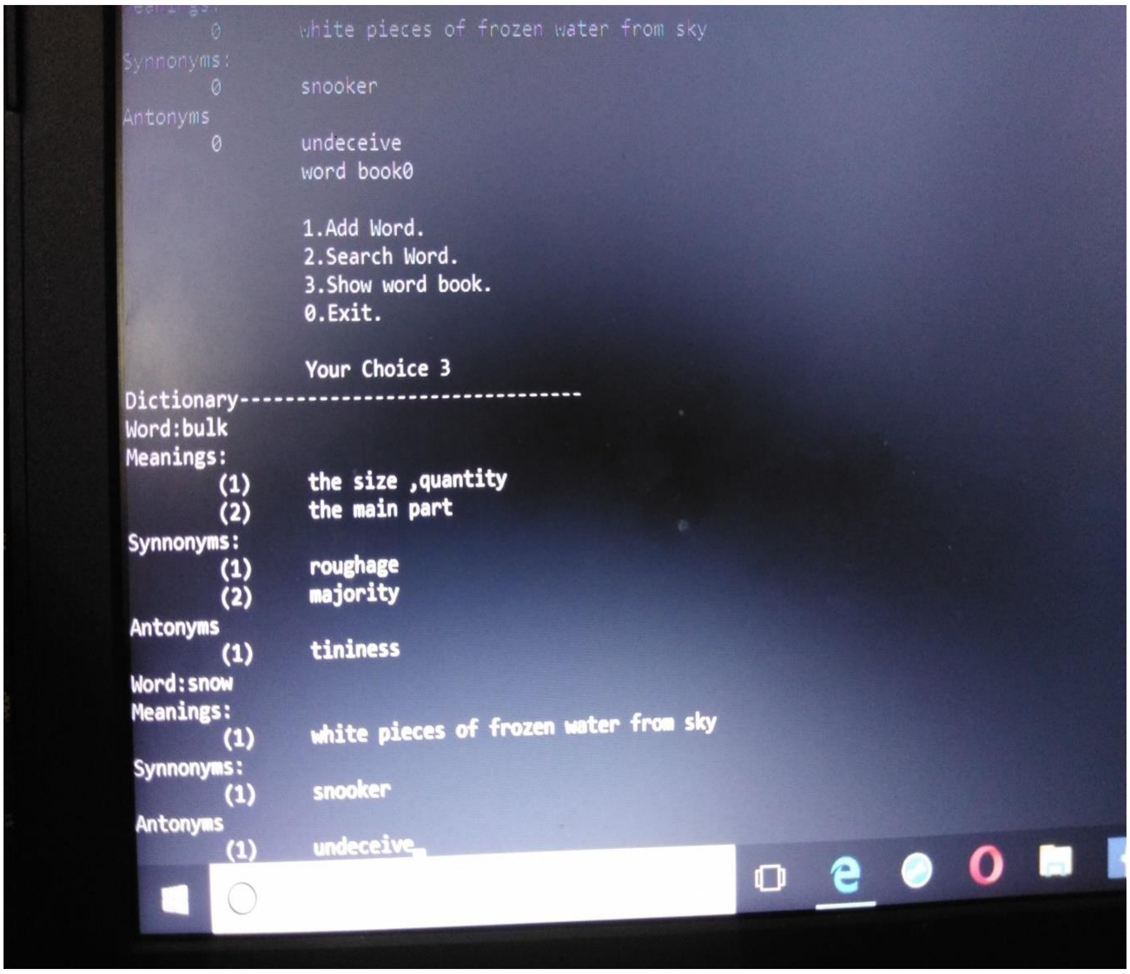
- Now to search a word press option 2

Fig 5.4



```
Add more  (y/n) n

Enter the antonyms  : undeceive

Add more  (y/n) n
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 2

Enter the word to search : snow
Word :  snow
Meanings:
        0       white pieces of frozen water from sky
Synnonyms:
        0       snooker
Antonyms
        0       undeceive_
```

-> show the dictionary

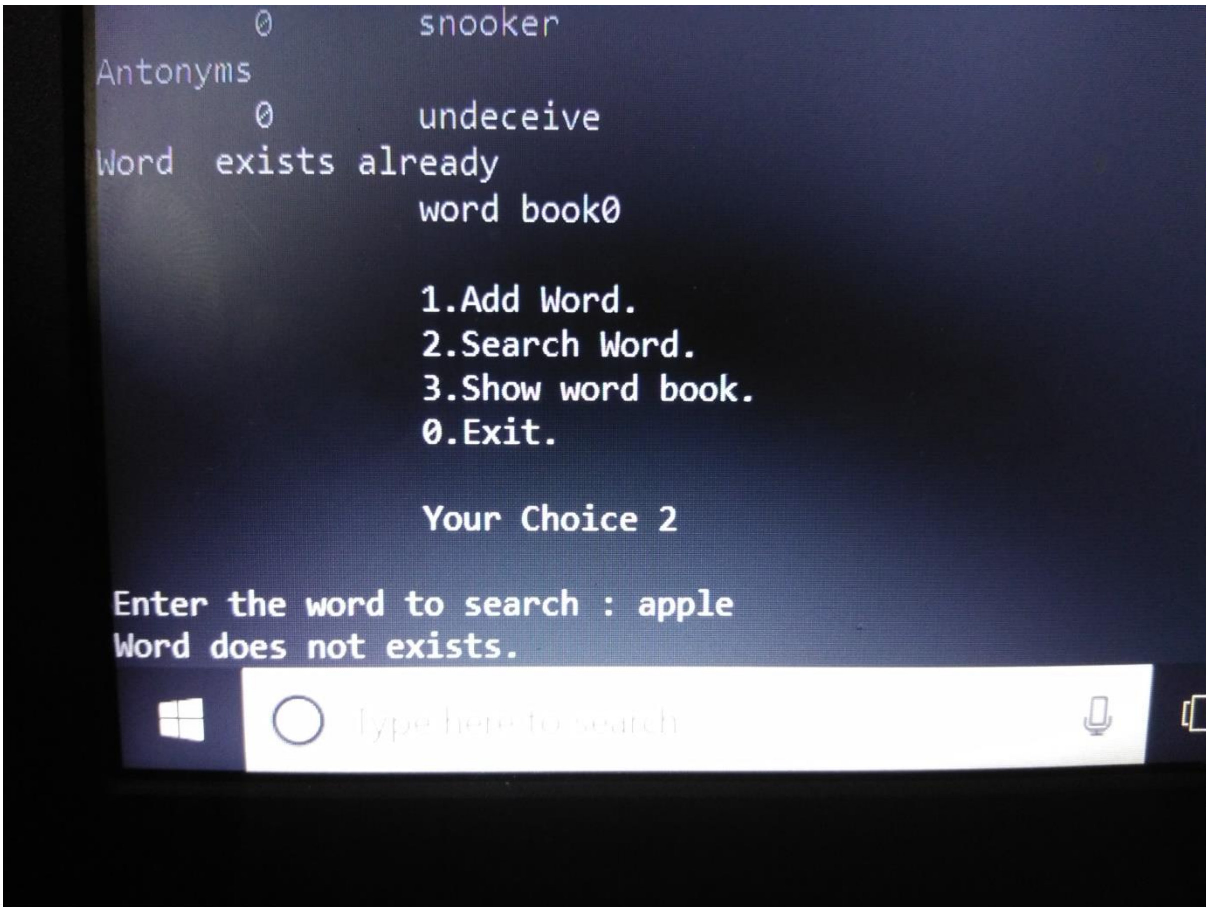


```
        0       white pieces of frozen water from sky
Synnonyms:
        0       snooker
Antonyms
        0       undeceive
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 3
Dictionary-----------------------------
Word:bulk
Meanings:
        (1)     the size ,quantity
        (2)     the main part
Synnonyms:
        (1)     roughage
        (2)     majority
Antonyms
        (1)     tininess
Word:snow
Meanings:
        (1)     white pieces of frozen water from sky
Synnonyms:
        (1)     snooker
Antonyms
        (1)     undeceive_
```

Fig 5.5

It displays all words in the dictionary.

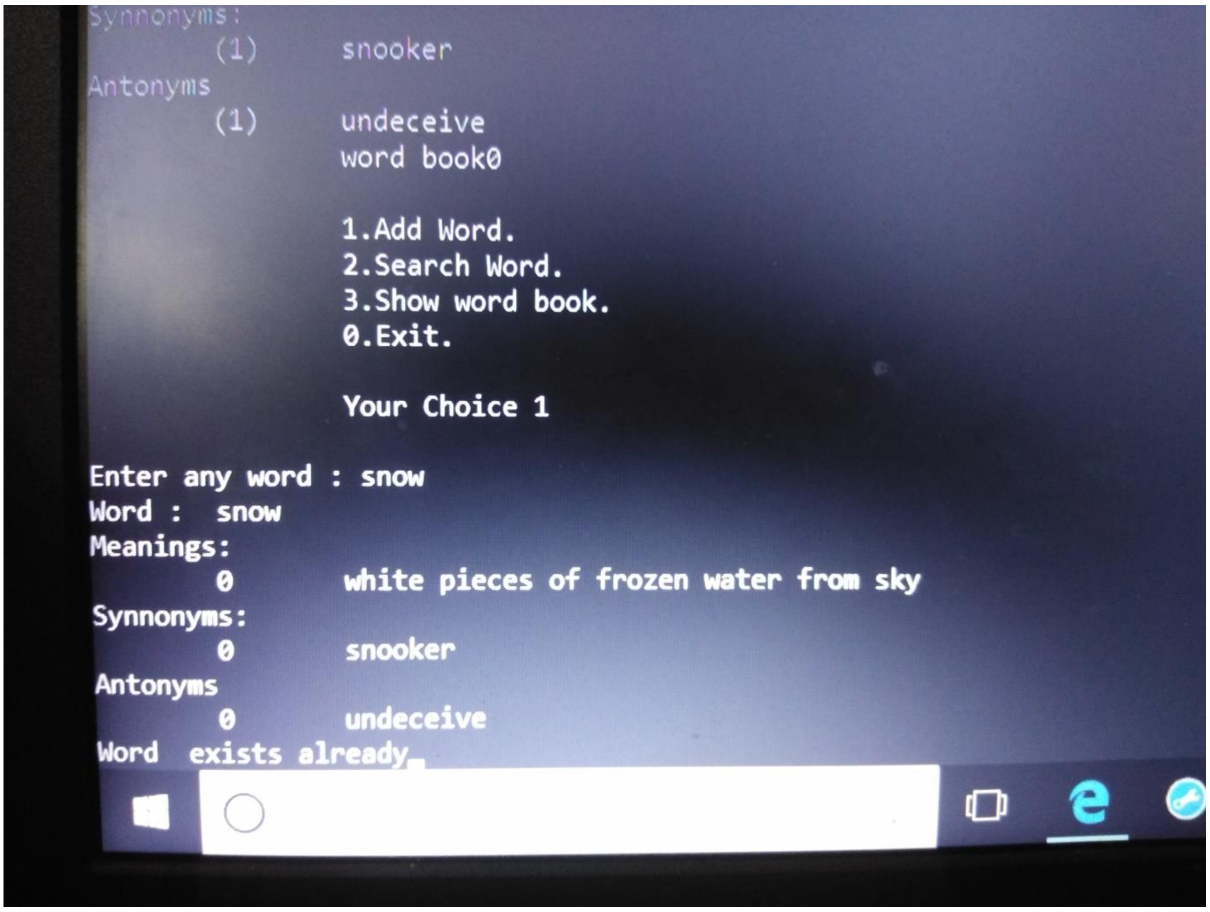
If u want to search a word which doesn't exist in the dictionary we get word doesn't exist



```
        0       snooker
Antonyms
        0       undeceive
Word  exists already
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 2

 Enter the word to search : apple
 Word does not exists.
```

fig 5.6

Now if we want to add a word which already exist we get word exists already.



```
Synnonyms:
        (1)     snooker
Antonyms
        (1)     undeceive
                word book0

                1.Add Word.
                2.Search Word.
                3.Show word book.
                0.Exit.

                Your Choice 1

Enter any word : snow
Word :  snow
Meanings:
        0       white pieces of frozen water from sky
Synnonyms:
        0       snooker
Antonyms
        0       undeceive
Word  exists already_
```

fig 5.7

CHAPTER 6
CONCLUSION

This can be effective tool in teaching English. It can be used as hand-out for the people who get confused in usage of words that are within the same sematic domain. We can create similar features bundle for nouns, adjectives, adverbs and prepositions.
Last but the not the least thing about this project is that it will make an effort to provide an effective and easily available application to all the dictionary users

REFERENCES

Professional c# 2nd Edition . NET v1.0 by Robinson
www.google.co.in
www.dotnetfunda.com
www.w3school.com