NEW HORIZON
COLLEGE OF ENGINEERING

AUTONOMOUS COLLEGE Permanently Affiliated to VTU, Approved by AICTE & UGC
Accredited by NAAC with 'A' Grade

A MINI PROJECT REPORT ON

DICTIONARY USING BINARY SEARCH TREE

Submitted for partial fulfilment for the award of degree

In Bachelor of Engineering

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by,

Name: Nikhil.S.Nambiar

USN: 1NH17BT018

Section: 3B

Reviewed by,

Name: Mr. Puneet.S.Palagi

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 titled

Dictionary using Binary Search tree

Submitted in partial fulfilment for the award of the degree

Of Bachelor of Engineering

Name: Nikhil.S. Nambiar

USN: 1NH17BT018

During the academic year,

2018-2019

Signature of the Reviewer

Signature of HOD

Semester End Examination

Name of the Examiner

Signature with date

1._____ _____

2._____ _____

ABSTRACT

A dictionary is nothing but a list of words from one or more languages, normally ordered alphabetically, explaining each word's meaning, and sometimes containing information on tis pronunciation, usage, translations, and other data.

This project basically allows the user to create the dictionary of its own. The user gets to choose what type of words he/she wants to add in the dictionary. The user can manipulate his dictionary in whichever way he/she wants to. He/she can either insert, delete, search, and print the whole dictionary which he has made. This project is possible only by using Binary search trees. Every word and its meaning is considered as a node of the trees, and based on the options selected by the user, performs the operations respectively. If the user selects insertion, the word and its meaning is considered as one node and then added to the tree based on the conditions specified in the program. If the user wants to delete, the user will be asked which word does he/she wants to delete and based on the cases provided, will delete the element from the tree. When the user wants to search, he/she will be asked the word which they want to search, and the compiler will print the word he/she has searched for. Finally, for traversing, once the option is selected, the whole database will be printed.

This has been one of the simplest methods to create a dictionary in C language using binary search tree. The user has been provided the opportunity to create his/her own dictionary with how many ever words he/she wants.

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 [Reviewer Name], 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.

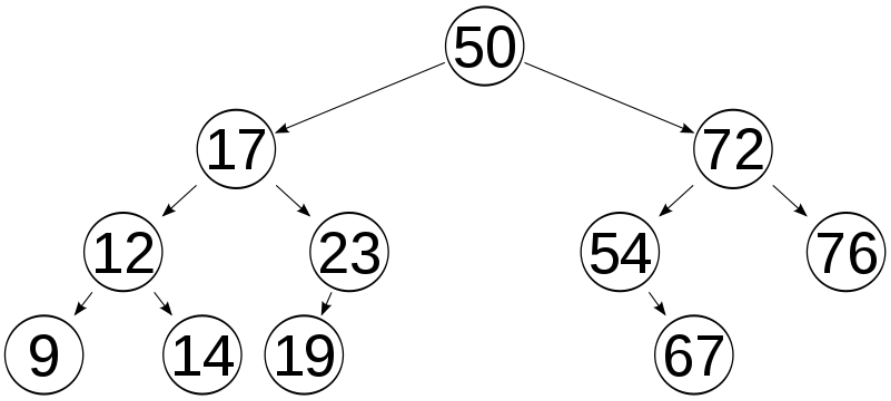
Nikhil.S.Nambiar

1NH17BT018

INDEX

INTRODUCTION

The topic of my mini project is Dictionary using Binary search tree. So let us understand what a binary search tree is.



A binary search tree is a binary tree in which the nodes to the left of the subtree is lesser than the nodes which are on the right of the subtree. Every node should be added while satisfying this condition if a right or a left subtree exists.

1. When inserting or searching for an element in a binary search tree, the key of each visited node should compare with the key of the element to be inserted or found.

2. The shape of the binary search tree depends entirely on the order of insertions and deletions.1

3. After a long sequence of random insertion and deletion, the expected height of the tree approaches square root of the number of keys.

In a binary search tree, 4 operations can be performed. They are,

- Insertion
- Deletion
- Searching
- Traversal

These operations have been elaborated in the Implementation part given below.

ANALYSIS

2.1. Objectives of this project:

The objectives of this mini project is as follows:

- I have basically allowed the user to create his own dictionary and store any number of data he/she requires in the dictionary.
- I have provided 5 options to the user which are, insert node, Delete Node, Search Node, traverse Node, and Exit so that the user can manipulate with his dictionary and can do operations such as inserting any kind of word he/she wants to add, delete any word he/she wants to delete from the dictionary, search any word which is added in the dictionary, and if the user wants, traverse the whole dictionary typed by the user.
- When the user hits Option 1, it asks the user to type a word which has to be inserted. When the word is typed and insert is hit, the node will be added. The node will check all the conditions before it adds the node. If there is no root node, the node added will be the first node. If root is already present, it will check conditions given, and add the node either to the right or to the left o the node.
- If he selects insertion, the program should ask him/her the word to be inserted and once enter is hit, the meaning of the particular word should be provided.
- If he chooses option 2, the program should ask him which word does he/she want to delete from the dictionary. Once the word is typed and the user hits enter, the word should be deleted from the dictionary. If there is no root node, deletion will not take place. Otherwise, it will delete the given word based on the conditions provided in the program.
- If option 3 is chosen, the program should ask the user for the word him/her wants to search. Once the word is typed, it should print the word and its meaning.
- At last, if he chooses option 4, the control should visit each node at least once and print all the nodes the control has gone through during traversing.

- If the user selects the option 5, the control comes out of the otput screen back to the program.
- If an invalid choice is typed, the control should come out of the program and display “invalid option”.

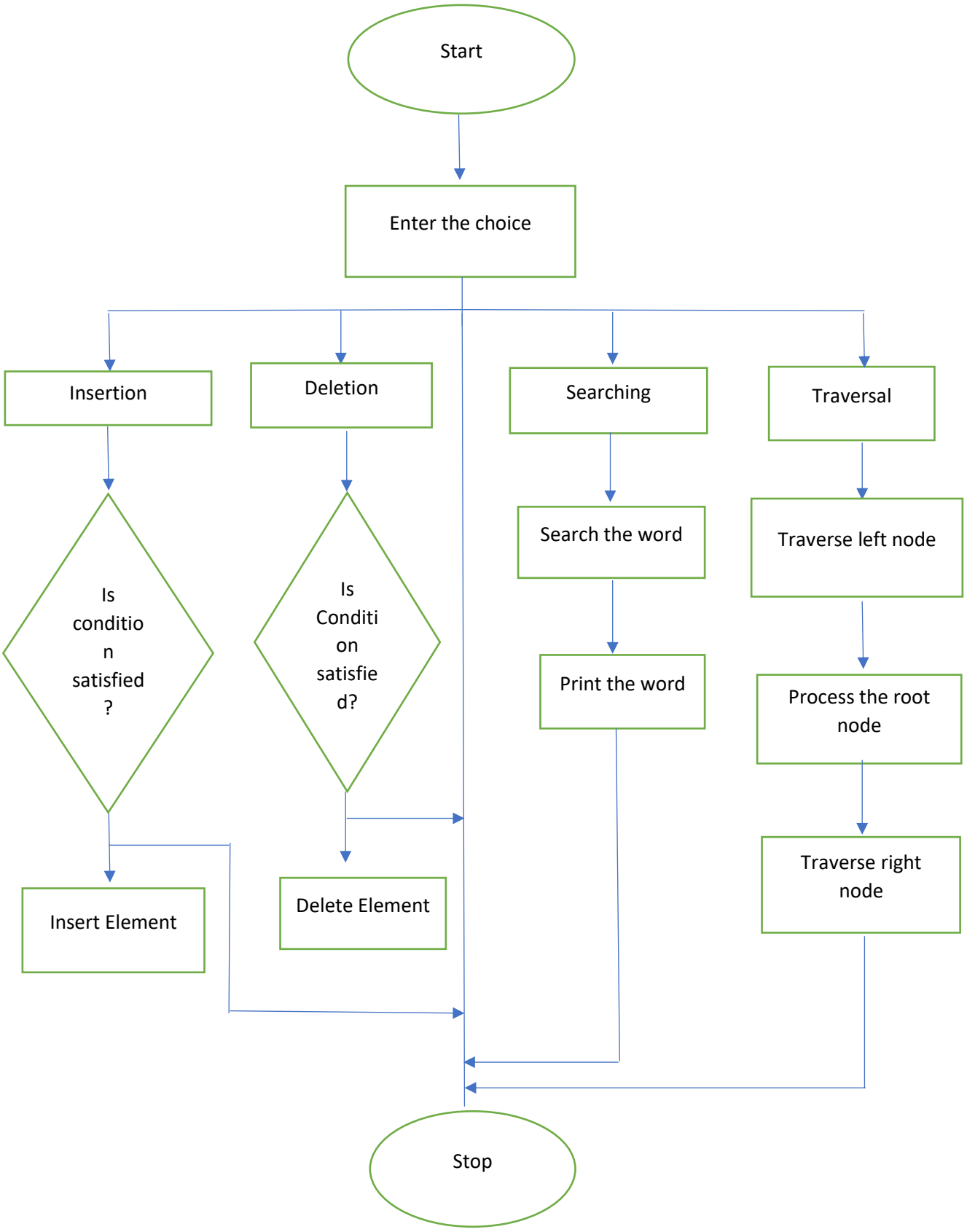
2.2. Requirements:

The requirements of this project are:

- Hardware requirements:
  1. Processor: X86 compatible processor with 1.7 Ghz clock speed
  2. RAM: 512 MB or more
  3. Hard disk: 250 GB or more

- Software requirements:
  1. Windows 7,8,10
  2. Turbo C/C++, Code blocks

Design


Start
Enter the choice
Insertion
Deletion
Searching
Traversal
Is conditio n satisfied ?
Is Conditi on satisfie d?
Search the word
Traverse left node
Print the word
Process the root node
Traverse right node
Insert Element
Delete Element
Stop

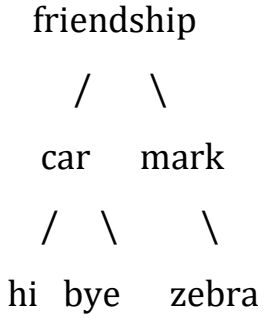
ALGORITHM

An algorithm is a set of rules used to solve problems using calculations through it by a computer. Algorithm is the thing which explains the logic of your whole program. There is no limiting of algorithms for C language. As for easier ways algorithms are implemented in different languages for different purposes. Algorithms are the core of the process task that needs to be implemented.

In my program, the algorithm explains how the insertion, deletion, searching, and traversing takes place in the program. There are 4 algorithms which are used in this program that need to be explained. They are:

1. Insertion algorithm
2. Deletion algorithm
3. Searching algorithm
4. Traversal algorithm

4.1. Insertion algorithm:

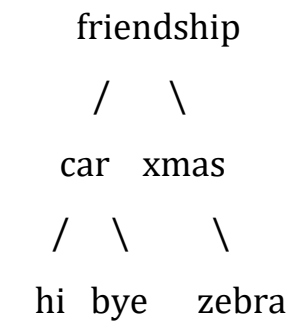


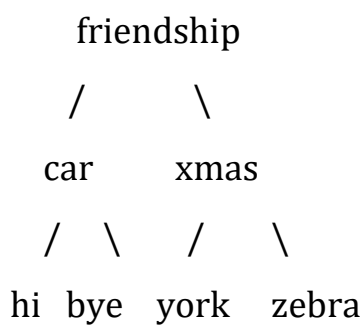
The steps to insert an element in BST are:

- To insert an element in a binary search tree, check whether the root is present or not. If root is absent, then the new node is the root.
- If root node is present, check whether the key in new node is greater than or lesser than the key in root node.

- If key in new node is less than the key in root, then traverse the left sub-tree recursively until we reach the leaf node. Then, insert the new node to the left.
- If the key in new node is greater than the key in root, then traverse the right sub-tree recursively until we reach the leaf node. Then, insert the new node to the right.

Now, suppose we need to insert "York" to the below binary search tree,



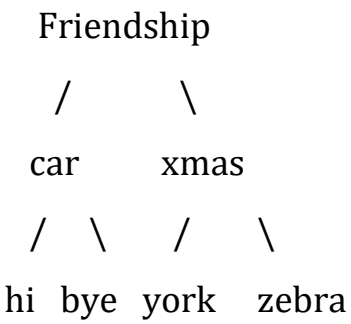


This is how insertion takes place. It is one of the easiest and the simplest way to add an element in a Binary Search tree. It checks for all the conditions before adding the element which helps the compiler add the element in the correct position.

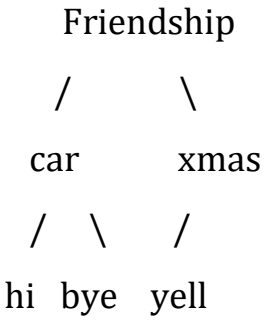
4.2 Deletion algorithm:

There are 3 cases to be seen before deleting a node. They are as follows:

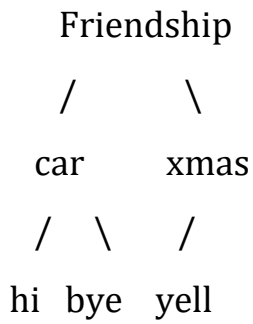
- **Case 1:** When the node has no child: When there is no children, user a pointer to point the node and then delete the node using free () function.



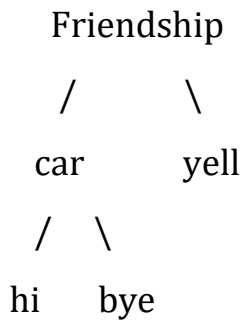
Delete “zebra” from above binary search tree.



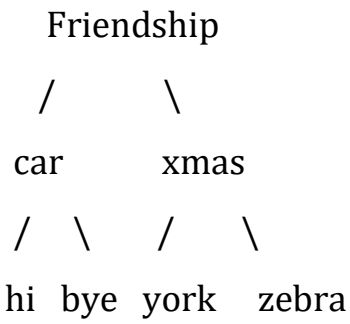
- Case 2: When the node has one child: This step is one of the simplest step in the process of deletion because when there are node nodes present, it is nothing but a leaf node. For this, just use a pointer which will point the leaf node you want to delete, delete the node and then point the child of the node which was deleted to the parent of the node which was deleted.



Now, if we have to delete the word “xmas” from the tree, then, after deletion, the tree becomes like this.



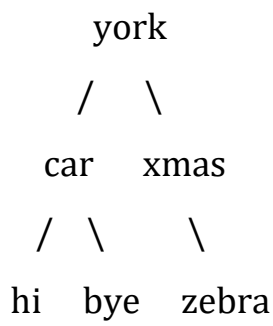
- Case 3: When the node has to 2 children : If the node that has to be deleted has 2 children, then find its successor 'S'. Remove 'S' from the binary search tree. And replace the node that has to be deleted with ‘S’.



Now, suppose we have to delete the word, “Friendship” from the given binary search tree,

- Firstly, Delete the word, “friendship”.
- Secondly, Find the in-order successor of the word, “Friendship”. i.e., York.
- Place “York” in the position of “friendship”.

After following all the rules, the above binary search tree will now become,

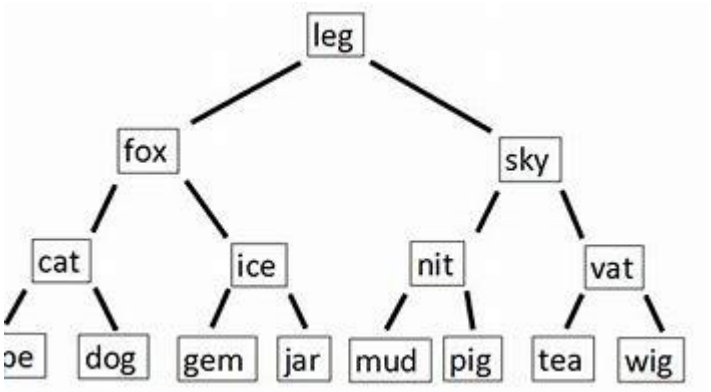


This is how deletion takes place.

4.3 Searching Algorithm:

Searching is one of the simplest operations in binary search tree. The element that has to be searched has to be assigned to a pointer which compares the elements in the tree. Ht pointer will first start traversing from the left subtree. If any of the node matches, the pointer will print the node. If not, The pointer will continue traversing the root node and then moves towards the right subtree until the element searched by the user is not found.

For example,



In the above binary search tree, the user wants to search the word nit, the compiler will start it's searching from the left. If "nit" is not found, the nthe compiler will go to the root node and compare the word with the root node. In this example, "nit" is not same as the root node. So, it shifts to the right, and starts searching. Here, the word "nit" is found to the left of the word "sky". Hence, it will print the word nit saying the word is found.

4.4. Traversal algorithm :

Once the binary search tree has been created, its elements can be retrieved in-order by recursively traversing the left subtree of the root node, accessing the node itself, then recursively traversing the right subtree of the node, continuing this pattern with each node in the tree as it's recursively accessed.

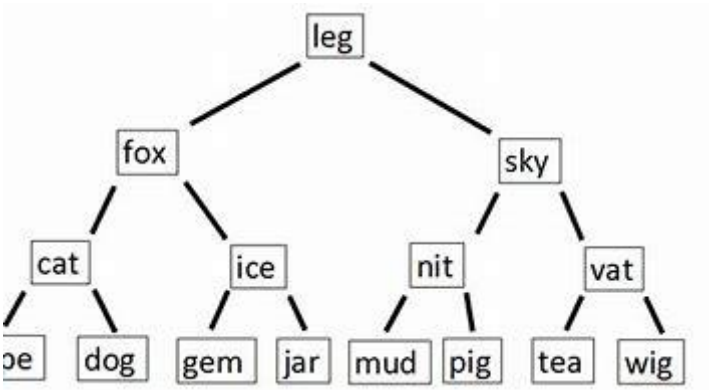
The trace of a traversal is called a sequentialization of the tree. The traversal trace is a list of each visited root. No one sequentialization according to pre-, in- or post-order describes the underlying tree uniquely. Given a tree with distinct elements, either pre-order or post-order paired with in-order is sufficient to describe the tree uniquely. However, pre-order with post-order leaves some ambiguity in the tree structure

As with all binary trees, one may conduct a pre-order traversal or a post-order traversal, but neither are likely to be useful for binary search trees.

An in-order traversal of a binary search tree will always result in a sorted list of node items. Therefore, In-order Traversal always follows these 3 points.

1. Traverse the left node – The compiler will start traversing from the leftmost element of the tree.
2. Process the root node – The compiler will them print the root node.
3. Traverse the right node – The compiler then prints all the elements to the right of the subtree.

Now, let us take the example given below.



If I had to perform In-order traversal for the above tree, firstly, the compiler will go to the word, joe. Then from that word, it will follow the procedure of left-root-right and keep traversing all the left subtrees present in the tree. After all the subtrees are done traversing, it will process the root node, leg. After processing root node, it will again go the word, mud and start performing in-order traversal on all right sub-trees.

So, here compiler will go from “joe” to “cat” and then to “dog”, and after finishing that subtree, moves to “fox” and then “gem”, “ice”, and “jar” get printed. Then it goes to “leg”. From there, The control goes to “mud”, “nit”,” pig ”, then to “sky” and then to “tea”, ”vat”, and “wig”

Therefore, after in-order traversal, the output will be,

Joe, cat, dog, fox, gem, ice, jar, leg, mud, nit, pig, sky, tea, vat, wig.

which prints the BST in an ascending order.

IMPLEMENTATION

I have used C language to implement the code for my mini project.

1. Firstly, I have declared two main pointers *left and *right and I have declared a global pointer called *root and initialized it to NULL.

```
struct BSTnode
{
 char word[150],meaning[250];
 struct BSTnode*left,*right;
}*root=NULL;
```

2. I have created a function named createnode () which creates a node once it's called.

```
struct BSTnode *createnode(char *word,char *meaning)
{
 struct BSTnode*newnode;
 newnode=(struct BSTnode*)malloc(sizeof(struct BSTnode));
 strcpy(newnode->word,word);
 strcpy(newnode->meaning,meaning);
 newnode->left=newnode->right=NULL;
 return newnode;
}
```

3. In the main () function, I have just printed the whole Menu which askes the user about Insertion, Deletion, Traversing, Searching, Exit
4. Which calls the functions once the option is hit.
5. Now in the function insertnode (), insertion takes place. Once the root node is created, when second node is created, I have used a string function called stricmp (), which compares and gives the value 0, -1, +1. If its value is less than 0, the node will be added to the left, or else will be added to the right of the root node. If the value is zero, it prints "Duplicate Entry!"

6. When the deletenode () function is called, the control will delete the node based on the given cases specified in the program.
7. When the function search () is called, the compiler will ask the user to enter the word that has to be searched and then once user types the word and hits enter, the control will compare the word typed and every node in the BST. If the stricmp () function passes the value 0, it will print the word and meaning. If there is some other value, the searching continues until the value zero is passed.
8. When the function traverse () is called, the control will start printing the words and its meaning from the left and then, will print the root node, and then will print all the values which are in the right of the root node.
9. Whne exit option is hit, the exit (0) function is executed, which brings the control back to the program.

Sample Output

6.1 Insertion :

```
Enter the word to be inserted
hi
enter the meaning
hello
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
1
Enter the word to be inserted
tiger
enter the meaning
animal_
```

```
 4.Search node
 5.Exit
1
Enter the word to be inserted
tiger
enter the meaning
animal
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
1
Enter the word to be inserted
sad
enter the meaning
depressed
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
```

```
depressed
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
2
Word : hi
Meaning : hello

Word : sad
Meaning : depressed

Word : tiger
Meaning : animal
```

6.2. Traversal:

```
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
1
Enter the word to be inserted
rat
enter the meaning
mice
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
1
Enter the word to be inserted
smooth
enter the meaning
gentle_
```

```
 5.Exit
1
Enter the word to be inserted
smooth
enter the meaning
gentle
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
1
Enter the word to be inserted
india
enter the meaning
country
Menu
```

```
    enter the meaning
    country
    Menu
     1.Insert Node
     2.Traverse Node
     3.Delete Node
     4.Search node
     5.Exit
    2
    Word : india
    Meaning : country

    Word : rat
    Meaning : mice

    Word : smooth
    Meaning : gentle
```

6.2. Deletion:

```
 4.Search node
 5.Exit
3
enter the word to be deleted
tiger
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
2
Word : hi
Meaning : hello

Word : sad
Meaning : depressed
```

```
Meaning : length

Word : rat
Meaning : mice

Word : tiger
Meaning : animal

Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
3
enter the word to be deleted
boy
The data entered is not found
Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
```

6.3. Searching:

```
Word : hi
Meaning : hello

Word : sad
Meaning : depressed

Menu
 1.Insert Node
 2.Traverse Node
 3.Delete Node
 4.Search node
 5.Exit
4
enter the word to be searched
sad
Word found
Word : sad
Meaning : depressed
```

CONCLUSION

After many efforts, my mini project i.e., Dictionary using binary search tree, is providing the 5 options, and based on the option given by the user, is performing each and every task successfully. On Insertion, the control is adding the word and its meaning in the correct positions satisfying the above conditions. On deletion, it is deleting the nodes following the 3 cases. On searching, it is printing the word which is searched, with its meaning. On traversal, all the nodes are printed in the form of in-order traversal. In this way, this program helps you to make your own dictionary with as many words as possible and perform various operations like deletion, searching a given word, and printing the whole dictionary entered by a user.

BIBLIOGRAPHY

The references I have used for this mini project are:

- www.geeksforgeeks.com
- Data Structures – Padma Reddy
- Some information from www.quora.com
- A concise introduction to data structures – Mark.J. Johnson