PREDICTED WORD DICTIONARY

Prepared by:

Syed Asad Abbas (Group leader)
Huzaifa Awan
Kiran khalid
Javid Ali
Alamgir khan
Samra

Abstract

The creation of a dictionary is very easy in C++. I will show you how to do it in simple steps. Inclusion of header File: The first step is to include the header file stack. In our case, we also use strings to store the names that act as keys so we will also include the header file string. #include<string> #include<string> 2.

➢ PROJECT AIMS AND OBJECTIVES

The objective and basic aim of developing this project is to implement the concepts of DSA. And one of the most important aim is that as nowadays the use of good vocabulary is badly needed and some people from backward areas are lacking in it so it's the initial thought to make things easier for them

➢ FUNCTIONALITY

The functionality of our project that will be achieved after completion of this project are as follows:

- User can add a new word.
- User can search that word by using numbers corresponding to that word.
- The user can view stored words in dictionary
- The user can also delete any particular word and can update the dictionary

➢ Methods and procedures used.

We have used the following functions in our project.

- **Menu()**
- **View dict.()**
- **Search word()**
- **Add word()**
- **Delete word()**
- **Update file()**
- **Renew file()**
- **Make num()**

-

➢ Source code.

```
#include <iostream>
#include <fstream>
#include <stack>
#include <string>
using namespace std;

stack <char> st;

struct Node
{
    string numRef;
    string word;
    Node *next;
};

string makeNum(string s)
{
    char read;
    string line = "";
    int i = 0;
    while(i <= s.length())
    {
        st.push(s[i]);
        read = st.top();
        if(read == 'a' || read == 'A' || read == 'b' || read == 'B' || read == 'c' || read ==
'C')
        {
            line = line+"2";
        }
        if(read == 'd' || read == 'D' || read == 'e' || read == 'E' || read == 'f' || read ==
'F')
        {
            line = line+"3";
        }
        if(read == 'g' || read == 'G' || read == 'h' || read == 'H' || read == 'i' || read ==
'I')
        {
            line = line+"4";
        }
        if(read == 'j' || read == 'J' || read == 'k' || read == 'K' || read == 'l' || read ==
'L')
        {
            line = line+"5";
        }
        if(read == 'm' || read == 'M' || read == 'n' || read == 'N' || read == 'o' || read ==
'O')
        {
            line = line+"6";
        }
        if(read == 'p' || read == 'P' || read == 'q' || read == 'Q' || read == 'r' || read ==
'R' || read == 's' || read == 'S')
        {
            line = line+"7";
        }
        if(read == 't' || read == 'T' || read == 'u' || read == 'U' || read == 'v' || read ==
'V')
        {
            line = line+"8";
        }
```

```
        if(read == 'w' || read == 'W' || read == 'x' || read == 'X' || read == 'y' || read ==
'Y' || read == 'z' || read == 'Z')
        {
            line = line+"9";
        }
        i++;
    }
    return line;
}

void renewFile(string s)
{
    ofstream myFile;
    myFile.open("list.txt", ios :: out);
            if(myFile.is_open())
            {
                myFile << "\n" << s;
                myFile.close();
            }
}

void updateFile(string s)
{
    ofstream myFile;
    myFile.open("list.txt", ios :: app);
            if(myFile.is_open())
            {
                myFile << "\n" << s;
                myFile.close();
            }
}

int searchWord(Node *start)
{
    system("cls");
    string find;
    string numRead;
    bool found = false;
    cout << "\n\t\t\t\t\tEnter Number to search a word : ";
    cin >> find;
    while(start!=NULL)
    {
        if(find == start->numRef)
        {
            cout << "\n\t\t\t\t\t----------------------------\n\t\t\t\t\tWord Found
Successfuly !!!\n";
            cout << "\t\t\t\t\tWord is : " << start->word << "\n\t\t\t\t\tAgainst Number : "
<< start->numRef << "\n\t\t\t\t\t----------------------------\n";
            found = true;
        }
        if(start->next == NULL && find != start->numRef && found == false)
        {
            string append;
            if(found == true)
            {
                return 0;
            }
            else
            {
                cout << "\n\t\t\t\t\tOOPS! Word Not Found.\n\t\t\t\t\tPlease Enter the word
you meant to Search : ";
            cin >> append;
            string NR;
            updateFile(append);
```

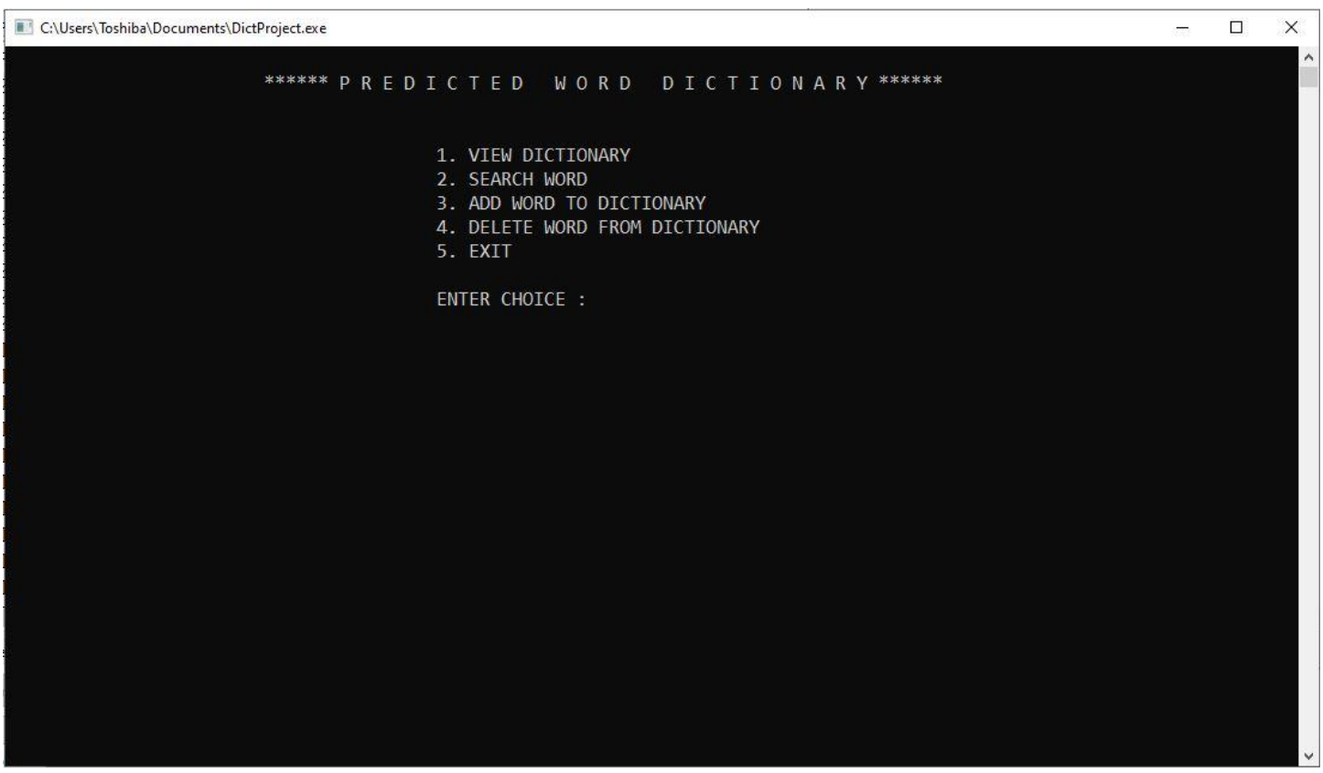
```
            ifstream myFile2;
            myFile2.open("list.txt", ios :: in);
            if(myFile2.is_open())
            {
                string line;
                while(getline(myFile2, line))
                {
                    NR = makeNum(line);
                    Node *curr = new Node;
                    curr->word = line;
                    curr->numRef = NR;
                    curr->next = start->next;
                    start->next = curr;
                }
                myFile2.close();
                return 0;
            }
            }


        }
        start = start->next;

    }
}

void viewDict(Node *start)
{
    system("cls");
    cout << "\n\t\t\t\t***   D I C T I O N A R Y   D I S P L A Y   ***\n\n";
    while(start!=NULL)
    {
            cout << "\n\t\t\t\t\t----------------------------\n";
            cout << "\t\t\t\t\tWord in Dictionary is : " << start->word <<
"\n\t\t\t\t\tAgainst Number : " << start->numRef << "\n\t\t\t\t\t----------------------------
\n";

        start = start->next;

    }
}

int addWord(Node *start)
{
    system("cls");
    cout << "\n\t\t\t\t***   A D D   W O R D   T O   D I C T I O N A R Y   ***\n\n";
    string find;
    bool found = false;
    cout << "\n\t\t\t\t\tEnter Word to Add : ";
    cin >> find;
    while(start!=NULL)
    {
        if(find == start->word)
        {
            cout << "\n\t\t\t\t\tWORD '" << find << "' ALREADY EXISTS IN THE
DICTIONARY!!!\n\n";
            found = true;
        }
        if(start->next == NULL && find != start->numRef && found == false)
        {
            string append = find;
            string NR;
            updateFile(append);

            ifstream myFile2;
```

```
            myFile2.open("list.txt", ios :: in);
            if(myFile2.is_open())
            {
                string line;
                while(getline(myFile2, line))
                {
                    NR = makeNum(line);
                    Node *curr = new Node;
                    curr->word = line;
                    curr->numRef = NR;
                    curr->next = start->next;
                    start->next = curr;
                }
                myFile2.close();
                cout << "\n\t\t\t\t\tWORD ADDED SUCCESSFULLY!\n";
                return 0;
            }
        }
        start = start->next;

    }
}

int deleteWord(Node *start)
{
    system("cls");
    string find, upd;
    Node *head = start;
    Node *save = start;
    cout << "\n\t\t\t\t\tEnter Word to Delete : ";
    cin >> find;

    while(start!=NULL)
    {
        if(find == start->word)
        {
            start = start->next;
            cout << "\n\t\t\t\t\tWORD DELETED SUCCESSFULLY!\n";
            start = head;

            while(start!=NULL)
            {
                upd = start->word;
                renewFile(upd);
                start = start->next;
            }

        }
        else
        {
            start = start->next;
        }
    }
    cout << "\n\t\t\t\t\tFILE UPDATED SUCCESSFULLY!\n\n";
}

void menu(Node *start)
{
    system("cls");
    int choice;
    cout << "\n\t\t\t*** P R E D I C T E D   W O R D   D I C T I O N A R Y ***\n\n\n";
    cout << "\t\t\t\t1. VIEW DICTIONARY\n\t\t\t\t\t2. SEARCH WORD\n\t\t\t\t\t3. ADD WORD TO
DICTIONARY\n\t\t\t\t4. DELETE WORD FROM DICTIONARY\n\t\t\t\t\t5. EXIT\n\n\t\t\t\tENTER
CHOICE : ";
    cin >> choice;
```

```
    switch(choice)
    {
        case 1:
            {
                viewDict(start);
                break;
            }
        case 2:
            {
                searchWord(start);
                break;
            }
        case 3:
            {
                addWord(start);
                break;
            }
        case 4:
            {
                deleteWord(start);
                break;
            }
        case 5:
            {
                cout << "\n\t\t\t\t\tE X I T I N G   P R O G R A M\n\n";
                exit(0);
            }
        default:
            {
                cout << "\n\t\t\t\t\tOOPS! WRONG INPUT...\n\n";
                menu(start);
            }
    }
    char opt;
    cout << "\n\n\t\t\t\tDO YOU WANT TO PERFORM MORE OPERATIONS? (y/n) : ";
    cin >> opt;
    if(opt == 'y' || opt == 'Y')
    {
        menu(start);
    }
    else
    {
        cout << "\n\t\t\t\t\tE X I T I N G   P R O G R A M\n\n";
        exit(0);
    }
}



int main()
{
    Node *head = NULL;
    char read;
    string name;
    string numRead;
    numRead = makeNum(name);
    cout << "\nPREDICTED WORD DICTIONARY\n";

    ifstream myFile;
    myFile.open("list.txt", ios :: in);
    if(myFile.is_open())
    {
        string line;
        while(getline(myFile, line))
```

```
        {
            name = line;
            numRead = makeNum(name);
            Node *node = new Node;
            node->numRef = numRead;
            node->word = name;
            node->next = head;
            head = node;
        }
        myFile.close();
    }

    menu(head);

}
```

➢ Output.



C:\Users\Toshiba\Documents\DictProject.exe

```
*****   A D D   W O R D   T O   D I C T I O N A R Y   *****

        Enter Word to Add : Hamdard

        WORD ADDED SUCCESSFULLY!

DO YOU WANT TO PERFORM MORE OPERATIONS? (y/n) :
```

C:\Users\Toshiba\Documents\DictProject.exe

```
****** P R E D I C T E D   W O R D   D I C T I O N A R Y ******

        1. VIEW DICTIONARY
        2. SEARCH WORD
        3. ADD WORD TO DICTIONARY
        4. DELETE WORD FROM DICTIONARY
        5. EXIT

        ENTER CHOICE : 5

        E X I T I N G   P R O G R A M

--------------------------------
Process exited after 4.674 seconds with return value 0
Press any key to continue . . .
```

C:\Users\Toshiba\Documents\DictProject.exe

```
*****  S E A R C H I N G   W O R D   F R O M   D I C T I O N A R Y   *****

       Enter Number to search a word : 526

       ----------------------------
       Word Found Successfuly !!!
       Word is : Jam
       Against Number : 526
       ----------------------------

DO YOU WANT TO PERFORM MORE OPERATIONS? (y/n) :
```

C:\Users\Toshiba\Documents\DictProject.exe

```
*****  D E L E T I N G   W O R D   F R O M   D I C T I O N A R Y   *****

       Enter Word to Delete : Jam

       WORD DELETED SUCCESSFULLY!
```

C:\Users\Toshiba\Documents\DictProject.exe

```
*****  D I C T I O N A R Y   D I S P L A Y   *****

       ----------------------------
       Word in Dictionary is : Chiku
       Against Number : 24458
       ----------------------------

       ----------------------------
       Word in Dictionary is : Jam
       Against Number : 526
       ----------------------------

       ----------------------------
       Word in Dictionary is : Banana
       Against Number : 226262
       ----------------------------

       ----------------------------
       Word in Dictionary is : Fruit
       Against Number : 37848
       ----------------------------

       ----------------------------
       Word in Dictionary is : Apple
       Against Number : 27753
       ----------------------------

       ----------------------------
```