Autocomplete

Project

Documentation

**Team 17**

**1-Khaled Mhmoud ELsayed sec(7)**

**2-Mostafa Ahmed Mostafa sec(16)**

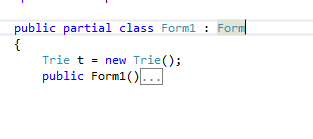
**3-Bassem Yossef Bekhit sec(6)**

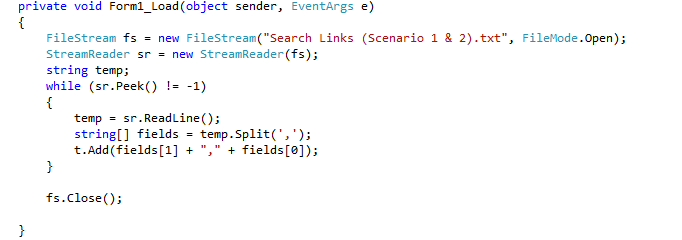
**4-Muhammed Mohy Hassan sec(15)**

* Prefix search

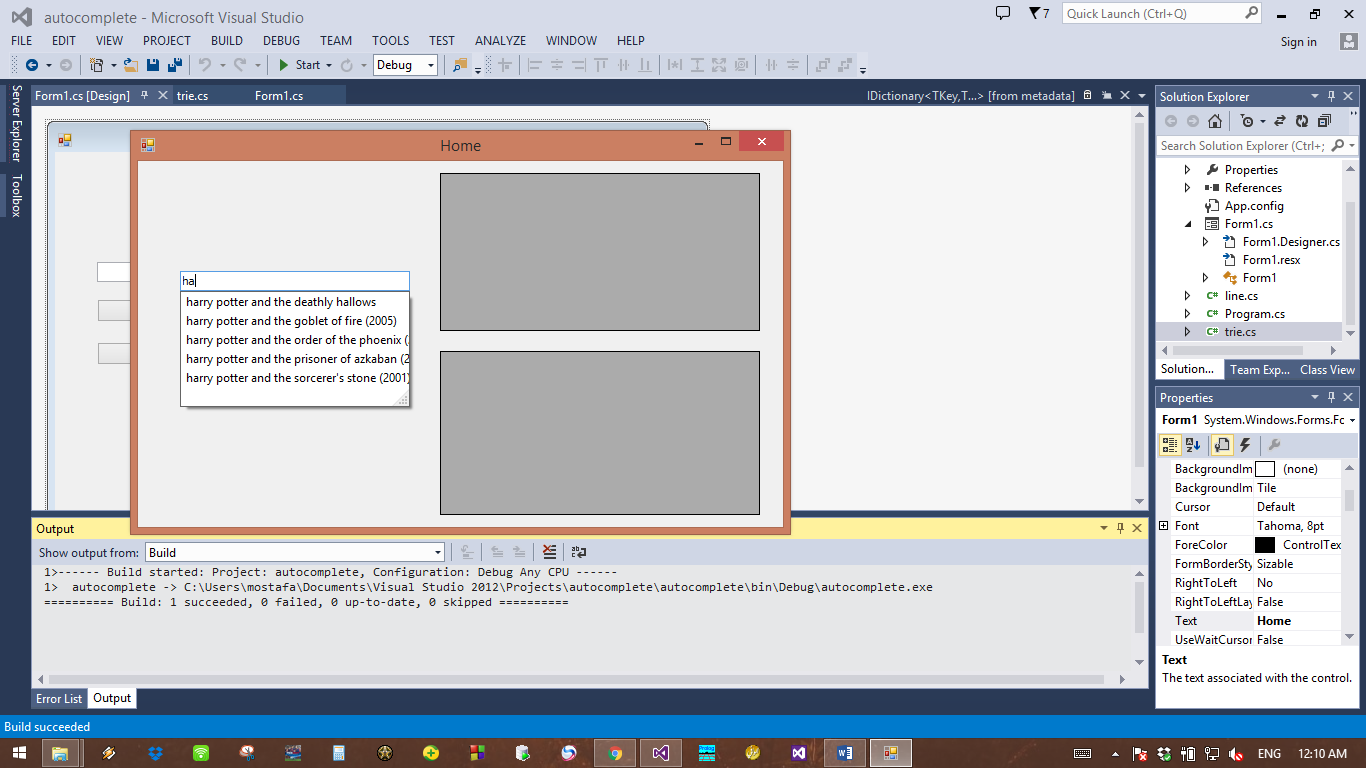
We used a trie data structure to store strings from a file.

At the running time we fill our trie’s object from a file

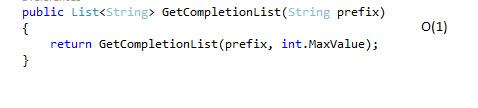


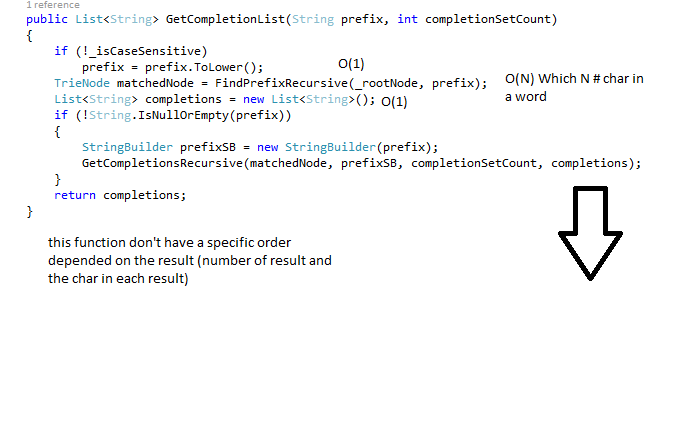


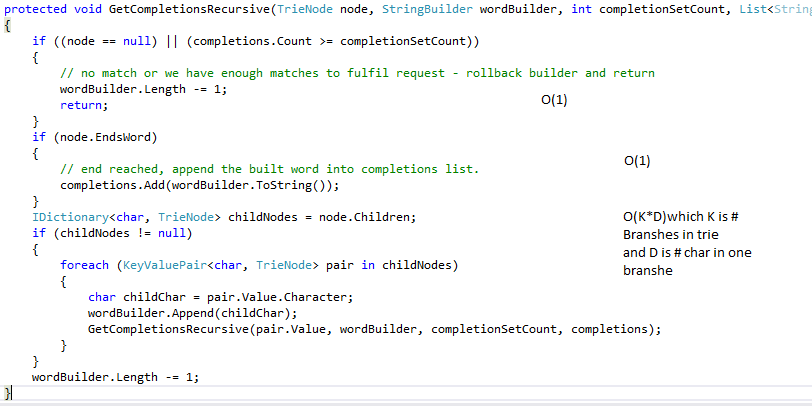
Then when the user begin to enter a word the autocomplete will give him results.

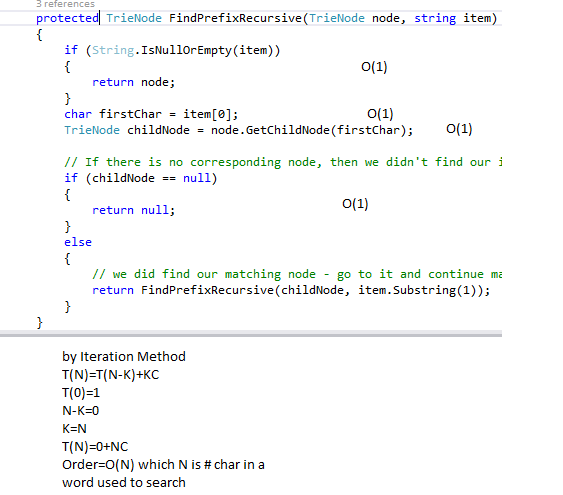


* **Analysis of prefix search code**



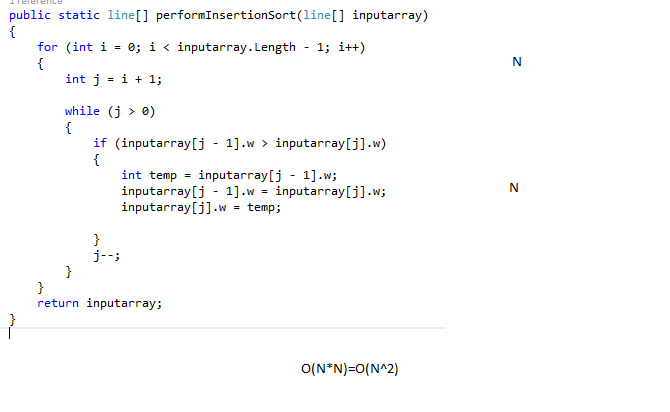




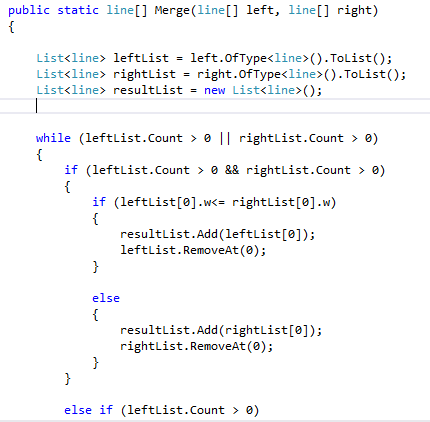


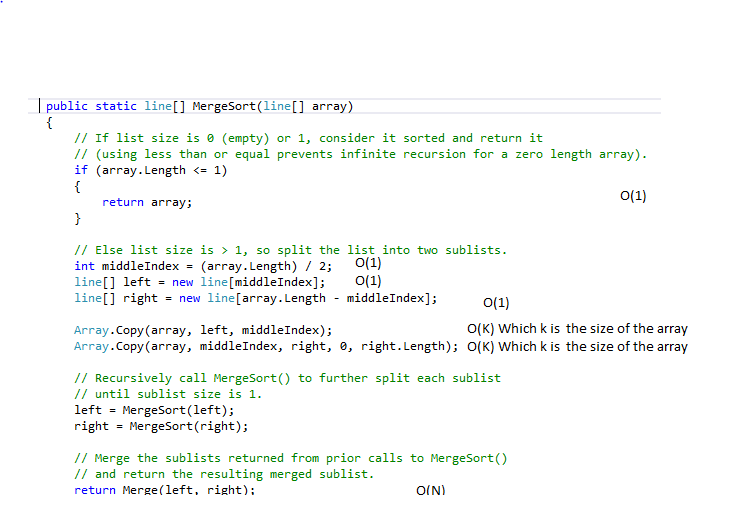
* Sorting

Insertion sort:



Merge sort:

O(N)



T(N)=2T(N/2)+N

F(n) vs N^log a base b

N vs N

Case2 in Master method

Order = f(n)logN=NlogN

O(NlogN)

\***Button of check**

-Check from dictionary:

-It take your sentence and split it with space

-Then

-Pass the word to check function

-And load dectionary at list to check word if found

-sorting dictionary with O(nlogn )

-if it founded the program show message box that your sentence is true

**\*Cont –Button of check**

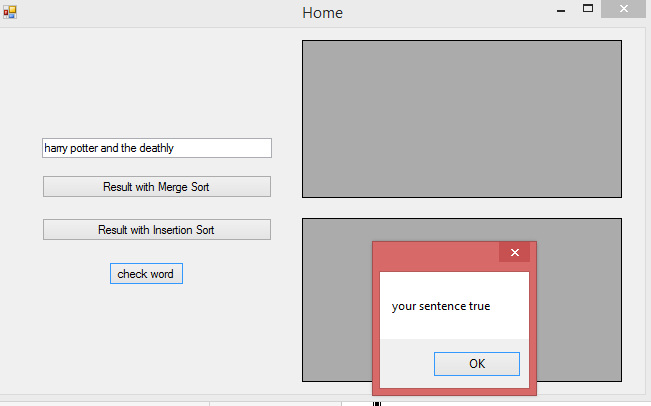
-else

- the program show message box that your sentence is false

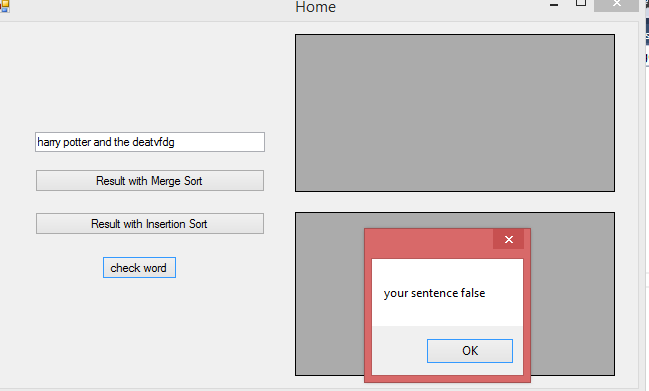
If there is at least one word in sentence not found in dictionary

-the function take time O(logn ) N is number of word in dictionary

\*\*\*true sentence



\*\*\* false sentence



* Spiliting word by spaces:-
* w = (textBox1.Text);

- string[] sss = w.Split(' ');//// O(1)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

* Reading file of dictionary in list
* FileStream fs = new FileStream("dictionary.txt", FileMode.Open); StreamReader fr = new StreamReader(fs);
* List<string> l = new List<string>();
* while (fr.Peek() != -1)
* {
* string st = fr.ReadLine();
* l.Add(st);
* }
* fr.Close();
* fs.Close(); //O(N) //Nis num of words in file
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* Sorting dictionary using list.sort()
* L.sort(); //O(nlogn)
* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*
* Searcing using binary search //O(logn)

int first = 0;

int last = l.Count - 1;

cont searching using binary search

if (w.Length > 0)

{

while (first <= last)

{

int mid = (last + first) / 2;

string com = l[mid];

if (com.CompareTo(w) < 0)

{

first = mid + 1;

}

else if (com.CompareTo(w) > 0)

{

last = mid - 1;

}

else

return true;

}

Checking true or false sentence //O(1)

bool f = check(sss[i]);

if (!f) //O(1)

{

count = 1;

}

}

if (count == 1) //O(1)

{

MessageBox.Show("your sentence false");

}

else

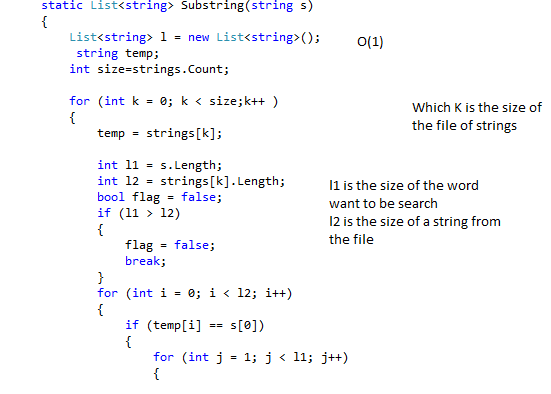
{ MessageBox.Show("your sentence true"); } //O(1)

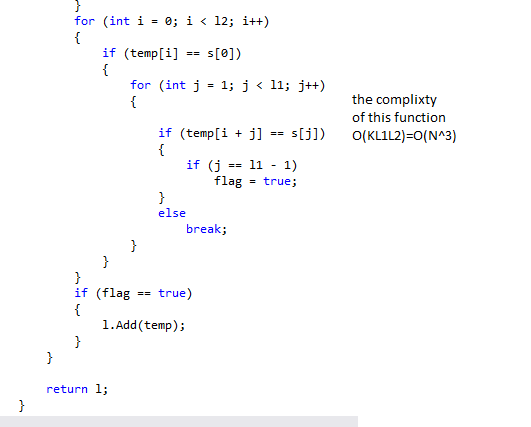
}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Substring function:





(Suggestion) edit word function :

