TRF Robospark'21

Final Task Report

Project Name: Spell Checker

Group: Tejas Katkade (SY)

Pratik Dhamal (SY)

Aditya Warghane (SY)

Project Github Link: FT Spell Checker

Project Algorithm:

- Get user input and store it in a string.
- Create a hash table with unordered set. In that make a set of strings called *dict* and store the contents of the dictionary in *dict* with file handling.
- Check the user input with all words in dictionary. If input is correct, display so, else go to all the suggestion functions.
- Type 1. Missing character in a word (eg. stck for stack).

  In a loop from 'a' to 'z', add each alphabet to the end of the input (eg. stcka, stckb and so on).

  Keeping first alphabet as it is, sort the remaining letters (eg. sackt).

  Check over whole dictionary if any word, which when sorted with first alphabet as it is, matches the word generated in above step. If matches, the return that word from dictionary as suggestion.

- Type 2. Extra letter in word (eg. staick for stack)

  In this case, in a loop, remove a letter from input starting from 1st index (eg. saick, stick etc.) and check if the remaining word is in the dictionary or not. If it is, suggest that word.

- **Type 3. One extra word and one missing word(eg. srack for stack)**

  Here we remove each alphabet (except 1st) in a loop and at the same time, add an alphabet from 'a' to 'z' to that word (eg. sacka).
  Then for each such iteration, we call the function used for suggestion 1.
  At any point if condition for suggestion 1 is satisfied, that word from dictionary is suggested.

- **Type 4. One letter relaced with another(eg. atack for stack).**

  Here we replace each letter of the input with the alphabet series and then compare with dictionary. (eg. btack ctack and so on).

- **Type 5. Incorrect Arrangement (eg. stkca for stack)**

  Sort all words of the input except the first. And match that sorted word with the sorted words in the dictionary. If they match then suggest that word.

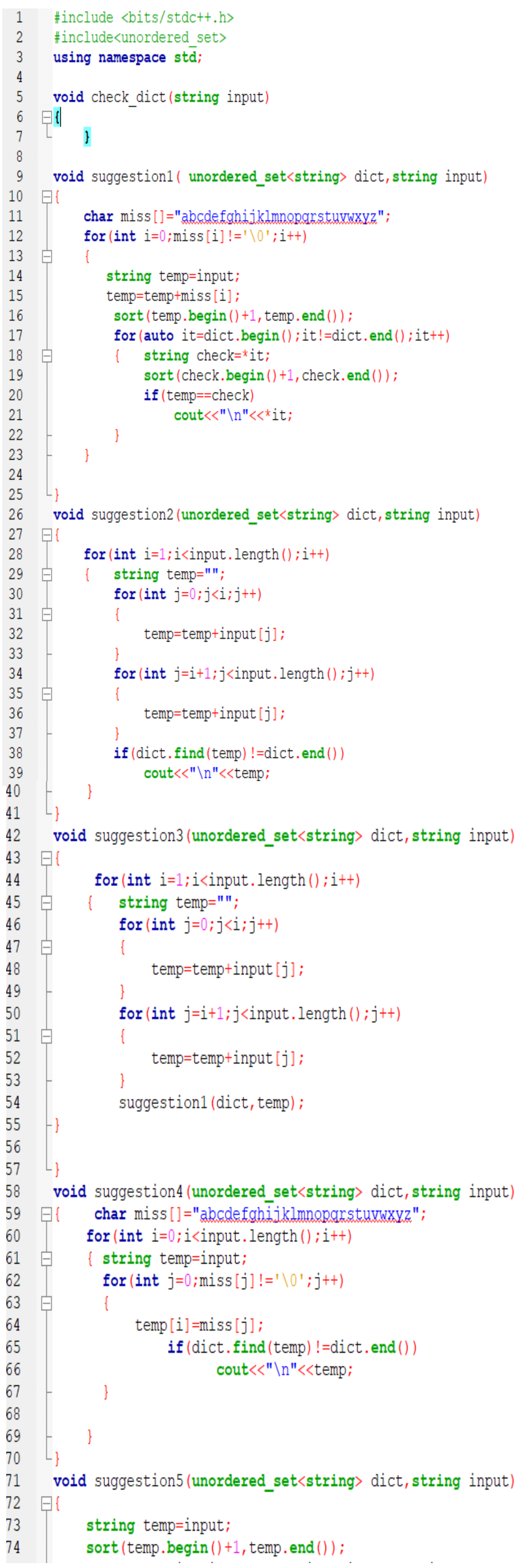
Problems Faced:

- **Thinking for various ways to write the program.**
- **As Python was not allowed, we had to use C/C++.**
- **We had to see tutorials for various topics like hashing, etc. before starting work.**

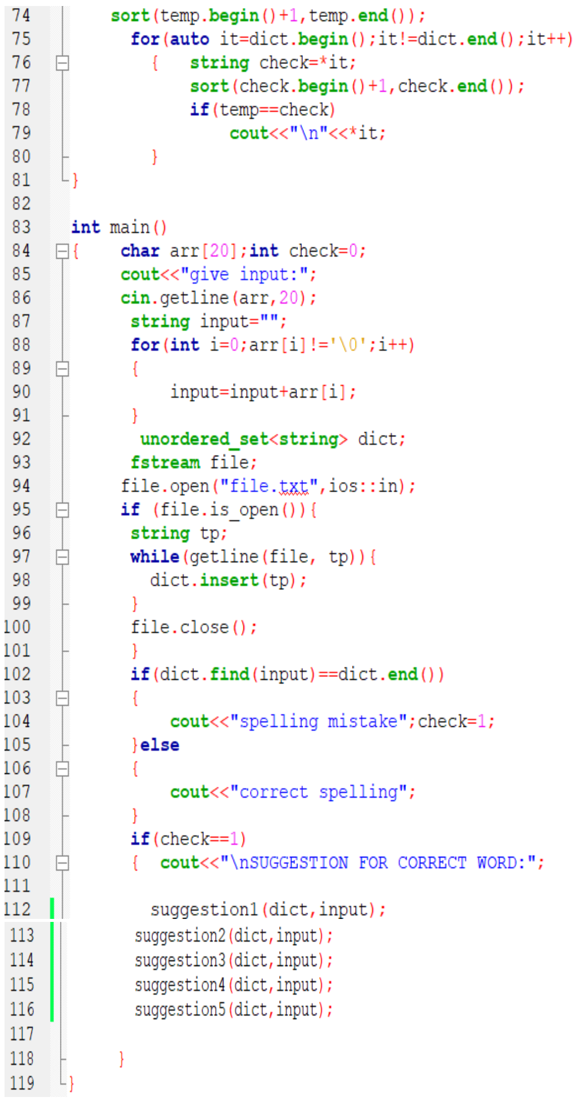
Alternative Solutions:

1) **Without using any special functions of C++, we made a spell checker that has only simple functions.**
2) **Using hash table. We can assign each string with its ascii value and add them to a bucket. Whenever an input is given, program will check the ascii value of the string and search it in hash table and give the nearest suggestion of that ascii value.**

Code Snippet ss:

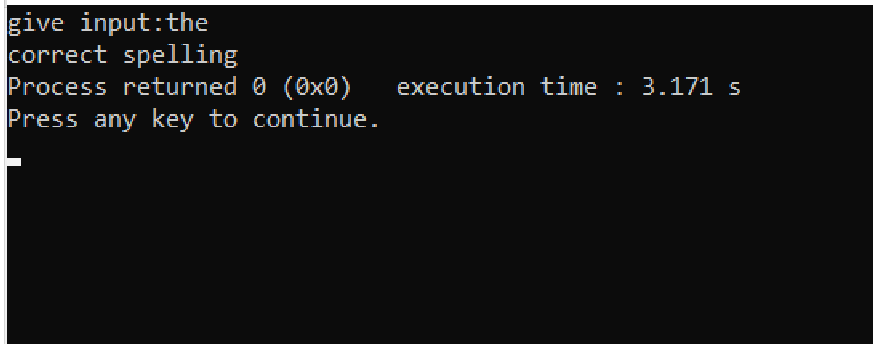


```cpp
#include <bits/stdc++.h>
#include<unordered_set>
using namespace std;

void check_dict(string input)
{
    }

void suggestion1( unordered_set<string> dict,string input)
{
    char miss[]="abcdefghijklmnopqrstuvwxyz";
    for(int i=0;miss[i]!='\0';i++)
    {
       string temp=input;
       temp=temp+miss[i];
        sort(temp.begin()+1,temp.end());
        for(auto it=dict.begin();it!=dict.end();it++)
        {   string check=*it;
            sort(check.begin()+1,check.end());
            if(temp==check)
                cout<<"\n"<<*it;
        }
    }

}
void suggestion2(unordered_set<string> dict,string input)
{
    for(int i=1;i<input.length();i++)
    {   string temp="";
        for(int j=0;j<i;j++)
        {
            temp=temp+input[j];
        }
        for(int j=i+1;j<input.length();j++)
        {
            temp=temp+input[j];
        }
        if(dict.find(temp)!=dict.end())
            cout<<"\n"<<temp;
    }
}
void suggestion3(unordered_set<string> dict,string input)
{
     for(int i=1;i<input.length();i++)
    {   string temp="";
        for(int j=0;j<i;j++)
        {
            temp=temp+input[j];
        }
        for(int j=i+1;j<input.length();j++)
        {
            temp=temp+input[j];
        }
        suggestion1(dict,temp);
}

}
void suggestion4(unordered_set<string> dict,string input)
{    char miss[]="abcdefghijklmnopqrstuvwxyz";
    for(int i=0;i<input.length();i++)
    { string temp=input;
      for(int j=0;miss[j]!='\0';j++)
      {
          temp[i]=miss[j];
              if(dict.find(temp)!=dict.end())
                     cout<<"\n"<<temp;
      }

    }
}
void suggestion5(unordered_set<string> dict,string input)
{
    string temp=input;
    sort(temp.begin()+1,temp.end());
```

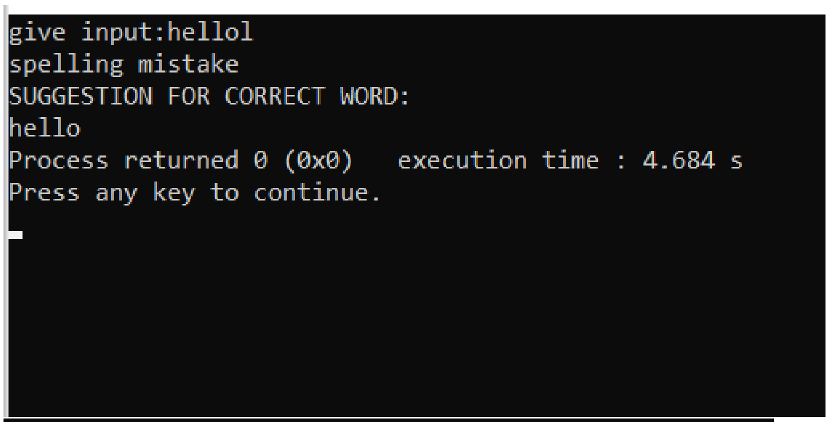


```cpp
    sort(temp.begin()+1,temp.end());
      for(auto it=dict.begin();it!=dict.end();it++)
        {   string check=*it;
            sort(check.begin()+1,check.end());
            if(temp==check)
                cout<<"\n"<<*it;
        }
}

int main()
{     char arr[20];int check=0;
      cout<<"give input:";
      cin.getline(arr,20);
       string input="";
       for(int i=0;arr[i]!='\0';i++)
       {
           input=input+arr[i];
       }
        unordered_set<string> dict;
       fstream file;
      file.open("file.txt",ios::in);
      if (file.is_open()){
       string tp;
       while(getline(file, tp)){
         dict.insert(tp);
       }
       file.close();
       }
       if(dict.find(input)==dict.end())
       {
           cout<<"spelling mistake";check=1;
       }else
       {
           cout<<"correct spelling";
       }
       if(check==1)
       {  cout<<"\nSUGGESTION FOR CORRECT WORD:";

          suggestion1(dict,input);
         suggestion2(dict,input);
         suggestion3(dict,input);
         suggestion4(dict,input);
         suggestion5(dict,input);

      }
}
```

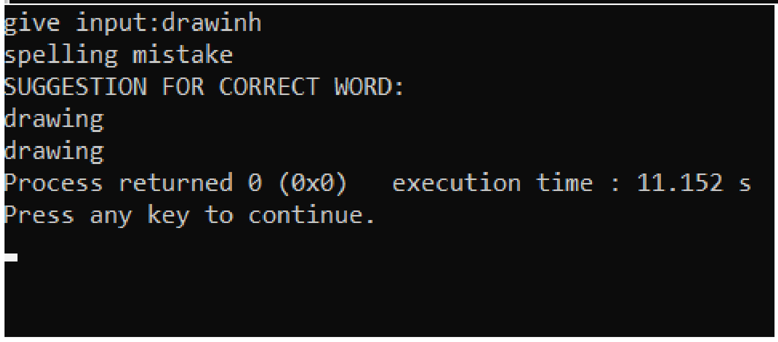
Output ss:



```
give input:the
correct spelling
Process returned 0 (0x0)   execution time : 3.171 s
Press any key to continue.
```



```
give input:hellol
spelling mistake
SUGGESTION FOR CORRECT WORD:
hello
Process returned 0 (0x0)   execution time : 4.684 s
Press any key to continue.
```



```
give input:drawinh
spelling mistake
SUGGESTION FOR CORRECT WORD:
drawing
drawing
Process returned 0 (0x0)   execution time : 11.152 s
Press any key to continue.
```