

**Department of Computer Science**

**Bahria University**

**CSC-221: Data structures & Algorithms**

**Semester 03(Spring 2021)**

**ASSIGNMENT 03**

Marks: 05

# NAME:Sajjad Ali

# CLASS: BSCS-3A

# REG #: 67756

**Marks Obtained:**

**Instructions.**

1. Dead line: Submit your assignment by 05-06-2021
2. Follow same format for assignment submission.
3. Copied/Plagiarized answers will be marked zero.
4. Write code and also attach output.

**Question no.1**

A shrinkable word is a word that can be reduced down to the empty string by deleting one letter at a time such that, at each stage, the remaining string is a word. For example, the word “startling” is shirnkable because of this sequence of words: startling  🡪 starting🡪   staring 🡪  string  🡪 sting 🡪  sing 🡪  sin 🡪  in 🡪  i   🡪 (empty)

Plate🡪late🡪ate🡪at🡪a—(empty)

Write a function that accepts as input a string and a set of all the words in English, then reports whether the input word is shrinkable.

#include<iostream>

#include<cstring>

using *namespace* std;

#define SIZE 100

*int* top=-1;

string Stack[SIZE];

*void* ShrinkedWords(string *word*){

    if(top==SIZE-1)

    cout<<"Stack overflow\n";

    else{

        top++;

        Stack[top]=*word*;

        }

}

string remain(*char* *word*[],*int* *size*){

*int* i=0;

    string shrinked="";

    while(i!=*size*){

        shrinked+=*word*[i];

        i++;

    }

    ShrinkedWords(shrinked);

    return shrinked;

}

*int* Delete(*char* *word*[],*char* *alpha*,*int* *size*,*int* *pos*,*bool* *confirm*){

    if(*confirm*==true){

            if(*pos*<0 || *pos*>*size*){

                cout<<"INVALID POSITION........\n";

            }

            else{

*int* count=0;

                for(*int* i=0;i<*size*;i++){

                    if(*alpha*==*word*[i]){

                        count++;

                    }

                    if(count==*pos*){

*int* deletePosition=i;

                    for(*int* j=deletePosition; j<*size*;j++)

                        {

*word*[j]=*word*[j+1];

                        }

                        break;

                    }

                }

*size*--;

                cout<<"\tRemaining word : "<<remain(*word*,*size*)<<"\n";

            }

    }

    else{

         for(*int* i=0;i<*size*;i++){

                    if(*alpha*==*word*[i]){

*int* deletePosition=i;

                         for(*int* j=deletePosition; j<*size*;j++)

                        {

*word*[j]=*word*[j+1];

                        }

                        break;

                    }

                }

*size*--;

                cout<<"\tRemaining word : "<<remain(*word*,*size*)<<"\n";

    }

    return *size*;

}

*int* check(*char* *word*[],*char* *alpha*,*int* *size*){

*int* count=0;

    for(*int* i=0;i<*size*;i++){

        if(*alpha*==*word*[i]){

            count++;

        }

    }

        if(count>1){

*bool* CorrectPosition=false;

*int* pos;

            while(CorrectPosition!=true)

            {

                cout<<"\tPosition of"<<*alpha*<<"( 1 to "<<count<<") to Delete : "; cin>>pos;

                if(pos>=1&&pos<=count) CorrectPosition=true;

                else CorrectPosition=false;

            }

            return pos;

        }

        else{

            return -1;

        }

}

*void* Shrink(*char* *word*[],*int* *size*){

    if(*size*>1){

        cout<<"Alphabet to Delete : "; *char* alpha; cin>>alpha;

*int* pos=check(*word*,alpha,*size*);

        if(pos==-1){

*size*=Delete(*word*,alpha,*size*,-1,false);

            Shrink(*word*,*size*);

        }

        else{

*size*=Delete(*word*,alpha,*size*,pos,true);

            Shrink(*word*,*size*);

        }

    }

}

*int* main(){

    string word;

    cout<<"Enter Word to Shrink : "; cin>>word;

*int* size=word.length();

*char* wordChar[size+1];

    strcpy(wordChar,word.c\_str());

    Shrink(wordChar,size+1);

    cout<<"SHRINKED WORDS : ";

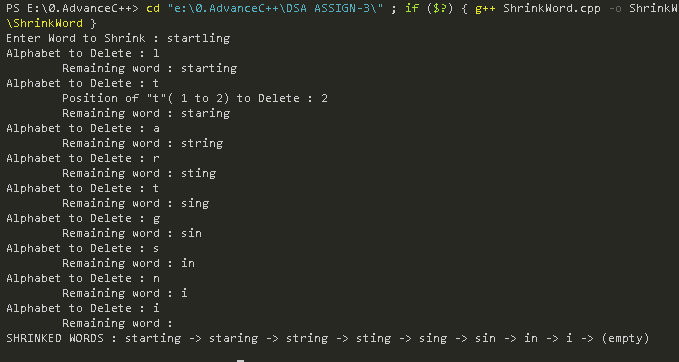
    for(*int* i=0;i<top;i++){

        cout<<Stack[i]<<" -> ";

    }

    cout<<"(empty)\n\n";

}



**Question no.2**

Suppose there are 5 friends ***A, B,C,D,E*** are playing a game. First time A will be the leader and give a word to all friends to guess the meaning of the word. Those who guess the correct answer will get 5 points. Next time B will be the leader and give one word to all friends for guessing the meaning of the word. Each will get 5 points for correct answer and same goes for all friends. Friend who get the more points will be the winner.

Design the solution for above problem with appropriate data structure/techniques. Write the code of your solution

#include<iostream>

using *namespace* std;

*struct* node{

    string name;

*int* points=0;

    node\* next;

};

node\* head=NULL;

string Questions[5]={"flat","rank","lock","yard","firm"};

string Answers[5]={"level","position","bolt","twig","secure"};

*void* Create(*int* *num*){

    node\* temp,\*ptr;

    for(*int* i=1;i<=*num*;i++){

         ptr=head;

         cout<<i<<") Friend Name  : "; string data; cin>>data;

        if(ptr==NULL){

                temp=new node();

                head=ptr=temp;

                temp->name=data;

                temp->next=NULL;

        }

        else{

                while(ptr->next!=NULL){

                    ptr=ptr->next;

                }

                temp=new node();

                temp->name=data;

                temp->next=NULL;

                ptr->next=temp;

        }

    }

}

*void* MakeLeader(*int* *LeaderNo*,*int* *totalFriends*){

   if(*LeaderNo*<=*totalFriends*){

            node\*ptr=head;

            node\*member=head;

            for(*int* i=1;i<*LeaderNo*;i++){

                ptr=ptr->next;

            }

            cout<<"\t------------\n";

            cout<<"\tLeader : "<<ptr->name<<endl;

            cout<<"\t------------\n";

            cout<<"Q: What is the the meaning of \""<<Questions[*LeaderNo*-1]<<"\"?"<<endl;

            while(member!=NULL){

                if(member!=ptr){

                    cout<<member->name<<" -> Answer : "; string ans; cin>>ans;

                    if(ans==Answers[*LeaderNo*-1]){

                        member->points +=5;

                    }

                }

                member=member->next;

            }

            MakeLeader(*LeaderNo*+1,*totalFriends*);

   }

}

*void* Display(){

       node \*ptr = head;

*int* winnerPoints=0;

       string winnerName="";

    cout<<"OVERAll Result : [ ";

            while(ptr!=NULL)

            {

                cout<<ptr->name<<" : ";

                cout<<ptr->points<<" , ";

                if(ptr->points>winnerPoints){

                    winnerPoints=ptr->points;

                    winnerName=ptr->name;

                }

                ptr=ptr->next;

            }

            cout<<" ]\n";

    cout<<"Result Announcment : \n";

    cout<<"Winner : "<<winnerName<<endl;

    cout<<"Points : "<<winnerPoints<<endl;

}

*int* main(){

cout<<"Enter Numbers of Friends(maximum 5) to play : ";*int* totalFriends; cin>>totalFriends;

Create(totalFriends);

MakeLeader(1,totalFriends);

Display();

}

