Data Structures C, D
FAST-NU, Lahore, Fall 2016

Homework 2

Strings as Linked Structures

Due on Thursday September 8 before class Marked out of 80 points.

It is sometimes useful to implement character strings as linked lists for applications requiring frequent update operations such as cut and paste, like a text editor. In that case an array would require too many shift and copy operations whereas a linked list, owing to its flexibility, would only require disconnections of old connections and formation of new ones. The list of characters in this program is Singly Linked.

Implement a LinkedString class, which is basically a linked list of characters and supports a few basic operations described below.

In order to facilitate the evaluation process, you will also provide a menu for this program on the console to test these operations.

The class LinkedString

Following is skeleton code of class definition for LinkedString followed by a sample usage of its functions to explain their working.

```
class LinkedString{
    struct node{
        char ch;
        node * next;
    };
    node * strhead, * strtail;
    int length;
public:
    LinkedString();
    LinkedString(const string&);
    LinkedString(const LinkedString&);
    const LinkedString& operator = (const LinkedString&);
    friend ostream & operator <<(ostream &,const LinkedString&);
    friend istream & operator >>(istream &,const LinkedString&);
    ~LinkedString();
    int getLength();
    LinkedString operator + (const LinkedString &);
    bool find(const LinkedString&);
    bool find(const string&);
    LinkedString findAndCopy(const string&);
    LinkedString findAndCut(const string&);
    void findAndReplace(const string&,const string&);
    void findAndInsertAfter(const string&,const string&);
    void findAndInvert(const string&);
    void reverse();
};
```

Notice in this code there are two types of “strings”. One is the character array based string from C++ STL, and the other is LinkedString, the class you’re making.

Following main program shows the usage. However, your program should provide a menu on the console with an option of each of the supported operations of the class.

```
void main(){
    string temp1="hello", temp2="world";
    LinkedString str1(temp1),str2(temp2), str3(" my ");
    //concatenation
    str3=str1+str3+str2;
    str3.findAndReplace("my", "bad");
    //"hello my world" becomes "hello bad world"
    str1=str3.findAndCut("bad ");
    //str1 contains "bad ", str3 becomes "hello world"
    str2=str1.findAndCopy("ba");
    //str2 contains "ba", str1 remains unchanged
    str3.findAndInvert("orl");
    //"hello world" becomes "hello wlrod"
    str3.findAndInsertAfter("hello ", "mad ");
    //str3 becomes "hello mad wlrod"
    str3.reverse();//whole string inverted
    cout<<str3<<endl;//string displayed

}
```

***

THE END