ENSF 337 Fall 2018: Some Solutions for Tutorial 8

M. Moussavi

Problem I: Consider the following definition and implementation of class MyString, and the given main function, then answer the following questions:

```
class MyString {
public:
    MyString();
    MyString(const char *s);
    ~MyString();
    MyString& MyString(const MyString& src);
    MyString& operator=(const MyString& rhs);
    const char* c_str()const();
    void append(const MyString& src);
    // PROMISES: to copy src.storageM to the end of this-> storage.
private:
    int lengthM;
    char *storageM;
};

MyString::MyString() : lengthM(0), storageM(new char[1]){
    storageM[0] = '\0';
}
```

```
MyString::MyString(const char *s):
lengthM(strlen(s)){
    storageM = new char[lengthM + 1];
    strcpy(storageM, s);
}

MyString::~MyString(){
    delete [] storageM;
}

int main(void){
   MyString* s2 = new MyString("Red");
   MyString s3 ("AB");
   s2.append(s3);
   cout << s2.c_str();
   s2 = s3;
   return 0;
}
```

Questions:

1. Write the definition of the member function append.
2. Write the definition of an assignment operator for class MyString

Problem II:

Consider the definitions of `struct` type called `Node` and a class type called `List`**:**

```
struct Node {
    int item;
    Node *next;
};

class List {
public:
    List():headM(0){//Point 1}
    void insert(int the_item);
    void display()
 private:
    Node *headM;
};
```

```
void List::insert(int the_item){
    Node *new_node = new Node;
    new_node->item = the_item;
    if(headM == NULL) {
        headM = new_node;
        new_node -> next = NULL;
    }
    else{
        new_node -> next = headM;
        headM = new_node;
    }
    // Point two
}
```

```
int main(){
  List n;
  n.insert(123);
  n.insert(189);
  n.insert(145);
  n.insert(167);
  // Point three
  n.display();
  return 0;
}
```

Questions:

- Draw an AR diagram for **point 1**
- Draw an AR diagram when the program reaches **point two** for the first time.
- Draw an AR diagram when program reaches **point three**.
- Write the definition of the member function display, that displays all values in a linked list, one value per line.

```
void append(const MyString& src)
    // PROMISES: to copy src.storageM to the end of this-> storage.
{
        if(src.lengthM == 0) return;
        new_length = lengthM + src.lengthM;
        char* new_storage = new char[ new_length + 1];
        assert(new_storage != NULL);
        int i;

        for(i =0; i < lengthM; i++)
                new_storage[i] = storageM[i];
        int j, k
        for(j = i, k=0; j < new_length; j++, k++)
                new_storage[j] = src.storageM[k];

        new_storage[j] = '\0';
        delete[]storageM;
        storageM = new_stroage;
}

MyString& MyString::operator= (const MyString& rhs)
{
        if(&rhs != this) {
                delete [] storageM;
                storageM = new char[rhs.lengthM +1];
                assert(storageM != NULL);
                strcpy(storageM, rhs.storageM);
        }
        return *this;
}
```