```
///Marc Pfeiffer
///Assignment 3 - Widgets Inventory
#include <iostream>
#include <fstream>

using namespace std;

ifstream dataIn("infile.txt");
ofstream dataOut("outfile.txt");


struct Node{

    int amount;
    double price;
    Node* next;

};

class Shipment{

public:

    friend ifstream & operator>> (ifstream & dataIn , Shipment& order);
    char getType(){return type;}
    int getAmount(){return amount;}
    double getPrice() {return price;}

private:

    char type;
    int amount;
    double price;

};

ifstream& operator >>(ifstream& dataIn, Shipment& order){

    dataIn>>order.type;

    if(order.type == 'R'){

        dataIn>>order.amount>>order.price;

    }
    else if(order.type == 'S'){

        dataIn>>order.amount;

    }
    else{

        dataIn>>order.price;

    }
    return dataIn;

}

bool isEmpty( Node* head);
void insertFirstNode(Node* &head, Node* &last, Shipment &order);
void insertNode(Node* &head, Node* &last, Shipment& obj);
void removeNode(Node *&head, Node* &last, Shipment &obj);
void printList(Node *current);

bool isEmpty(Node* head){
```

```
67
68      if(head == NULL){
69          return true;
70      }
71      else {
72          return false;
73      }
74
75
76  }
77  void insertFirstNode(Node* &head, Node* &last, Shipment &obj){
78
79      Node *temp = new Node;
80      temp->amount = obj.getAmount();
81      temp->price = obj.getPrice();
82      temp->next = NULL;
83      head = temp;
84      last = temp;
85  }
86  void insertNode(Node* &head, Node* &last, Shipment& obj){
87
88      if( isEmpty(head) ){
89
90          insertFirstNode(head, last , obj);
91      }
92      else{
93
94          Node *temp = new Node;
95          temp->amount = obj.getAmount();
96          temp->price = obj.getPrice();
97          temp->next = NULL;
98          last->next = temp;
99          last = temp;
100
101     }
102 }
103 void removeNode(Node* &head, Node* &last, Shipment& obj){
104
105     if(isEmpty(head)){
106
107         dataOut<<"The List is Empty\n";
108     }
109     else if(head == last){
110
111         delete head;
112         head = NULL;
113         last = NULL;
114     }
115     else {
116
117         Node *temp = head;
118         head = head->next;
119         delete temp;
120     }
121 }
122 void printList(Node* current){
123
124     if(isEmpty(current)){
125
126         dataOut<<"The List is empty\n";
127     }
128     else{
129
130         dataOut<<"CURRENT STOCK\n\n";
131         while(current != NULL){
132
```

```
133             dataOut<<current->amount<<" at "<<current->price<<endl;
134             current = current->next;
135         }
136     }
137 
138 }
139 
140 int main(){
141 
142     Shipment order;
143     Node *List = NULL, *Rear=NULL;
144 
145     dataOut.setf(ios::fixed ,ios::floatfield);
146     dataOut.precision(2);
147 
148     int amountNeeded = 0, promotionVal= 0;
149     double totalPrice, promotion=1.00;
150 
151     while(!dataIn.eof()){
152 
153         dataIn>>order;
154 
155         if(order.getType()=='R'){
156             dataOut<<"Order of "<<order.getAmount()<<" Widgets at $"<<order.getPrice
()<<" per piece has been received\n\n\n\n";
157             insertNode(List, Rear, order);
158 
159         }
160         else if(order.getType() == 'S'){
161 
162             amountNeeded = order.getAmount();
163             if(isEmpty(List)){
164                 dataOut<<"\nRemainder of "<<amountNeeded<<" Widgets, not
available\n\n";
165             }
166             else{
167 
168                 totalPrice = 0.00;
169 
170                 if(order.getAmount() > List->amount){
171 
172 
173                     dataOut<<amountNeeded<<" widgets ordered\n";
174                     dataOut<<List->amount<<" at "<<(List->price*1.30)<<" each\t
Sales: $"<<(List->price*1.30)*List->amount<<endl;
175                     totalPrice += ((List->price*1.30)*List->amount);
176                     amountNeeded -= List->amount;
177 
178                     while(amountNeeded != 0){
179 
180                         removeNode(List, Rear , order);
181 
182                         if(!isEmpty(List)){
183 
184                             if(amountNeeded > List->amount){
185 
186                                 dataOut<<List->amount<<" at "<<(List->price*1.30)<<"
each\t Sales: $"<<(List->price*1.30)*List->amount<<endl;
187                                 totalPrice += ((List->price*1.30)*List->amount);
188                                 amountNeeded -= List->amount;
189                             }
190                             else{
191 
192                                 List->amount -= amountNeeded;
193                                 dataOut<<amountNeeded<<" at "<<List->price*1.30<<"
each\t Sales: $"<<(List->price*1.30)*amountNeeded<<endl;
```

```
194                                 totalPrice += ((List->price*1.30)*amountNeeded);
195                                 amountNeeded = 0;
196 
197                             }
198                         }
199                         else{
200                             dataOut<<"\nRemainder of "<<amountNeeded<<" Widgets, not
available\n\n";
201                             amountNeeded = 0;
202                         }
203                     }
204                     if(promotionVal != 0){
205                         dataOut<<"\t\t   Total Sales: $"<<totalPrice*promotion<<endl
<<endl<<endl;
206                         promotionVal --;
207                     }
208                     else{
209                         dataOut<<"\t\t   Total Sales: $"<<totalPrice<<endl<<endl<<
endl;
210                     }
211                 }
212 
213 
214             else if(order.getAmount() == List->amount){
215 
216                 dataOut<<amountNeeded<<" widgets sold\n";
217                 dataOut<<List->amount<<" at "<<(List->price*1.30)<<" each\t Sales:
$"<<(List->price*1.30)*List->amount<<endl;
218                 totalPrice += ((List->price*1.30)*List->amount);
219                 removeNode(List, Rear, order);
220                 if(promotionVal != 0){
221                         dataOut<<"\t\t   Total Sales: $"<<totalPrice*promotion<<endl
<<endl<<endl;
222                         promotionVal --;
223                     }
224                     else{
225                         dataOut<<"\t\t   Total Sales: $"<<totalPrice<<endl<<endl<<
endl;
226                     }
227                 }
228 
229 
230             else{
231                 dataOut<<amountNeeded<<" widgets sold\n";
232                 List->amount -= amountNeeded;
233                 dataOut<<amountNeeded<<" at "<<List->price*1.30<<" each\t Sales: $"
<<(List->price*1.30)*amountNeeded<<endl;
234                 totalPrice += ((List->price*1.30)*amountNeeded);
235                 if(promotionVal != 0){
236                         dataOut<<"\t\t   Total Sales: $"<<totalPrice*promotion<<endl
<<endl<<endl;
237                         promotionVal --;
238                     }
239                     else{
240                         dataOut<<"\t\t   Total Sales: $"<<totalPrice<<endl<<endl<<
endl;
241                     }
242                 }
243             }
244 
245 
246         }
247         else{
248 
249             promotion = order.getPrice();
250             promotionVal = 2;
```

```
251             dataOut<<"Promotional discount of "<<promotion*100<<"% has been applied
for the next 2 orders out\n\n";
252
253         }
254     }
255
256
257  printList(List);
258
259     return 0;
260  }
261
```