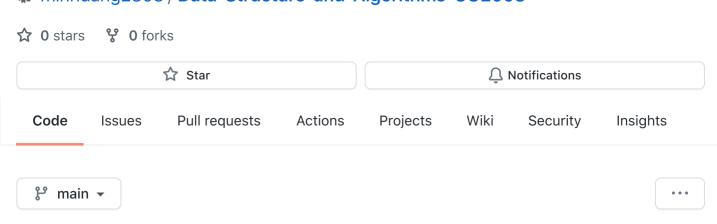
minhdang2803 / Data-Structure-and-Algorithms-CO2003-



Data-Structure-and-Algorithms-CO2003- / dsastudents / dsacpp / include / sorting / SLinkedListSE.h

```
minhdang2803 DSA 203 Asm3

At 1 contributor
```

```
142 lines (133 sloc) 3.75 KB
  1
      /*
  2
       * To change this license header, choose License Headers in Project Properties.
       * To change this template file, choose Tools | Templates
       * and open the template in the editor.
  4
  5
       */
  6
  7
      /*
  8
       * File:
                 SLinkedListSE.h
  9
       * Author: LTSACH
 10
 11
       * Created on 31 August 2020, 21:00
 12
 13
 14
      #ifndef SLINKEDLISTSE H
 15
      #define SLINKEDLISTSE H
 16
 17
      #include "list/SLinkedList.h"
 18
      #include "sorting/ISort.h"
 19
 20
      template <class T>
 21
      class SLinkedListSE : public SLinkedList<T>
 22
      {
 23
      public:
 24
          SLinkedListSE(
 25
              void (*removeData)(SLinkedList<T> *) = 0,
              bool (*itemEQ)(T &, T &) = 0) : SLinkedList<T>(removeData, itemEQ){
 26
 27
```

```
28
29
         void sort(int (*comparator)(T &, T &) = 0)
30
31
             if (this->count > 0)
             {
33
                  typename SLinkedList<T>::Node *first = this->head->next; //first user's dat
                                                                              //to tail => to 0
34
                  this->tail->next->next = 0;
35
                  mergeSort(first, comparator);
36
37
                  //find the last user's data
38
                  typename SLinkedList<T>::Node *last = first;
                  while (last->next != 0)
                      last = last->next:
40
41
                  //attach to head&tail of the list
42
                  last->next = this->tail;
43
44
                  this->tail->next = last;
45
                  this->head->next = first;
             }
46
47
         };
48
49
     protected:
         static int compare(T &lhs, T &rhs, int (*comparator)(T &, T &) = 0)
50
51
         {
             if (comparator != 0)
53
                  return comparator(lhs, rhs);
54
             else
55
             {
56
                  if (lhs < rhs)</pre>
57
                      return -1;
                  else if (lhs > rhs)
59
                      return +1;
60
                  else
61
                      return 0;
             }
62
63
         void mergeSort(typename SLinkedList<T>::Node *&head, int (*comparator)(T &, T &) =
64
65
             if ((head != 0) && (head->next != 0))
66
67
68
                  typename SLinkedList<T>::Node *second;
69
                  devide(head, second);
70
                  mergeSort(head, comparator);
71
                  mergeSort(second, comparator);
72
                  merge(head, second, comparator);
             }
73
74
         };
75
         void devide(typename SLinkedList<T>::Node *&first, typename SLinkedList<T>::Node *&
76
77
             //YOUR CODE HERE
             typename SLinkedList<T>::Node *slow, *fast;
79
             if (first == NULL)
```

```
80
                   return;
              }
 82
              else
 84
               {
 85
                   slow = first;
                   fast = slow->next;
                   while (fast != NULL)
 87
                   {
89
                       fast = fast->next;
 90
                       if(fast!=NULL)
                       {
91
 92
                           slow = slow->next;
 93
                           fast = fast->next;
                       }
                   }
              }
97
              second = slow->next;
               slow->next = NULL;
99
          }
          void merge(typename SLinkedList<T>::Node *&first, typename SLinkedList<T>::Node *&s
100
101
               if(first == NULL || second == NULL)
102
103
                   return;
104
              typename SLinkedList<T>::Node *temp = NULL;
105
               if(compare(first->data,second->data,comparator) == -1)
               {
107
                   temp = first;
108
                   first = first->next;
109
              }
110
              else
               {
111
112
                   temp = second;
                   second = second->next;
113
              }
114
              typename SLinkedList<T>::Node *ptr = temp;
115
              while(first!=NULL && second!=NULL)
116
               {
117
                   if(compare(first->data, second->data, comparator) == -1)
118
119
120
                       ptr->next = first;
121
                       first = first->next;
122
                   }
123
                   else
124
                   {
125
                       ptr->next = second;
126
                       second = second->next;
127
128
                   ptr = ptr->next;
              }
129
130
              if(first == NULL && second!=NULL)
131
               {
```

```
7/5/2021
```

```
132
                  ptr->next = second;
133
              }
              if(second == NULL && first!=NULL)
134
135
136
                  ptr->next = first;
137
              }
138
              first = temp;
          }
139
140
      };
141
142
      #endif /* SLINKEDLISTSE_H */
```