POS1 – Übungsbeispiel SortedDoubleLinkedList

Schuljahr: 2023/24

Erstellen Sie ein cpp-Programm, welches eine Klasse für eine aufsteigend sortierte, doppelte verkettete Liste bestehend aus ganzen Zahlen implementiert!

Folgende Funktionalitäten bzw. Merkmale sollen vorhanden sein:

a) Einfügen eines Elements in die Liste
   Es soll sowohl das iterative als auch das rekursive Einfügen in die Liste zur Verfügung stehen (beide Varianten sowohl beginnend beim Anfang als auch beginnend am Ende)!
b) Entfernen eines beliebigen Elements aus der Liste
   Es soll sowohl das iterative als auch das rekursive Löschen aus der Liste zur Verfügung stehen (beide Varianten sowohl beginnend beim Anfang als auch beginnend am Ende)!
c) iterative und rekursive Ausgabe sämtlicher Elemente der Liste sowohl von links nach rechts als auch umgekehrt.
d) iterative und rekursive Ermittlung der Summe aller in der Liste gespeicherten Zahlen

Die Lösung der Aufgabenstellung gliedert sich in folgende fünf Dateien:

- `node.h`
  - Definition der Klasse `node`
- `node.cpp`
  - Implementierung der Methoden der Klasse `node`
- `sorteddll.h`
  - Definition der Klasse `sdll`
- `sorteddll.cpp`
  - Implementierung der Methoden der Klasse `sdll`
- `main.cpp`
  - Implementierung der `main()`-Funktion

Die Klasse `sdll` soll folgende öffentlichen Member besitzen:

a) `sdll()`
b) `~sdll()`
c) `void add_iterative_begin(int value)`
d) `void add_iterative_end(int value)`
e) `void add_recursive_begin(int value)`
f) `void add_recursive_end(int value)`
g) `void remove_iterative_begin(int value)`
h) `void remove iterative_end(int value)`
i) `void remove recursive_begin(int value)`
j) `void remove recursive_end(int value)`
k) `void print_iterative_begin()`
l) `void print_iterative_end()`
m) `void print_recursive_begin()`
n) `void print_iterative_end()`
o) `void print_reverse_recursive()`
p) `int sum_iterative()`
q) `int sum_recursive()`

POS1 – Übungsbeispiel SortedDoubleLinkedList **Schuljahr:** 2023/24

Hinweise:

- Halten Sie sich an die üblichen Programmierrichtlinien (Kommentare, Header, Einrückungen, Datenkapselung…)!
- Überlegen Sie, welche Member (`public/private`) die Klasse `node` sinnvollerweise besitzen sollte! Beachten Sie, dass in der Klasse `sdll` sowohl der Beginn als auch das Ende der Liste als private Objektvariable (`head` und `tail`) gespeichert werden muss!
- Erstellen Sie zum Testen Ihrer Lösung die Datei `main.cpp` in welcher die `main`()-Funktion enthalten ist! Die main()-Funktion soll sämtliche erstellten Methoden verwenden!
- Beachten Sie, dass sämtlicher mit `new` angeforderter Speicher vor Beendigung des Programms wieder freigegeben werden muss!
- Erstellen Sie ein `makefile` zur automatisierten Erstellung des Projekts. Halten Sie sich bez. Tags und Abhängigkeiten an die Vorgaben aus dem Unterricht (Stack mit Klassen und dynamischer Liste)