DAP2 Praktikum für ETIT und IKT, SoSe 2018, Kurzaufgabe K5

Fällig am 18.06. um 18:00

Die Regeln für das Programmieren gelten erbarmungslos. `assert` können Sie weglassen.

Kurzaufgabe K5 (1 Punkt): Traversieren von Binärbäumen

Ein Suchbaum ist ein binärer Baum mit der Eigenschaft, dass für jeden Knoten v der Wert des linken Kindes von v kleiner und der Wert des rechten Kindes von v größer als der Wert von v ist.

Implementieren Sie den Suchbaum indem sie ihn in das Programmgerüst **`Baum.cpp`** einbauen. In dem Klassentemplate

```
template<class T> class Baum
```

müssen mindestens der Konstruktor

```
Baum()
```

sowie die Methoden zum Einfügen neuer Elemente

```
void Insert(T v)
```

und die drei Methoden zum Traversieren des Baumes vorhanden sein:

```
void InOrder()
void PreOrder()
void PostOrder()
```

Beim Traversieren geben diese drei Methoden (ausnahmsweise) die Elemente entsprechend der gewählten Reihenfolge direkt aus, machen aber sonst nichts. Andere Methoden oder Klassen, die für eine korrekte und/oder Funktionierende Implementierung notwendig sind, sollten Sie selbst hinzufügen. Die Nutzung von STL-Containern oder anderen Bibliotheken ist nicht erlaubt.

Ein korrekt implementiertes Programm sollte folgende Ausgabe liefern:

```
Baum 5 4 6 3 1 2 0 7 9 8 10
  InOrder: 0 1 2 3 4 5 6 7 8 9 10
 PreOrder: 5 4 3 1 0 2 6 7 9 8 10
PostOrder: 0 2 1 3 4 8 10 9 7 6 5
```

Kurzaufgabe K5 (1 Punkt): Rucksackproblem

Vervollständigen sie den Quelltext Backpack.cpp um den in der Vorlesung vorgestellten auf der dynamischen Programmierung aufbauenden Algorithmus zum Rucksackproblem. Das Programm beinhaltet bereits einen Greedy-Algorithmus. Ihr verbesserter Algorithmus soll an die im Quelltext gekennzeichnete Stelle eingebaut werden. Außerhalb dieses Bereiches ist keine Änderung nötig und auch nicht erlaubt.

Beachten Sie, dass Ihr Algorithmus nicht nur den Wert und das Gewicht des Rucksackes berechnen muss, sondern auch eine Liste der eingepackten Waren.

Aufruf mit Greedy Algorithmus	Algorithmus mit dynamischer Programmierung

`Backpack 16 -o -i  6 9 3 2 6 7`
`Initial items:`
 	
`Value / Weight`
`6 9`
`3 2`
`6 7`
	`Backpack 16 -o -i -b  6 9 3 2 6 7`
	`Initial items:`
	
	`Value / Weight`
	`6 9`
	`3 2`
	`6 7`

Bagpack filled with: Value / Weight 3 2 6 7 Backpack in total: Value / Weight 9 9	Bagpack filled with: Value / Weight 6 7 6 9 Backpack in total: Value / Weight 12 16