Abstrakte Datentypen

14. Aufgabe *10 Punkte*

Der binäre Baum soll modifiziert werden. Bisher hatten die Knoten des Baumes als Elemente einen Integer-Wert, jetzt sollen die Elemente einen abstrakten Datentyp erhalten. Dazu werden weitere Funktionen benötigt:

- **Vergleichsfunktion** für je zwei Knotenwerte:
  int (*compare)(const void *key1, const void *key2)
  Zurückgeliefert wird im Gleichheitsfall *(key1 = key2) 0*, ist *key1 > key2* wird ein positiver Wert zurückgeliefert und ansonsten ein negativer Wert.

- **Löschfunktion,** die zur Speicherfreigabe für dynamisch angelegte Elemente zum Einsatz kommt:
  void(*destroy)(void *data)
  Besitzt der Knoten beispielsweise Daten *data*, für welche dynamisch Speicherplatz mit *malloc* angelegt wurde, so sollte *destroy* auf die Funktion *free* zeigen.

- **Ausgabefunktion** zur Ausgabe der Knotenwerte:
  void (*print)(FILE * filepointer, const void * key)
  Zur Ausgabe der Knotenwerte müssen die Funktionen aus *traverse.h* angepasst werden. Beispielsweise:
  *int preorder(const BiTreeNode *node, void (*print)(const void * key));*

Ändern bzw. erweitern Sie die bisherigen Schnittstellen und den Programmcode entsprechend. Schreiben Sie auch ein Programm mit welchem Sie den neuen Binären Baum testen können (analog zu Bitree-ex-1.c). Die Wahl der Knotenwerte in Ihrem Testprogramm bleibt Ihnen freigestellt.

```
typedef struct BiTreeNode_ {
    void                *data;
    struct BiTreeNode_  *left;
    struct BiTreeNode_  *right;
} BiTreeNode;

typedef struct BiTree_ {
    int                 size;
    int                 (*compare)(const void *key1, const void *key2);
    void                (*destroy)(void *data);
    BiTreeNode          *root;
} BiTree;
```

Alle Mitarbeiter der Sai wünschen Ihnen
ein frohes Weihnachtsfest und ein erfolgreiches Jahr 2000.