```c
#include <limits.h>
#include <stdlib.h>
#include <stdio.h>
#include "mapa.h"
#include "arvore.h"



struct smapa {
  int chave;
  int conteudo;
  short int bf;
  struct smapa* esq;
  struct smapa* dir;
};


static Mapa* cria_no(int c, int novodado);
static Mapa* insere2(Mapa *m, int chave, int novodado, int *cresceu);
static Mapa* corrige_dir(Mapa *m);
static Mapa* corrige_esq(Mapa *m);
static Mapa* rotaciona_dir(Mapa *m);
static Mapa* rotaciona_esq(Mapa *m);


static Mapa* cria_no(int c, int novodado) {
  Mapa *m = (Mapa*)malloc(sizeof(Mapa));
  if (m != NULL) {
    m->esq = m->dir = NULL;
    m->chave = c;
    m->conteudo = novodado;
    m->bf = 0;
  }
  return m;
}


Mapa* cria() {
  return NULL;
}


Mapa* insere (Mapa* m, int chave, int novodado) {
  int cresceu = 0;
  return insere2(m, chave, novodado, &cresceu);
}


static Mapa* insere2(Mapa *m, int chave, int novodado, int *cresceu) {
  if (m == NULL) {
    m = cria_no(chave, novodado);
    *cresceu = 1;
  }
  else if (chave < m->chave) {
   m->esq = insere2(m->esq, chave, novodado, cresceu);
    if(*cresceu) {
      switch (m->bf) {
      case 1:
        m->bf = 0;
        *cresceu = 0;
        break;
      case 0:
        m->bf = -1;
        break;
      case -1:
        m = corrige_esq(m);
        *cresceu = 0;
        break;
      }

  }
    }
  else if (chave > m->chave) {
    m->dir = insere2(m->dir, chave, novodado, cresceu);
    if(*cresceu) {
      switch (m->bf) {
      case 1:
        m = corrige_dir(m);
        *cresceu = 0;
        break;
      case 0:
```

```
85          m->bf = 1;
86          break;
87        case -1:
88          m->bf = 0;
89          *cresceu = 0;
90          break;
91        }
92      }
93    }
94    return m;
95  }
96
97
98  static Mapa* corrige_esq(Mapa *m) {
99    if (m->esq->bf == -1) {
100     m->bf = m->esq->bf = 0;
101     return rotaciona_dir(m);
102   }
103   else if (m->esq->bf == 1) {
104     Mapa *n = m->esq->dir;
105
106     switch (n->bf) {
107     case -1:
108       m->bf = 1;
109       m->esq->bf = n->bf = 0;
110       break;
111     case 0:
112       m->bf = m->esq->bf = n->bf = 0;
113       break;
114     case 1:
115       m->bf = n->bf = 0;
116       m->esq->bf = -1;
117       break;
118     }
119     m->esq = rotaciona_esq(m->esq);
120     return rotaciona_dir(m);
121   }
122
123 }
124
125
126
127 static Mapa* corrige_dir(Mapa *m) {
128   if (m->dir->bf == 1) {
129     m->bf = m->dir->bf = 0;
130     return rotaciona_esq(m);
131   }
132   else if (m->dir->bf == -1) {
133     Mapa *n = m->dir->esq;
134
135     switch (n->bf) {
136     case -1:
137       m->bf = n-> bf= 0;
138       m->dir->bf = 1;
139       break;
140     case 0:
141       m->bf = m->dir->bf = n->bf = 0;
142       break;
143     case 1:
144       m->bf = -1;
145       m->dir->bf = n->bf = 0;
146       break;
147     }
148     m->dir = rotaciona_dir(m->dir);
149     return rotaciona_esq(m);
150   }
151 }
152
153
154
155 static Mapa* rotaciona_dir(Mapa *m) {
156     Mapa* antesq = m->esq;
157     m->esq = antesq->dir;
158     antesq->dir = m;
159   return antesq;
160 }
161
162
163 static Mapa* rotaciona_esq(Mapa *m) {
164   Mapa* dir = m->dir;
165   m->dir = dir->esq;
166   dir->esq = m;
167   return dir;
168 }
169
```

```c

int busca (Mapa *m, int chave) {
  if (m == NULL) {
    return -1;
  }
  else if (m->chave == chave) {
    return m->conteudo;
  }
  else if (chave < m->chave) {
  return busca(m->esq, chave);
  }
  else if (chave > m->chave) {
    return busca(m->dir, chave);
  }
  else {
    return -1;
  }
}



void destroi (Mapa *m) {
  if (m != NULL) {
    destroi(m->esq);
    destroi(m->dir);
    free(m);
  }
}



int iguais (Mapa* m1, Mapa* m2) {

  if (m1==NULL) return (m2==NULL);
  if (m2==NULL) return 0;

  return (m1->chave == m2->chave) &&
          iguais (m1->esq, m2->esq) &&
          iguais (m1->dir, m2->dir);
}


int altura (Mapa* m) {
  if (m==NULL) return 0;
  printf ("chave %d\n", m->chave);
  return ((m->bf==-1)?(altura(m->esq)):altura(m->dir))+1;
}


static void mostra_mapa_int (Mapa* m) {

  printf("[");
  if (m != NULL) {
    printf("%d:%d", m->chave,m->bf);
    mostra_mapa_int(m->esq);
    mostra_mapa_int(m->dir);
  }
  printf("]");
}


void mostra (Mapa* m) {
  mostra_mapa_int (m); printf ("\n");
}

```