acm.mipt.ru

олимпиады по программированию на Физтехе

Раздел «Алгоритмы» . DecartTreesCPP :

Декартово дерево на C

- теория

```
#include <stdio.h>
#include <malloc.h>
#include "dt.h"

#define UNIQ_KEYS 1

typedef int mvalue_t;
typedef int mkey_t;

typedef enum {
   BT_ADDED,
   BT_UPDATED,
   BT_DELETED,
   BT_FOUND,
   BT_NOTFOUND,
   BT_ERROR,
} dt_result_t;


typedef struct dt {
   mvalue_t v;
      mkey_t x;
   int  y;
      struct dt  *l, *r;
} dt_t;

#include <stdio.h>
#include <malloc.h>
#include "dt.h"

dt_t*
dt_search(dt_t *t, mkey_t x) {
    if ( t == NULL ) return NULL;
    if ( x < t->x  ) return dt_search(t->l, x);
    if ( x > t->x  ) return dt_search(t->r, x);
    return t;
}


void
dt_split(dt_t *t, mkey_t x, dt_t **l, dt_t **r) {
    if(t == NULL) {
        *r = *l = NULL;
    } else {
        if(x < t->x) {
            dt_split(t->l, x, l, r);
            t->l = *r; *r = t;
        } else {
            dt_split(t->r, x, l, r);
            t->r = *l; *l = t;
        }
    }
}

dt_t*
dt_merge(dt_t *l, dt_t *r) {
```

Поиск
Поиск
Раздел «Алгоритмы»
Главная
Форум
Ссылки
El Judge
Инструменты:
Поиск
Изменения
Index
Статистика
Разделы
Информация
Алгоритмы
Язык Си
Язык Ruby
Язык Ассемблера
El Judge
Парадигмы
Образование
Сети
Objective C
Logon>>

```
    if (l == NULL) return r;
    if (r == NULL) return l;
    if (l->y > r->y) {
        l->r = dt_merge(l->r,r);
        return l;
    } else {
        r->l = dt_merge(l,r->l);
        return r;
    }
}


inline dt_t* dt_newnode(mkey_t x, int y, mvalue_t v) {
    dt_t *t = (dt_t*) malloc(sizeof(dt_t));
    t->x = x; t->y = y, t->v = v;
    return t;
}

int
dt_insert(dt_t **t, mkey_t x, int y, mvalue_t v) {
    dt_t *tmp;
#ifdef UNIQ_KEYS
    tmp = dt_search(*t, x);
    if(tmp) {
        //  printf("Replaced key %d  value %d changed to %d\n", x, (*t)->v, v);
        tmp->v = v;
        return BT_UPDATED;
    }
#endif
    if (*t == NULL ||  y > (*t)->y) {
        tmp = dt_newnode(x, y, v);
        dt_split(*t, x, &(tmp->l), &(tmp->r));
        *t    = tmp;
        return BT_ADDED;
    } else if ( x < (*t)->x) {
        return dt_insert(&((*t)->l), x, y, v);
    } else {
        return dt_insert(&((*t)->r), x, y, v);
    }
}

int
dt_delete(dt_t **t, mkey_t x) {
    if ( *t == NULL ) return BT_NOTFOUND;
    if ( x < (*t)->x ) {
        return dt_delete( &((*t)->l), x);
    } else if ( x > (*t)->x) {
        return dt_delete( &((*t)->r), x);
    } else {
        dt_t *tmp = *t;
        *t = dt_merge((*t)->l, (*t)->r);
        free(tmp);
        return BT_ADDED;
    }
}

dt_t* dt_left(dt_t *t) {
    while(t->l != NULL) t = t->l;
    return t;
}

dt_t* dt_right(dt_t *t) {
    while(t->r != NULL) t = t->r;
    return t;
}
```

- Реализация на Pascal

-- ArtemVoroztsov - 02 Mar 2005