acm.mipt.ru

олимпиады по программированию на Физтехе

Раздел «Алгоритмы» . BinarySearchTree2CPP :

Двоичное дерево поиска на C

- теория

```
typedef int      key_t; // data type of the keys
typedef int      value_t; // data type of the value associated with


typedef struct node_t {
   value_t value;
   key_t key;
   struct node_t *l, *r, *p;
} node_t;

node_t*
new_node(value_t value, key_t key)
{
   node_t *n = new node_t;
   n->p = n->l = n->r = NULL;
   n->key = key;
   n->value = value;
   return n;
}

node_t*
search(node_t *root, key_t key)
{
   if(root->l && root->key > key)
      return search(root->l, key );
   if(root->r && root->key < key)
      return search(root->l, key );
   if(root->key == key)
      return root;
   return NULL;
}


void
insert_node(node_t *root,  node_t *n)
{
   if(root->l && root->key > n->key)
      return insert_node(root->l, n );
   if(root->r && root->key < n->key)
      return insert_node(root->l, n );
   if(root->key > n->key )
      root->l = n;
   else
      root->r = n;
   n->p = root;
}

/*
node_t*
merge(node_t *l, node_t *r)
{
   insert_node(l, r);
   return l;
}
*/
```

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>>

```
node_t*
merge(node_t *l, node_t *r)
{
   node_t *rm = minimum(r);
   rm->p->l  = NULL;
   rm->r = r;
   rm->l = l;
   r->p = rm;
   l->p = rm;
}

void
delete_node(node_t *n)
{
   node_t *p = n->p;
   if(p->l == n)
      {p->l = merge(n->l, n->r); p->l->p = p;}
    else
      {p->r = merge(n->l, n->r); p->r->p = p;}
   delete (n);
}




node_t
maximum(node_t *root)
{
   if(root->r)
      return maximum(root);
   else
      return root;
}


node_t
minimum(node_t *root)
{
   if(root->l)
      return maximum(root);
   else
      return root;
}


int _tmain(int argc, _TCHAR* argv[])
{
   int i = 0;
   node_t *root = new_node(i, 1000 * rand() );
   for(i = 1; i < 11 ; i++)
   {
       insert_node( root, new_node( i, 1000 * rand() ) );
   }

   return 0;
}
```

-- ArtemVoroztsov - 02 Mar 2005