NAME

nbox_t – Multi-dimensional box class

SYNOPSIS

```
#include <nbox.h>

class nbox_t {

    nbox_t();
    nbox_t(int dimension);
    nbox_t(int dimension, int box[]);
    nbox_t(const nbox_t& nbox);
    nbox_t(const char* s, int len); // for conversion from tuple key
    nbox_t(const char* s);          // for conversion from ASCII for Tcl

    virtual ~nbox_t() {}

    int dimension() const      { return dim; }
    int bound(int n) const     { return array[n]; }
    int side(int n) const      { return array[n+dim]-array[n]; }
    int center(int n) const { return (array[n+dim]-array[n])/2+array[n]; }

    bool    empty() const;    // test if box is empty
    void    squared();  // make the box squared
    void    nullify();  // make the box empty

    int hvalue(const nbox_t& universe, int level=0) const; // Hilbert value
    int hcmp(const nbox_t& other, const nbox_t& universe,
                int level=0) const; // Hilbert value comparison

    void print(int level) const;
    void draw(int level, FILE* DrawFile, const nbox_t& CoverAll) const;

    //
    // area of a box :
    //      >0 : valid box
    //      =0 : a point
    //      <0 : null box
    //
    double area() const;

    //
    // margin of a Rectangle
    //
    int margin();

    //
    // some binary operations:
    //      ^: intersection  ->  box
    //      +: bounding box  ->  box (result of addition)
    //      +=: enlarge by adding the new box
    //      ==: exact match  ->  boolean
    //      /: containment   ->  boolean
    //      ||: overlap   ->  boolean
```

```
        //      >: bigger (compare low values) -> boolean
        //      <: smaller (compare low values) -> boolean
        //      *: square of distance between centers of two boxes
        //
        nbox_t  operator^(const nbox_t& other) const;
        nbox_t  operator+(const nbox_t& other) const;

        nbox_t&       operator+=(const nbox_t& other);
        nbox_t&       operator=(const nbox_t& other);
        bool          operator==(const nbox_t& other) const;
        bool          operator/(const nbox_t& other) const;
        bool    operator||(const nbox_t& other) const;
        bool    operator>(const nbox_t& other) const;
        bool    operator<(const nbox_t& other) const;
        double  operator*(const nbox_t& other) const;

        //
        // for tcl use only
        //
        operator      char*();
        void put(const char*);  // conversion from ASCII for tcl

        //
        // conversion between key and box
        //
        void  bytes2box(const char* key, int klen);
        const void* kval() const { return (void *) array; }
        int   klen() const { return 2*sizeof(int)*dim; }

    };
```

DESCRIPTION

TODO

VERSION

This manual page applies to Version 2.0 of the Shore Storage Manager.

SPONSORSHIP

The Shore project is sponsored by the Advanced Research Project Agency, ARPA order number 018 (formerly 8230), monitored by the U.S. Army Research Laboratory under contract DAAB07-91-C-Q518. Further funding for this work was provided by DARPA through Rome Research Laboratory Contract No. F30602-97-2-0247.

COPYRIGHT

SEE ALSO

rtree(ssm)