Q: How do I use the arguments to the various HashTable methods? I don't understand the mechanics of HashTable storage for my data.

A: The following code snippet shows how the arguments to the various HashTable methods should be used. Note that even though data can be stored in the hash table as **int**, **char ***, **id**, **void ***, or any other 32-bit quantity that can be described by a type string, the arguments of the hash table methods, such as
insertKey:value: , **nextState:key:value**, **valueForKey:** have to be cast to void * or const void * respectively. See the documentation on HashTable for more details.

```
static char *month_name[]={
       "january", "february", "march",
       "april", "may", "june",
       "july", "august", "september",
       "october", "november", "december" };

- hashTableTest:sender
{
```

```
NXHashState state;
const void *key;
void *val;
id table;

unsigned count = 0;


/* Allocate a hash table with the capacity of 12 entries. The Key
 * type is char *, the Value type is int.
 */

table =[[HashTable alloc] initKeyDesc:"*" valueDesc:"i" capacity:12];

/* Insert data into the table. Note that count ranges from 0 to 11 */
while ( count < 12 )
    {
    [table insertKey:(const void *)month_name[count] value:(void *)count];
    count++;
    }
```

```
/* Retrieve the key/value pairs from the table
 * and display them.
 * Note that you can only enumerate entries in the table once it has been
 * initialized with the insertKey:value: method.
 */
state = [table initState];

while ( [table nextState:&state key:(void *)&key value:(void *)&val] )
    {
    fprintf(stderr, "key %s value %d\n", key, val);
    }

/* Retrieve a particular value from the table
 * knowing its key
 */
if ( [table isKey:(const void *)"may"] )
       val = [table valueForKey: (void*)"may"];
```

```
        return self;
    }
```

Valid for 1.0, 2.0, 3.0
N.B. (The Release 2 method **initKeyDesc:valueDesc: capacity** should be replaced by the Release 1 method **newKeyDesc:valueDesc:capacity**).

QA714