

CS480

Hash Tables, Dynamic Memory & the Eclipse Debugger

January 30, 2015

Debug - CS480_0/src/CS480_0.c - Eclipse SDK

File Edit Refactor Navigate Search Run Project Window Help

Debug Variables Breakpoints Registers Modules

Name	Value
theArray	0x00661b98
*theArray	0x00661bc8
(x)= **theArray	0
(x)= i	10
(x)= k	100

CS480_0.c

Notice the array declaration and mallocs. The array is not shown properly in the variable listing.

Right click on the array variable and select Display as Array

You must specify the size.

```
const int arraySize = 10;
int **theArray;
int i, k;
int slot, stride, size;
theArray = (int**) malloc (sizeof(int *) * arraySize);

for (i = 0; i < arraySize; i++)
{
    theArray[i] = (int*) malloc (sizeof(int) * (i + 1) * 10);
    for (k = 0; k < (i + 1) * 10; k++)
    {
        theArray[i][k] = k;
    }
}
```

Select All Ctrl+A

Copy Variables Ctrl+C

Enable

Disable

***[] Display As Array...**

Cast To Type...

Revert Original Type

Debug - CS480_0/src/CS480_0.c - Eclipse SDK

File Edit Refactor Navigate Search Run Project Window Help

Debug (x)= Variables Breakpoints Registers Modules

Name	Value
(x) argc	2
(x) argv	0x00661b18
(x) arraySize	10
theArray	0x00661b98
theArray[0]	0x00661bc8
theArray[1]	0x00661bf8
theArray[2]	0x00661c50
theArray[3]	0x00661cd0
theArray[4]	0x00661d78
theArray[5]	0x00661e48
theArray[6]	0x00661f40
theArray[7]	0x00662060
theArray[8]	0x006621a8

Now the first dimension of the array is shown.

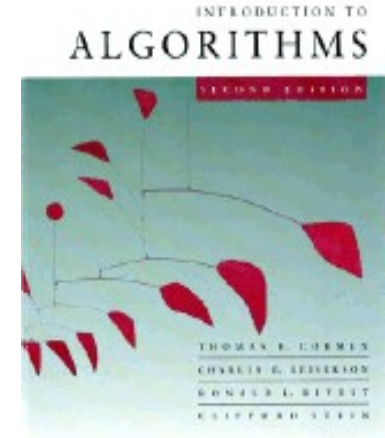
Right click on theArray[0] and repeat the above process.



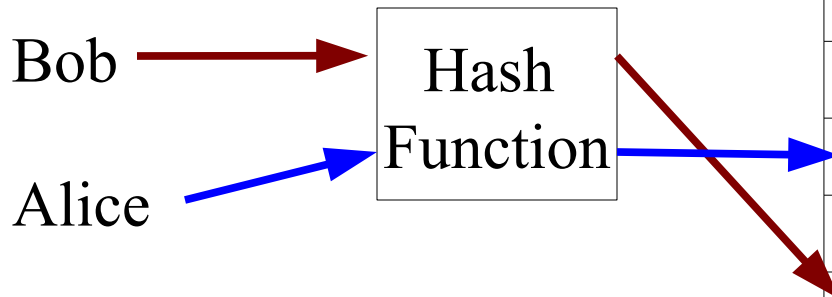
Name	Value
argv	0x00661b18
arraySize	10
theArray	0x00661b98
theArray[0]	0x00661bc8
theArray[0][0]	0
theArray[0][1]	1
theArray[0][2]	2
theArray[0][3]	3
theArray[0][4]	4
theArray[0][5]	5
theArray[0][6]	6
theArray[0][7]	7
theArray[0][8]	8
theArray[0][9]	9
theArray[1]	0x00661bf8
theArray[2]	0x00661c50
theArray[3]	0x00661cd0

Hash Tables!

- Turn data into a numeric key
 - Hash function
- Use that key to index into a table



Cormen,
Leiserson,
Rivest
ISBN-13: 978-0262032933



- The data entry in the table can contain the meaningful data

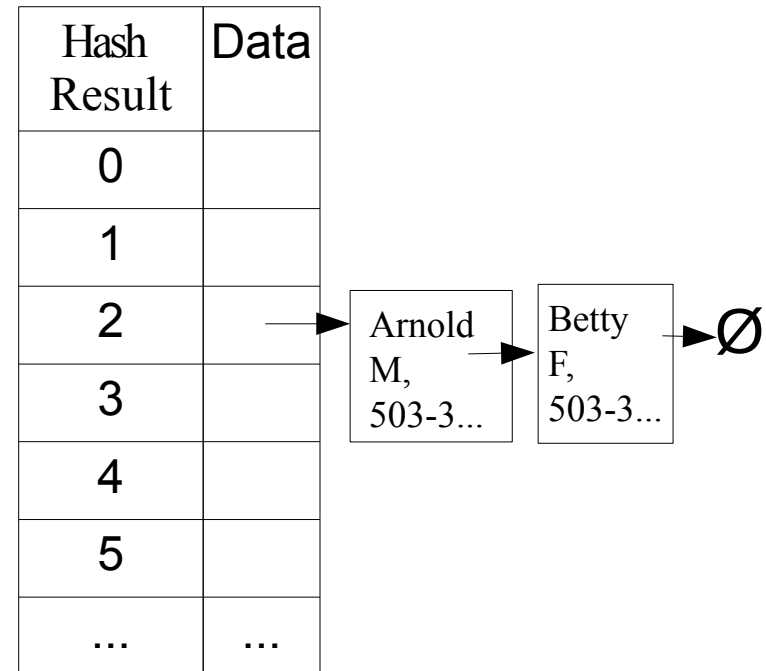
Hash Result	Data
0	
1	Alice, F, 503-352-..
2	
3	Bob, M, 503-352-..
4	
5	
...	...

http://en.wikipedia.org/wiki/Hash_tables

Hash Function

- Good hash function: spread data across the table (hash results) evenly
 - Few collisions
- Many good algorithms available
 - Check CLR

- Collision
 - Two pieces of data produce the same hash value
 - Resolve by *chaining*
 - Have table *data* entry point to linked list



/*

- How does gcc handle:

```
int *pInteger;  
int value    =4;  
pInteger = &value;  
  
printf("%d", 8/*pInteger);
```