The aim of this report is to show my progress on the first Programming Project and to demonstrate my understanding of the code I wrote. I am going to write about each of the tasks individually and attach screenshots where necessary. In the end I am also going to reflect on those aspects of the Project that I found most difficult and the things I could have done better.

**Task 1: Destroy the tree**

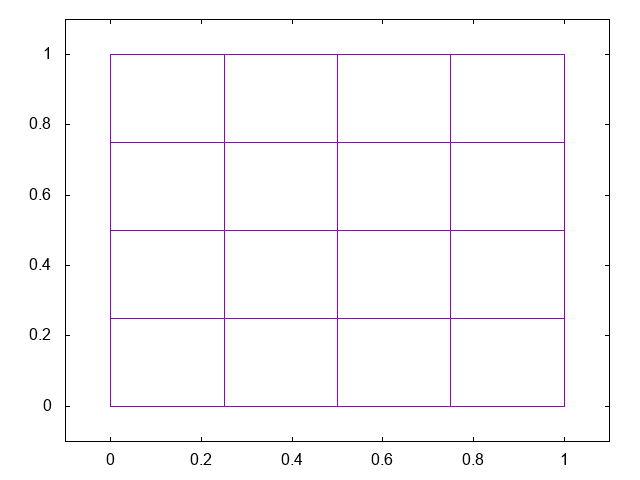
It is of prime significance to free all the memory which has been allocated. To ensure that there are no memory leaks the tree has to be destroyed (recursively).

1. A full tree at Level 2  
   Expected results: Creation of a full tree of level 2, memory leaks when the memory isn’t freed, no memory leaks after destroying the tree.

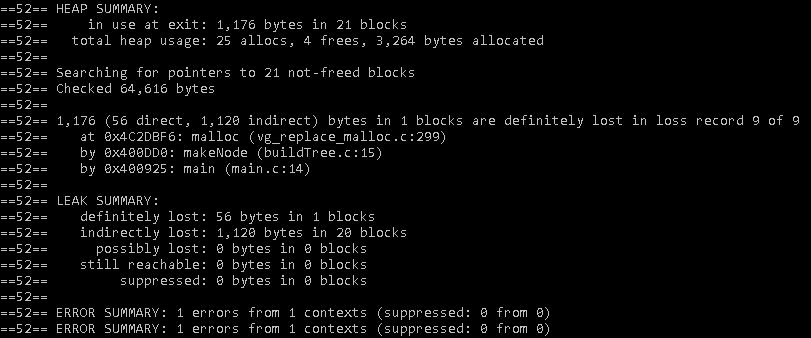
Expected input:

1  
1  
Final results:

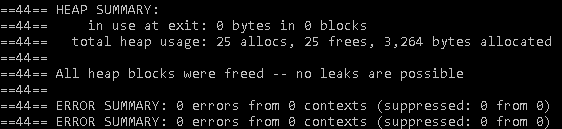
A full level 2 tree.



Before destroying the tree



After destroying the tree



1. A non-uniform Level 3 tree that you define.

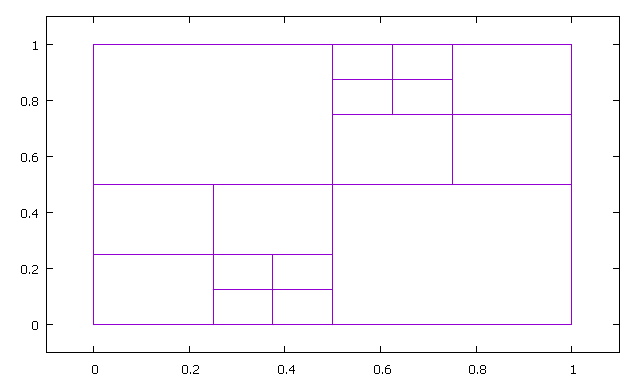
Expected results: Creation of a non-uniform tree of Level 3, memory leaks when the memory isn’t freed, no memory leaks after destroying the tree.  
Expected input:

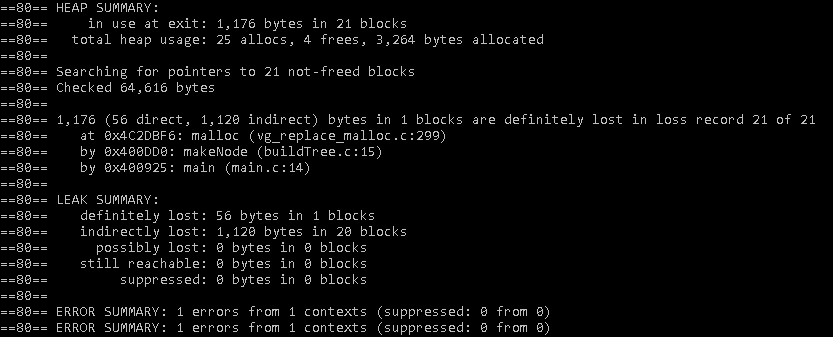
1

2

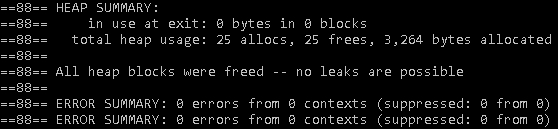
Final results:

A level 3 tree



Before destroying the tree

After destroying the tree:



Code

Checks if the node is a leaf node and frees the allocated memory if it is. Otherwise, the recursive function goes through all 4 children of the node until a leaf is reached. After freeing the children, the memory allocated to the parent is also freed.

**void** destroyTree(Node \*node) {  
 **int** i;  
  
 **if**( node->child[0] == NULL )  
 free(node);

**else** {  
 **for** ( i=0; i<4; ++i ) {  
 destroyTree( node->child[i] );  
 }  
 free(node);  
 }  
}

### Task 2: Growing the Quadtree

This function recursively grows the tree uniformly overall by one level.