**Assignment #3: DEBUGGING TIPS**

(Activate and run the Simple Test Routine built within the start-up codes)

Dear class,

In order to help the class debug the codes more accurately for Assignment #3, I have prepared a guide here for examining the results of calculations in a step-by-step manner.

The start-up codes have a built-in Simple Test Routine that can be activated for debugging purposes. This will help simplify the analysis of your codes.

---------------------------------------------------------------------------

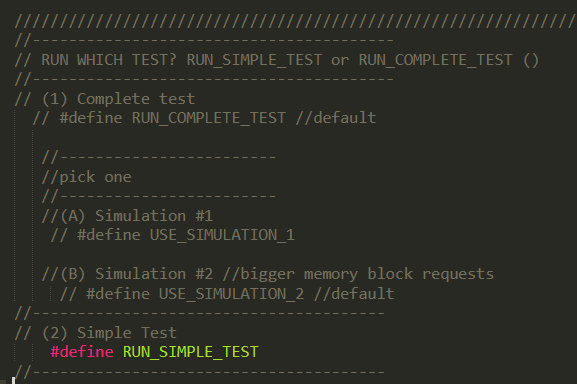
Firstly, it would be helpful in debugging your codes, if you could display the number of nodes created inside your Free List (also called Buddy List).

You could also display the base address and the size of each of the nodes in the Free List - this will tell us if memory is correctly being segmented (when allocating) and coalesced (when freeing).

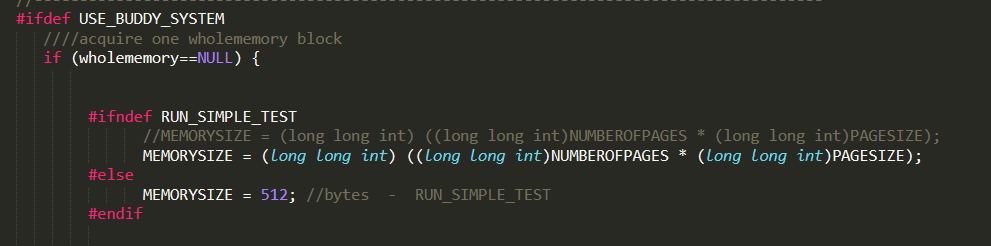
---------------------------------------------------------------------------

For debugging purposes, I have actually created a **simple test routine** (inside the start-up codes) that will make it easier.

In order to activate this simple test, modify the **auxillary.h** file to enable **RUN\_SIMPLE\_TEST** as follows:



Inside **main.cpp**, I have actually made a provision for this Simple Test already, it will automatically switch to using **512** bytes for the MEMORYSIZE.



**Next, given 512** bytes, **this Simple Test will execute the following sequence of memory allocation and deallocation requests:**

**malloc(13), malloc(3), malloc(110), free(3), free(13)**

**//-- this is due to my codes:**

**char actions[] = {'m' , 'm', 'm', 'f', 'f'}; //1 = MALLOC, 0 = FREE**

**int requests[] = {13, 3, 110, 3, 13}; //if NUMBEROFPAGES\_BUDDY is set to 64**

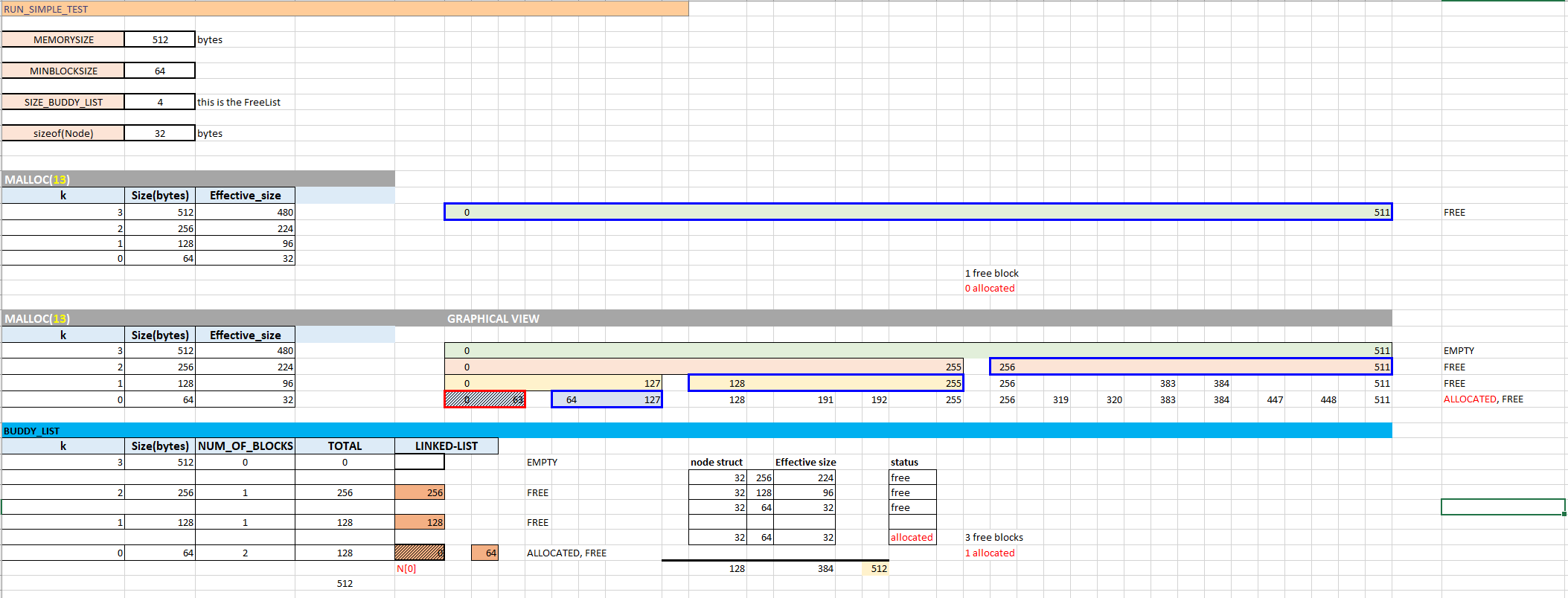
**//---**

**Set the Free List to have 4 rows, so that we can easily trace what the Buddy System is doing:**

|  |  |
| --- | --- |
| **k** | **Size(bytes)** |
| 3 | 512 |
| 2 | 256 |
| 1 | 128 |
| 0 | 64 |

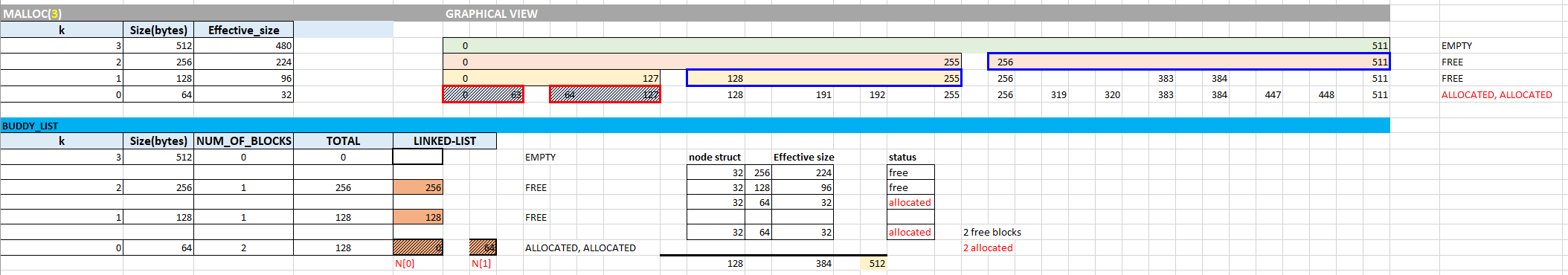
**To debug, I suggest that you display the number of Nodes created in your Free List. I am providing (see diagrams below) a break-down of what the Buddy System should do with the Free List/Buddy List (Note that the addresses are all relative):**

**Malloc(13) should cause the Buddy System to create the following Nodes inside Free List:**

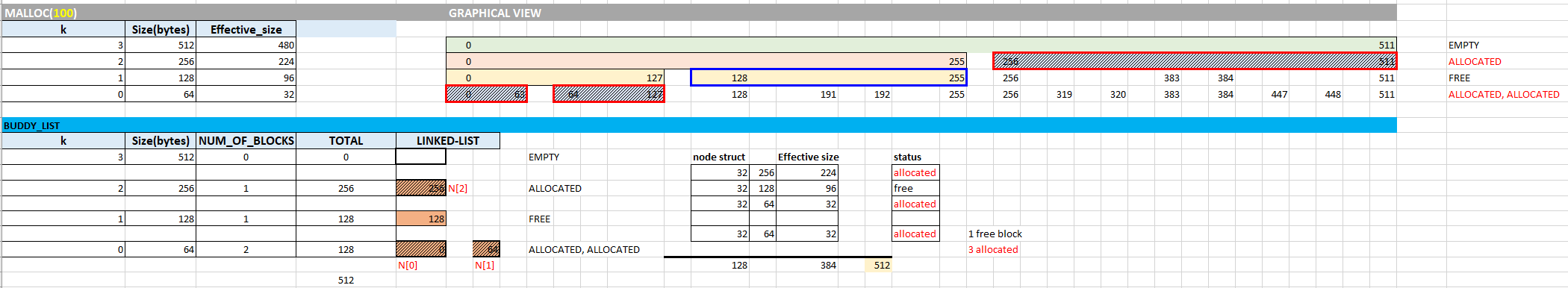


**As can be seen in the diagram** (look for the Graphical View header)**, Malloc(13) should cause the Buddy System to create 3 Nodes in the Free List: one of size 256 ( minus the Node size), another of size 128 (minus the Node size) and lastly, a node of size 64 ( minus the Node size). The allocated block is of size 64 and is returned to the user.**

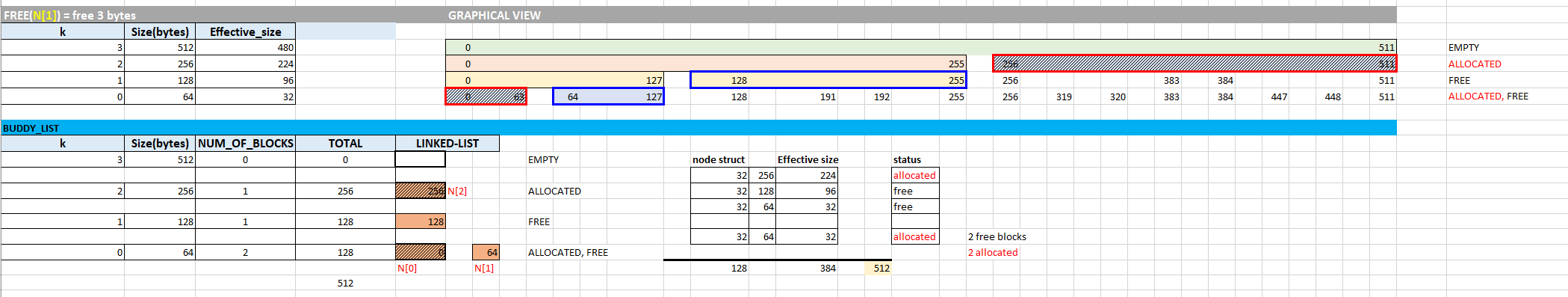
**Next, Malloc(3) should cause the Buddy System to create the following Nodes inside Free List:**



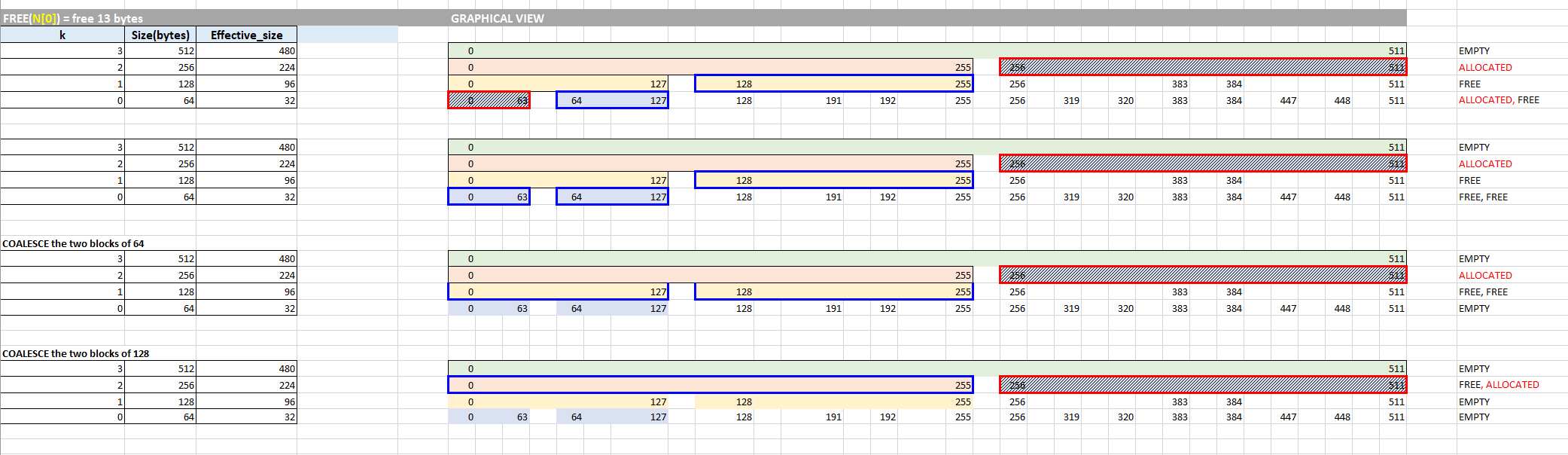
**Next, Malloc(100) should cause the Buddy System to create the following Nodes inside Free List:**



**Next, Free(3) should cause the Buddy System to create the following Nodes inside Free List. We are interested in seeing how free blocks are coalesced:**



**Lastly, Free(13) should cause the Buddy System to create the following Nodes inside Free List:**



I hope this helps.

Cheers,

Napoleon