Name: Yu Siying

ZJU ID: 3210110858

intl ID: siying.21

UIUC NetID: siyingy3

a. analysis of memory allocation and running time in your original MP5 implementation

Memory Allocation: For each tile image, allocate memory to store its original PNG, resized PNG and other parameter(e.g. averageColor_). Can't get numerical allocate data since valgrind would crash when running MP5.

Running Time: O(w*h+n*w'*h')

b. analysis of memory allocation and running time in your new MP6 implementation

Memory Allocation: For each tile image, allocate memory to store its resized PNG and other parameter averageColor_. (No longer need to store its original PNG image). Numerical data: 4,075,093 bytes allocs.

```
ysy@ubuntu:~/UserFile/CS225/Drafts/MP6$ valgrind ./mp6 tests/source.png ../MP5/mp5_pngs/ 400 5 mosaic.png
==19022== Memcheck, a memory error detector
==19022== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==19022== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==19022== Command: ./mp6 tests/source.png ../MP5/mp5_pngs/ 400 5 mosaic.png
==19022==
Loading Tile Images... (4730/4730)... 4479 unique images loaded
Populating Mosaic: setting tile (399, 532)
Drawing Mosaic: resizing tiles (213200/213200)
Saving Output Image... Done
==19022==
==19022== HEAP SUMMARY:
==19022==     in use at exit: 0 bytes in 0 blocks
==19022==   total heap usage: 4,075,093 allocs, 4,075,093 frees, 24,538,818,120 bytes allocated
==19022==
==19022== All heap blocks were freed -- no leaks are possible
==19022==
==19022== For lists of detected and suppressed errors, rerun with: -s
==19022== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

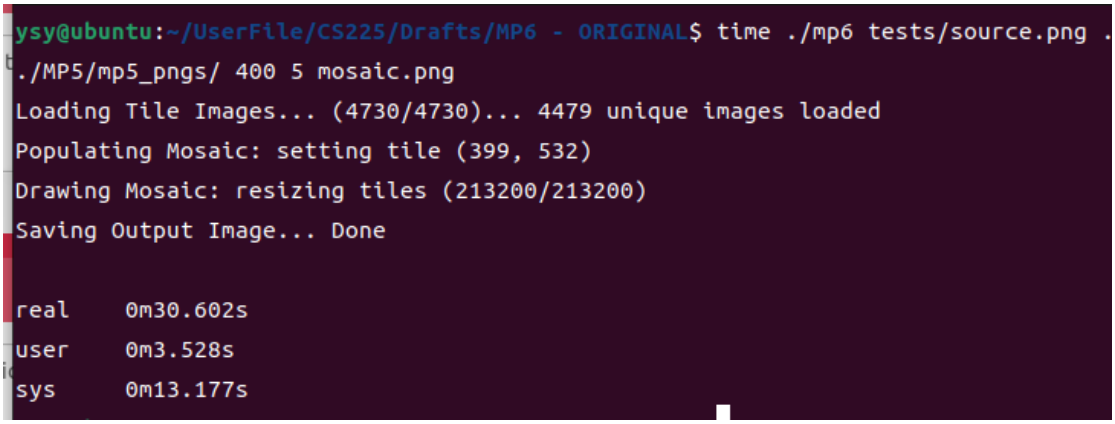
Running Time: The same time complexity as the one in MP5, but cost a little longer since MP6 need to open tile image source file several (generally 3) times more when resizing.

c. description of changes made to reduce memory footprint and running time.

Change: Delete "PNG image" variable in the private field of TileImage, add "string path" in the private field of TileImage. In this way, memory footprint would be reduced (since PNG is always big). However, as a sacrifice, the program would cost slightly more time.

For Running Time, the original running time is the suggested one, so it makes no sense to make other modification. n*w'*h' is for resizing tile images, and w*h is used to generated the final mosaic canvas. It's already the best. Since we need to improve space managing, running time is sacrificed a little. Here's the result.

MP5:



MP6: Mainly took more time in second part