1. Use the simulator to perform some basic RAID mapping tests. Run with different levels (0, 1, 4, 5) and see if you can figure out the mappings of a set of requests. For RAID-5, see if you can figure out the difference between left-symmetric and left-asymmetric layouts. Use some different random seeds to generate different problems than above.

nickliu@Nicks-Mac chapter38:$ ./raid.py -L 5 -5 LS -W seq -c -n 12

Left Symmetric			
Disk 0	Disk1	Disk2	Disk3
0	1	2	p
4	5	p	3
8	p	6	7
p	9	10	11

nickliu@Nicks-Mac chapter38:$ ./raid.py -L 5 -5 LA -W seq -c -n 12

left-asymmetric			
Disk 0	Disk1	Disk2	Disk3
0	1	2	p
3	4	p	5
6	p	7	8
p	9	10	11

2. Do the same as the first problem, but this time vary the chunk size with -C. How does chunk size change the mappings?

./raid.py -L 5 -5 LS -W seq -n 12 -c -C 8192

Left-symmetric			
Disk 0	Disk1	Disk2	Disk3
0	2	4	p
1	3	5	p
8	10	p	6
9	11	p	7

./raid.py -L 5 -5 LA -W seq -n 12 -c -C 8192

Left-asymmetric			
Disk 0	Disk1	Disk2	Disk3
0	2	4	p
1	3	5	p
6	8	p	10
7	9	p	11

3. Do the same as above, but use the -r flag to reverse the nature of each problem.
./raid.py -L 5 -5 LA -W seq -n 12 -c -C 8192 -r

4. Now use the reverse flag but increase the size of each request with the -S flag. Try specifying sizes of 8k, 12k, and 16k, while varying the RAID level. What happens to the underlying I/O pattern when the size of the request increases? Make sure to try this with the sequential workload too (-W sequential); for what request sizes are RAID-4 and RAID-5 much more I/O efficient?

When is 16k the , it use all 4 disk to do I/O operation which is the maximum parallelism.

5. Use the timing mode of the simulator (-t) to estimate the performance of 100 random reads to the RAID, while varying the RAID levels, using 4 disks.

./raid.py -t -n 100 -D 4 -W rand -c -L 0
Total time 275.69

./raid.py -t -n 100 -D 4 -W rand -c -L 1
Total time 278.69

./raid.py -t -n 100 -D 4 -W rand -c -L 4
Total time 386.1

./raid.py -t -n 100 -D 4 -W rand -c -L 5
Total time 276.7

6. Do the same as above, but increase the number of disks. How does the performance of each RAID level scale as the number of disks increases?

For each level, when increase the number of the disk, the performance also increases.

7. Do the same as above, but use all writes (-w 100) instead of reads. How does the performance of each RAID level scale now? Can you do a rough estimate of the time it will take to complete the workload of 100 random writes?

./raid.py -t -n 100 -W rand -c -L 0 -w 100 -D 4
STAT totalTime 275.69999999999993

./raid.py -t -n 100 -W rand -c -L 1 -w 100 -D 4
STAT totalTime 509.80000000000047

./raid.py -t -n 100 -W rand -c -L 4 -w 100 -D 4
STAT totalTime 982.5000000000013

./raid.py -t -n 100 -W rand -c -L 5 -w 100 -D 4
STAT totalTime 497.40000000000043

Level 0, 1, 5 have almost equal performance, but level 4 have worst performance

8. Run the timing mode one last time, but this time with a sequential workload (-W sequential). How does the performance vary with RAID level, and when doing reads versus writes? How about when varying the size of each request? What size should you write to a RAID when using RAID-4 or RAID-5?

./raid.py -t -n 100 -W sequential -c -L 0 -w 100 -D 4
STAT totalTime 12.499999999999991

./raid.py -t -n 100 -W sequential -c -L 1 -w 100 -D 4
STAT totalTime 14.999999999999982

./raid.py -t -n 100 -W sequential -c -L 4 -w 100 -D 4
STAT totalTime 13.399999999999988

./raid.py -t -n 100 -W sequential -c -L 5 -w 100 -D 4
STAT totalTime 13.399999999999988

All levels have almost equal performance for sequential.

The size for RAID-4 and RAID-5 should number of disk time size of the block to achieve max performance, by parallelism all 4 disks.