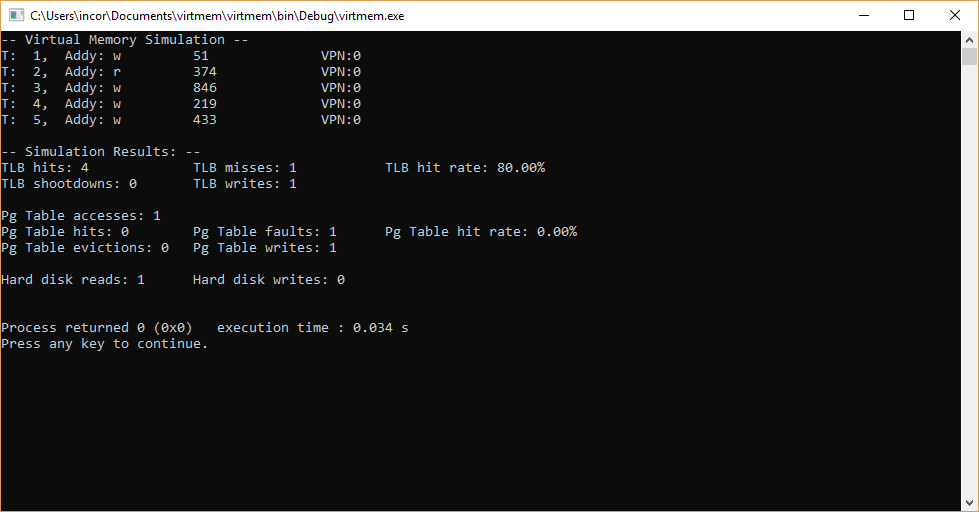
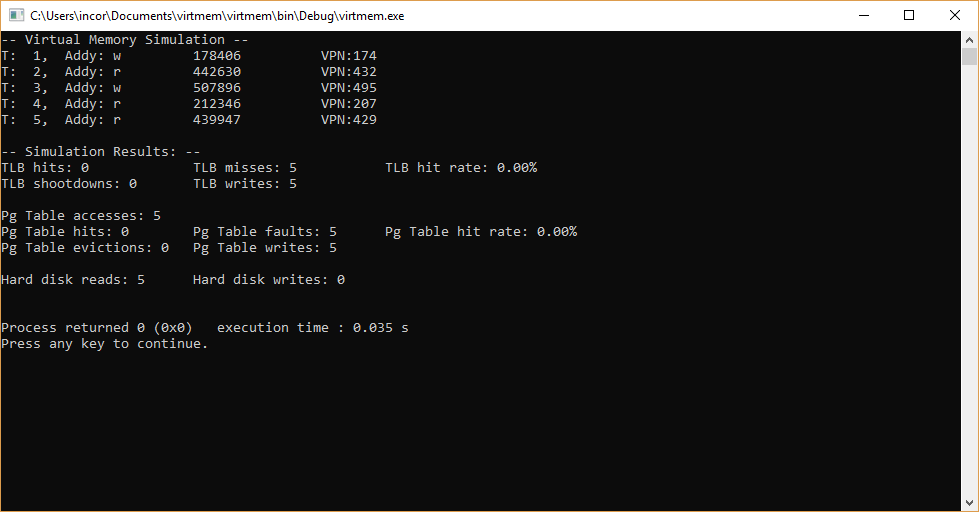
**Case Study #1** **Access 5 addresses from the same page**



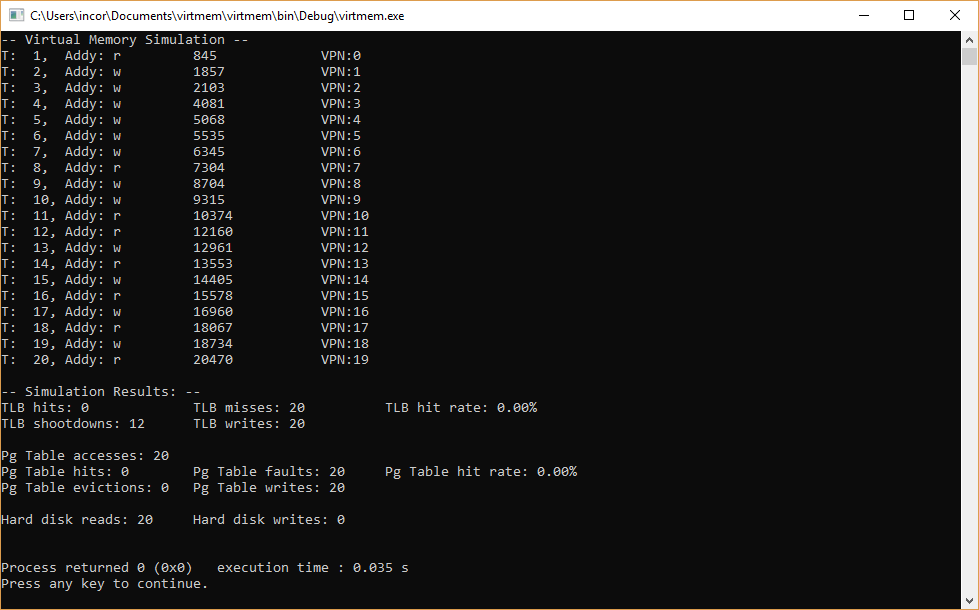
Since all these memory access are from the same page, there is one miss and one TLB write, followed by 4 successive hits since the page is stored in the TLB and never evicted. There is one PT access, in the form of a write to the page table after the initial fault. Much like with the TLB, since all accesses are from the same page, there only needs to be one write to satisfy all subsequent accesses. One entry is read from the HD as it is entered into the cache hierarchy.

**Case Study #2** **Access 5 addresses from different pages**



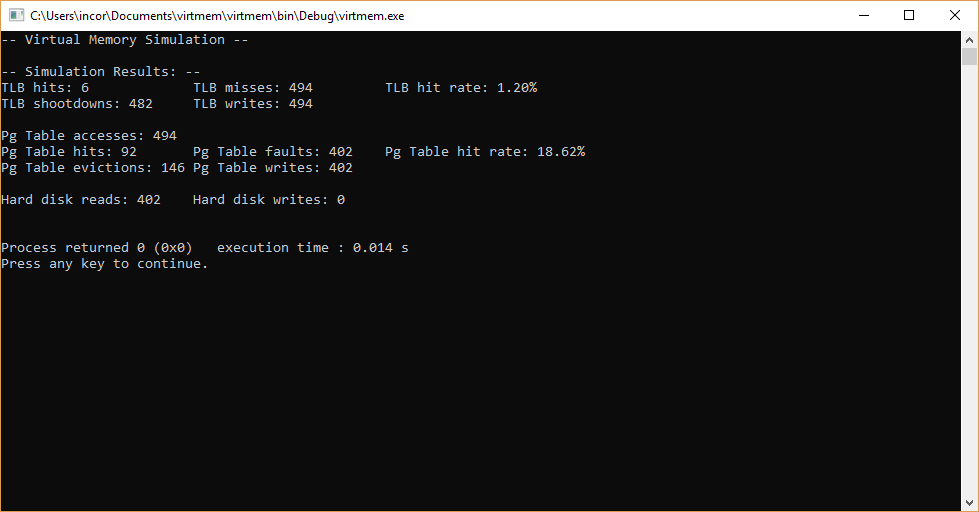
Here, each page is being written into the TLB for the first time upon each access, so there are 5 misses and 5 writes. Since the size of the TLB is 8, these entries are now ready for future accesses, with no evictions having taken place. If any of these pages were accessed again, the TLB hit rate would increase. Similarly, there are 5 page faults on each new page access, but 5 writes when these entries are allocated a frame. Further accesses from these pages will result in hits in both the TLB and PT. These 5 pages are read from the HD into memory which is why there are 5 HD reads.

**Case Study #3** **Access 20 addresses from different pages**



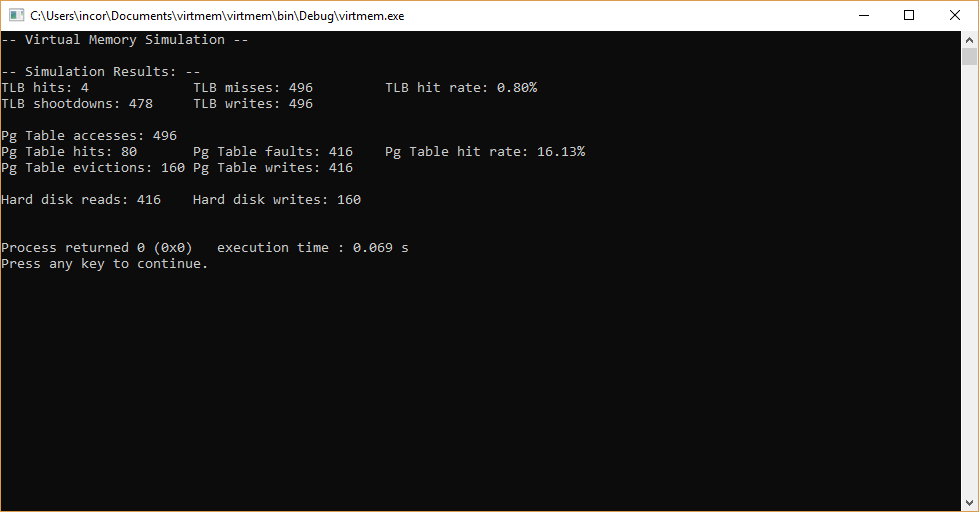
Since these are all different pages, all 20 will be TLB misses. The capacity of the TLB is 8, so there will be 20-8 shootdowns as all these new pages are read into the TLB, which is what we see: 12 TLB shootdowns. The same goes for the page table, all accesses are faults and there are no hits. Similarly, as these 20 locations are read into memory from the HD, there is a read per page, resulting in a total of 20 reads.

**Case Study #4** **Access 500 random addresses, r/w probability 50%**



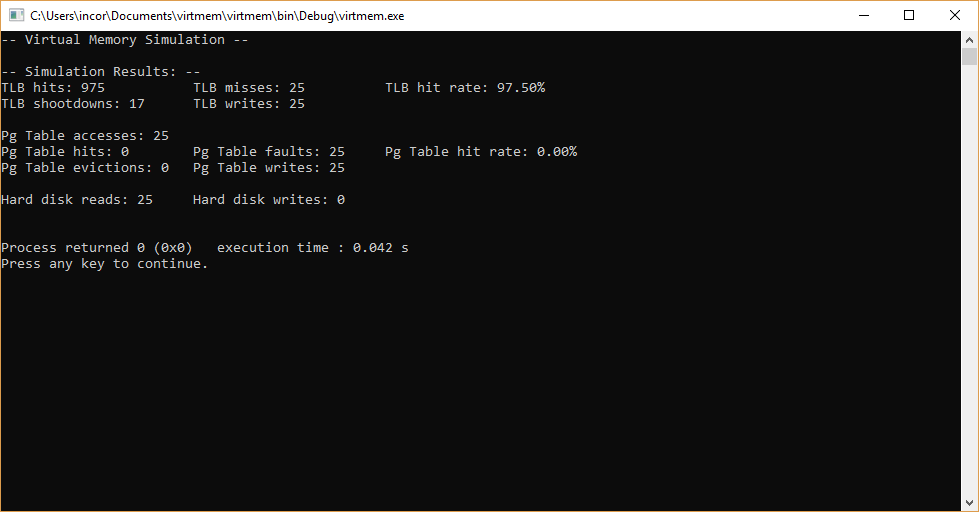
Here the TLB hit rate average of ~.8% is due to some of the randomly generated pages already being cached, even though this number is very low. Most VPNs are new and cause a TLB shootdown to replace an unneeded entry, but this will, rarely, match an entry already in the TLB, causing a hit. Page evictions happen when a frame is needed but is not currently assigned a slot in the RAM. Hd writes are low because the change of a page being written to, then written to again, is likely to be very low without the write chance being higher than 50%.

**Case Study #5** **Access 500 random addresses, all accesses are writes**



This somewhat mirrors the same results above, where due to the “random” nature of the addresses generated, TLB hits average just below a percentage point. Page evictions are high compared to hits as previously, and due to the 100% nature of writes happening here, each page eviction needs to be written back to the hard disk in order to make sure the data is represented correctly in the hierarchy, so these two values match completely.

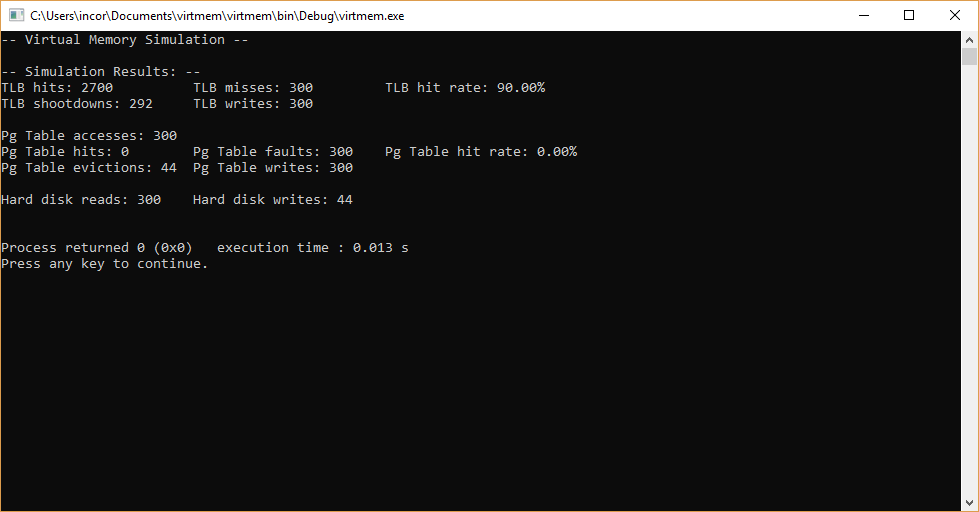
**Case Study #6** **1000 memory accesses; each 40 consecutive addresses from the same page**



Here, there are 25 unique pages being accessed, so 17 shootdowns arises from 25-8, which is the size of the TLB. This explains the high hit rate, sequential accesses are checking the TLB first and immediately getting a hit once the page is entered the first time after the mandatory cache miss. Pages are not being evicted because the frame table has not exceeded its capacity, and the page table is only accessed when the TLB fails to report success.

The hit rate can be calculated by determining number of TLB hits – that is, entries already in the buffer upon request – divided by (tlb hits + tlb misses), then multiplying by 100 to obtain a percentage. Since there are 40 sequential accesses from each page, this number will be very high as there is only one mandatory cache miss per new page in the TLB.

**Case Study #7** **3000 memory accesses; each 10 consecutive addresses from the same page**

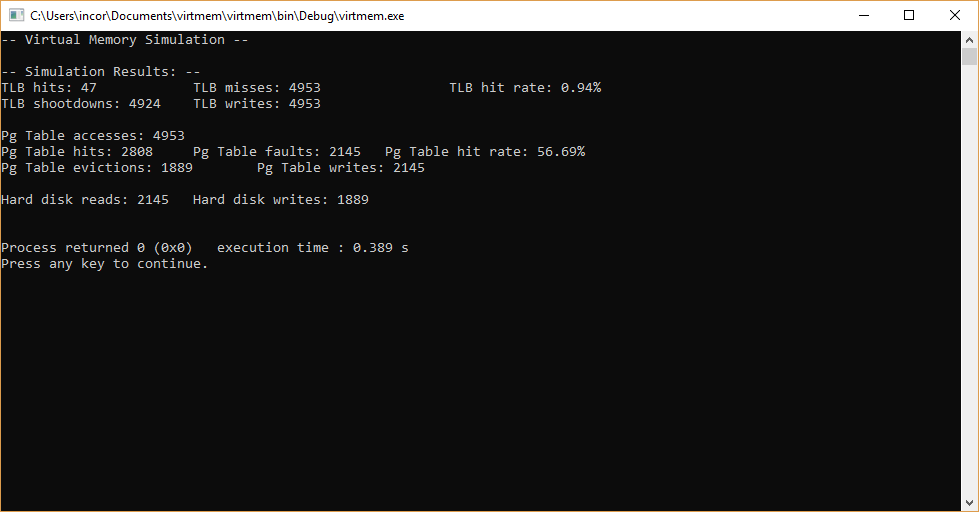


Here we again see a high TLB hit rate due to sequential accesses from each page, although the number of accesses is lower so there is a higher number of shootdowns as new pages are loaded into the TLB. The hit rate can be calculated by calculating the number of unique pages, then subtracting the capacity of the TLB. This is 300-8=292. The percentage can be found, as in #6, by dividing hits by (hits + misses) and multiplying by 100 to get a percentage.

Page evictions happen when there is not an available frame, so the 300 unique accesses can fit into the 256 slots of the frame table with 44 evictions. 300-256=44. (unique pages minus frame slots equals evictions).

There are 300 hard disk reads as each page is read from the HD into the memory hierarchy. The hard disk writes of 44 corresponds with the evictions of 44 because, on average, a write rate of 50% will leave at least one memory access as a write, therefore each page that is not needed anymore by the system will need to be written back to the HD to maintain data integrity and make sure that the work done on that data is reflected and saved for future use. If the write rate was much lower, such that all accesses on a page did not result in a single write, writes to the drive would not be a high number.

**Case Study #8** **Access 5 addresses from the same page**



Here, fully random addresses are accessed. We see a slightly higher hit rate on average than in the earlier examples due to a higher n of 5000, which results in about a 1% success rate on actually finding the entry where it should be in the TLB. With a write rate of 50%, just below 40% of accesses result in a dirty page being written back to the HD. Due to the fully random nature of the address, slightly higher than 40% of accesses result in a HD read in order to populate the data into the memory cache hierarchy, since many pages are already loaded at the time of access.