HW 4 Solution

1. 1) If the block is in the cache, return the copy of the block.
   2) If a write request in the request queue refers to the same block, return the copy of the block.
   3) Create a read request and add it to the queue, and block the current process. After the request is satisfied, remove a previous entry (in the cache) that refers to the same block (if it exists), and unblock the waiting process. This is because a read request transforms into a cache entry after the request is satisfied.

2. 1) If any write request in the request queue refers to the same block, overwrite the content and return.
   2) If the cache holds an entry that refers to the same block, remove it from the cache.
   3) Create a write request and add it to the queue.

3. The order of the request queue goes from the top to bottom (head to tail).

Item	**Type**	**Block number**	**pid**
1	Read	760	2
2	Write	200	-

4. The order is the same as above.

Item	**Type**	**Block number**	**pid**
1	Write	307	-

   * The event at time 7 is satisfied without entering the request queue, since the content for block 307 was in the write request node.

5. Two different cases are marked as correct. The first case is when the event at time 14 is removed (since process 3 is blocked at that time). The second case is when the event did occur (but requested by some other unblocked process).

   - Case 1
     - Cache has four entries: 118, 333, 611, and 900.
     - Number of reads: 4.
     - Number of writes: 0.

     * At time 9, block 900 was already in the cache.
   - Case 2
     - Cache has three entries: 333, 611, and 900 *(or four: it could contain 118).
     - Number of reads: 4.
     - Number of writes: 1.

     * I accepted two different answers. The first case is when we have not considered what rdsprocess does (it actually places a copy to the cache just before sending a message, but Module 11 did not cover this). The second case is when we did take rdsprocess into consideration.