Report

Creating cache:

The details of cache are extracted from the config file i.e the first input file.
The cache implemented is a 2D array of blocks(structure defined) of sets and associativity
My approach was to create an array of linked lists based on replacement policy for FIFO and LRU.
If it is fully associative then the set is made 1 with all others as ways whereas for direct mapped way is 1 and the set is found accordingly.
In set associative cache sets(s) are calculated based on block size(b),cache size(c) and associativity(a) as s=c/(b*a).
Now,first valid bits in all blocks are initialised with 0.

Checking hit and Miss:

After reading access file input, all the different addresses are read each time and their set index,tag are found.
This tag is compared each time with elements in the set and also checked if the valid bit is 0.
If tag matches then it is considered a hit else miss.

Handling hit and miss:

If it has valid bit 0 and tag matches then it is a hit and in LRU policy this way is moved to head (symbolising that it is most recent).
In FIFO and random no change is made.

Replacing policy:

In case of miss, if there is a empty block present in set then it is placed in it. This way is determined by placein variable.

If there is no empty space for a block in its respective set then it is replaced according to policy.

In case of LRU,it is replaced with tail way and is made head of the linked list as it is recently accessed.

In the case of FIFO, it is replaced with the head which is first accessed(old in terms of time).

In case of RANDOM, it is replaced with a randomly using rand function whose seed is (time xor pid).

In the case of WB (allocate)or R it is the same as above.
If it is WT (no allocate) then hit handling is the same but miss is not brought to cache.

Testing correctness:

I have used some trace files to test my code and calculated,checked manually if it is correct.
I checked with my friends' test cases as well.
I have used cases where output it is different for different replacement policies (LRU,FIFO,RANDOM) and checked with various possible combinations.