DAY 21 - 111 DAYS VERIFICATION CHALLENGE

Topic: Cache replacement algorithms

Skill: Cache memories

DAY 21 CHALLENGE:

1. Why does direct mapping not need any replacement algorithm?

Direct mapping doesn't need a replacement algorithm because each memory block maps to exactly one predetermined cache line, leaving no flexibility or choice in placement. If a new block needs to be loaded, it simply overwrites the block currently in that specific cache line.

2. Explain the cache replacement algorithms:

Least Recently used – LRU

- **Concept**: Replaces the cache block that has not been used for the longest time.
- **Operation**: Tracks the order of usage for cache blocks and replaces the one that hasn't been accessed the longest.
- **Pros**: Effective when past usage patterns are indicative of future usage, often providing a good approximation of optimal replacement.
- **Cons**: Implementation can be complex, requiring tracking of access order.

Most recently used – MRU

- **Concept**: Replaces the cache block that was accessed most recently.
- **Operation**: The most recently used cache block is considered the least likely to be needed again soon and is replaced first.
- **Pros**: Can be effective in certain access patterns where the most recently used data is unlikely to be reused immediately.
- **Cons**: Less commonly used as it often performs worse than LRU in typical scenarios.

First in first out – FIFO

- **Concept**: Replaces the oldest cache block, regardless of how often or recently it was accessed.

- **Operation**: Tracks the order in which blocks are added to the cache and replaces the one that has been there the longest.
- **Pros**: Simple to implement, requiring only a queue to track the order of insertion.
- **Cons**: Can perform poorly if the oldest data is still frequently used, as it doesn't consider access patterns.

3. A cache memory can hold 8 frames from RAM. Consider the cache is initially EMPTY. RAM has 128 frames of 1 byte each in a byte addressable system. The processor requested frames in following sequence: 127, 8, 0, 127, 3, 5, 7, 9, 6, 3, 8, 0, 5, 11, 19, 8. Calculate the no. of cache misses for Fully associative cache mapping in

Least Recently Used (LRU)

1. **127** - Miss (Cache: 127)
2. **8** - Miss (Cache: 127, 8)
3. **0** - Miss (Cache: 127, 8, 0)
4. **127** - Hit (Cache: 8, 0, 127)
5. **3** - Miss (Cache: 8, 0, 127, 3)
6. **5** - Miss (Cache: 8, 0, 127, 3, 5)
7. **7** - Miss (Cache: 8, 0, 127, 3, 5, 7)
8. **9** - Miss (Cache: 8, 0, 127, 3, 5, 7, 9)
9. **6** - Miss (Cache: 8, 0, 127, 3, 5, 7, 9, 6)
10. **3** - Hit (Cache: 8, 0, 127, 5, 7, 9, 6, 3)
11. **8** - Hit (Cache: 0, 127, 5, 7, 9, 6, 3, 8)
12. **0** - Hit (Cache: 127, 5, 7, 9, 6, 3, 8, 0)
13. **5** - Hit (Cache: 127, 7, 9, 6, 3, 8, 0, 5)
14. **11** - Miss (Cache: 7, 9, 6, 3, 8, 0, 5, 11)
15. **19** - Miss (Cache: 9, 6, 3, 8, 0, 5, 11, 19)
16. **8** - Hit (Cache: 9, 6, 3, 0, 5, 11, 19, 8)

Cache Misses (LRU): 8

Most Recently Used (MRU)

1. **127** - Miss (Cache: 127)
2. **8** - Miss (Cache: 127, 8)
3. **0** - Miss (Cache: 127, 8, 0)
4. **127** - Hit (Cache: 8, 0, 127)
5. **3** - Miss (Cache: 8, 0, 3)

6. **5** - Miss (Cache: 8, 0, 3, 5)
7. **7** - Miss (Cache: 8, 0, 3, 5, 7)
8. **9** - Miss (Cache: 8, 0, 3, 5, 7, 9)
9. **6** - Miss (Cache: 8, 0, 3, 5, 7, 9, 6)
10. **3** - Hit (Cache: 8, 0, 5, 7, 9, 6, 3)
11. **8** - Hit (Cache: 0, 5, 7, 9, 6, 3, 8)
12. **0** - Hit (Cache: 5, 7, 9, 6, 3, 8, 0)
13. **5** - Hit (Cache: 7, 9, 6, 3, 8, 0, 5)
14. **11** - Miss (Cache: 7, 9, 6, 3, 8, 0, 11)
15. **19** - Miss (Cache: 7, 9, 6, 3, 8, 11, 19)
16. **8** - Hit (Cache: 7, 9, 6, 3, 11, 19, 8)

Cache Misses (MRU): 10

First In, First Out (FIFO)

1. **127** - Miss (Cache: 127)
2. **8** - Miss (Cache: 127, 8)
3. **0** - Miss (Cache: 127, 8, 0)
4. **127** - Hit (Cache: 127, 8, 0)
5. **3** - Miss (Cache: 127, 8, 0, 3)
6. **5** - Miss (Cache: 127, 8, 0, 3, 5)
7. **7** - Miss (Cache: 127, 8, 0, 3, 5, 7)
8. **9** - Miss (Cache: 127, 8, 0, 3, 5, 7, 9)
9. **6** - Miss (Cache: 127, 8, 0, 3, 5, 7, 9, 6)
10. **3** - Hit (Cache: 127, 8, 0, 3, 5, 7, 9, 6)
11. **8** - Hit (Cache: 127, 8, 0, 3, 5, 7, 9, 6)
12. **0** - Hit (Cache: 127, 8, 0, 3, 5, 7, 9, 6)
13. **5** - Hit (Cache: 127, 8, 0, 3, 5, 7, 9, 6)
14. **11** - Miss (Cache: 8, 0, 3, 5, 7, 9, 6, 11)
15. **19** - Miss (Cache: 0, 3, 5, 7, 9, 6, 11, 19)
16. **8** - Miss (Cache: 3, 5, 7, 9, 6, 11, 19, 8)

Cache Misses (FIFO)**: 10**