Operating Systems Lab

<u>Under guidance of:</u>

Dr. Hari Om
Assistant Professor
Dept. Of Computer Science and Engineering
IIT(ISM) Dhanbad

Title: Caching Simulation

Project guide: Mr. Pankaj Kumar

PhD Scholar ,
IIT(ISM) Dhanbad

Objective

To Simulate how cache memory improves performance.

To simulate how page replacement is done in cache memory.

Caching Simulation: Introduction

- This program simulates how caching works in a computer.
- The program retrieves the cache memory from a *file*.
- The program takes page references as input from user.
- The program then checks for the referenced page in cache memory.
- If a *page fault* occurs then the program fetches that page from memory and loads it to cache memory.
- The program determines where the new page should be loaded in cache memory.
- If cache memory is full then the program follows Optimal algorithm to determine which page should be replaced with the new page.
- The program closes when you have no further pages to reference. (I.e 0 pages to reference)
- The program shows the final state of cache memory and save it in a file for next time.

What is Cache memory?

- It is a memory placed between CPU and Main memory.
- It is faster than Main memory.
- It is smaller than Main memory.
- The purpose of cache memory is to enhance performance of the program by providing faster access to the memory.
- When a data is referenced, the program first looks inside cache memory for the data , if the data is there then it is a *cache hit* otherwise it is *cache miss*.
- In event of a cache miss (also known as *Page faults*), the page containing the data is fetched from main memory and written to cache memory.

Why Cache Memory?

- Large memories are slow, fast memories are small.
- Also faster memories are expensive. I.e. An SSD costs twice as much as a Hard Disk.
- There is a trade-off between memory size and memory speed.
- To overcome this problem, the concept of memory hierarchy is introduced.
- Memory Hierarchy: Instead of using a single memory, one may use multiple memories arranged in decreasing order of their speed.
- Registers -> Cache -> Main Memory(RAM) -> Disk

Memory Hierarchy Design:

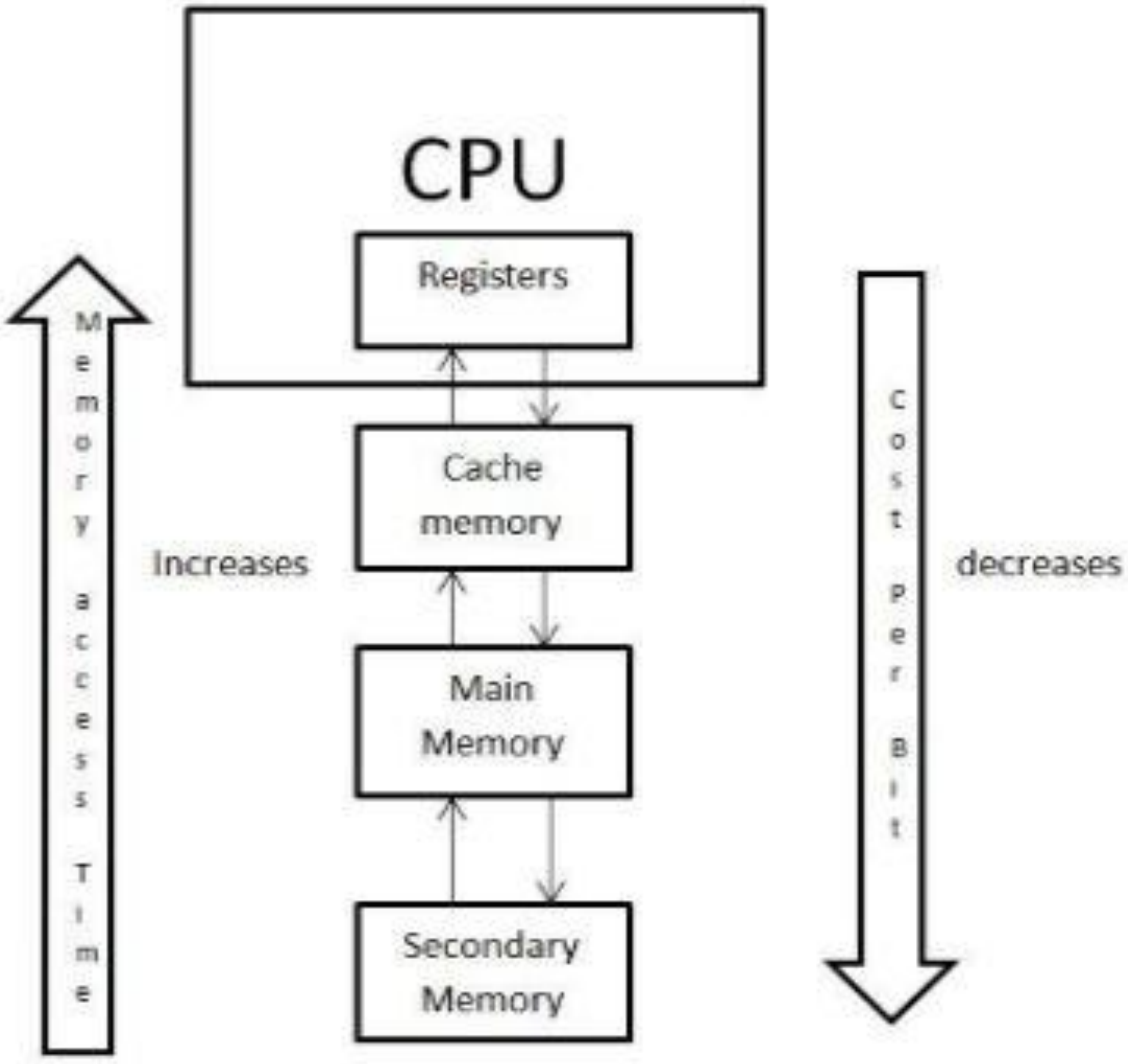


Figure 1

What are Pages and Frames?

- Physical memory divided into fixed sizes blocks called Frames.
- Logical memory divided into blocks of same size called pages.
- The frame may not be contiguous but the page will be.
- Frames are also called virtual page and frames are referred to as page frames.
- All memory chunks in physical address space are identified with frame numbers and logical address with page numbers.
- The page table consists of page number with its corresponding offset.
    - Physical address = (page size * frame number) + page offset

Why Paging?

- Paging is a memory management technique in which the memory is divided into fixed size blocks called pages.

- **Paging is used for faster access to data.**

- When a program needs a page, it is available in the main memory as the OS copies a certain number of pages from your storage device to main memory.

- Paging allows the physical address space of a process to be noncontiguous. Earlier, the whole program had to fit into storage contiguously.

What is Principle of Locality?

- Principle of locality, also known as the locality of reference, is the tendency of a processor to access the same set of memory locations repetitively over a short period of time.

- On an abstract level there are two types of localities: **Temporal locality** and **Spatial locality**.

  - **Temporal locality** includes bringing in frequently accessed memory references to a nearby memory location for a short duration of time to make future accesses faster. It is implemented through cache memory.

  - **Spatial locality** includes bringing in memory references along with their nearby memory references too. It is implemented through paging.

What is Page Replacement?

- In event of a cache miss, the page containing the data is fetched from main memory and written to cache memory.
- New pages are written in empty blocks (also known as frames).
- If no empty frame is available, then a non-empty frame is overwritten with the new page. This is called Page Replacement.
- There are various algorithms to determine which page should be replaced.
- The best Page replacement algorithm tries to minimize future page faults.
- There are some known page replacement algorithms :
  1. FIFO
  2. LRU
  3. Optimal
- In our program , we will use Optimal algorithm.

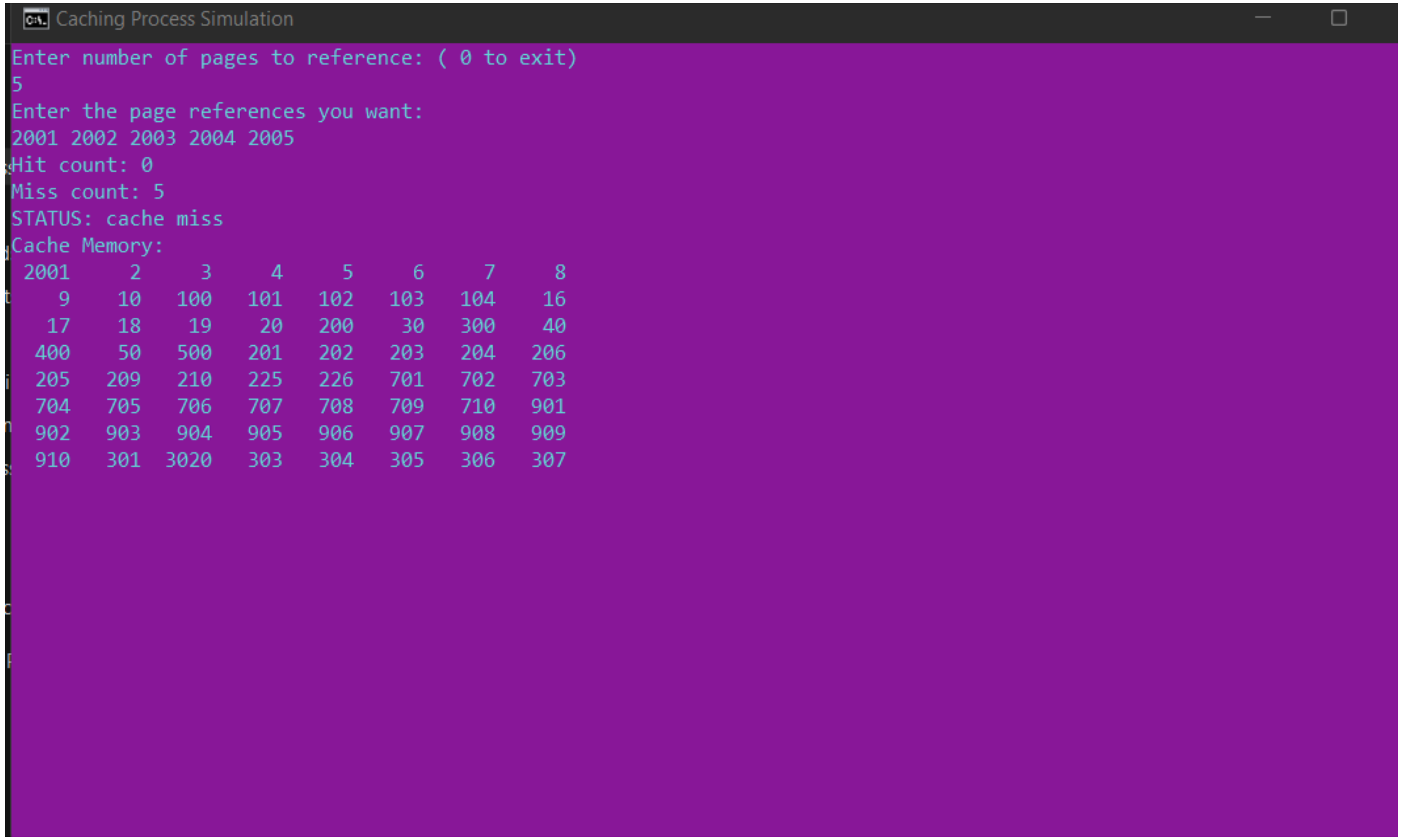
What is Optimal algorithm?

- Start
- Frame[1:n] // an array referring to page stored in each frame.
- For each page in cache memory traverse the page reference array and find the location when it will be needed next.
- Take the page which will wait longest for next referencing.
- Replace the page in cache memory with the new page.
- End

- **Advantage:**
  1. Minimum number of page faults.
  2. Less complex and easy to implement.

- **Disadvantage:**
  1. Program needs to know future page references in advance, which is not the case generally.

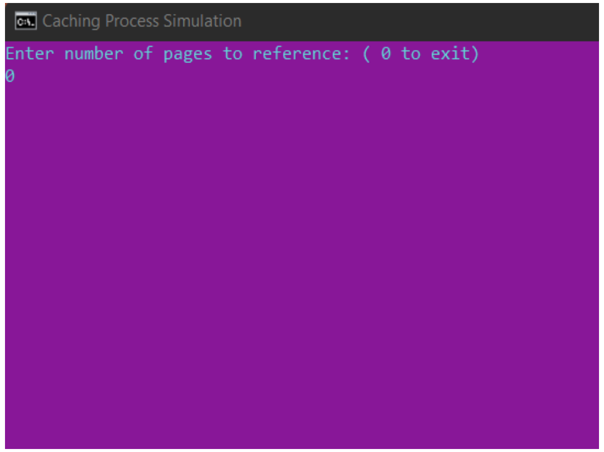
Output:



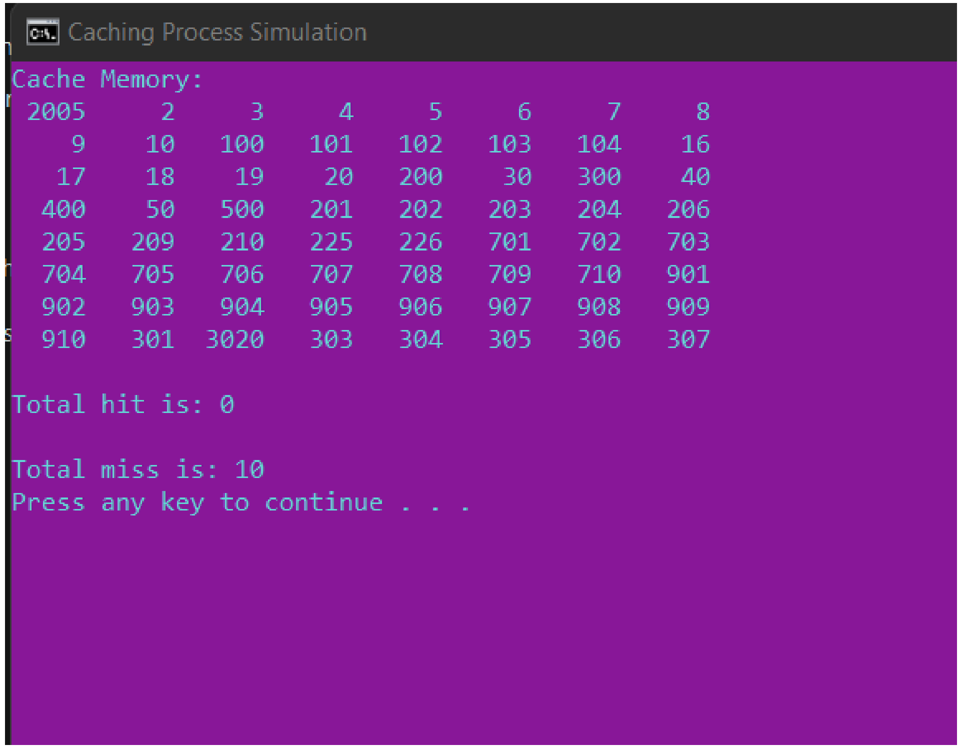
```
Enter number of pages to reference: ( 0 to exit)
5
Enter the page references you want:
2001 2002 2003 2004 2005
Hit count: 0
Miss count: 5
STATUS: cache miss
Cache Memory:
 2001     2     3     4     5     6     7     8
    9    10   100   101   102   103   104    16
   17    18    19    20   200    30   300    40
  400    50   500   201   202   203   204   206
  205   209   210   225   226   701   702   703
  704   705   706   707   708   709   710   901
  902   903   904   905   906   907   908   909
  910   301  3020   303   304   305   306   307
```

For normal page reference

Output:



```
Enter number of pages to reference: ( 0 to exit)
0
```



```
Cache Memory:
 2005     2     3     4     5     6     7     8
    9    10   100   101   102   103   104    16
   17    18    19    20   200    30   300    40
  400    50   500   201   202   203   204   206
  205   209   210   225   226   701   702   703
  704   705   706   707   708   709   710   901
  902   903   904   905   906   907   908   909
  910   301  3020   303   304   305   306   307

Total hit is: 0

Total miss is: 10
Press any key to continue . . .
```

When program terminates

Project Team:

- Yash Kumar , 20JE1110
- Asharam Meena, 20JE0202
- Aman Kumar , 20JE0106
- Amish Kumar, 20JE0115

Thank You