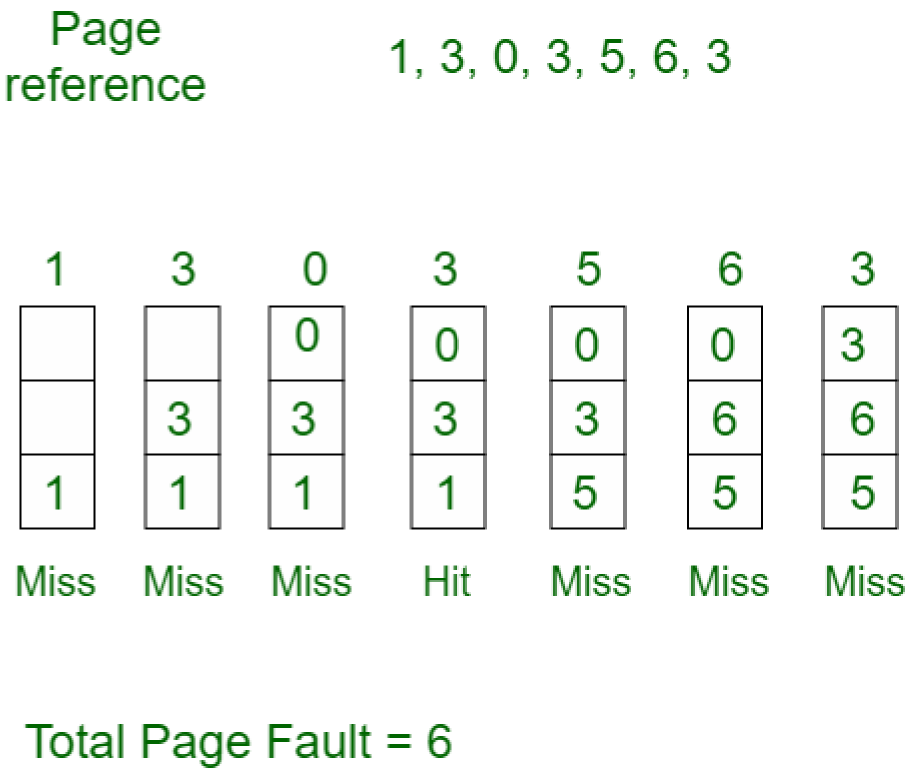
Page Replacement Algorithms in Operating Systems
In an operating system that uses paging for memory management, a page replacement algorithm is needed to decide which page needs to be replaced when new page comes in.

Page Fault – A page fault happens when a running program accesses a memory page that is mapped into the virtual address space, but not loaded in physical memory.

Since actual physical memory is much smaller than virtual memory, page faults happen. In case of page fault, Operating System might have to replace one of the existing pages with the newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce the number of page faults.

Page Replacement Algorithms :
1. First In First Out (FIFO) –
This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.
Example-1 Consider page reference string 1, 3, 0, 3, 5, 6 with 3 page frames.Find number of page faults.



Initially all slots are empty, so when 1, 3, 0 came they are allocated to the empty slots —> **3 Page Faults.**
when 3 comes, it is already in memory so —> **0 Page Faults.**
Then 5 comes, it is not available in memory so it replaces the oldest page slot i.e 1. —>**1 Page Fault.**

6 comes, it is also not available in memory so it replaces the oldest page slot i.e 3 —>**1 Page Fault.**

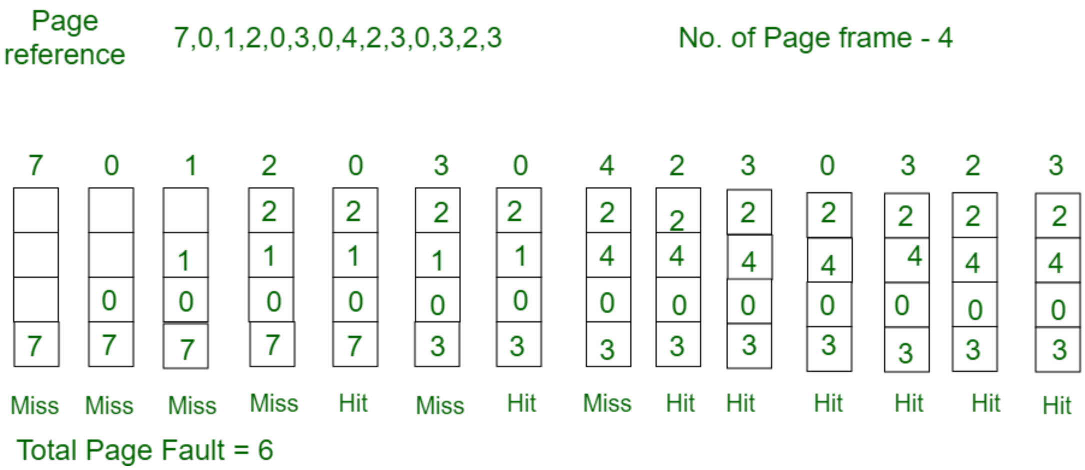
Finally when 3 come it is not available so it replaces 0 **1 page fault**

Belady's anomaly – Belady's anomaly proves that it is possible to have more page faults when increasing the number of page frames while using the First in First Out (FIFO) page replacement algorithm. For example, if we consider reference string 3, 2, 1, 0, 3, 2, 4, 3, 2, 1, 0, 4 and 3 slots, we get 9 total page faults, but if we increase slots to 4, we get 10 page faults.

2. Optimal Page replacement –

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future.

Example-2: Consider the page references 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, with 4 page frame. Find number of page fault.



Initially all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**

0 is already there so —> **0 Page fault.**

when 3 came it will take the place of 7 because it is not used for the longest duration of time in the future.—>**1 Page fault.**

0 is already there so —> **0 Page fault.**.

4 will takes place of 1 —> **1 Page Fault.**

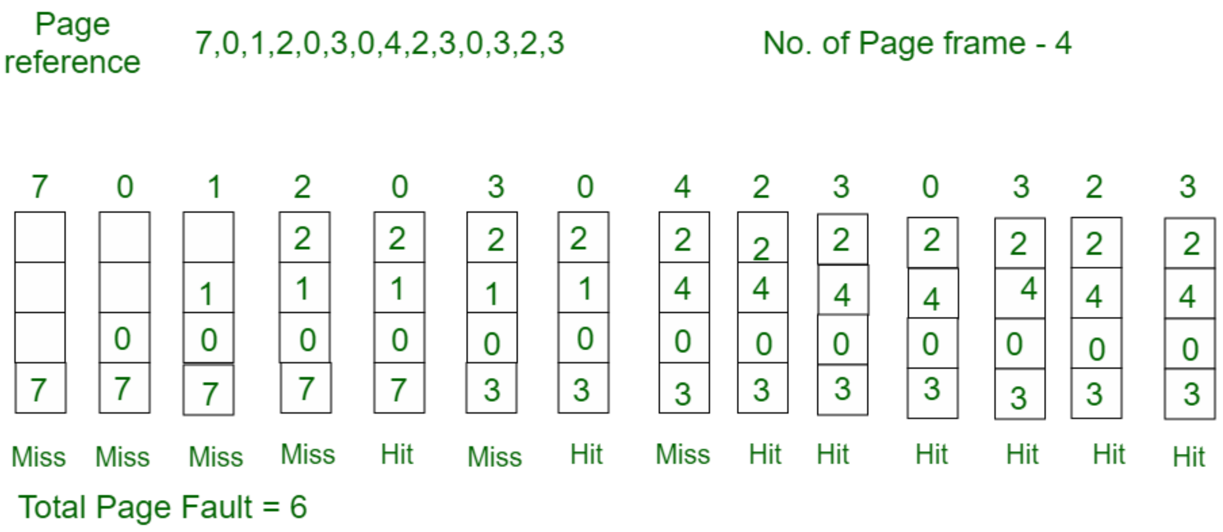
Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

Optimal page replacement is perfect, but not possible in practice as the operating system cannot know future requests. The use of Optimal Page replacement is to set up a benchmark so that other replacement algorithms can be analyzed against it.

3. Least Recently Used –

In this algorithm page will be replaced which is least recently used.

Example-3 Consider the page reference string 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2 with 4 page frames.Find number of page faults.



Page reference 7,0,1,2,0,3,0,4,2,3,0,3,2,3 No. of Page frame - 4

7	0	1	2	0	3	0	4	2	3	0	3	2	3
			2	2	2	2	2	2	2	2	2	2	2
		1	1	1	1	1	4	4	4	4	4	4	4
	0	0	0	0	0	0	0	0	0	0	0	0	0
7	7	7	7	7	3	3	3	3	3	3	3	3	3
Miss	Miss	Miss	Miss	Hit	Miss	Hit	Miss	Hit	Hit	Hit	Hit	Hit	Hit

Total Page Fault = 6

Here LRU has same number of page fault as optimal but it may differ according to question.

Initially all slots are empty, so when 7 0 1 2 are allocated to the empty slots —> **4 Page faults**
0 is already their so —> **0 Page fault.**
when 3 came it will take the place of 7 because it is least recently used —>**1 Page fault**
0 is already in memory so —> **0 Page fault**.
4 will takes place of 1 —> **1 Page Fault**
Now for the further page reference string —> **0 Page fault** because they are already available in the memory.

Page Replacement-

Page replacement is a process of swapping out an existing page from the frame of a main memory and replacing it with the required page.

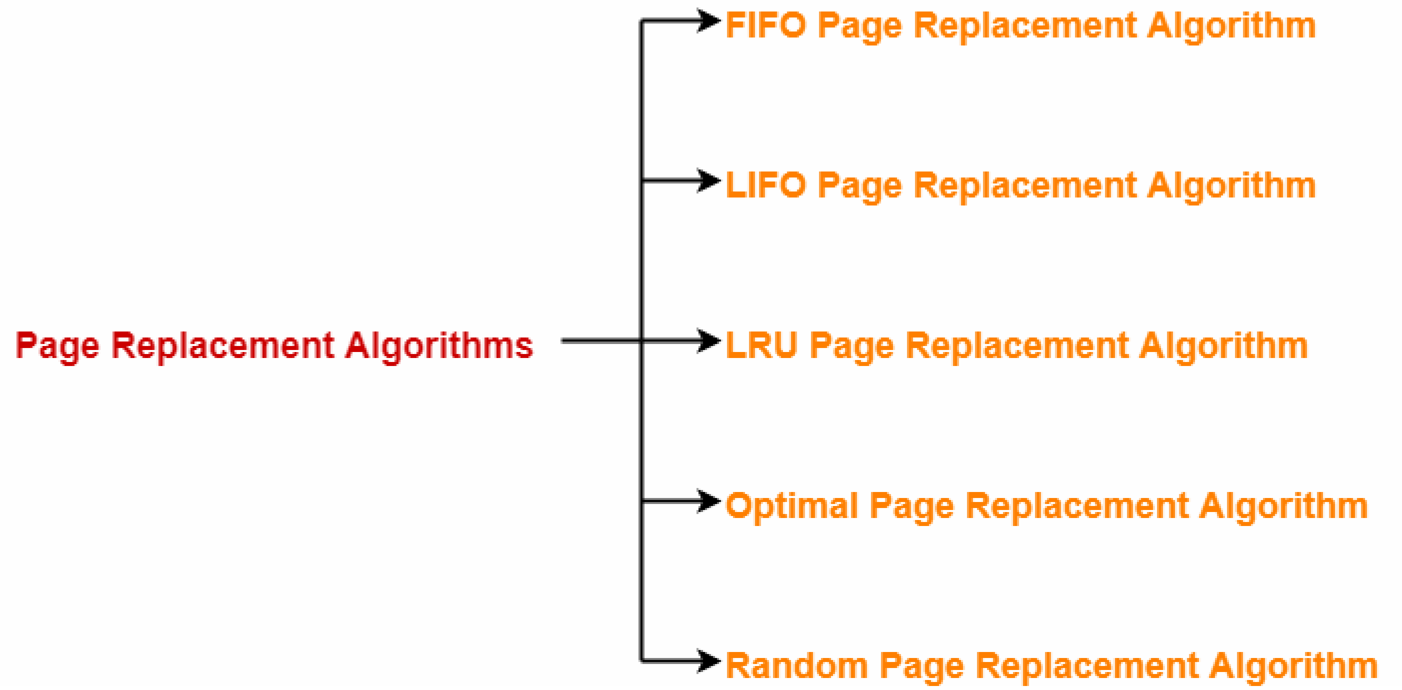
Page replacement is required when-

- All the frames of main memory are already occupied.
- Thus, a page has to be replaced to create a room for the required page.

Page Replacement Algorithms-

Page replacement algorithms help to decide which page must be swapped out from the main memory to create a room for the incoming page.

Various page replacement algorithms are-


Page Replacement Algorithms
FIFO Page Replacement Algorithm
LIFO Page Replacement Algorithm
LRU Page Replacement Algorithm
Optimal Page Replacement Algorithm
Random Page Replacement Algorithm

1. FIFO Page Replacement Algorithm
2. LIFO Page Replacement Algorithm
3. LRU Page Replacement Algorithm
4. Optimal Page Replacement Algorithm
5. Random Page Replacement Algorithm

A good page replacement algorithm is one that minimizes the number of page faults.

FIFO Page Replacement Algorithm-

- As the name suggests, this algorithm works on the principle of "**First in First out**".
- It replaces the oldest page that has been present in the main memory for the longest time.
- It is implemented by keeping track of all the pages in a queue.

LIFO Page Replacement Algorithm-

- As the name suggests, this algorithm works on the principle of "**Last in First out**".
- It replaces the newest page that arrived at last in the main memory.
- It is implemented by keeping track of all the pages in a stack.

NOTE

Only frame is used for page replacement during entire procedure after all the frames get occupied.

LRU Page Replacement Algorithm-

- As the name suggests, this algorithm works on the principle of "**Least Recently Used**".
- It replaces the page that has not been referred by the CPU for the longest time.

Optimal Page Replacement Algorithm-

- This algorithm replaces the page that will not be referred by the CPU in future for the longest time.

- It is practically impossible to implement this algorithm.
- This is because the pages that will not be used in future for the longest time can not be predicted.
- However, it is the best known algorithm and gives the least number of page faults.
- Hence, it is used as a performance measure criterion for other algorithms.

Random Page Replacement Algorithm-

- As the name suggests, this algorithm randomly replaces any page.
- So, this algorithm may behave like any other algorithm like FIFO, LIFO, LRU, Optimal etc.

PRACTICE PROBLEMS BASED ON PAGE REPLACEMENT ALGORITHMS-

Problem-01:

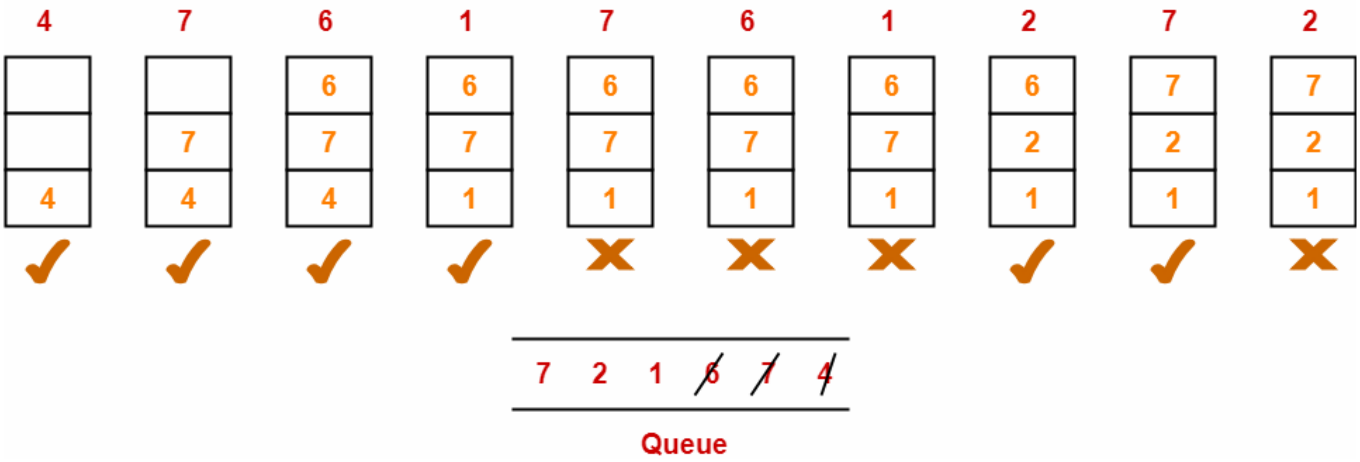
A system uses 3 page frames for storing process pages in main memory. It uses the First in First out (FIFO) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4 , 7, 6, 1, 7, 6, 1, 2, 7, 2

Also calculate the hit ratio and miss ratio.

Solution-

Total number of references = 10



From here,

Total number of page faults occurred = 6

Calculating Hit ratio-

Total number of page hits

= Total number of references – Total number of page misses or page faults

= 10 – 6

= 4

Thus, Hit ratio

= Total number of page hits / Total number of references

= 4 / 10

= 0.4 or 40%

Calculating Miss ratio-

Total number of page misses or page faults = 6

Thus, Miss ratio

= Total number of page misses / Total number of references

= 6 / 10

= 0.6 or 60%

Alternatively,

Miss ratio

= 1 – Hit ratio

= 1 – 0.4

= 0.6 or 60%

Problem-02:

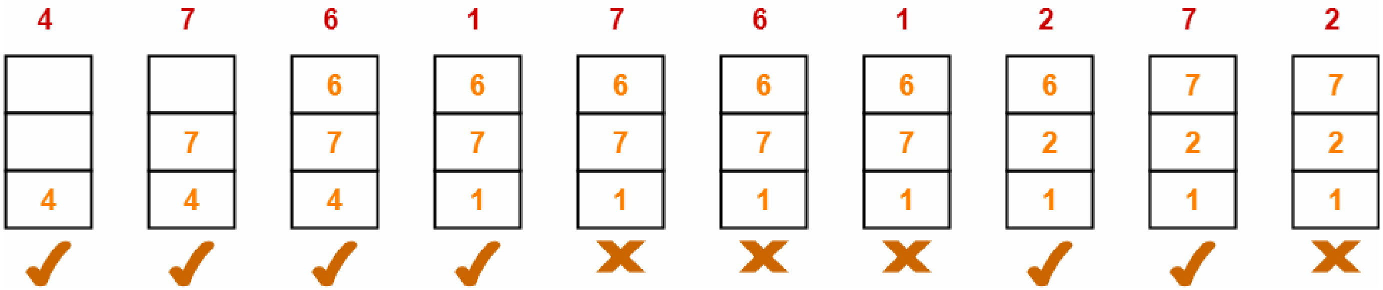
A system uses 3 page frames for storing process pages in main memory. It uses the Least Recently Used (LRU) page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4 , 7, 6, 1, 7, 6, 1, 2, 7, 2

Also calculate the hit ratio and miss ratio.

Solution-

Total number of references = 10



From here,

Total number of page faults occurred = 6

In the similar manner as above-

- Hit ratio = 0.4 or 40%
- Miss ratio = 0.6 or 60%

Problem-03:

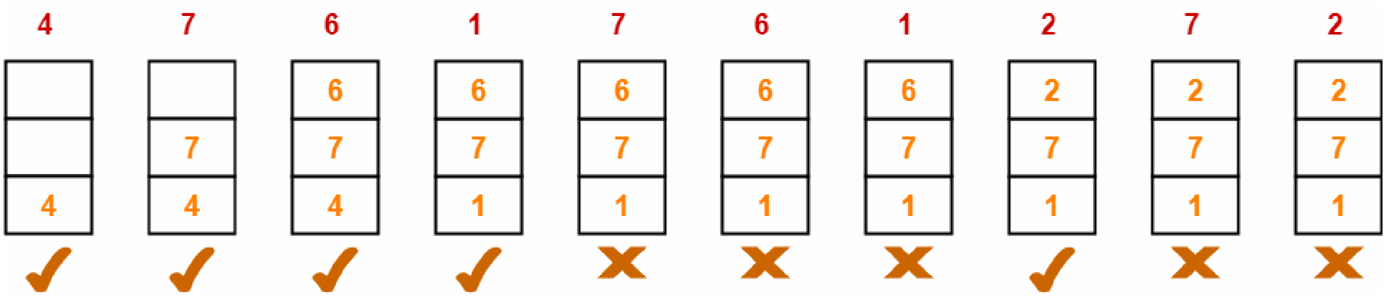
A system uses 3 page frames for storing process pages in main memory. It uses the Optimal page replacement policy. Assume that all the page frames are initially empty. What is the total number of page faults that will occur while processing the page reference string given below-

4 , 7, 6, 1, 7, 6, 1, 2, 7, 2

Also calculate the hit ratio and miss ratio.

Solution-

Total number of references = 10



4	7	6	1	7	6	1	2	7	2
		6	6	6	6	6	2	2	2
	7	7	7	7	7	7	7	7	7
4	4	4	1	1	1	1	1	1	1
✓	✓	✓	✓	✗	✗	✗	✓	✗	✗

From here,

Total number of page faults occurred = 5

In the similar manner as above-

- Hit ratio = 0.5 or 50%
- Miss ratio = 0.5 or 50%

What is Page Replacement Algorithm and why it is needed?

One of the techniques which are used for memory management is paging. In paging, processes are divided into pages and main memory is divided into frames. Pages of a process are loaded into frames of main memory when required.

Page Replacement Algorithm is used when a page fault occurs. Page Fault means the page referenced by the CPU is not present in the main memory.

When the CPU generates the reference of a page, if there is any vacant frame available in the main memory then the page is loaded in that vacant frame. In another case, if there is no vacant frame available in the main memory, it is required to replace one of the pages in the main memory with the page referenced by the CPU.

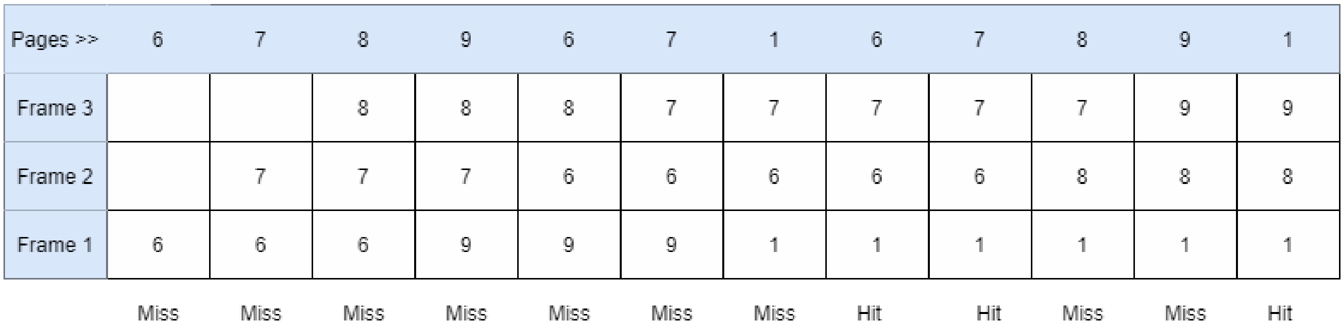
Page Replacement Algorithm is used to decide which page will be replaced to allocate memory to the current referenced page.

Different Page Replacement Algorithms suggest different ways to decide which page is to be replaced. The main objective of these algorithms is to reduce the number of page faults.

Page Replacement Algorithms

First In First Out (FIFO) -This algorithm is similar to the operations of the queue. All the pages are stored in the queue in the order they are allocated frames in the main memory. The one which is allocated first stays in the front of the queue. The one which is allocated the memory first is replaced first. The one which is at the front of the queue is removed at the time of replacement.

Example: Consider the Pages referenced by the CPU in the order are 6, 7, 8, 9, 6, 7, 1, 6, 7, 8, 9, 1



Pages >>	6	7	8	9	6	7	1	6	7	8	9	1
Frame 3			8	8	8	7	7	7	7	7	9	9
Frame 2		7	7	7	6	6	6	6	6	8	8	8
Frame 1	6	6	6	9	9	9	1	1	1	1	1	1
	Miss	Miss	Miss	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Miss	Hit

- As in the above figure shown, Let there are 3 frames in the memory.

- 6, 7, 8 are allocated to the vacant slots as they are not in memory.

- When 9 comes page fault occurs, it replaces 6 which is the oldest in memory or front element of the queue.

- Then 6 comes (Page Fault), it replaces 7 which is the oldest page in memory now.

- Similarly, 7 replaces 8, 1 replaces 9.

- Then 6 comes which is already in memory (Page Hit).

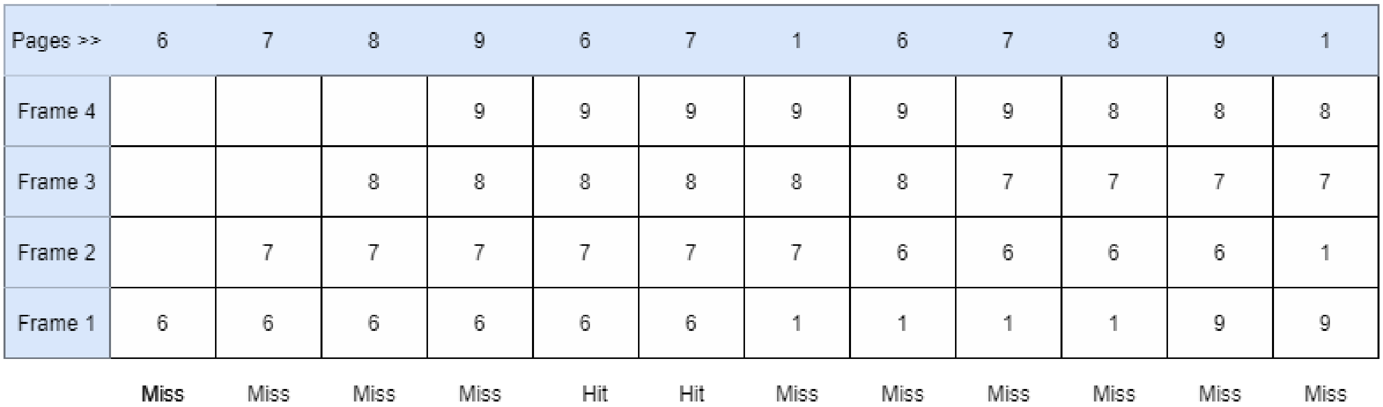
- Then 7 comes (Page Hit).

- Then 8 replaces 6, 9 replaces 7. Then 1 comes (Page Hit).

Number of Page Faults = 9

- While using the First In First Out algorithm, the number of page faults increases by increasing the number of frames. This phenomenon is called **Belady's Anomaly**.

- Let's take the same above order of pages with 4 frames.

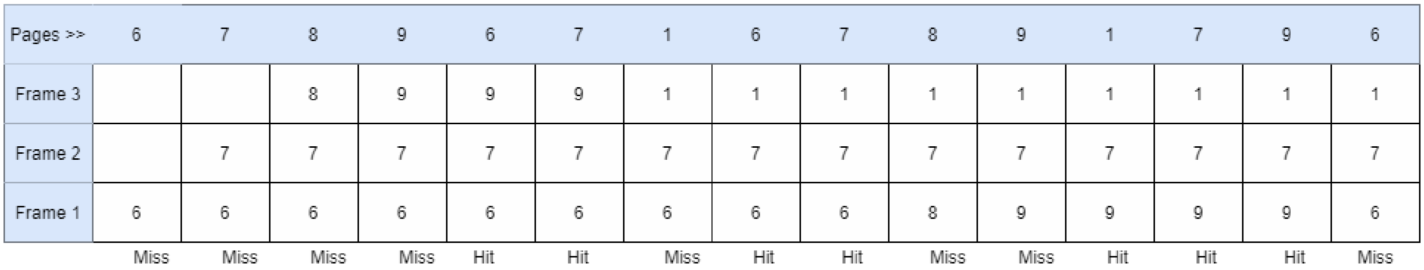


Pages >>	6	7	8	9	6	7	1	6	7	8	9	1
Frame 4				9	9	9	9	9	9	8	8	8
Frame 3			8	8	8	8	8	8	7	7	7	7
Frame 2		7	7	7	7	7	7	6	6	6	6	1
Frame 1	6	6	6	6	6	6	1	1	1	1	9	9
	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Miss	Miss	Miss	Miss	Miss

- In the above picture shown, it can be seen that the number of page faults is 10.
- There were 9 page faults with 3 frames and 10 page faults with 4 frames.
- The number of page faults increased by increasing the number of frames.

Optimal Page Replacement - In this algorithm, the page which would be used after the longest interval is replaced. In other words, the page which is farthest to come in the upcoming sequence is replaced.

Example: Consider the Pages referenced by the CPU in the order are 6, 7, 8, 9, 6, 7, 1, 6, 7, 8, 9, 1, 7, 9, 6



Pages >>	6	7	8	9	6	7	1	6	7	8	9	1	7	9	6
Frame 3			8	9	9	9	1	1	1	1	1	1	1	1	1
Frame 2		7	7	7	7	7	7	7	7	7	7	7	7	7	7
Frame 1	6	6	6	6	6	6	6	6	6	8	9	9	9	9	6
	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Hit	Hit	Miss	Miss	Hit	Hit	Hit	Miss

- First, all the frames are empty. 6, 7, 8 are allocated to the frames (Page Fault).
- Now, 9 comes and replaces 8 as it is the farthest in the upcoming sequence. 6 and 7 would come earlier than that so not replaced.
- Then, 6 comes which is already present (Page Hit).
- Then 7 comes (Page Hit).
- Then 1 replaces 9 similarly (Page Fault).
- Then 6 comes (Page Hit), 7 comes (Page Hit).
- Then 8 replaces 6 (Page Fault) and 9 replaces 8 (Page Fault).

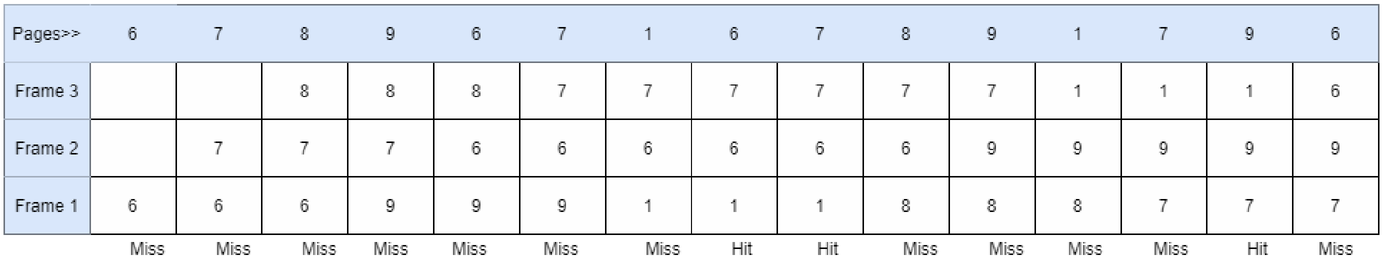
- Then 1, 7, 9 come respectively which are already present in the memory.
- Then 6 replaces 9 (Page Fault), it can also replace 7 and 1 as no other page is present in the upcoming sequence.

The number of Page Faults = 8

- This is the most optimal algorithm but is impractical because it is impossible to predict the upcoming page references.

Least Recently Used - This algorithm works on previous data. The page which is used the earliest is replaced or which appears the earliest in the sequence is replaced.

Example: Consider the Pages referenced by the CPU in the order are 6, 7, 8, 9, 6, 7, 1, 6, 7, 8, 9, 1, 7, 9, 6



Pages>>	6	7	8	9	6	7	1	6	7	8	9	1	7	9	6
Frame 3			8	8	8	7	7	7	7	7	7	1	1	1	6
Frame 2		7	7	7	6	6	6	6	6	6	9	9	9	9	9
Frame 1	6	6	6	9	9	9	1	1	1	8	8	8	7	7	7
	Miss	Miss	Miss	Miss	Miss	Miss	Miss	Hit	Hit	Miss	Miss	Miss	Miss	Hit	Miss

- First, all the frames are empty. 6, 7, 8 are allocated to the frames (Page Fault).
- Now, 9 comes and replaces 6 which is used the earliest (Page Fault).
- Then, 6 replaces 7, 7 replaces 8, 1 replaces 9 (Page Fault).
- Then 6 comes which is already present (Page Hit).
- Then 7 comes (Page Hit).
- Then 8 replaces 1, 9 replaces 6, 1 replaces 7, and 7 replaces 8 (Page Fault).
- Then 9 comes (Page Hit).
- Then 6 replaces 1 (Page Fault).

The number of Page Faults = 12

A computer system has a limited amount of memory. Adding more memory physically is very costly. Therefore most modern computers use a combination of both hardware and software to allow the computer to address more memory than the amount physically present on the system. This extra memory is actually called **Virtual Memory.**

Virtual Memory is a storage allocation scheme used by the Memory Management Unit(MMU) to compensate for the shortage of physical memory by transferring data from RAM to disk storage. It addresses secondary memory as though it is a part of the main memory. Virtual Memory makes the memory appear larger than actually present which helps in the execution of programs that are larger than the physical memory.

Virtual Memory can be implemented using two methods :

- Paging
- Segmentation

In this blog, we will learn about the paging part.

Paging

Paging is a process of reading data from, and writing data to, the secondary storage. It is a memory management scheme that is used to retrieve processes from the secondary memory in the form of pages and store them in the primary memory. The main objective of paging is to divide each process in the form of pages of fixed size. These pages are stored in the main memory in frames. Pages of a process are only brought from the secondary memory to the main memory when they are needed.

When an executing process refers to a page, it is first searched in the main memory. If it is not present in the main memory, a page fault occurs.

*** Page Fault is the condition in which a running process refers to a page that is not loaded in the main memory.*

In such a case, the OS has to bring the page from the secondary storage into the main memory. This may cause some pages in the main memory to be replaced due to limited storage. A Page Replacement Algorithm is required to decide which page needs to be replaced.

Page Replacement Algorithm

Page Replacement Algorithm decides which page to remove, also called swap out when a new page needs to be loaded into the main memory. Page Replacement happens when a requested page is not present in the main memory and the available space is not sufficient for allocation to the requested page.

When the page that was selected for replacement was paged out, and referenced again, it has to read in from disk, and this requires for I/O completion. This process determines the quality of the page replacement algorithm: the lesser the time waiting for page-ins, the better is the algorithm.

A page replacement algorithm tries to select which pages should be replaced so as to minimize the total number of page misses. There are many different page replacement algorithms. These algorithms are evaluated by running them on a particular string of memory reference and computing the number of page faults. The fewer is the page faults the better is the algorithm for that situation.

*** If a process requests for page and that page is found in the main memory then it is called* ***page hit****, otherwise* ***page miss*** *or* ***page fault****.*

Some Page Replacement Algorithms :

- First In First Out (FIFO)
- Least Recently Used (LRU)
- Optimal Page Replacement

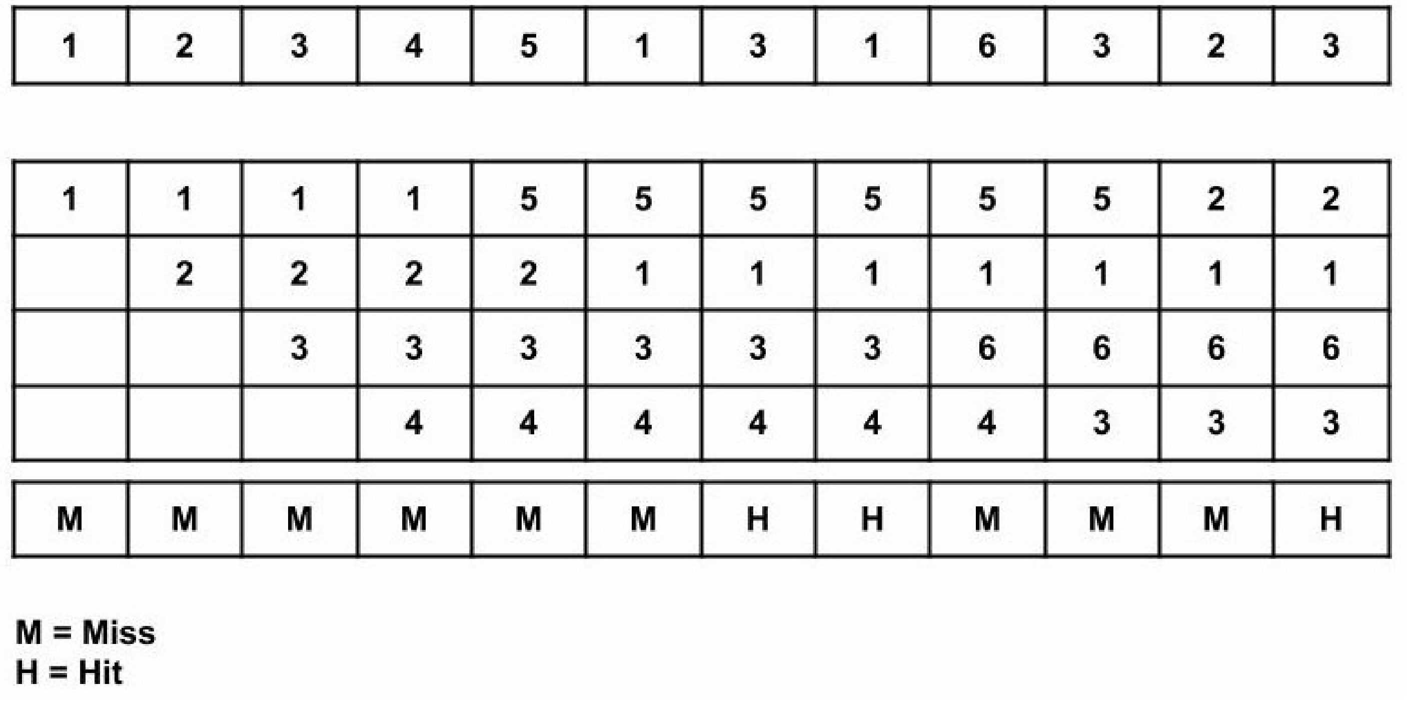
First In First Out (FIFO)

This is the simplest page replacement algorithm. In this algorithm, the OS maintains a queue that keeps track of all the pages in memory, with the oldest page at the front and the most recent page at the back.

When there is a need for page replacement, the FIFO algorithm, swaps out the page at the front of the queue, that is the page which has been in the memory for the longest time.

For Example:

Consider the page reference string of size 12: 1, 2, 3, 4, 5, 1, 3, 1, 6, 3, 2, 3 with frame size 4(i.e. maximum 4 pages in a frame).



1	2	3	4	5	1	3	1	6	3	2	3

1	1	1	1	5	5	5	5	5	5	2	2
	2	2	2	2	1	1	1	1	1	1	1
		3	3	3	3	3	3	6	6	6	6
			4	4	4	4	4	4	3	3	3

M	M	M	M	M	M	H	H	M	M	M	H

M = Miss
H = Hit

Total Page Fault = 9

Initially, all 4 slots are empty, so when 1, 2, 3, 4 came they are allocated to the empty slots in order of their arrival. This is page fault as 1, 2, 3, 4 are not available in memory.

When 5 comes, it is not available in memory so page fault occurs and it replaces the oldest page in memory, i.e., 1.

When 1 comes, it is not available in memory so page fault occurs and it replaces the oldest page in memory, i.e., 2.

When 3,1 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

When 6 comes, it is not available in memory so page fault occurs and it replaces the oldest page in memory, i.e., 3.

When 3 comes, it is not available in memory so page fault occurs and it replaces the oldest page in memory, i.e., 4.

When 2 comes, it is not available in memory so page fault occurs and it replaces the oldest page in memory, i.e., 5.

When 3 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

Page Fault ratio = 9/12 i.e. total miss/total possible cases

Advantages

- Simple and easy to implement.
- Low overhead.

Disadvantages

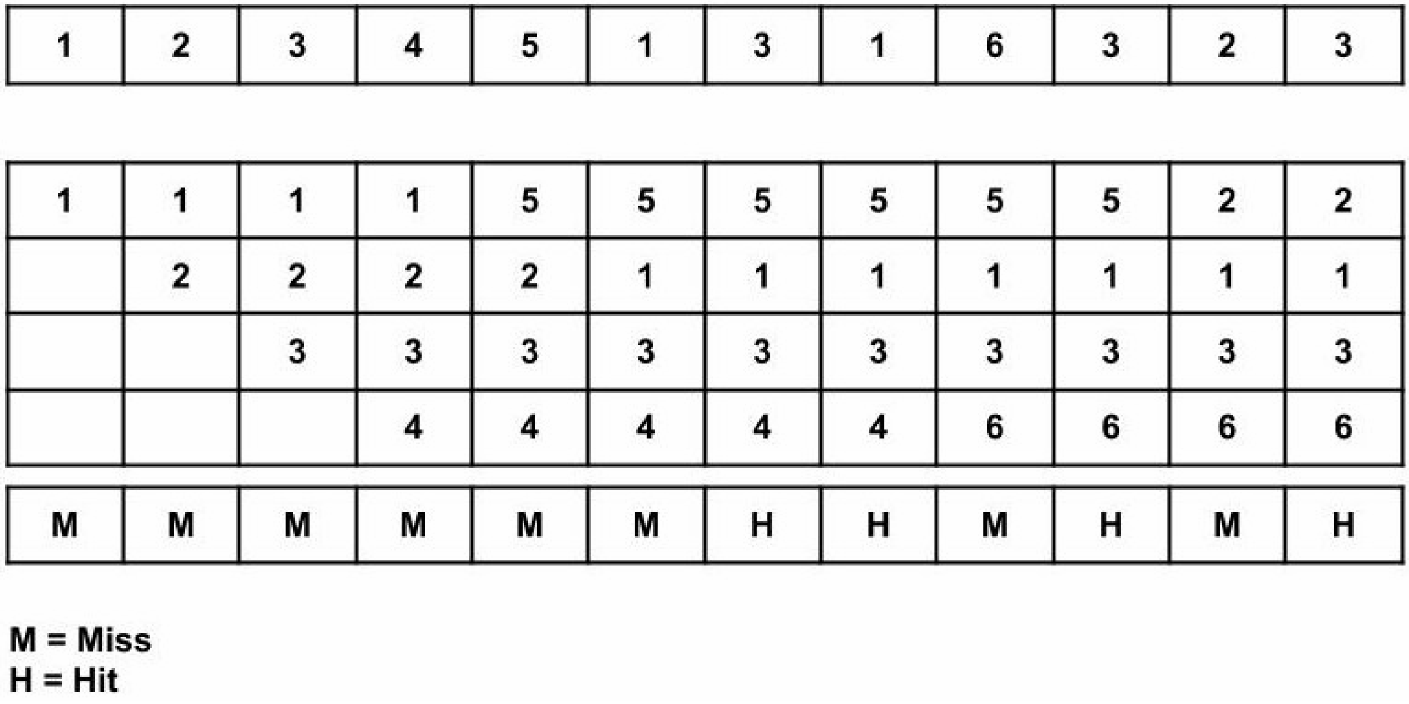
- Poor performance.
- Doesn’t consider the frequency of use or last used time, simply replaces the oldest page.
- Suffers from Belady’s Anomaly(i.e. more page faults when we increase the number of page frames).

Least Recently Used (LRU)

Least Recently Used page replacement algorithm keeps track of page usage over a short period of time. It works on the idea that the pages that have been most heavily used in the past are most likely to be used heavily in the future too.

In LRU, whenever page replacement happens, the page which has not been used for the longest amount of time is replaced.

For Example



1	2	3	4	5	1	3	1	6	3	2	3

1	1	1	1	5	5	5	5	5	5	2	2
	2	2	2	2	1	1	1	1	1	1	1
		3	3	3	3	3	3	3	3	3	3
			4	4	4	4	4	6	6	6	6

M	M	M	M	M	M	H	H	M	H	M	H

M = Miss
H = Hit

Total Page Fault = 8

Initially, all 4 slots are empty, so when 1, 2, 3, 4 came they are allocated to the empty slots in order of their arrival. This is page fault as 1, 2, 3, 4 are not available in memory.

When 5 comes, it is not available in memory so page fault occurs and it replaces 1 which is the least recently used page.

When 1 comes, it is not available in memory so page fault occurs and it replaces 2.

When 3,1 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

When 6 comes, it is not available in memory so page fault occurs and it replaces 4.

When 3 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

When 2 comes, it is not available in memory so page fault occurs and it replaces 5.

When 3 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

Page Fault ratio = 8/12

Advantages

- Efficient.
- Doesn't suffer from Belady's Anomaly.

Disadvantages

- Complex Implementation.
- Expensive.
- Requires hardware support.

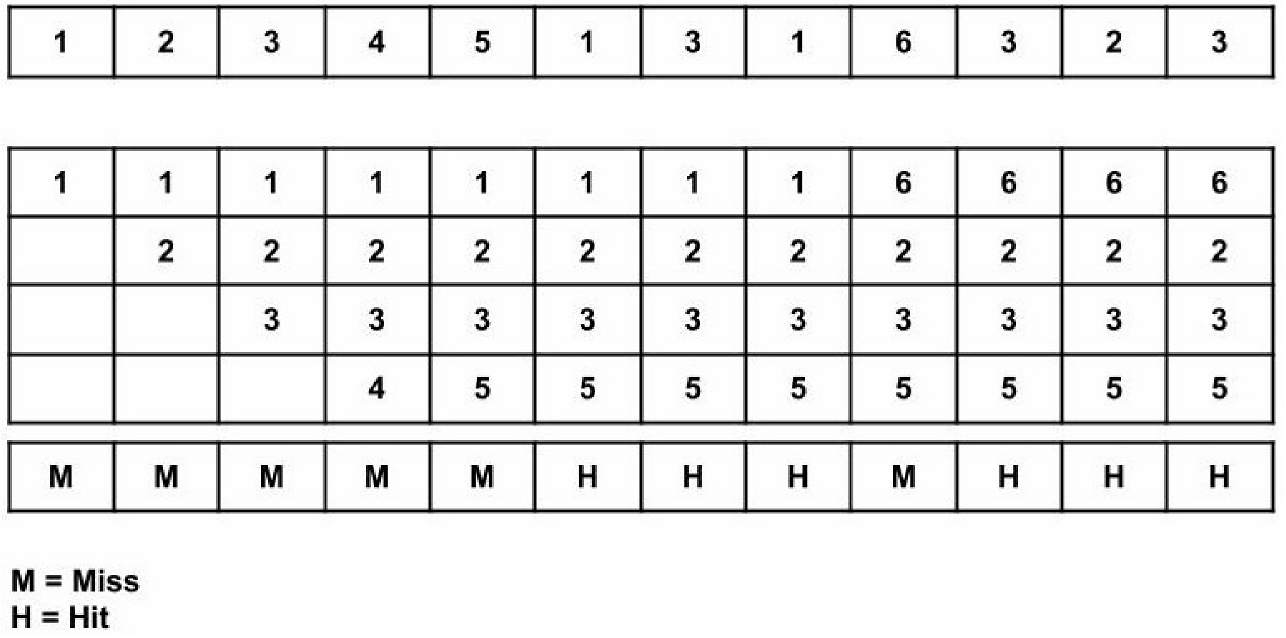
Optimal Page Replacement

Optimal Page Replacement algorithm is the best page replacement algorithm as it gives the least number of page faults. It is also known as OPT, clairvoyant replacement algorithm, or Belady's optimal page replacement policy.

In this algorithm, pages are replaced which would not be used for the longest duration of time in the future, i.e., the pages in the memory which are going to be referred farthest in the future are replaced.

This algorithm was introduced long back and is difficult to implement because it requires future knowledge of the program behaviour. However, it is possible to implement optimal page replacement on the second run by using the page reference information collected on the first run.

For Example



1	2	3	4	5	1	3	1	6	3	2	3

1	1	1	1	1	1	1	1	6	6	6	6
	2	2	2	2	2	2	2	2	2	2	2
		3	3	3	3	3	3	3	3	3	3
			4	5	5	5	5	5	5	5	5

M	M	M	M	M	H	H	H	M	H	H	H

M = Miss
H = Hit

Total Page Fault = 6

Initially, all 4 slots are empty, so when 1, 2, 3, 4 came they are allocated to the empty slots in order of their arrival. This is page fault as 1, 2, 3, 4 are not available in memory.

When 5 comes, it is not available in memory so page fault occurs and it replaces 4 which is going to be used farthest in the future among 1, 2, 3, 4.

When 1,3,1 comes, they are available in the memory, i.e., Page Hit, so no replacement occurs.

When 6 comes, it is not available in memory so page fault occurs and it replaces 1.

When 3, 2, 3 comes, it is available in the memory, i.e., Page Hit, so no replacement occurs.

Page Fault ratio = 6/12

Advantages

- Easy to Implement.
- Simple data structures are used.
- Highly efficient.

Disadvantages

- Requires future knowledge of the program.

- Time-consuming.

So, these are some of the page replacement algorithms that are used in paging.