Assignment - 9

Aim : Implement paging technique

Theory :

The paging technique is a memory management scheme used by OS to manage memory allocation for processes. It allows a computer to allocate memory in fixed-size blocks called "pages," which are typically smaller than the total size of physical memory.

The paging technique involves dividing both the physical memory (RAM) and the logical memory (used by processes) into fixed-size blocks. These blocks in physical memory are called "frames," while those in logical memory are called "pages." The size of a page is the same as the size of a frame.

When a process requests memory, the operating system allocates memory for the process in terms of pages. The pages do not need to be contiguous in physical memory; they can be scattered across different frames. The operating system maintains a data structure called a page table to keep track of the mapping between logical pages and physical frames.

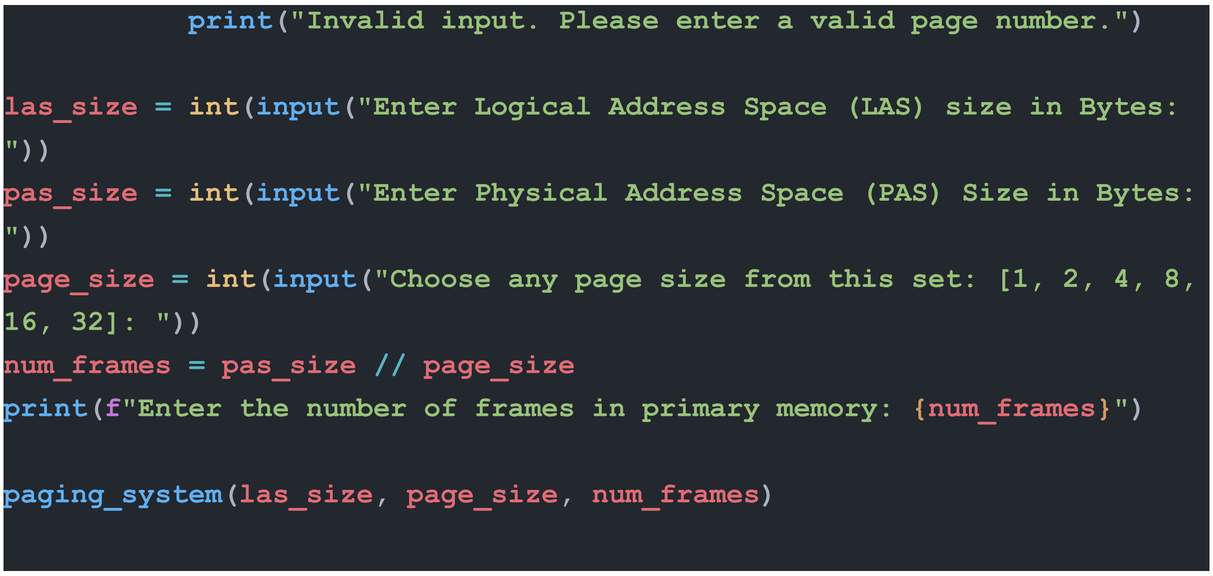
Paging provides several benefits, including:

1. **Efficient memory utilization:** Paging allows for more efficient memory usage because it allows processes to use memory space more flexibly. Processes do not need to be loaded into contiguous blocks of memory.
2. **Simplified memory management:** Paging simplifies memory management for the operating system since it does not need to manage contiguous blocks of memory for each process.
3. **Memory protection:** Paging provides a level of memory protection by enforcing access permissions on a per-page basis. This helps prevent processes from accessing memory that they are not authorized to access.

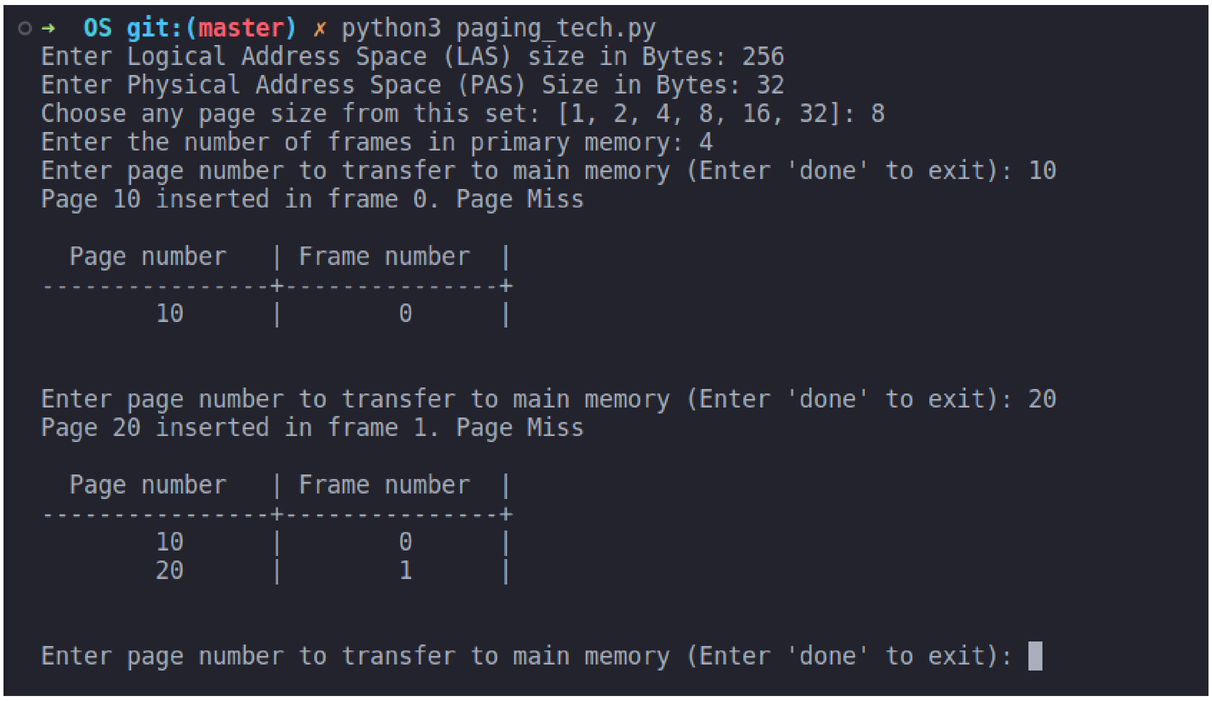
Overall, paging is a fundamental technique used in modern operating systems to efficiently manage memory allocation and provide memory protection for processes.

Code :

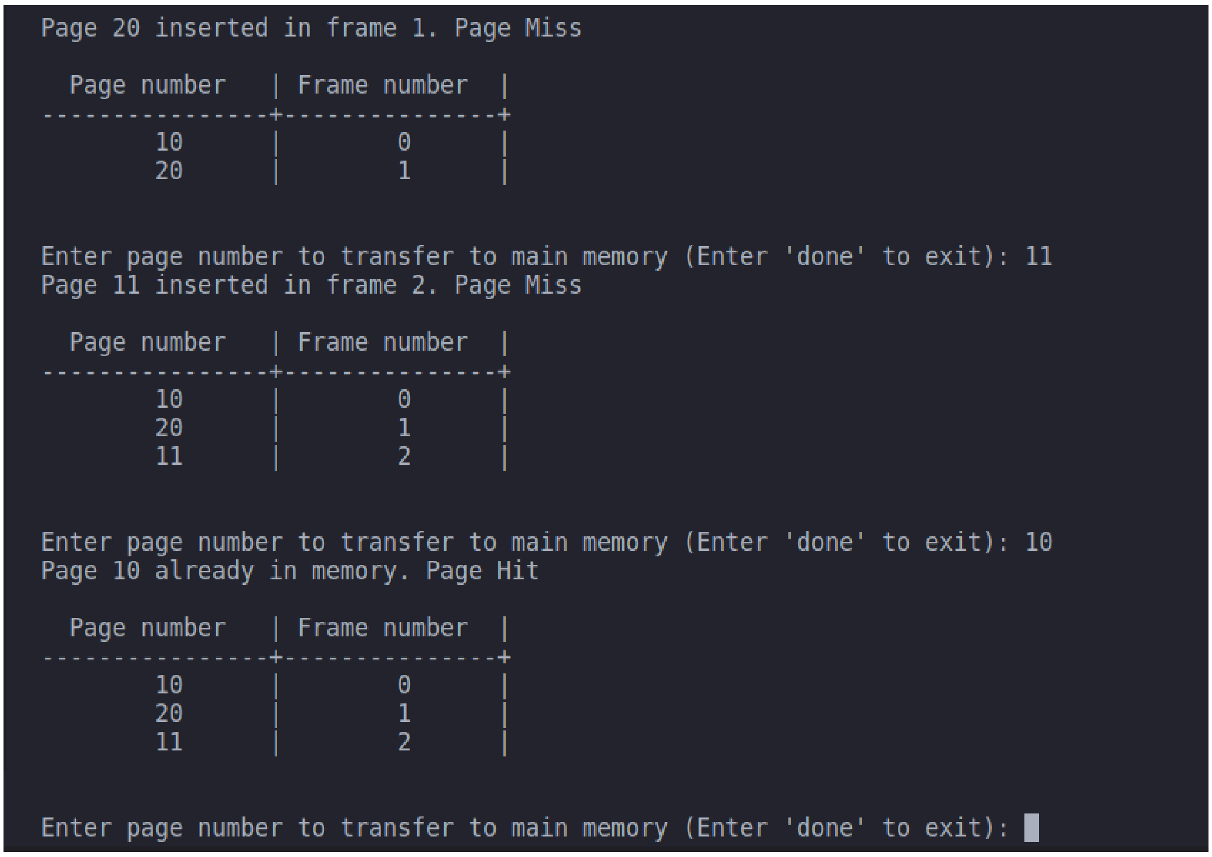
```
def fifo_replace(frames, page_number, page_table):
    if page_number in frames:
        return f"Page {page_number} already in memory. Page Hit"
    for i, frame in enumerate(frames):
        if frame is None:
            frames[i] = page_number
            page_table[page_number] = i
            return f"Page {page_number} inserted in frame {i}. Page
Miss"
    removed_page = frames.pop(0)
    frames.append(page_number)
    removed_frame = page_table.pop(removed_page)
    page_table[page_number] = removed_frame
    return f"Page {page_number} inserted in frame {removed_frame}. Page
{removed_page} replaced. Page Miss"

def paging_system(las_size, page_size, num_frames):
    num_pages = las_size // page_size
    frames = [None] * num_frames
    page_table = {}

    while True:
        page_input = input("Enter page number to transfer to main memory
(Enter 'done' to exit): ")
        if page_input.lower() == 'done':
            break
        try:
            page_number = int(page_input)
            if page_number < 0 or page_number >= num_pages:
                print("Invalid page number. Please enter a number
between 0 and", num_pages - 1)
                continue
            print(fifo_replace(frames, page_number, page_table))

            print("\n  Page number \t| Frame number\t|")
            print("----------------+---------------+")
            for page, frame in page_table.items():
                print(f"\t{page} \t|\t {frame} \t|")
            print("\n")

        except ValueError:
```



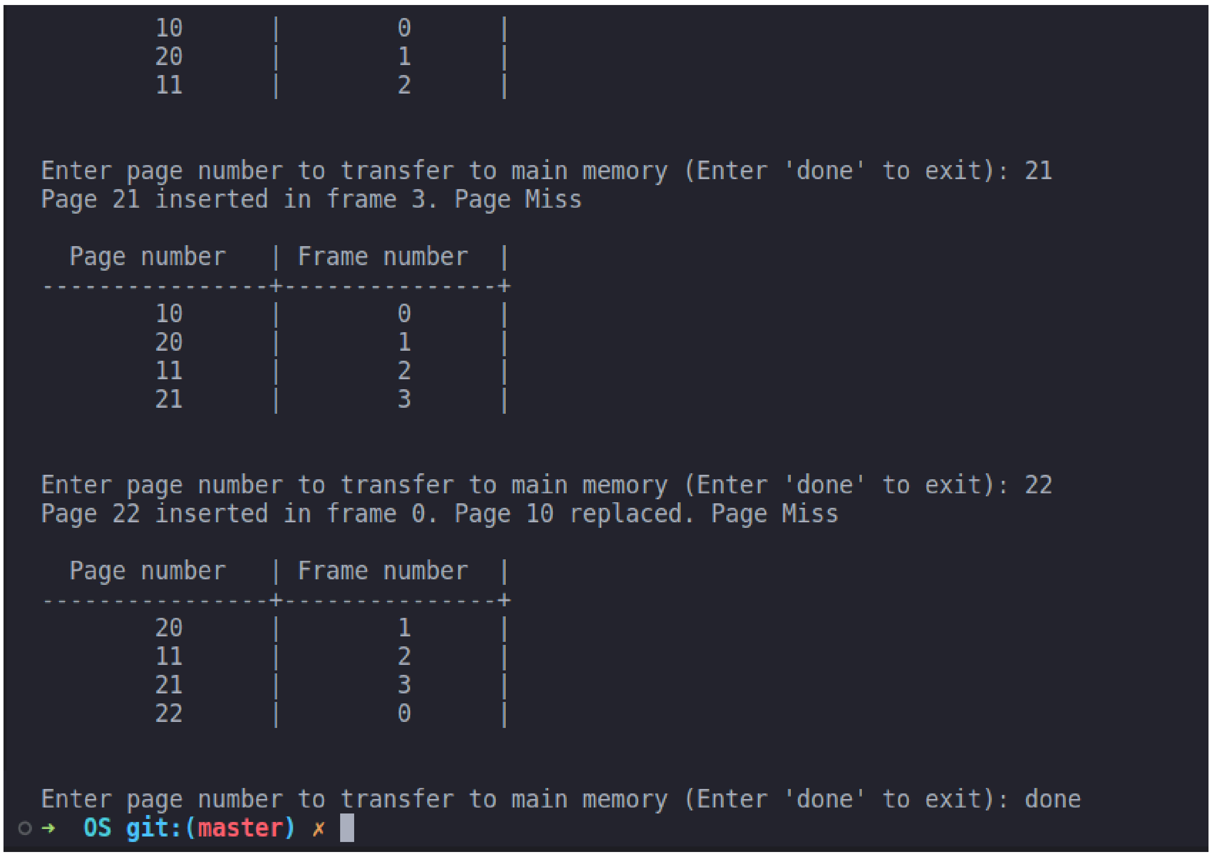
```python
            print("Invalid input. Please enter a valid page number.")

las_size = int(input("Enter Logical Address Space (LAS) size in Bytes: 
"))
pas_size = int(input("Enter Physical Address Space (PAS) Size in Bytes: 
"))
page_size = int(input("Choose any page size from this set: [1, 2, 4, 8, 
16, 32]: "))
num_frames = pas_size // page_size
print(f"Enter the number of frames in primary memory: {num_frames}")

paging_system(las_size, page_size, num_frames)
```



```
○ → OS git:(master) ✗ python3 paging_tech.py
  Enter Logical Address Space (LAS) size in Bytes: 256
  Enter Physical Address Space (PAS) Size in Bytes: 32
  Choose any page size from this set: [1, 2, 4, 8, 16, 32]: 8
  Enter the number of frames in primary memory: 4
  Enter page number to transfer to main memory (Enter 'done' to exit): 10
  Page 10 inserted in frame 0. Page Miss

    Page number   | Frame number  |
  ----------------+---------------+
          10      |        0      |


  Enter page number to transfer to main memory (Enter 'done' to exit): 20
  Page 20 inserted in frame 1. Page Miss

    Page number   | Frame number  |
  ----------------+---------------+
          10      |        0      |
          20      |        1      |


  Enter page number to transfer to main memory (Enter 'done' to exit): 
```



```
Page 20 inserted in frame 1. Page Miss

  Page number    | Frame number  |
----------------+--------------+
        10      |       0      |
        20      |       1      |


Enter page number to transfer to main memory (Enter 'done' to exit): 11
Page 11 inserted in frame 2. Page Miss

  Page number    | Frame number  |
----------------+--------------+
        10      |       0      |
        20      |       1      |
        11      |       2      |


Enter page number to transfer to main memory (Enter 'done' to exit): 10
Page 10 already in memory. Page Hit

  Page number    | Frame number  |
----------------+--------------+
        10      |       0      |
        20      |       1      |
        11      |       2      |


Enter page number to transfer to main memory (Enter 'done' to exit):
```



```
        10      |       0      |
        20      |       1      |
        11      |       2      |


Enter page number to transfer to main memory (Enter 'done' to exit): 21
Page 21 inserted in frame 3. Page Miss

  Page number    | Frame number  |
----------------+--------------+
        10      |       0      |
        20      |       1      |
        11      |       2      |
        21      |       3      |


Enter page number to transfer to main memory (Enter 'done' to exit): 22
Page 22 inserted in frame 0. Page 10 replaced. Page Miss

  Page number    | Frame number  |
----------------+--------------+
        20      |       1      |
        11      |       2      |
        21      |       3      |
        22      |       0      |


Enter page number to transfer to main memory (Enter 'done' to exit): done
○ →  OS git:(master) ✗
```

Conclusion : In this experiment we have implemented the paging technique to simulate memory allocation and management in an Operating System.