CHAPTER 5

MEMORY MANEGEMENT

Memory Allocation

- dynamic storage allocation problem concerns how to satisfy a request of size n from a list of free holes.
- The solutions :
  - **First fit**.
    - Allocate the first hole that is big enough.
    - Searching can start either at the beginning of the set of holes or at the location where the previous first-fit search ended.
    - We can stop searching as soon as we find a free hole that is large enough.

Memory Allocation

- **Best fit**.
  - Allocate the smallest hole that is big enough.
  - We must search the entire list, unless the list is ordered by size.
- **Worst fit**.
  - Allocate the largest hole.
  - we must search the entire list, unless it is sorted by size.
  - This strategy produces the largest leftover hole, which may be more useful than the smaller leftover hole from a best-fit approach.

Fragmentation

- is a phenomenon in which storage space is used inefficiently, reducing capacity or performance and often both.
- There are two different fragmentation:
  - external fragmentation and
  - internal fragmentation.

External fragmentation

- exists when there is enough total memory space to satisfy a request but the available spaces are not contiguous:
  - storage is fragmented into a large number of small holes.
  - That happen when the processes are loaded and removed from memory,
  - the free memory space is broken into little pieces.
  - If all these small pieces of memory were in one big free block instead, we might be able to run several more processes this solution called compaction.

Internal fragmentation

- exists when the memory allocated to a process may be slightly larger than the requested memory.
- Consider a hole of 18,464 bytes.
  - Suppose that the next process requests 18,462 bytes.
  - If we allocate exactly the requested block, we are left with a hole of 2 bytes.
  - to avoiding this problem is to break the physical memory into fixed-sized blocks and allocate memory in units based on block size.

Segmentation

- is a memory-management scheme that supports programmer view of memory.
- programmer thinks of it as a main program with a set of methods, procedures, or functions.
- It may also include various data structures: objects, arrays, stacks, variables, and so on.
- Each of these modules or data elements is referred to by name.

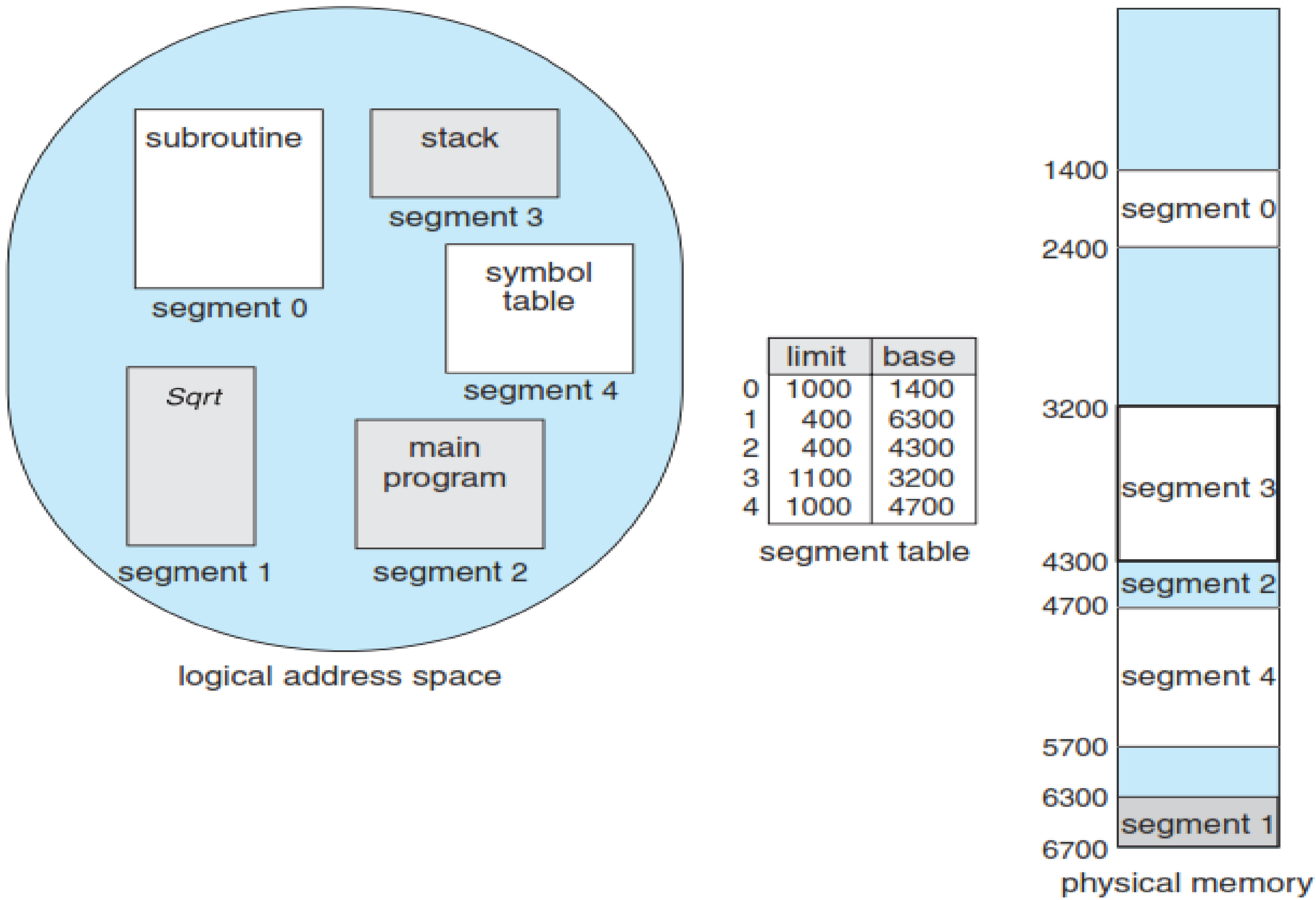
Segmentation

- Segments vary in length,
- the length of each is defined by its purpose in the program.
- Each segment has a name and a length.
- Segments are numbered and are referred to by a segment number, rather than by a segment name.
- Each segment has a segment base and a segment limit stored in segment table.

Segmentation

- Consider we have five segments numbered from 0 through 4.
- The segment table has a separate entry for each segment,
  - the beginning address of the segment in physical memory (or base)
  - and the length of that segment (or limit).
  - For example, segment 2 is 400 bytes long and begins at location 4300.

Segmentation



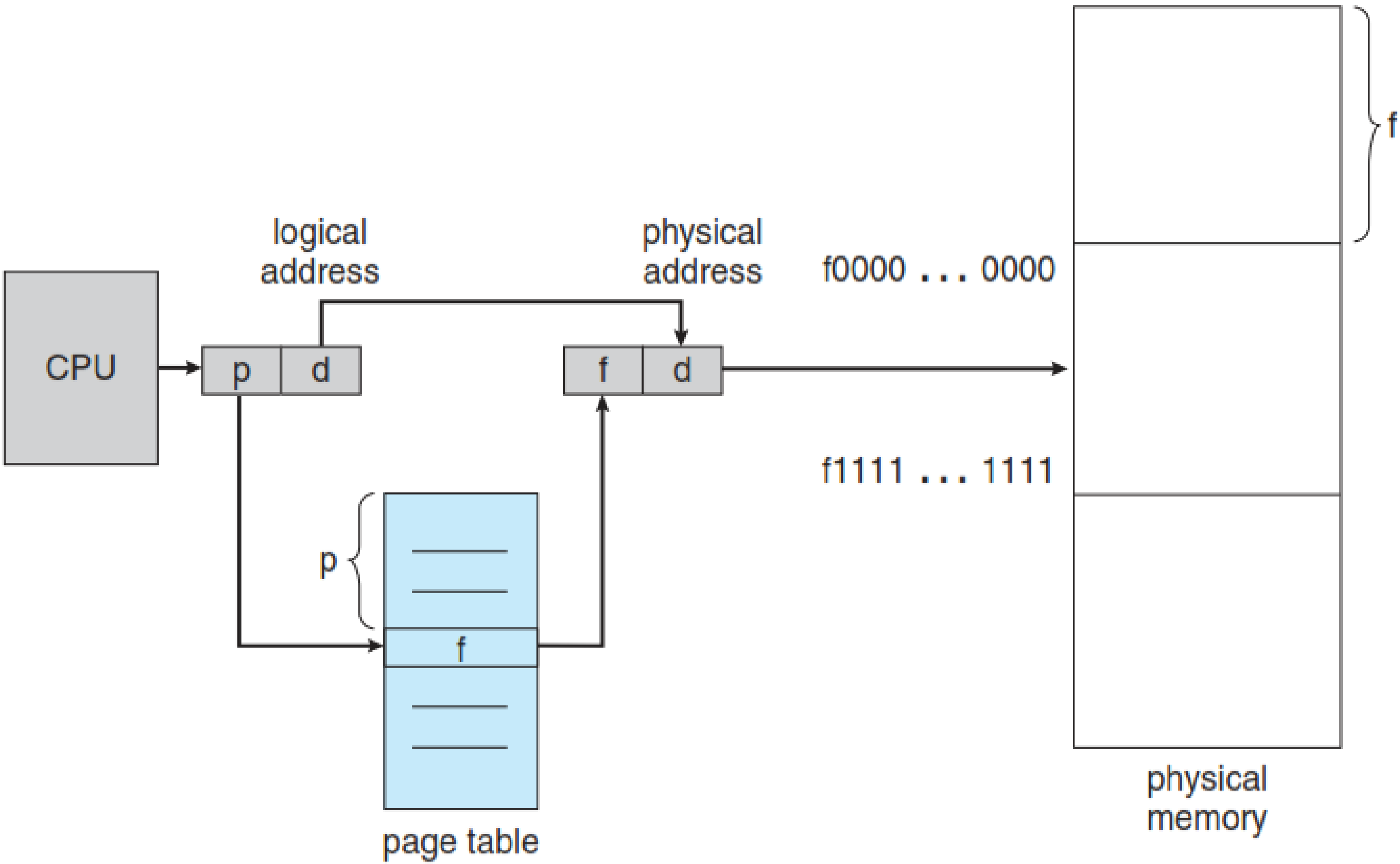
Paging

- breaking physical memory into fixed-sized blocks called **frames**
- breaking logical memory into blocks of the same size called **pages**.
- When a process is to be executed, its pages are loaded into any available memory frames from their source.
- The backing store is divided into fixed-sized **blocks** that are the same size as the memory frames or clusters of multiple frames.

Paging

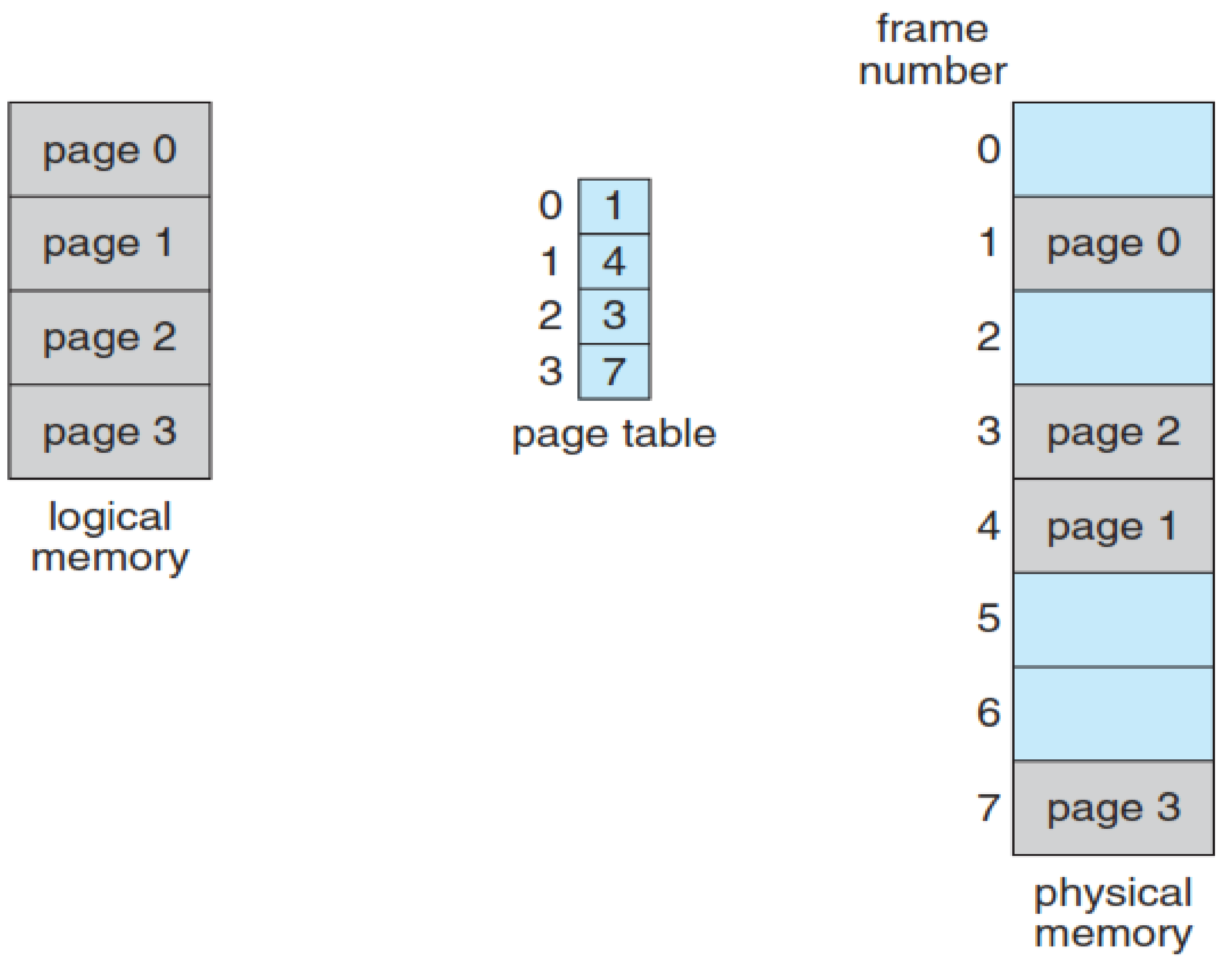
- Every address generated by the CPU is divided into two parts:
  - a page number (p) and
  - a page offset (d).
  - The page number is used as an index into a page table.
  - The page table contains the base address of each page in physical memory.
  - This base address is combined with the page offset to define the physical memory address that is sent to the memory unit

Paging



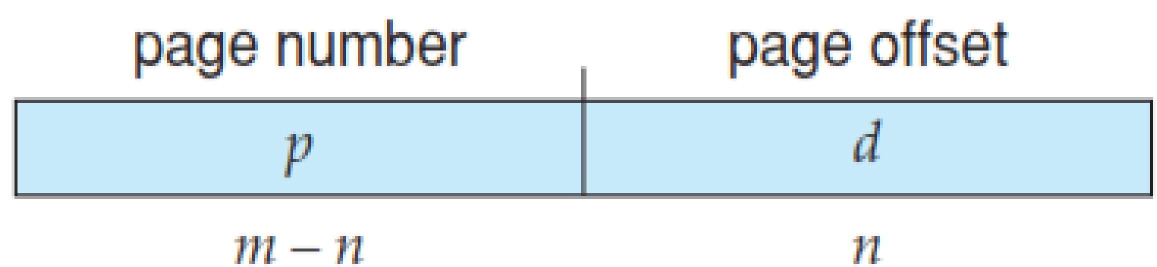
logical
address
physical
address
f0000 . . . 0000
f1111 . . . 1111
f
CPU
p
d
f
d
p
f
page table
physical
memory

Paging

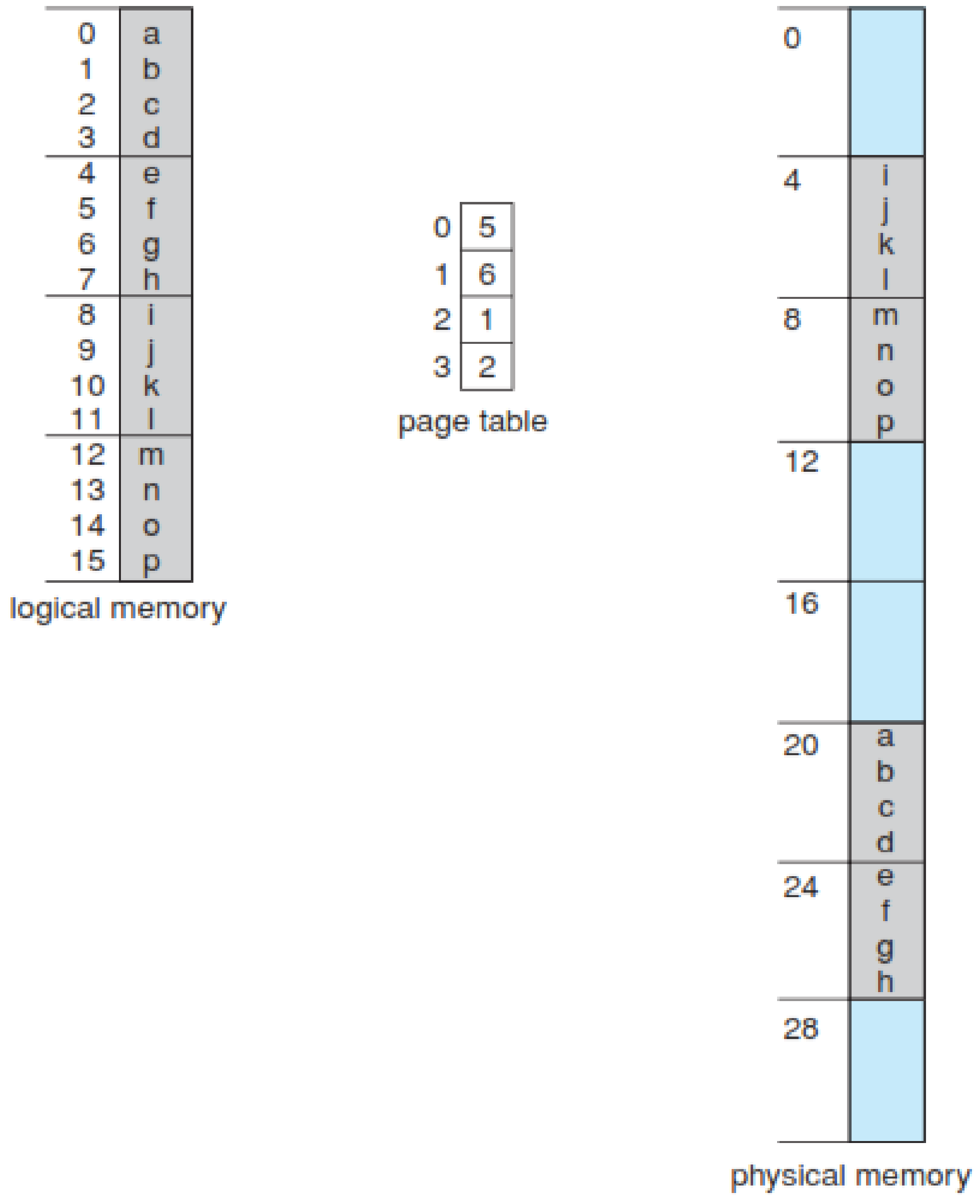

frame number
page 0
page 1
page 2
page 3
logical memory
0 1
1 4
2 3
3 7
page table
0
1 page 0
2
3 page 2
4 page 1
5
6
7 page 3
physical memory

Paging

- The size of a page is a power of 2,
- varying between 512 bytes and 1 GB per page, depending on the computer architecture.
- If the size of the logical address space is 2^m, and a page size is 2^n bytes,
  - then the high-order m - n bits of a logical address designate the page number,
  - and the n low-order bits designate the page offset.



Paging


0 a
1 b
2 c
3 d
4 e
5 f
6 g
7 h
8 i
9 j
10 k
11 l
12 m
13 n
14 o
15 p
logical memory
0 5
1 6
2 1
3 2
page table
0
4 i j k l
8 m n o p
12
16
20 a b c d
24 e f g h
28
physical memory

Paging

- the logical address, n = 2 and m = 4.
- Using a page size of 4 bytes and
- a physical memory of 32 bytes (8 pages),
- Logical address 0 is page 0, offset 0.
- we find that page 0 is in frame 5.
  - logical address 0 maps to physical address 20 [= (5 × 4) + 0].
- Logical address 3 (page 0, offset 3)
  - maps to physical address 23 [= (5 × 4) + 3].
- Logical address 4 is page 1, offset 0;
  - page 1 is mapped to frame 6.
  - logical address 4 maps to physical address 24 [= (6 × 4) + 0].
- Logical address 13 maps to physical address 9.

Paging

- we have no external fragmentation
- we may have some internal fragmentation.
- If the memory requirements of a process do not match page boundaries, the last frame allocated may not be completely full.
  - For example, if page size is 2,048 bytes,
  - a process of 72,766 bytes will need 35 pages plus 1,086 bytes.
  - It will be allocated 36 frames, resulting in internal fragmentation of 2,048 − 1,086 = 962 bytes.

Paging

- small page sizes are desirable.
- overhead is involved in each page-table entry,
- and this overhead is reduced as the size of the pages increases.
- disk I/O is more efficient when the amount data being transferred is larger.
- page sizes have grown over time as processes, data sets, and main memory have become larger.
- Today, pages are between 4 KB and 8 KB in size.
- Some CPUs and kernels even support multiple page sizes.
  - For instance, Solaris uses page sizes of 8 KB and 4 MB, depending on the data stored by the pages.

References

- ABRAHAM SILBERSCHATZ, PETER BAER GALVIN, GREG GAGNE “OPERATING SYSTEM CONCEPTS” NINTH EDITION, Wiley