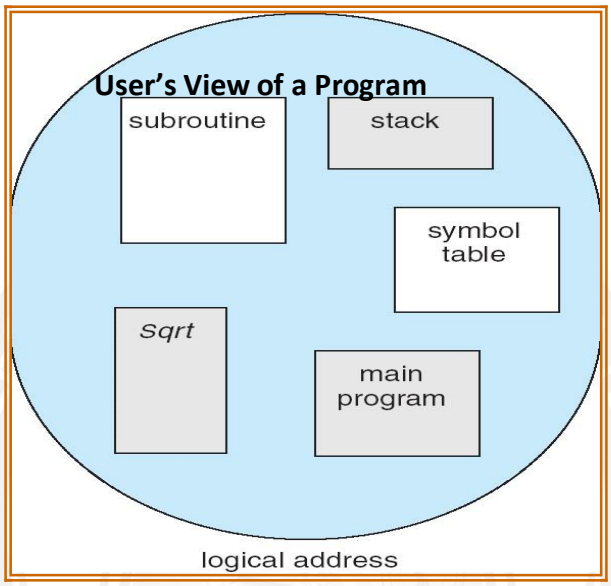
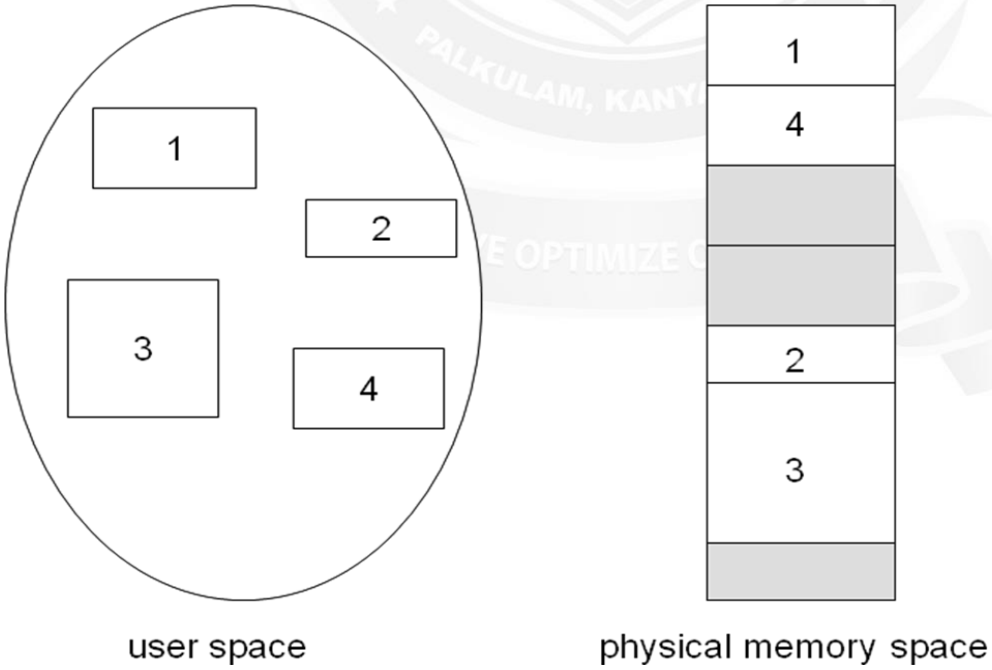
SEGMENTATION

Memory-management scheme that supports user view of memory

A program is a collection of segments. A segment is a logical unit such as: Main program, Procedure, Function, Method, Object, Local variables, global variables, Common block, Stack, Symbol table, arrays



Logical View of Segmentation



Segmentation Hardware

Logical address consists of a two tuple :

<Segment-number, offset>

Segment table – maps two-dimensional physical addresses; each table entry has:

Base – contains the starting physical address where the segments reside in memory

Limit – specifies the length of the segment

Segment-table base register (STBR) points to the segment table's location in memory

Segment-table length register (STLR) indicates number of segments used by a program;

Segment number s is legal, if $s <$ STLR

Relocation.

Dynamic

By segment table

Sharing.

- shared segments
- same segment number

Allocation.

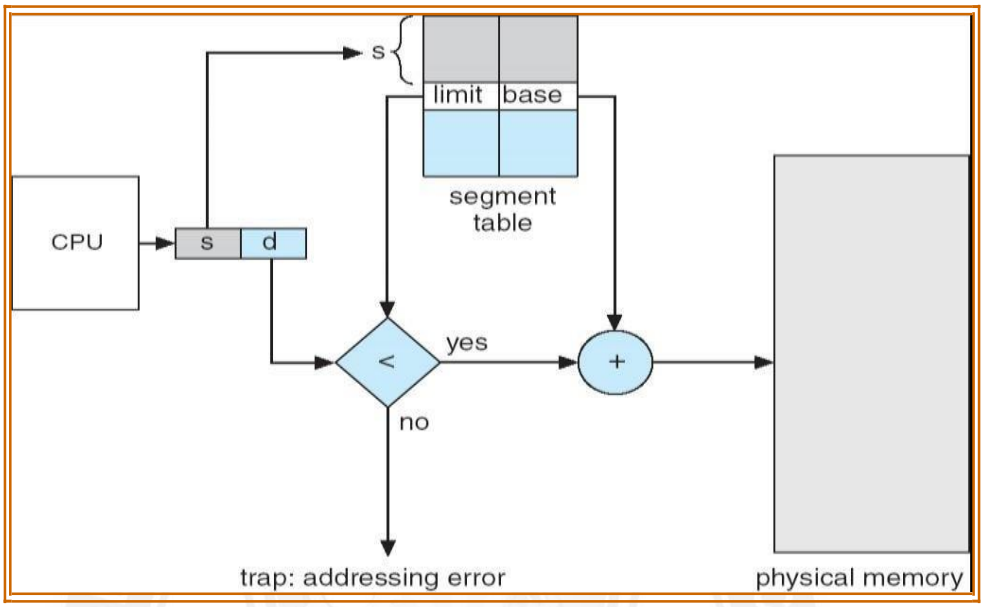
- first fit/best fit
- external fragmentation

Protection: With each entry in segment table associate:

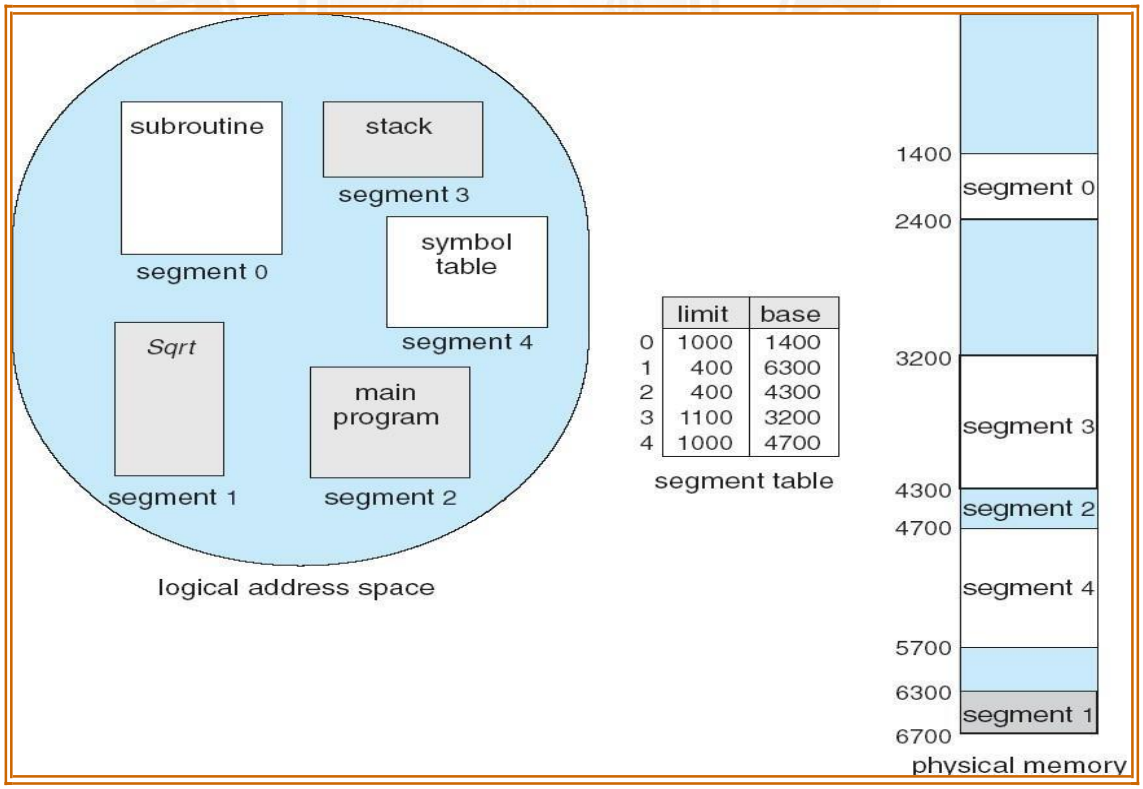
  - validation bit = 0 illegal segment
  - read/write/execute privileges

- Protection bits associated with segments; code sharing occurs at segment level
- Since segments vary in length, memory allocation is a dynamic storage- allocation problem
- A segmentation example is shown in the following diagram

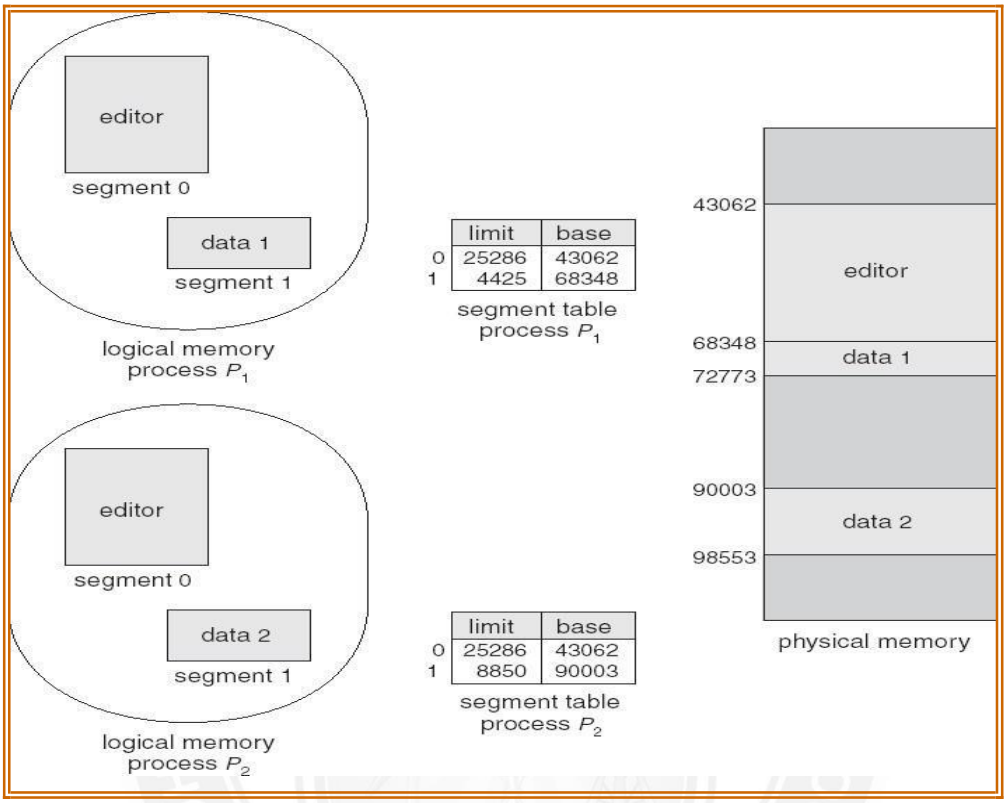
Address Translation scheme



EXAMPLE:



Sharing of Segments



- Another advantage of segmentation involves the sharing of code or data.
- Each process has a segment table associated with it, which the dispatcher uses to define the hardware segment table when this process is given the CPU.
- Segments are shared when entries in the segment tables of two different processes point to the same physical location.

Segmentation with paging

The IBM OS/ 2.32 bit version is an operating system running on top of the Intel 386 architecture. The 386 uses segmentation with paging for memory management. The maximum number of segments per process is 16 KB, and each segment can be as large as 4 gigabytes.

The local-address space of a process is divided into two partitions.

- The first partition consists of up to 8 KB segments that are private to that process.
- The second partition consists of up to 8KB segments that are shared among all the processes.

- Information about the first partition is kept in the **local descriptor table (LDT)**, information about the second partition is kept in the **global descriptor table (GDT)**.
- Each entry in the LDT and GDT consist of 8 bytes, with detailed information about a particular segment including the base location and length of the segment.
- The logical address is a pair (selector, offset) where the selector is a16-bit number:

s	g	p
13	1	2

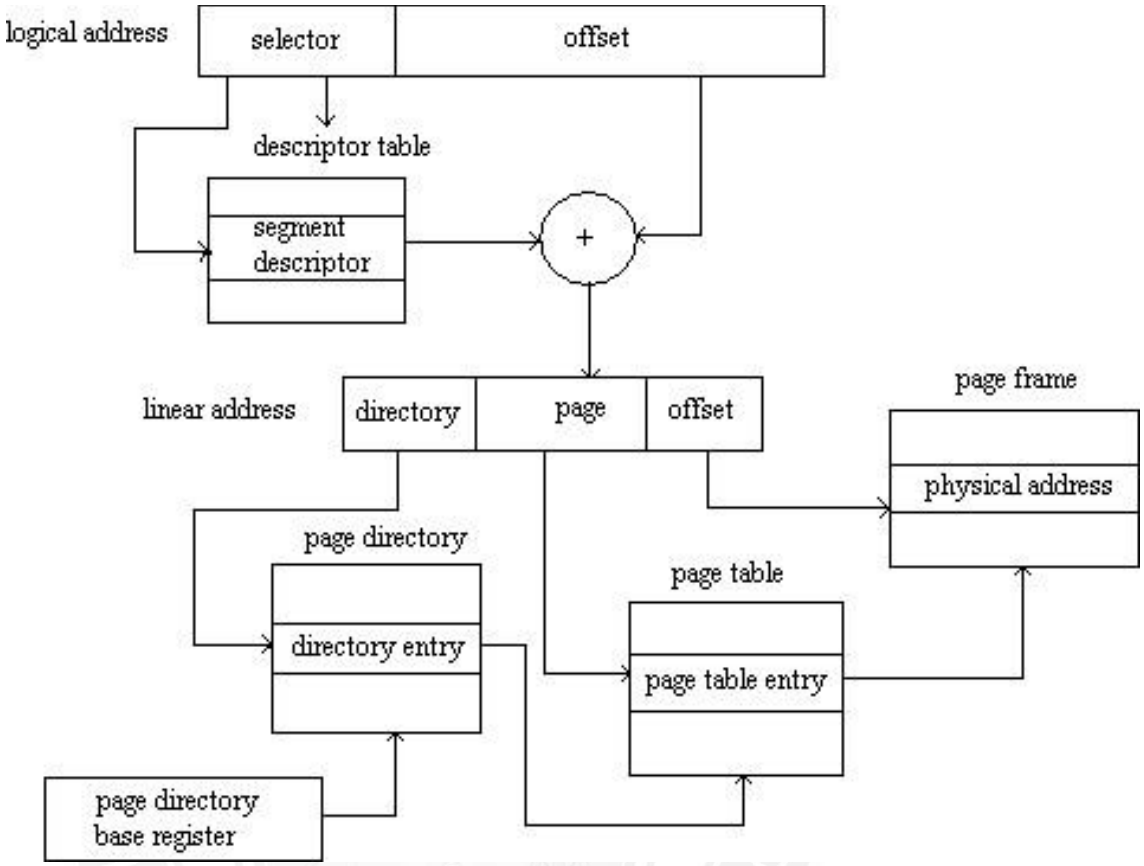
Where s designates the segment number, g indicates whether the segment is in the GDT or LDT, and p deals with protection. The offset is a 32-bit number specifying the location of the byte within the segment in question.

- The base and limit information about the segment in question are used to generate a linear-address.

- First, the limit is used to check for address validity. If the address is not valid, a memory fault is generated, resulting in a trap to the operating system. If it is valid, then the value of the offset is added to the value of the base, resulting in a 32-bit linear address. This address is then translated into a physical address.

- The linear address is divided into a page number consisting of 20 bits, and a page offset consisting of 12 bits. Since we page the page table, the page number is further divided into a 10-bit page directory pointer and a 10-bit page table pointer. The logical address is as follows.

p1	p2	d
10	10	12



- To improve the efficiency of physical memory use. Intel 386 page tables can be swapped to disk. In this case, an invalid bit is used in the page directory entry to indicate whether the table to which the entry is pointing is in memory or on disk.

- If the table is on disk, the operating system can use the other 31 bits to specify the disk location of the table; the table then can be brought into memory on demand.