information (pages or segments) that can be shared. Sharing is a means of running many processes with a limited amount of memory, but shared programs and data must be designed carefully.

- **Protection.** If paging or segmentation is provided, different sections of a user program can be declared execute-only, read-only, or read-write. This restriction is necessary with shared code or data and is generally useful in any case to provide simple run-time checks for common programming errors.

**Practice Exercises**

8.1 Name two differences between logical and physical addresses.

8.2 Consider a system in which a program can be separated into two parts: code and data. The CPU knows whether it wants an instruction (instruction fetch) or data (data fetch or store). Therefore, two base-limit register pairs are provided: one for instructions and one for data. The instruction base-limit register pair is automatically read-only, so programs can be shared among different users. Discuss the advantages and disadvantages of this scheme.

8.3 Why are page sizes always powers of 2?

8.4 Consider a logical address space of 64 pages of 1,024 words each, mapped onto a physical memory of 32 frames.

   a. How many bits are there in the logical address?
   b. How many bits are there in the physical address?

8.5 What is the effect of allowing two entries in a page table to point to the same page frame in memory? Explain how this effect could be used to decrease the amount of time needed to copy a large amount of memory from one place to another. What effect would updating some byte on the one page have on the other page?

8.6 Describe a mechanism by which one segment could belong to the address space of two different processes.

8.7 Sharing segments among processes without requiring that they have the same segment number is possible in a dynamically linked segmentation system.

   a. Define a system that allows static linking and sharing of segments without requiring that the segment numbers be the same.
   b. Describe a paging scheme that allows pages to be shared without requiring that the page numbers be the same.

8.8 In the IBM/370, memory protection is provided through the use of **keys.** A key is a 4-bit quantity. Each 2-k block of memory has a key (the storage key) associated with it. The CPU also has a key (the protection key) associated with it. A store operation is allowed only if both keys
are equal or if either is 0. Which of the following memory-management schemes could be used successfully with this hardware?

a. Bare machine
b. Single-user system
c. Multiprogramming with a fixed number of processes
d. Multiprogramming with a variable number of processes
e. Paging
f. Segmentation

Exercises

8.9 Explain the difference between internal and external fragmentation.

8.10 Consider the following process for generating binaries. A compiler is used to generate the object code for individual modules, and a linkage editor is used to combine multiple object modules into a single program binary. How does the linkage editor change the binding of instructions and data to memory addresses? What information needs to be passed from the compiler to the linkage editor to facilitate the memory-binding tasks of the linkage editor?

8.11 Given six memory partitions of 300 KB, 600 KB, 350 KB, 200 KB, 750 KB, and 125 KB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 115 KB, 500 KB, 358 KB, 200 KB, and 375 KB (in order)? Rank the algorithms in terms of how efficiently they use memory.

8.12 Most systems allow a program to allocate more memory to its address space during execution. Allocation of data in the heap segments of programs is an example of such allocated memory. What is required to support dynamic memory allocation in the following schemes?

a. Contiguous memory allocation
b. Pure segmentation
c. Pure paging

8.13 Compare the memory organization schemes of contiguous memory allocation, pure segmentation, and pure paging with respect to the following issues:

a. External fragmentation
b. Internal fragmentation
c. Ability to share code across processes

8.14 On a system with paging, a process cannot access memory that it does not own. Why? How could the operating system allow access to other memory? Why should it or should it not?
Chapter 8  Main Memory

8.15 Explain why mobile operating systems such as iOS and Android do not support swapping.

8.16 Although Android does not support swapping on its boot disk, it is possible to set up a swap space using a separate SD nonvolatile memory card. Why would Android disallow swapping on its boot disk yet allow it on a secondary disk?

8.17 Compare paging with segmentation with respect to how much memory the address translation structures require to convert virtual addresses to physical addresses.

8.18 Explain why address space identifiers (ASIDs) are used.

8.19 Program binaries in many systems are typically structured as follows. Code is stored starting with a small, fixed virtual address, such as 0. The code segment is followed by the data segment that is used for storing the program variables. When the program starts executing, the stack is allocated at the other end of the virtual address space and is allowed to grow toward lower virtual addresses. What is the significance of this structure for the following schemes?
   a. Contiguous memory allocation
   b. Pure segmentation
   c. Pure paging

8.20 Assuming a 1-KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers):
   a. 3085
   b. 42095
   c. 215201
   d. 650000
   e. 2000001

8.21 The BTV operating system has a 21-bit virtual address, yet on certain embedded devices, it has only a 16-bit physical address. It also has a 2-KB page size. How many entries are there in each of the following?
   a. A conventional, single-level page table
   b. An inverted page table

8.22 What is the maximum amount of physical memory?

8.23 Consider a logical address space of 256 pages with a 4-KB page size, mapped onto a physical memory of 64 frames.
   a. How many bits are required in the logical address?
   b. How many bits are required in the physical address?
8.24 Consider a computer system with a 32-bit logical address and 4-KB page size. The system supports up to 512 MB of physical memory. How many entries are there in each of the following?

8.25 Consider a paging system with the page table stored in memory.
   a. If a memory reference takes 50 nanoseconds, how long does a paged memory reference take?
   b. If we add TLBs, and 75 percent of all page-table references are found in the TLBs, what is the effective memory reference time? (Assume that finding a page-table entry in the TLBs takes 2 nanoseconds, if the entry is present.)

8.26 Why are segmentation and paging sometimes combined into one scheme?

8.27 Explain why sharing a reentrant module is easier when segmentation is used than when pure paging is used.

8.28 Consider the following segment table:

<table>
<thead>
<tr>
<th>Segment</th>
<th>Base</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>219</td>
<td>600</td>
</tr>
<tr>
<td>1</td>
<td>2300</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>3</td>
<td>1327</td>
<td>580</td>
</tr>
<tr>
<td>4</td>
<td>1952</td>
<td>96</td>
</tr>
</tbody>
</table>

What are the physical addresses for the following logical addresses?
   a. 0,430
   b. 1,10
   c. 2,500
   d. 3,400
   e. 4,112

8.29 What is the purpose of paging the page tables?

8.30 Consider the hierarchical paging scheme used by the VAX architecture. How many memory operations are performed when a user program executes a memory-load operation?

8.31 Compare the segmented paging scheme with the hashed page table scheme for handling large address spaces. Under what circumstances is one scheme preferable to the other?

8.32 Consider the Intel address-translation scheme shown in Figure 8.22.
   a. Describe all the steps taken by the Intel Pentium in translating a logical address into a physical address.
   b. What are the advantages to the operating system of hardware that provides such complicated memory translation?
8.33 Assume that a system has a 32-bit virtual address with a 4-KB page size. Write a C program that is passed a virtual address (in decimal) on the command line and have it output the page number and offset for the given address. As an example, your program would run as follows:

```
./a.out 19986
```

Your program would output:

```
The address 19986 contains:
page number = 4
offset = 3602
```

Writing this program will require using the appropriate data type to store 32 bits. We encourage you to use `unsigned` data types as well.

**Bibliographical Notes**

Dynamic storage allocation was discussed by [Knuth (1973)](Section 2.5), who found through simulation that first fit is generally superior to best fit. [Knuth (1973)] also discussed the 50-percent rule.

The concept of paging can be credited to the designers of the Atlas system, which has been described by [Kilburn et al. (1961)](Section 2.5) and by [Howarth et al. (1961)]. The concept of segmentation was first discussed by [Dennis (1965)]. Paged segmentation was first supported in the GE 645, on which MULTICS was originally implemented ([Organick (1972)] and [Daley and Dennis (1967)]).

Inverted page tables are discussed in an article about the IBM RT storage manager by [Chang and Mergen (1988)].

[Hennessy and Patterson (2012)] explains the hardware aspects of TLBs, caches, and MMUs. [Talluri et al. (1995)] discusses page tables for 64-bit address spaces. [Jacob and Mudge (2001)] describes techniques for managing the TLB. [Fang et al. (2001)] evaluates support for large pages.


**Bibliography**