**Student name:** Federico Watkins

**CSIS 3810:** Operating Systems

**Chapter:** 8 Main Memory

**Assignment:** 6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. 🡪 8.11 | 1. 🡪 8.15 | 1. 🡪 8.20 | 1. 🡪 8.24 | 1. 8.28 |

1. 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.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **First-fit** | | | | | | | | | |
| 115 KB | 🡪 | 300 KB | **=** | 185 KB | 600 KB | 350 KB | 200 KB | 750 KB | 125 KB |
| 500 KB | 🡪 | 600 KB | **=** | 185 KB | 100 KB | 350 KB | 200 KB | 750 KB | 125 KB |
| 358 KB | 🡪 | 750 KB | **=** | 185 KB | 100 KB | 350 KB | 200 KB | 392 KB | 125 KB |
| 200 KB | 🡪 | 350 KB | **=** | 185 KB | 100 KB | 150 KB | 200 KB | 392 KB | 125 KB |
| 375 KB | 🡪 | 392 KB | **=** | 185 KB | 100 KB | 150 KB | 200 KB | 17 KB | 125 KB |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Best-fit** | | | | | | | | | |
| 115 KB | 🡪 | 125 KB | **=** | 300 KB | 600 KB | 350 KB | 200 KB | 750 KB | 10 KB |
| 500 KB | 🡪 | 600 KB | **=** | 300 KB | 100 KB | 350 KB | 200 KB | 750 KB | 10 KB |
| 358 KB | 🡪 | 750 KB | **=** | 300 KB | 100 KB | 350 KB | 200 KB | 392 KB | 10 KB |
| 200 KB | 🡪 | 200 KB | **=** | 300 KB | 100 KB | 350 KB | 0 KB | 392 KB | 10 KB |
| 375 KB | 🡪 | 392 KB | **=** | 300 KB | 100 KB | 350 KB | 0 KB | 17 KB | 10 KB |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Worst-fit** | | | | | | | | | |
| 115 KB | 🡪 | 750 KB | **=** | 300 KB | 600 KB | 350 KB | 200 KB | 635 KB | 125 KB |
| 500 KB | 🡪 | 635 KB | **=** | 300 KB | 600 KB | 350 KB | 200 KB | 135 KB | 125 KB |
| 358 KB | 🡪 | 600 KB | **=** | 300 KB | 242 KB | 350 KB | 200 KB | 135 KB | 125 KB |
| 200 KB | 🡪 | 350 KB | **=** | 300 KB | 242 KB | 150 KB | 200 KB | 135 KB | 125 KB |
| 375 KB | 🡪 | Has to wait! | | | | | | | |

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

* Mobile devices have a **space constraint** in that they usually use **flash memory** with **limited capacity** hence swapping is just avoided.
* Flash memory becomes **unreliable** after so many **write operations**.
* More often than not, there **is low throughput** between **main memory and flash** **memory**.

1. Assuming a 1-KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers):
2. 3085

|  |  |
| --- | --- |
| **Page** | **Offset** |
| 3 | 13 |

1. 42095

|  |  |
| --- | --- |
| **Page** | **Offset** |
| 41 | 111 |

1. 215201

|  |  |
| --- | --- |
| **Page** | **Offset** |
| 210 | 161 |

1. 650000

|  |  |
| --- | --- |
| **Page** | **Offset** |
| 634 | 784 |

1. 2000001

|  |  |
| --- | --- |
| **Page** | **Offset** |
| 1953 | 129 |

1. 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?
2. A conventional single-level page table

|  |  |
| --- | --- |
| 220 | Entries |

1. An inverted page table
2. Consider the following segment table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Segment** |  | **Base** |  | **Length** |
| 0 |  | 219 |  | 600 |
| 1 |  | 2300 |  | 14 |
| 2 |  | 90 |  | 100 |
| 3 |  | 1327 |  | 580 |
| 4 |  | 1952 |  | 96 |

What are the physical addresses for the following logical addresses?

1. 0,430
2. 1,10
3. 2,500

Illegal reference, trap to operating system.

1. 3,400
2. 4,112

Illegal reference, trap to operating system.

References:

1. Operating System Concepts, Ninth Edition, Abraham Silberschartz, Peter bear Galvin, Greg Gagne. Entire Chapter 8 for each question in its respective section.
2. http://www.tutorialspoint.com/operating\_system/os\_memory\_management.htm

In this chapter I learned in detail about various ways of organizing memory hardware. How we improve both the utilization of the CPU and the speed of the computer’s response to its users by sharing memory. Explored various techniques of allocating memory to processes. The memory-management algorithms varied form a primitive bare-machine approach to paging and segmentation strategies. Although we touched on it in Chapter 6, most interesting of all is the details of how paging works in contemporary computer systems. It is now clear that selection of a memory-management method for a specific system depends on many factors, especially on the hardware design of the system. Understanding why many algorithms require hardware support explains why many systems closely integrate hardware and operating-system memory management.