CSC 369 Worksheet 8 Solution

August 25, 2020

1. I need to translate the addresses in the following sets of parameters

   - ./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 0
   - ./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 1
   - ./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 2

   Running each command results as follows, with the following sets of valid and invalid addresses

   - ./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 0

     – VA 0: 0x0000006c (decimal: 108) → Segment violation
     – VA 1: 0x00000061 (decimal: 97) → Segment violation
     – VA 2: 0x00000035 (decimal: 53) → Segment violation
     – VA 3: 0x00000021 (decimal: 33) → Segment violation
     – VA 4: 0x00000041 (decimal: 65) → Segment violation
   - ./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 1

- `./segmentation.py -a 128 -p 512 -b 0 -l 20 -B 512 -L 20 -s 2`

Notes

- I need help on this question
- How do we know tell which VA goes to which segmentation?
- How could VA 0 (decimal 108) be valid at SEG 1 (decimal 492) when limit is 20?
- **Segmentation**
  - **Segment** is a contigous portion of the address space of a particular length
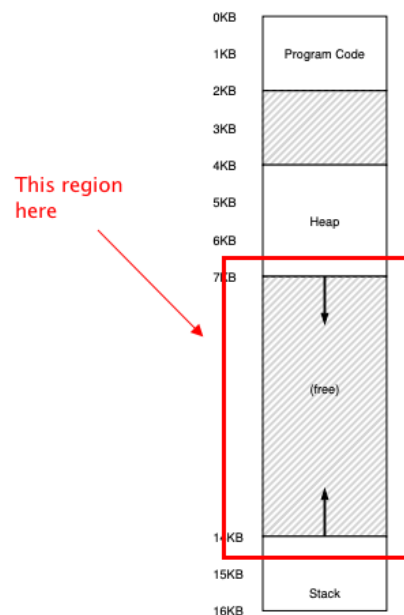 - Is about the big chunk of space in the middle

Figure 16.1: **An Address Space (Again)**

  - **Segmentation** allows the OS to place each one of the logical segments (i.e. stack, heap, program code) in different parts of physical memory, and avoid filling physical memory with unused virtual address space

0KB

Operating System

Segment

16KB

(not in use)

Stack

(not in use)

32KB

Code

Heap

48KB

(not in use)

64KB