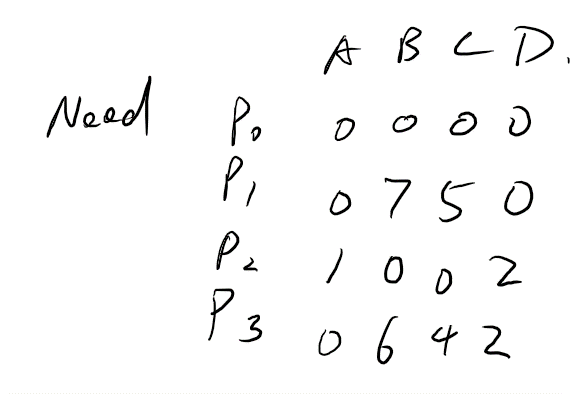
# Homework4

Name Yu Yang Student ID 892449550

# 7.3

## a.



## b.

yes. P0 runs and returns all the allocated resources and p2. Then Available becomes 2 8 8 6, then p1 and p3 could finish.

## c.

It could. Then the order of exection is p0 p2 p3 p1

# 7.7

We could define a max allowed time for every process. Timer starts when a process starts, when it reaches a certain amount of time, it means that process is starving.

# 7.17

We could always find a way out in this system. Because in the worst case, each process has 1 resource and the system still has 1 left. No matter the left 1 is allocated to which process, this process could finish and return its 2 resources. Then other 2 processes could finish

# 8.9

Internal fragmentation is the difference between memory allocated to a process and the memory requested by process – unused memory that is internal to a partition.

External fragmentation is exists when there is enough total space to satisfy a request by the that space is not contiguous, storage is fragmented into a large number of small holes – free memory external to partitions.

# 8.11

## First fit:

115 KB allocated to 300 KB partition

500 KB allocated to 600 KB partition

358 KB allocated to 750 KB partition

200 KB allocated to 350 KB partition

375 KB need to wait former one finishing

## Best fit

115 KB allocated to 125 KB partition

500 KB allocated to 600 KB partition

358 KB allocated to 750 KB partition

200 KB allocated to 200 KB partition

375 KB need to wait former one finishing

## Worst fit

115 KB allocated to 750 KB partition

500 KB allocated to 600 KB partition

358 KB need to wait former one finishing

In this case both Best fit and first fit could allocate 4 process in the sequence without pause, more efficient than worst fit.

Internal fragmentation of best fit = 10 + 100 + 392 = 502

Internal fragmentation of first fit = 185 + 100 + 392 + 150 > 502, in terms of internal fragmentation, best fit is better that first fit.

# 8.13

A.B external fragmentation and internal fragmentation

Contiguous memory allocation could not eliminate external fragmentation, that is the reason of introducing segmentation and paging, because they could allow logical address space of the processes to be noncontiguous. There is also possibility that external fragmentation occurs in pure segmentation because that each segment in a process is contiguous in physical memory.

Pure paging doesn’t suffer from external fragmentation, but suffers from internal fragmentation if a page is not used completely in a process, because the granularity is a page size.

C. Share code:

Contiguous memory allocation doesn’t allow code share because memory segment is not broken into non-contiguous part. Segmentation and paging allow it because common code and distinct data segments could be in different segments.

# 8.23

a. logical addr: log(256) + log(4K) = 8 + 12 = 20

b. physical addr: log(64) + log(4K) = 18

# 8.25

a. 50 \* 2 = 100, 50 for accessing page table and 50 for accessing actual memory content

b. 0.75 \* (200 + 2) + 0.25\*(200 + 200) = 251.5

# 8.29

Most modern computer systems have large logical address space. In such an environment, the page table becomes excessively large. We don’t want allocate that large page table contiguously in main memory so we page the page table itself.

# Virtual Machine

