Hung-Hsueh Shih (UIN: 132006330) CSCE611 Operating System
Machine Problem 2 : Frame Management

Introduction
The objective of this machine problem is to design two frame pools. One kernel frame pool from (2MB -> 4MB) and one process frame pool from (4MB to 32MB).

Design:
Since one frame is 4KB, the kernel frame pool will have 512 frames. I used bit map to store the state of each frames.
00 -> Frame is free
01-> Frame is head of sequence
10 -> Frame is not accessible
11-> Frame is allocated
Since one bytes can store 8 bits and I used two bits to represent the state of frame, one frame can be used to store 4KB*4 = 16K frames information in bit map.

Implementation:

1) cont_frame_pool.H

Header file for contiguous frame pool, I define the private variables in this header file included the basic information for the frame pool (such as bit map, total number of free frames in this pool, total size of the frame pool, and the pointer point to next frame.) Beside, I also define two method for cont_frame_pool class, one getter and one setter.

2) cont_frame _pool.C

Constructor : declare all the information for this frame pool, included the number of first frame in this frame pool, the size, in frames, in this frame pool, the total number of free frames, and the information frame number.
In addition, I initialize the bit map to 0x0 to initialize each frame in this frame pool is free. Then, based on the frame pool to set up the storing location for this frame pool (in kernel frame pool I store the bit map in the first frame, thus, I need to change the state of first frame to 01 which is head. Last, I link the linked list like structure for the frame pool.

get_state() : use the frame number as an input return the state of the frame from 00, 01, 10 to11. Since one byte can store state for four frames, I need to based

on the frame number to find out the index of frame in bit map and the offset in this one byte. Eventually, I do the bit manipulation and return the state of frame.

set_state() the setter for the frame pool, take frame number as input, then change the state of frame number based on the input state we want to set up. Once again, do the bit manipulation to change the state of frame.

get_frames() input is how many frames we want to allocated. Iterating the Bit map first and using the getting to find out the state of this frame until we find out contiguous frames which have enough space to be allocated. Set the first frame in this sequence to 01 by using the setter and set the rest of frames in sequence to 11 as allocated.

mark_ inaccessiable() given the frame number and how many contiguous of frames, use setter to change the state of given input to inaccessible which is 10.

Release_frames() first checker whether the frame pool possess this frame number. Then use getter to check this frame number is the head of sequence. In the end, use setter to reset all frames to free.

needed_info_frames()
Return the number of info frames required to store the frames. In this case, we use one frame to 16KB.