Elijah Andrushenko

CS360

K.C. Wang

LAB#3 PRE-work 1: Multitasking System

outfile:

pid=0 priority=3

readyQ = [0 3] -> NULL

pid=1 priority=6

readyQ = [1 6] -> [0 3] -> NULL

pid=2 priority=7

readyQ = [2 7] -> [1 6] -> [0 3] -> NULL

pid=3 priority=5

readyQ = [2 7] -> [1 6] -> [3 5] -> [0 3] -> NULL

pid=4 priority=3

readyQ = [2 7] -> [1 6] -> [3 5] -> [4 3] -> [0 3] -> NULL

pid=5 priority=5

readyQ = [2 7] -> [1 6] -> [5 5] -> [3 5] -> [4 3] -> [0 3] -> NULL

pid=6 priority=6

readyQ = [2 7] -> [6 6] -> [1 6] -> [5 5] -> [3 5] -> [4 3] -> [0 3] -> NULL

pid=7 priority=2

readyQ = [2 7] -> [6 6] -> [1 6] -> [5 5] -> [3 5] -> [4 3] -> [0 3] -> [7 2] -> NULL

pid=8 priority=9

readyQ = [8 9] -> [2 7] -> [6 6] -> [1 6] -> [5 5] -> [3 5] -> [4 3] -> [0 3] -> [7 2] -> NULL

queue.c:

#include <stdio.h>

#include "type.h"

int enqueue(PROC \*\*queue, PROC \*p)

{

PROC\* Cur = \*queue;

if (\*queue == NULL || (\*queue)->priority <= p->priority)

{

p->next = \*queue;

\*queue = p;

}

else

{

while (Cur->next != NULL && Cur->next->priority > p->priority)

{

Cur = Cur->next;

}

p->next = Cur->next;

Cur->next = p;

}

}

PROC \*dequeue(PROC \*\*queue)

{

// remove and return first PROC from queue

PROC\* Cur = \*queue;

PROC\* Prev = \*queue;

if (Cur->pid == 1)

{

return NULL;

}

else

{

Cur = Cur->next;

Prev->next = Cur->next;

return Cur;

}

}

queue.c (cont.):

int printList(char \*name, PROC \*p)

{

printf("%s = ", name);

PROC\* Cur = p;

while (Cur)

{

printf("[%d %d] -> ", Cur->pid, Cur->priority);

Cur = Cur->next;

}

printf("NULL\n");

return 0;

// print list elements as [pid priority] -> ....

}