NAME_____

ASSIGNMENT #7 91.308 OPERATING SYSTEMS

DUE APRIL 15, 2005

BUDDY SYSTEM. Using the buddy system of memory allocation indicate the **starting addresses** for each of the following **memory allocation requests** as they enter an initially empty memory allocation region which has a memory size of 2^{16} (64K) words. (Addresses run from 0 to 64k-1, and can be given in K form, i.e. location 4096 = 4K.) Assume (1) that when memory is allocated from a list the available block of memory closest to address 0 (shallow end of memory) is always given for the request and (2) the smallest allocated block is 4K. Give the **address** of each allocation in the space provided below:

TIME	JOB REQUESTING	JOB RETURNED	REQUEST SIZE(WORDS)
|	A		12K
|	B		3K
v	C		17K
|		A	
|	D		1K
v	E		6K
|		B	
|		D	
v	F		8K
|		E	
|		C	
v	G		15K

ANSWERS

Request A at _____ **Request E at _____**

Request B at _____ **Request F at _____**

Request C at _____ **Request G at _____**

Request D at _____

MAKE SURE TO HAND IN THE DIAGRAM THAT SHOWS YOUR WORK.

VIRTUAL MEMORY. The following problem deals with a virtual memory system with **a 12 bit address space (from 0 to 4096 (4K) locations).** The system is byte addressable and uses a **256 byte per page** organization. The real memory, therefore, is organized into **16 page frames of 256 bytes** each. Assume the operating system itself occupies the last six pages permanently, and that user programs will page against the **first 10 pages** as they run. Remember, the 12 bit address space will allow a user to have a virtual address space of **4096 bytes** (16 pages) even though only 10 real pages will be available for all running users to share during execution. The current status of this system is shown below for a time when 3 processes, **A, B and C**, are active in the system. All process are full size programs using all 16 virtual pages available. **A is presently in the running state** while B and C are in the ready state. As you look at the current CPU registers you can see that **process A has just fetched a JUMP instruction** from its code path. The **PROGRAM COUNTER (PC)** value shown is the (binary) **VIRTUAL address** of the JUMP instruction itself, which is now in the INSTRUCTION REGISTER (**IR**), and the JUMP instruction shows a (binary) **VIRTUAL address to jump to** as it executes.

A. From what **REAL physical byte address** did the current JUMP instruction in the **IR** come from ? (You can give a <page, offset> combination or the single number actual address, but use base 10 numbers either way)

Give a base 10 answer_____

B. To what **REAL physical byte address** will control be transferred when the current JUMP instruction executes ?? (Remember, **a page fault can occur** if a process thread references an invalid page, and faults are satisfied by connecting a virtual page to an available free physical page.) (Again, you can give a <page, offset> combination or the single number actual address, but use base 10 numbers either way)

Give a base 10 answer_____

Tables on next page ➔

SYSTEM PAGE FRAME TABLE AND CURRENT PAGE TABLE FOR RUNNING PROCESS A ARE SHOWN BELOW (THE OS KEEPS THESE IN ITS SPACE)

SYSTEM PAGE FRAME TABLE (PHYSICAL PAGES)

PAGE #	STATUS	FRAME # (BASE 2)	VALID BIT
0	OWNED BY A	NONE	0
1	OWNED BY B	NONE	0
2	OWNED BY C	NONE	0
3	OWNED BY B	0000	1
4	OWNED BY A	NONE	0
5	FREE	0110	1
6	OWNED BY A	NONE	0
7	OWNED BY C	1001	1
8	OWNED BY C	NONE	0
9	OWNED BY A	NONE	0
10	OP SYS	0100	1
11	OP SYS	NONE	0
12	OP SYS	NONE	0
13	OP SYS	NONE	0
14	OP SYS	NONE	0
15	OP SYS	NONE	0

PAGE TABLE FOR PROCESS A

CPU

PC(BASE 2) 101010010110
IR(BASE 2) JUMP 111011011011

Credits: These problems were developed by Prof. Moloney, UML CS Dept.