CS471: Operating System Concepts
Fall 2006
(Lecture: TR 11:25-12:40 PM)
Homework #5
Points: 20
Due: October 10, 2006

Q1) Question 1 [Points 4] Compare the differences between fork() and vfork() by running the program in Exercise 3.3 (pages 116-117). Print the values using fork() (as shown in 3.3) and by running again by replacing fork() with vfork(). Summarize the differences you have noticed with a brief explanation.

A1) The output by running the program 3.3 as is

PARENT: value = 5

The output by running the program 3.3 using vfork() is

PARENT: value = 20

When used fork(), both the child and parent has their own different address space. Both the parent and child has its own variable value.

With vfork(), both the child and parent share the same address space. After child thread starts, it executes the statement value += 15; the same value will be printed when parent excutes the printf statement because both the parent and children share the same set of variables and address space.

Question 2 [Points 8] Write a program that reads a sequence of address references of a process from an input file (provided to you under testdata5.txt). Assume that this process has been allocated 5 frames. It uses pure demand paging. Using FCFS page replacement algorithm, determine the total number of page faults. The page size is 2048 bytes.

Question 3 [Points 8] For the given sequence of page references, determine the number of page faults under theses page replacement algorithms: (i) FCFS (ii) Optimal (iii) LRU Each process has 4 buffers.
Reference string: 1,4,5,6,7,8,1,2,4,5,7,8,3,6,5,3,6,4,3,7,8,9,4,7,5,8,9

(i) FCFS:

The first row is the incoming page. Next four rows are the current pages that are in buffer. FCFS results in 20 page faults. * in the last row indicates page-faults.

1	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**4**	**5**	**7**	**8**	**3**	**6**	**5**	**3**	**6**	**4**	**3**	**7**	**8**	**9**	**4**	**7**	**5**	**8**	**9**
1	1	1	1	4	5	6	7	8	1	2	4	5	7	8	8	8	3	3	6	5	4	4	4	7	7	7
	4	4	4	5	6	7	8	1	2	4	5	7	8	3	3	3	6	6	5	4	7	7	7	8	8	8
		5	5	6	7	8	1	2	4	5	7	8	3	6	6	6	5	5	4	7	8	8	8	9	9	9
			6	7	8	1	2	4	5	7	8	3	6	5	5	5	4	4	7	8	9	9	9	5	5	5
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*			*		*	*	*			*		

(2) Optimal:

The first row is the incoming page. Next four rows are the current pages that are In buffer. Optimal results in 14 page faults.

1	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**4**	**5**	**7**	**8**	**3**	**6**	**5**	**3**	**6**	**4**	**3**	**7**	**8**	**9**	**4**	**7**	**5**	**8**	**9**
1	1	1	1	1	1	1	2	2	2	7	7	7	6	6	6	6	6	6	6	8	9	9	9	9	9	9
	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	8	8
		5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
			6	7	8	8	8	8	8	8	8	3	3	3	3	3	3	3	7	7	7	7	7	7	7	7
*	*	*	*	*	*		*			*		*	*						*	*	*				*	

(3) LRU:

The first row is the incoming page. Next four rows are the current pages that are In buffer. LRU results in 23 page faults.

1	**4**	**5**	**6**	**7**	**8**	**1**	**2**	**4**	**5**	**7**	**8**	**3**	**6**	**5**	**3**	**6**	**4**	**3**	**7**	**8**	**9**	**4**	**7**	**5**	**8**	**9**
1	1	1	1	7	7	7	7	4	4	4	4	3	3	3	3	3	3	3	3	3	3	4	4	4	4	9
	4	4	4	4	8	8	8	8	5	5	5	5	6	6	6	6	6	6	6	8	8	8	8	5	5	5
		5	5	5	5	1	1	1	1	7	7	7	7	5	5	5	5	5	7	7	7	7	7	7	7	7
			6	6	6	6	2	2	2	2	8	8	8	8	8	8	4	4	4	4	9	9	9	9	8	8
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*			*		*	*	*	*		*	*	*