Brandon Fowler **CSCD340** **Lab 11**

FIFO Algorithm (Showing work in new table for each eviction)

Stack Head

Page	Frame	Faults
1	0	
3	1	
4	2	
5	3	

Stack End

4

Page	Frame	Faults
2	0	
3	1	
4	2	
5	3	

5

Page	Frame	Faults
2	0	
8	1	
4	2	
5	3	

6

Page	Frame	Faults
2	0	
8	1	
6	2	
5	3	

7

Page	Frame	Faults
2	0	
8	1	
6	2	
7	3	

8

Page	Frame	Faults
10	0	
8	1	
6	2	
7	3	

9

Page	Frame	Faults
10	0	
3	1	
6	2	
7	3	

10

Page	Frame	Faults
10	0	
3	1	
9	2	
7	3	

11

Page	Frame	Faults
10	0	
3	1	
9	2	
4	3	

12

Page	Frame	Faults
8	0	
3	1	
9	2	
4	3	

13

Page	Frame	Faults
8	0	
1	1	
9	2	
4	3	

14

Page	Frame	Faults
8	0	
1	1	
5	2	
4	3	

15

Page	Frame	Faults
8	0	
1	1	
5	2	
2	3	

16

Page	Frame	Faults
3	0	
1	1	
5	2	
2	3	

17

Page	Frame	Faults
3	0	18
7	1	
5	2	
2	3	

Page	Frame	Faults
3	0	19
7	1	
4	2	
2	3	

Page	Frame	Faults
3	0	20
7	1	
4	2	
5	3	

Page	Frame	Faults
2	0	21
7	1	
4	2	
5	3	

Page	Frame	Faults
2	0	22
6	1	
4	2	
5	3	

Pages in memory at end : 2, 6, 4, 5

Total page faults: 22

LRU Time Counter Algorithm (Showing work in new table for each eviction)

Frames	0	1	2	3		Page	Faults
0	0	1	1	0		1	4
1	0	0	0	0		3	
2	0	1	0	0		4	
3	1	1	1	0		5	

	0	1	2	3		Page	Faults
0	0	0	1	0		1	5
1	1	0	1	1		2	
2	0	0	0	0		4	
3	1	0	1	0		5	

	0	1	2	3		Page	Faults
0	0	0	0	0		1	6
1	1	0	0	1		2	
2	1	1	0	1		3	
3	1	0	0	0		5	

	0	1	2	3		Page	Faults
0	0	1	1	1		8	7
1	0	0	0	1		2	
2	0	1	0	1		3	
3	0	0	0	0		5	

	0	1	2	3		Page	Faults
0	0	1	1	0		8	8
1	0	0	0	0		2	
2	0	1	0	0		3	
3	1	1	1	0		6	

	0	1	2	3		Page	Faults
0	0	1	0	0		8	9
1	1	1	0	1		7	
2	0	1	0	0		3	
3	1	1	0	0		6	

	0	1	2	3		Page	Faults
0	0	1	0	1		10	10
1	0	1	0	1		7	
2	1	1	0	1		3	
3	0	1	0	0		6	

	0	1	2	3		Page	Faults
0	0	1	0	0		10	11
1	0	1	0	0		7	
2	1	1	0	0		3	
3	1	1	1	0		9	

	0	1	2	3		Page
0	0	1	1	1		4
1	0	1	0	0		7
2	0	1	0	0		3
3	0	1	1	0		9

Faults 12

	0	1	2	3		Page
0	0	1	1	1		4
1	0	0	0	1		8
2	0	1	0	1		3
3	0	0	0	0		9

Faults 13

	0	1	2	3		Page
0	0	1	1	0		4
1	0	0	0	0		8
2	0	1	0	0		3
3	1	1	1	0		1

Faults 14

	0	1	2	3		Page
0	0	0	1	0		4
1	1	0	1	1		9
2	0	0	0	0		3
3	1	0	1	0		1

Faults 15

	0	1	2	3		Page
0	0	0	0	0		4
1	1	0	0	1		9
2	1	1	0	1		5
3	1	0	0	0		1

Faults 16

	0	1	2	3		Page
0	0	1	1	0		2
1	0	0	0	0		9
2	0	1	0	0		5
3	1	1	1	0		1

Faults 17

	0	1	2	3		Page
0	0	1	1	1		2
1	0	0	1	0		3
2	0	0	0	0		5
3	0	1	1	0		1

Faults 18

	0	1	2	3		Page
0	0	1	0	1		2
1	0	0	0	0		3
2	1	1	0	1		7
3	0	1	0	0		1

Faults 19

	0	1	2	3		Page
0	0	1	1	1		2
1	0	0	1	1		4
2	0	0	0	1		7
3	0	0	0	0		1

Faults 20

	0	1	2	3		Page
0	0	0	1	1		2
1	1	0	1	1		4
2	0	0	0	0		7
3	0	0	1	0		5

Faults 21

	0	1	2	3		Page
0	0	0	0	1		2
1	1	0	0	1		4
2	1	1	0	1		6
3	0	0	0	0		5

Faults 22

Pages in memory at end : 2, 4, 6, 5

Total page faults: 22

LRU Stack Algorithm (Showing work in new table for each reference and eviction)

Stack Head

Stack End

Page	Frame	Faults
5	3	4
1	0	
4	2	
3	1	

Page	Frame	Faults
2	1	5
5	3	
1	0	
4	2	

Page	Frame	Faults
3	2	6
2	1	
5	3	
1	0	

Page	Frame	Faults
8	0	7
3	2	
2	1	
5	3	

Page	Frame	Faults
6	3	8
8	0	
3	2	
2	1	

Page	Frame	Faults
7	1	9
6	3	
8	0	
3	2	

Page	Frame	Faults
10	2	10
7	1	
6	3	
8	0	

Page	Frame	Faults
3	0	11
10	2	
7	1	
6	3	

Page	Frame	Faults
9	3	12
3	0	
10	2	
7	1	

Page	Frame	Faults
4	1	13
9	3	
3	0	
10	2	

Page	Frame	Faults
8	2	14
4	1	
9	3	
3	0	

Page	Frame	Faults
3	0	14
8	2	
4	1	
9	3	

Page	Frame	Faults
9	3	14
3	0	
8	2	
4	1	

Page	Frame	Faults
4	1	14
9	3	
3	0	
8	2	

Page	Frame	Faults
1	2	15
4	1	
9	3	
3	0	

Page	Frame	Faults
9	3	15
1	2	
4	1	
3	0	

Page	Frame	Faults
5	0	16
9	3	
1	2	
4	1	

Page	Frame	Faults
2	1	17
5	0	
9	3	
1	2	

Page	Frame	Faults
1	2	17
2	1	
5	0	
9	3	

Page	Frame	Faults
3	3	18
1	2	
2	1	
5	0	

Page	Frame	Faults
5	0	18
3	3	
1	2	
2	1	

Page	Frame	Faults
1	2	18
5	0	
3	3	
2	1	

Page	Frame	Faults
5	0	18
1	2	
3	3	
2	1	

Page	Frame	Faults
1	2	18
5	0	
3	3	
2	1	

Page	Frame	Faults
3	3	18
1	2	
5	0	
2	1	

Page	Frame	Faults
2	1	18
3	3	
1	2	
5	0	

Page	Frame	Faults
3	3	18
2	1	
1	2	
5	0	

Page	Frame	Faults
1	2	18
3	3	
2	1	
5	0	

Page	Frame	Faults
2	1	18
1	2	
3	3	
5	0	

Page	Frame	Faults
7	0	19
2	1	
1	2	
3	3	

Page	Frame	Faults
4	3	20
7	0	
2	1	
1	2	

Page	Frame	Faults
2	1	20
4	3	
7	0	
1	2	

Page	Frame	Faults
4	3	20
2	1	
7	0	
1	2	

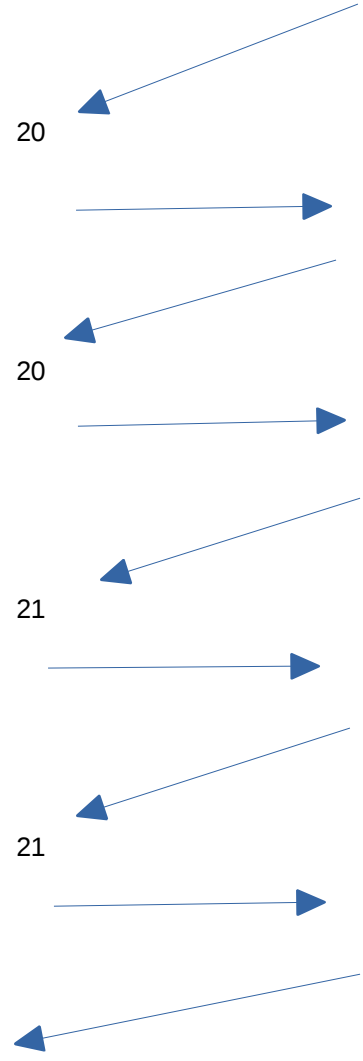
Page	Frame	Faults
2	1	20
4	3	
7	0	
1	2	

Page	Frame	Faults
5	2	21
2	1	
4	3	
7	0	

Page	Frame	Faults
2	1	21
5	2	
4	3	
7	0	

Page	Frame	Faults
4	3	21
2	1	
5	2	
7	0	

Page	Frame	Faults
6	0	22
4	3	
2	1	
5	2	



Pages in memory at end : 6, 4, 2, 5

Total page faults: 22

Stack Head

Stack End

Stu's Second Chance Algorithm (Showing work in new table for each reference and eviction)

Page	R – Bit	Frame
3	1	1
4	1	2
5	1	3
1	1	0

Faults: 4

Page	R – Bit	Frame
4	0	2
5	0	3
1	0	0
2	1	1

Faults: 5

Page	R – Bit	Frame
5	0	3
1	0	0
2	1	1
3	1	2

Faults: 6

Page	R – Bit	Frame
1	0	0
2	1	1
3	1	2
8	1	3

Faults: 7

Page	R – Bit	Frame
2	1	1
3	1	2
8	1	3
6	1	0

Faults: 8

Page	R – Bit	Frame
3	0	2
8	0	3
6	0	0
7	1	1

Faults: 9

Page	R – Bit	Frame
8	0	3
6	0	0
7	1	1
10	1	2

Faults: 10

Page	R – Bit	Frame
6	0	0
7	1	1
10	1	2
3	1	3

Faults: 11

Page	R – Bit	Frame
7	1	1
10	1	2
3	1	3
9	1	0

Faults: 12

Page	R – Bit	Frame
10	0	2
3	0	3
9	0	0
4	1	1

Faults: 13

Page	R – Bit	Frame
3	0	3
9	0	0
4	1	1
8	1	2

Faults: 14

Page	R – Bit	Frame
9	0	0
4	1	1
8	1	2
3	1	3

Faults: 14

Page	R – Bit	Frame
9	0	0
8	1	2
3	1	3
4	1	1

Faults: 14

Page	R – Bit	Frame
8	1	2
3	1	3
4	1	1
1	1	0

Faults: 15

Page	R – Bit	Frame
3	0	3
4	0	1
1	0	0
9	1	2

Faults: 16

Page	R – Bit	Frame
4	0	1
1	0	0
9	1	2
5	1	3

Faults: 17

Page	R – Bit	Frame
1	0	0
9	1	2
5	1	3
2	1	1

Faults: 18

Page	R – Bit	Frame
9	1	2
5	1	3
2	1	1
1	1	0

Faults: 18

Page	R – Bit	Frame
5	0	3
2	0	1
1	0	0
3	1	2

Faults: 19

Page	R – Bit	Frame
2	0	1
1	0	0
3	1	2
5	1	3

Faults: 19

Page	R – Bit	Frame
2	0	1
3	1	2
5	1	3
1	1	0

Faults: 19

Page	R – Bit	Frame
2	0	1
3	1	2
1	1	0
5	1	3

Faults: 19

Page	R – Bit	Frame
2	0	1
3	1	2
5	1	3
1	1	0

Faults: 19

Page	R – Bit	Frame
2	0	1
5	1	3
1	1	0
3	1	2

Faults: 19

Page	R – Bit	Frame
5	1	3
1	1	0
3	1	2
2	1	1

Faults: 19

Page	R – Bit	Frame
5	1	3
1	1	0
2	1	1
3	1	2

Faults: 19

Page	R – Bit	Frame
5	1	3
2	1	1
3	1	2
1	1	0

Faults: 19

Page	R – Bit	Frame
5	1	3
3	1	2
1	1	0
2	1	1

Faults: 19

Page	R – Bit	Frame
3	0	2
1	0	0
2	0	1
7	1	3

Faults 20

Page	R – Bit	Frame
1	0	0
2	0	1
7	1	3
4	1	2

Faults 21

Page	R – Bit	Frame
1	0	0
7	1	3
4	1	2
2	1	1

Faults 21

Page	R – Bit	Frame
1	0	0
7	1	3
2	1	1
4	1	2

Faults 21

Page	R – Bit	Frame
1	0	0
7	1	3
4	1	2
2	1	1

Faults 21

Page	R – Bit	Frame
7	1	3
4	1	2
2	1	1
5	1	0

Faults 22

Page	R – Bit	Frame
7	1	3
4	1	2
5	1	0
2	1	1

Faults 22

Page	R – Bit	Frame
7	1	3
5	1	0
2	1	1
4	1	2

Faults 23

Page	R – Bit	Frame
5	0	0
2	0	1
4	0	2
6	1	3

Pages in memory at end : 5, 2, 4, 6

Total page faults: 23

Clock Page Replacement Algorithm (Showing work in new table for each reference and eviction)

Page	R – Bit	Frame	Clock	Faults
1	1	0	x	4
3	1	1		
4	1	2		
5	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		5
3	0	1	x	
4	0	2		
5	0	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		5
3	1	1		
4	0	2	x	
5	0	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		6
3	1	1		
8	1	2		
5	0	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0	x	7
3	1	1		
8	1	2		
6	1	3		

Page	R – Bit	Frame	Clock	Faults
7	1	0		8
3	0	1	x	
8	0	2		
6	0	3		

Page	R – Bit	Frame	Clock	Faults
7	1	0		9
10	1	1		
8	0	2	x	
6	0	3		

Page	R – Bit	Frame	Clock	Faults
7	1	0		10
10	1	1		
3	1	2		
6	0	3	x	

Page	R – Bit	Frame	Clock	Faults
7	1	0	x	11
10	1	1		
3	1	2		
9	1	3		

Page	R – Bit	Frame	Clock	Faults
4	1	0		12
10	0	1	x	
3	0	2		
9	0	3		

Page	R – Bit	Frame	Clock	Faults
4	1	0		13
8	1	1		
3	0	2	x	
9	0	3		

Page	R – Bit	Frame	Clock	Faults
4	1	0		13
8	1	1		
3	0	2		
9	0	3	x	

Page	R – Bit	Frame	Clock	Faults
4	1	0		13
8	1	1	x	
3	0	2		
9	0	3		

Page	R – Bit	Frame	Clock	Faults
4	1	0		14
8	0	1		
1	1	2		
9	0	3	x	

Page	R – Bit	Frame	Clock	Faults
4	1	0	x	14
8	0	1		
1	1	2		
9	1	3		

Page	R – Bit	Frame	Clock	Faults
4	0	0		15
5	1	1		
1	1	2	x	
9	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		16
5	1	1	x	
1	0	2		
9	0	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		16
5	1	1		
1	1	2		
9	0	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0	x	17
5	1	1		
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2	x	
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2		
3	1	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2	x	
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2		
3	1	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0	x	17
5	1	1		
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1	x	
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0	x	17
5	1	1		
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2		
3	1	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1	x	
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0	x	17
5	1	1		
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1		
1	1	2		
3	1	3	x	

Page	R – Bit	Frame	Clock	Faults
2	1	0		17
5	1	1	x	
1	1	2		
3	1	3		

Page	R – Bit	Frame	Clock
2	0	0	
7	1	1	
1	0	2	x
3	0	3	

Faults: 18

Page	R – Bit	Frame	Clock
2	0	0	
7	1	1	
4	1	2	
3	0	3	x

Faults: 19

Page	R – Bit	Frame	Clock
2	1	0	
7	1	1	x
4	1	2	
3	0	3	

Faults: 19

Page	R – Bit	Frame	Clock
2	1	0	
7	1	1	
4	1	2	
3	0	3	x

Faults: 19

Page	R – Bit	Frame	Clock
2	1	0	
7	1	1	x
4	1	2	
3	0	3	

Faults: 19

Page	R – Bit	Frame	Clock
2	1	0	x
7	0	1	
4	0	2	
5	1	3	

Faults: 20

Page	R – Bit	Frame	Clock
2	1	0	
7	0	1	x
4	0	2	
5	1	3	

Faults: 20

Page	R – Bit	Frame	Clock
2	1	0	
7	0	1	
4	1	2	
5	1	3	x

Faults: 20

Page	R – Bit	Frame	Clock
2	0	0	
6	1	1	
4	1	2	x
5	0	3	

Faults: 21

Pages in memory at end : 2, 6, 4, 5

Total page faults: 21

Analysis

From the above experiment results I can see that FIFO, LRU Time Counter, and LRU Stack all end with 22 total page faults. Stu's Second Chance has the most page faults at 23, and Clock Page Replacement has the least amount of page faults at 21. All of the algorithms end with the same pages in memory. I believe the overall lack of variance in results is largely due to the low number of frames in the experiment.

By analyzing how the algorithms work, in an experiment with many more frames and page references, I would expect FIFO to have the most page faults. I would also expect Clock Page Replacement to have the least amount of page faults. FIFO should be the fastest algorithm just in terms of quickly deciding what to evict, followed by LRU Time Counter, and LRU Stack. Based on the above example, I would expect Stu's Second Chance and Clock Page Replacement to be the slowest in terms of deciding what to evict; because they frequently have to walk through large sections of the list to reset bits, before finding the correct page to evict. Overall, I would choose Clock Page Replacement as the most generally efficient algorithm. Since Clock has the lowest page faults, it will have to do less writing to disk/re-loading pages, which should more than make up for the longer search for eviction; and it doesn't have to do as much re-arranging as Stu's second chance does when a page is referenced.

Because FIFO is by definition a stack, it should be immune to Belady's Anomaly. LRU Stack is also obviously a stack, and thus immune to Belady's Anomaly. Stu's Second Chance is an enhanced variant of FIFO, and still very much a stack as made even more evident by the extra swapping of order on the stack, so it is also immune to Belady's Anomaly. Neither LRU Time Counter nor Clock Page Replacement employ stacks, and both should be susceptible to Belady's Anomaly.