UNIT 3 – Page replacement Algorithm

1) Consider the following page reference string: 1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 4, 5, 3. How many page faults would occur for the following replacement algorithms? Assume four frames and all frames are initially empty. **(16)** **(May 2015)**

   i) LRU replacement

   ii) FIFO replacement

   iii) Optimal replacement

****

a) **One Frame**

As there is only one page referenced more than once, all the LRU, FIFO and Optimal algorithms will result in **21 page faults.** 1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, **4, 4,** 5, 3.

b) **Two Frames**

i) **LRU(Least Recently Used)**

- Replace the page that **has not been** used for the longest period of time. **(past)**

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

Total Page faults = 18

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

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

Total Page faults = 18

iii) **Optimal**

- Replace the page that **will not be used** for the longest period of time.**(future)**

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

Total Page faults = 15

c) **Three Frames**

i) **LRU**

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

Total Page faults = 13

ii) **FIFO**

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

Total Page faults = 13

iii) **Optimal**

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

Total Page faults = 13

d) **Four Frames**

i) **LRU**

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

Total Page faults = 13

ii) **FIFO**

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

Total Page faults = 13

iii) **Optimal**

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

Total Page faults = 11

Result

No. of Frames	LRU	FIFO	Optimal
1	21	21	21
2	18	18	15
3	13	13	13
4	13	13	11

****

2) Consider the following page reference string: 1, 2, 3, 2, 5, 6, 3, 4, 6, 3, 7, 3, 1, 5, 3, 6, 3, 4, 2, 4, 3, 4, 5, 1. Indicate page faults and calculate total number of page faults and successful ration for FIFO, Optimal and LRU. Assume there are four frames and initially all the frames empty. **(12)** **(Nov 2015)**

****

a) **One Frame**

As there is no page referenced more than once, all the LRU, FIFO and Optimal algorithms will result in **24 page faults.** 1, 2, 3, 2, 5, 6, 3, 4, 6, 3, 7, 3, 1, 5, 3, 6, 3, 4, 2, 4, 3, 4, 5, 1.

b) **Two Frames**

i) **LRU(Least Recently Used)**

- Replace the page that has not been used for the longest period of time. **(past)**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	3		5	5	3	3	6	6	7		1	1	3	3		3	2		3		5	5
	2	2		2	6	6	4	4	3	3		3	5	5	6		4	4		4		4	1

Total Page faults = 19

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

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

Total Page faults = 20

iii) **Optimal**

- Replace the page that **will not be used** for the longest period of time.**(future)**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	3		3	3		4		3	3		3	3		3		3	2		3		3	1
	2	2		5	6		6		6	7		1	5		6		4	4		4		5	5

Total Page faults = 16

c) **Three Frames**

i) **LRU**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1		5	5	5	4			7		7	5		5		4	4				4	4
	2	2		2	2	3	3			3		3	3		3		3	3				3	1
		3		3	6	6	6			6		1	1		6		6	2				5	5

Total Page faults = 15

ii) **FIFO**

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

Total Page faults = 17

iii) **Optimal**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1		1	1		4			7		1	5				5	2				2	2
	2	2		5	6		6			6		6	6				4	4				4	1
		3		3	3		3			3		3	3				3	3				5	5

Total Page faults = 13

d) **Four Frames**

i) **LRU**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1		1	6		6			6		6	5		5		5	2				2	1
	2	2		2	2		4			4		1	1		1		4	4				4	4
		3		3	3		3			3		3	3		3		3	3				3	3
				5	5		5			7		7	7		6		6	6				5	5

Total Page faults = 14

ii) **FIFO**

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

Total Page faults = 16

iii) **Optimal**

1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1		1	1		1			1			1				1	2					2
	2	2		2	6		6			6			6				4	4					4
		3		3	3		3			3			3				3	3					1
				5	5		4			7			5				5	5					5

Total Page faults = 11

Result

No. of Frames	LRU	FIFO	Optimal
1	24	24	24
2	19	20	16
3	15	17	13
4	14	16	11

****

3) Consider the following page reference string: 1, 2, 3, 4, 2, 1, 5, 6, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6. How many page faults would occur for the LRU, FIFO, LFU and Optimal page replacement algorithms, assuming two and five frames. **(8)** **(May 2014)**

****

a) **Two Frames**

i) **LRU(Least Recently Used)**

- Replace the page that **has not been** used for the longest period of time. **(past)**

1	2	3	4	2	1	5	6	1	2	3	7	6	3	2	1	2	3	6
1	1	3	3	2	2	5	5	1	1	3	3	6	6	2	2		2	6
	2	2	4	4	1	1	6	6	2	2	7	7	3	3	1		3	3

Total Page faults = 18

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

1	2	3	4	2	1	5	6	1	2	3	7	6	3	2	1	2	3	6
1	1	3	3	2	2	5	5	1	1	3	3	6	6	2	2		3	3
	2	2	4	4	1	1	6	6	2	2	7	7	3	3	1		1	6

Total Page faults = 18

iii) **Optimal**

- Replace the page that **will not be used** for the longest period of time.**(future)**

1	2	3	4	2	1	5	6	1	2	3	7	6	3	2	1	2	3	6
1	1	3	4		1	1	1		2	3	7		3	3	1		3	3
	2	2	2		2	5	6		6	6	6		6	2	2		2	6

Total Page faults = 15

b) **Five Frames**

i) **LRU**

1	2	3	4	2	1	5	6	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1			1	1							
	2	2	2			2	2			2	2							
		3	3			3	6			6	6							
			4			4	4			3	3							
						5	5			5	7							

Total Page faults = 8

ii) **FIFO**

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

Total Page faults = 10

iii) **Optimal**

1	2	3	4	2	1	5	6	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1				1							
	2	2	2			2	2				2							
		3	3			3	3				3							
			4			4	4				7							
						5	6				6							

Total Page faults = 7

Result

No. of Frames	LRU	FIFO	Optimal
2	18	18	15
5	8	10	7

****

4) Consider the following page reference string: 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6. How many page faults would occur for the LRU, FIFO and Optimal replacement algorithms, assuming one, two, three, four, five, six and seven frames? Remember all frames are initially empty, so your first unique pages will cost one fault each. **(12)** **(May 2009, 2012)**

****

a) **One Frame**

As there is no page referenced more than once, all the LRU, FIFO and Optimal algorithms will result in **20 page faults.** 1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

b) **Two Frames**

i) **LRU(Least Recently Used)**

- Replace the page that **has not been** used for the longest period of time. **(past)**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	3	3	2	2	5	5	2	2		2	7	7	3	3	1		3	3
	2	2	4	4	1	1	6	6	1		3	3	6	6	2	2		2	6

Total Page faults = 18

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

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

Total Page faults = 18

iii) **Optimal**

- Replace the page that **will not be used** for the longest period of time.**(future)**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	3	4		1	5	6		1		3	3	3		3	1		1	6
	2	2	2		2	2	2		2		2	7	6		2	2		3	3

Total Page faults = 15

c) **Three Frames**

i) **LRU**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	4		4	5	5	5	1		1	7	7		2	2			2
	2	2	2		2	2	6	6	6		3	3	3		3	3			3
		3	3		1	1	1	2	2		2	2	6		6	1			6

Total Page faults = 15

ii) **FIFO**

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

Total Page faults = 16

iii) **Optimal**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1				3	3			3	3			6
	2	2	2			2	2				2	7			2	2			2
		3	4			5	6				6	6			6	1			1

Total Page faults = 11

d) **Four Frames**

i) **LRU**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1				1	1	6			6			
	2	2	2			2	2				2	2	2			2			
		3	3			5	5				3	3	3			3			
			4			4	6				6	7	7			1			

Total Page faults = 10

ii) **FIFO**

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

Total Page faults = 14

iii) **Optimal**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					7				1			
	2	2	2			2	2					2				2			
		3	3			3	3					3				3			
			4			5	6					6				6			

Total Page faults = 8

e) **Five Frames**

i) **LRU**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1				1	1							
	2	2	2			2	2				2	2							
		3	3			3	6				6	6							
			4			4	4				3	3							
						5	5				5	7							

Total Page faults = 8

ii) **FIFO**

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

Total Page faults = 10

iii) **Optimal**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					1							
	2	2	2			2	2					2							
		3	3			3	3					3							
			4			4	4					7							
						5	6					6							

Total Page faults = 7

f) **Six Frames**

i) **LRU**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					1							
	2	2	2			2	2					2							
		3	3			3	3					3							
			4			4	4					7							
						5	5					5							
							6					6							

Total Page faults = 7

ii) **FIFO**

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

Total Page faults = 10

iii) **Optimal**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					1							
	2	2	2			2	2					2							
		3	3			3	3					3							
			4			4	4					7							
						5	5					5							
							6					6							

Total Page faults = 7

g) **Seven Frames**

i) **LRU**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					1							
	2	2	2			2	2					2							
		3	3			3	3					3							
			4			4	4					4							
						5	5					5							
							6					6							
												7							

Total Page faults = 7

ii) **FIFO**

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

Total Page faults = 7

iii) **Optimal**

1	2	3	4	2	1	5	6	2	1	2	3	7	6	3	2	1	2	3	6
1	1	1	1			1	1					1							
	2	2	2			2	2					2							
		3	3			3	3					3							
			4			4	4					4							
						5	5					5							
							6					6							
												7							

Total Page faults = 7

Result

No. of Frames	LRU	FIFO	Optimal
1	20	20	20
2	18	18	15
3	15	16	11
4	10	14	8
5	8	10	7
6	7	10	7
7	7	7	7

****

5) Consider the following page reference string: 2, 3, 4, 5, 3, 2, 6, 7, 3, 2, 3, 4, 1, 7, 1, 4, 3, 2, 3, 4, 7. Calculate the number of page faults would occur for the LRU, FIFO and Optimal page replacement algorithms with frame size of 3 and 5. **(16)** **(Nov 2006)**

****

a) Three Frames

i) **LRU(Least Recently Used)**

- Replace the page that **has not been** used for the longest period of time. **(past)**

2	3	4	5	3	2	6	7	3	2	3	4	1	7	1	4	3	2	3	4	7
2	2	2	5		5	6	6	6	2		2	1	1			1	2			7
	3	3	3		3	3	7	7	7		4	4	4			4	4			4
		4	4		2	2	2	3	3		3	3	7			3	3			3

Total Page faults = 15

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

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

Total Page faults = 16

iii) Optimal

- Replace the page that **will not be used** for the longest period of time.**(future)**

2	3	4	5	3	2	6	7	3	2	3	4	1	7	1	4	3	2	3	4	7
2	2	2	2			2	2				4	4				4	4			7
	3	3	3			3	3				3	1				3	3			3
		4	5			6	7				7	7				7	2			2

Total Page faults = 11

b) Five Frames

i) LRU

2	3	4	5	3	2	6	7	3	2	3	4	1	7	1	4	3	2	3	4	7
2	2	2	2			2	2				2	2								
	3	3	3			3	3				3	3								
		4	4			4	7				7	7								
			5			5	5				4	4								
						6	6				6	1								

Total Page faults = 8

ii) FIFO

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

Total Page faults = 8

iii) Optimal

2	3	4	5	3	2	6	7	3	2	3	4	1	7	1	4	3	2	3	4	7
2	2	2	2			2	2					2								
	3	3	3			3	3					3								
		4	4			4	4					4								
			5			5	7					7								
						6	6					1								

Total Page faults = 7

Result

No. of Frames	LRU	FIFO	Optimal
3	15	16	11
5	8	8	7

****

6) Consider the following page reference string: 2, 3, 4, 2, 1, 5, 6, 4, 1, 2, 3, 7, 6, 3, 2, 1. Calculate the number of page faults would occur for the LRU, FIFO and Optimal page replacement algorithms with frame size of 4 and 5. **(16)** **(Nov 2008)**

****

a) Four Frames

i) LRU(Least Recently Used)

- Replace the page that **has not been** used for the longest period of time. **(past)**

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	2	2	4		4	4	7	7			1
	3	3		3	5	5	5		2	2	2	2			2
		4		4	4	6	6		6	3	3	3			3
				1	1	1	1		1	1	1	6			6

Total Page faults = 12

ii) **FIFO(First In First Out)**

- The page which comes first is replaced first.

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	5	5			5	5	7				7
	3	3		3	3	6			6	6	6				1
		4		4	4	4			2	2	2				2
				1	1	1			1	3	3				3

Total Page faults = 10

iii) **Optimal**

- Replace the page that **will not be used** for the longest period of time.**(future)**

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	2	2				2	2				1
	3	3		3	5	6				6	6				6
		4		4	4	4				3	3				3
				1	1	1				1	7				7

Total Page faults = 9

b) **Five Frames**

i) **LRU**

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	2	2				2	2	2			
	3	3		3	3	6				6	7	7			
		4		4	4	4				4	4	6			
				1	1	1				1	1	1			
					5	5				3	3	3			

Total Page faults = 9

ii) **FIFO**

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	2	6			6	6	6				6
	3	3		3	3	3			2	2	2				2
		4		4	4	4			4	3	3				3
				1	1	1			1	1	7				7
					5	5			5	5	5				1

Total Page faults = 10

iii) **Optimal**

2	3	4	2	1	5	6	4	1	2	3	7	6	3	2	1
2	2	2		2	2	2					2				
	3	3		3	3	3					3				
		4		4	4	4					7				
				1	1	1					1				
					5	6					6				

Total Page faults = 7

Result

No. of Frames	LRU	FIFO	Optimal
4	12	10	9
5	9	10	7

UNIT 3 – Segmentation

1) Consider the following segment table:

Segment	Base	Length
0	219	600
1	2300	14
2	90	100
3	1327	580
4	1952	96

What are the physical addresses for the following logical addresses? **(8)** **(May 2012)**

a) 0, 430 **b)** 1, 10 **c)** 2, 500 **d)** 3, 400

****

a) 0, 430

Logical address (s, d) is (0, 430)
Segment number(s) = 0
Offset(d) = 430
Sement length (or) limit = 600
Offset 430 < Segment limit 600
Physical address = Offset + Segment Base
= 430 + 219
= **649**

b) 1, 10

Logical address (s, d) is (1, 10)
Segment number(s) = 1
Offset(d) = 10
Sement length (or) limit = 14
Offset 10 < Segment limit 14
Physical address = Offset + Segment Base
= 10 + 2300
= **2310**

c) 2, 500

Logical address (s, d) is (2, 500)
Segment number(s) = 2
Offset(d) = 500
Sement length (or) limit = 100
Offset 500 > Segment limit 100
Segment fault, illegal address – trap to operating system is sent.

d) 3, 400

Logical address (s, d) is (3, 400)
Segment number(s) = 3
Offset(d) = 400
Sement length (or) limit = 580
Offset 400 < Segment limit 580
Physical address = Offset + Segment Base
= 400 + 1327
= **1727**

Result

S. No.	Logical Address	Physical Address
1	(0, 430)	649
2	(1, 10)	2310
3	(2, 500)	Segment fault
4	(3, 400)	1727

2) On a system using simple segmentation, compute the physical address for each of the logical addresses. Logical address is given in the following segment table. If the address generates a segment fault, indicate so.

Segment	Base	Length
0	330	124
1	876	211
2	111	99
3	498	302

a) 0, 99 b) 2, 78 c) 1, 265 d) 3, 222 e) 0, 111

****

a) 0, 99

Logical address (s, d) is (0, 99)
Segment number(s) = 0
Offset(d) = 99
Sement length (or) limit = 124
Offset 99 < Segment limit 124
Physical address = Offset + Segment Base
= 99 + 330 = **429**

b) 2, 78

Logical address (s, d) is (2, 78)
Segment number(s) = 2
Offset(d) = 78
Sement length (or) limit = 99
Offset 78 < Segment limit 99
Physical address = Offset + Segment Base
= 78 + 111 = **189**

c) 1, 265

Logical address (s, d) is (1, 265)
Segment number(s) = 1
Offset(d) = 265
Sement length (or) limit = 211
Offset 265 > Segment limit 211
Segment fault, illegal address – trap to operating system is sent.

d) 3, 222

Logical address (s, d) is (3, 222)
Segment number(s) = 3
Offset(d) = 222
Sement length (or) limit = 302
Offset 222 < Segment limit 302
Physical address = Offset + Segment Base
= 222 + 498 = **720**

e) 0, 111

Logical address (s, d) is (0, 111)
Segment number(s) = 0
Offset(d) = 111
Sement length (or) limit = 124
Offset 111 < Segment limit 124
Physical address = Offset + Segment Base
= 111 + 330 = **441**

Result

S. No.	Logical Address	Physical Address
1	(0, 99)	429
2	(2, 78)	189
3	(1, 265)	Segment fault
4	(3, 222)	720
5	(0, 111)	441

UNIT 3 – Dynamic Storage Allocation Problem

1) Given memory partitions of 100 KB, 500 KB, 200 KB and 600 KB(in order). Show with neat sketch how would each of the first-fit, best-fit and worst-fit algorithms place processes of 412 KB, 317 KB, 112 KB and 326 KB(in order). Which algorithm is most efficient in memory allocation? **(16)** **(Nov 2010)**

****

- Initially, all memory is available for user processes and one **large block of available memory** is considered as a **hole.**
- The memory blocks available comprise a set of holes of various sizes scattered throughout memory.
- **Dynamic storage allocation problem** concerns how to satisfy a request of size *n* from a list of free holes. The set of holes is searched to determine which hole is best to allocate. The first-fit, best-fit, and worst-fit are the most commonly used strategies to select a free hole from the set of available holes.

  First fit: Allocate the first hole that is big enough.
  Best fit: Allocate the smallest hole that is big enough.
  Worst fit: Allocate the largest hole.

- Both first fit and best fit are better than worst fit in terms of decreasing time and storage utilization.
- Both the **first-fit and best-fit** strategies for memory allocation suffer from **external fragmentation.**
- **External fragmentation** exists when there is **enough total memory space** to satisfy a request **but** the available spaces are **not contiguous:** storage is fragmented into a large number of small holes.
- The **solutions** to **external** fragmentation problem are **compaction, paging and segmentation**
- The **memory allocated** to a process may be **slightly larger** than the requested memory. It results in **internal fragmentation**– unused memory that is internal to a partition.

a) First Fit algorithm:

Allocates the first hole that is big enough.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
412	500	88
317	600	283
112	200	88
326	Not Available	-

b) Best Fit algorithm:

Allocates the smallest hole that is big enough.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
412	500	88
317	600	283
112	200	88
326	Not Available	-

c) Worst Fit algorithm:

Allocates the largest hole.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
412	600	188
317	500	183
112	200	88
326	Not Available	-

Best fit algorithm is most efficient in memory allocation. But for these given memory partitions and processes request, all the three algorithms(first fit, best fit and worst fit) results in the same wastage of memory.

2) Given memory partitions of 100 KB, 500 KB, 200 KB, 300 KB and 600 KB(in order). How would each of the first-fit, best-fit and worst-fit algorithms place processes of 212 KB, 417 KB, 112 KB and 426 KB(in order). Which algorithm makes the most efficient use of memory? **(10)** **(May 2007)**

****

- Initially, all memory is available for user processes and one **large block of available memory** is considered as a **hole.**
- The memory blocks available comprise a set of holes of various sizes scattered throughout memory.
- **Dynamic storage allocation problem** concerns how to satisfy a request of size *n* from a list of free holes. The set of holes is searched to determine which hole is best to allocate. The first-fit, best-fit, and worst-fit are the most commonly used strategies to select a free hole from the set of available holes.
  - **First fit:** Allocate the first hole that is big enough.
  - **Best fit:** Allocate the smallest hole that is big enough.
  - **Worst fit:** Allocate the largest hole.
- Both first fit and best fit are better than worst fit in terms of decreasing time and storage utilization.
- Both the **first-fit and best-fit** strategies for memory allocation suffer from **external fragmentation.**
- **External fragmentation** exists when there is **enough total memory space** to satisfy a request **but** the available spaces are **not contiguous:** storage is fragmented into a large number of small holes.
- The **solutions** to **external** fragmentation problem are **compaction, paging and segmentation**
- The **memory allocated** to a process may be **slightly larger** than the requested memory. It results in **internal fragmentation**– unused memory that is internal to a partition.

a) **First Fit algorithm:**

Allocates the first hole that is big enough.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
212	500	288
417	600	183
112	200	88
426	Not Available	-

b) **Best Fit algorithm:**

Allocates the smallest hole that is big enough.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
212	300	88
417	500	83
112	200	88
426	600	174

c) **Worst Fit algorithm:**

Allocates the largest hole.

Processes(kb)	Memory Allocation(kb)	Wastage of Memory(kb)
412	600	388
317	500	83
112	300	188
326	Not Available	-

Since the wastage (or) left over memory is less in best fit algorithm, Best fit **algorithm** uses the memory most efficiently.