University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Science

CS 162 I. Stoica
Spring 2011

FINAL EXAM
Friday, May 13, 2011

INSTRUCTIONS—READ THEM NOW! This examination is CLOSED BOOK/CLOSED NOTES. There is no need for calculations, and so you will not require a calculator, Palm Pilot, laptop computer, or other calculation aid. Please put them away. You MAY use one 8.5" by 11" double-sided crib sheet, as densely packed with notes, formulas, and diagrams as you wish. The examination has been designed for 80 minutes/80 points (1 point = 1 minute, so pace yourself accordingly). All work should be done on the attached pages.

In general, if something is unclear, write down your assumptions as part of your answer. If your assumptions are reasonable, we will endeavor to grade the question based on them. If necessary, of course, you may raise your hand, and a TA or the instructor will come to you. Please try not to disturb the students taking the examination around you.

_____ SID: _____
(Signature)

_____ Discussion Section (Day/Time): _____
(Name—Please Print!)

QUESTION	POINTS ASSIGNED	POINTS OBTAINED
1	20	
2	15	
3	10	
4	20	
5	10	
6	10	
7	15	
TOTAL	100	

Question 1. *Miscellaneous (20 points)*
For each of the following statements, indicate whether the statement is True or False, and provide a very short explanation of your selection (2 points each).

a. Starvation implies deadlock. T **F**
Rationale:

With starvation at least one job makes progress. In case of deadlock, no job makes progress.

b. A two-way set associative cache has a higher hit rate than a direct cache of same size. **T** F
Rationale:

Any access pattern that fits in a direct cache also fits in a two-way set associative cache of the same size.

We also considered **F** as there are access patterns for which both the two-way associative cache and a direct cache have the **same** hit rate.

c. With datagram packet switching each packet is individually routed. **T** F
Rationale:

Each router independently forwards each packet based on its destination address. If the conditions in the network change (i.e., failures, congestions), a router can forward packets for the same destination along different paths.

d. A store-and-forward router starts forwarding the packet as soon as it gets packet's header. T **F**
Rationale:

No. The router waits to receive the **entire** packet before forwarding it, hence the store-and-forward name. This allows a router to enque packets if the output link is congested, and drop a packet if it has been corrupted.

e. In a layered architecture, a layer can only use the service provided by the layer below it. **T** F
Rationale:

A layer only uses the service provided by the layer below it.

f. Flow control ensures that the sender does not overwhelm the network. T **F**
Rationale:

No. Flow control ensures that the sender does not overwhelm the receiver. (The congestion control ensures that the sender does not overwhelm the network.)

g. RPC provides semantics *identical* to a local procedure call. T **F**
Rationale:

In the presence of failures, RPC cannot provide the same semantics as a local procedure call as it cannot completely mask the failure, i.e., if a server is unreachable it needs to return a special error (see slides 17-24 in lecture 22).

h. Public key cryptography requires participants to distribute a secrete keys. T **F**
Rationale:

No. With public key cryptography a participant does not need to distribute her private key; it needs to distribute only the public key (which is not a secret).

i. Statistical multiplexing increases resource utilization. **T** F
Rationale:

Statistical multiplexing allows multiple flows/processes sharing the same resources even though the sum of their peak demands exceeds resource capacity. This is because with a high probability their peak demands do not coincide.

j. A buffer overflow attack is caused by overwriting the stack. **T** F
Rationale:

In general, with a buffer overflow attack, the attacker overwrites the stack to get control over the victim, by causing the program to jump to the infected code upon returning from a procedure call.

F was accepted if rationale talks about other types of buffer overflow such as heap overflow etc.

Question 2. *Synchronization (15 points)*

Consider a deposit of $100 and a withdrawal of $50 from account A. Initially, account A holds $1000. Assume the two operations are implemented by two threads, and assume that each instruction is atomic.

```
Thread1:              Thread2:
    t1 = A;               t2 = A;
    t1 = t1 + $100;       t2 = t2 – 50;
    A = t1;               A = t2;
```

a) What are the possible outcomes of the above transfer? For each outcome give a possible execution of the threads leading to that outcome.

Case 1: A = 1050

```
t1 = A;
t1 = t1 + 100;
A = t1;
                      t2 = A;
                      t2 = t2 - 50;
                      A = t2
```

Case 2: A = 950 (Thread1's update is lost)

```
t1 = A;
t1 = t1 + 100;
                      t2 = A;
A = t1;
                      t2 = t2 - 50;
                      A = t2
```

Case 3: A = 1000 (Thread2's update is lost)

```
                      t2 = A;
t1 = A;
                      t2 = t2 - 50;
                      A = t2
t1 = t1 + 100;
A = t1;
```

b) Assume Thread1 repeats the deposit operation 3 times and Thread2 executes the withdraw operation 2 times (i.e., there are three deposits of $100 each, and 2 withdrawals of $50 each). What are the potential outcomes?

900, 1000, 1050, 1100, 1150, 1200, 1250, 1300

c) Give a simple solution to avoid incorrect outcomes.

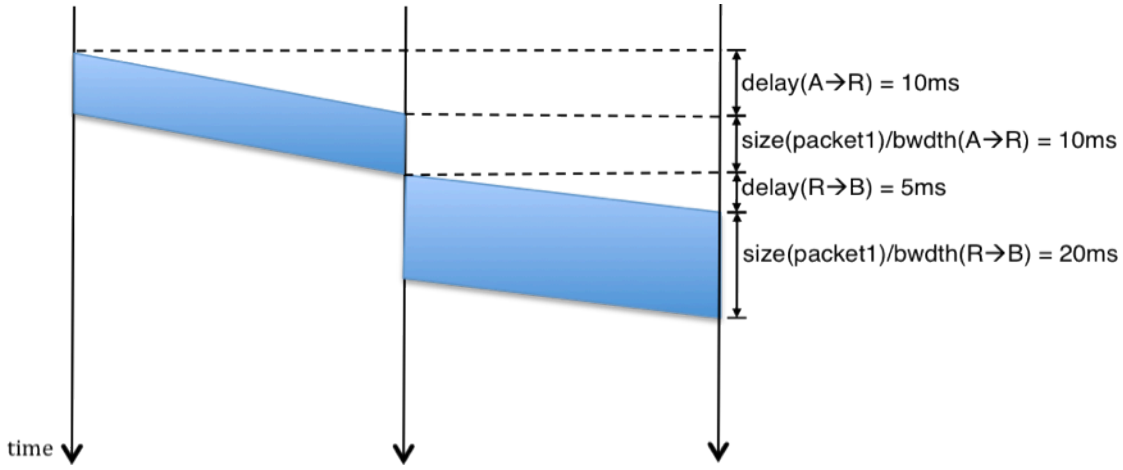
Use a lock to access and modify variable A. Each thread acquires lock before starting and releases after finishing (i.e., after writing back A).

Question 3. *Packet transmission (10 points)*

Hosts A and B are connected to each other via router R. R is a store-and-forward router. The bandwidth from A to R is 2Mbps, and the bandwidth from R to B is 1Mbps. The latency of link A-R is 10ms and the latency if link R-B is 5ms. Assume A sends a packet of 2500 bytes to R. Unless otherwise specified, there are no other packets in the network, and the *arrival time* of the packet is the time the receiver gets the *last* bit of the packet.

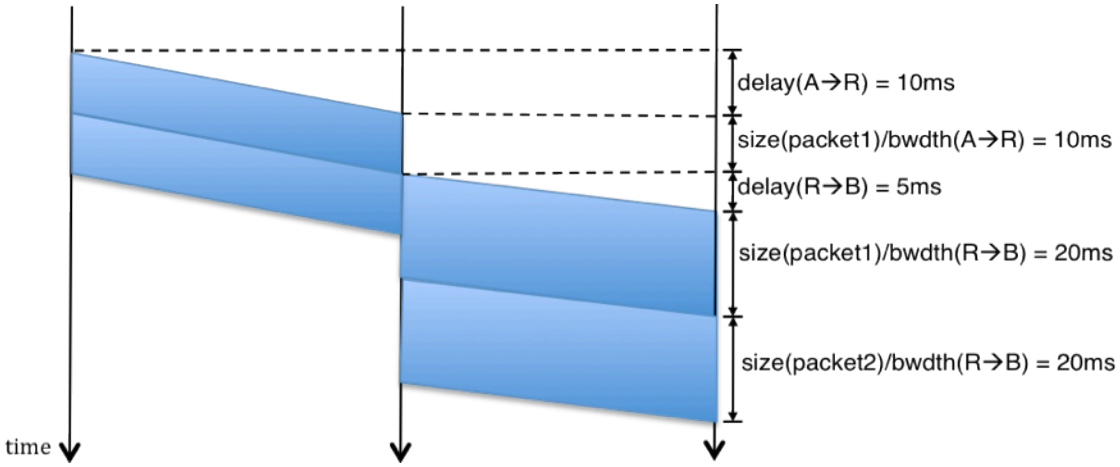
a) Assuming A starts sending the packet at time t=0 (i.e., the first bit is sent at time t=0), what is the arrival time of the packet at B?

45ms



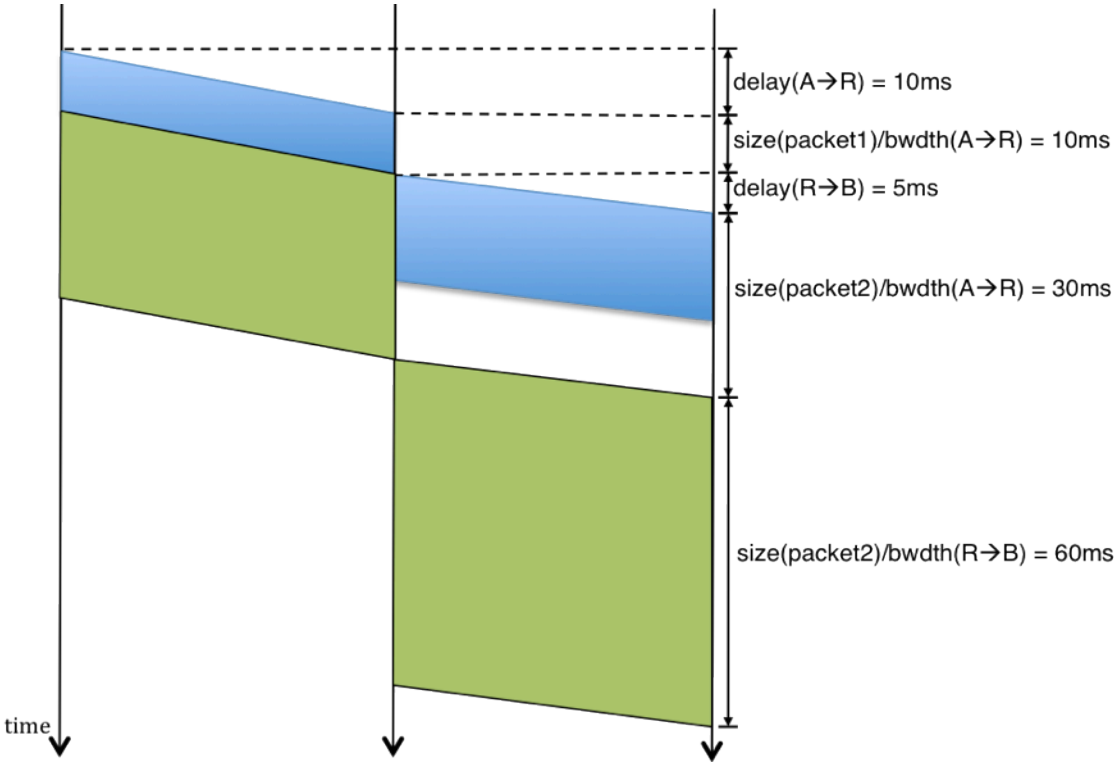
b) Assume A sends a second packet of same size right after the first one. What is the arrival time of the 2nd packet at B?

65ms



c) Repeat (b) if the size of the 2nd packet is 7500 bytes (i.e., 60 Kbits).

115ms



d) Repeat (b) if the size of the 2nd packet is x bytes, i.e., give a formula for the arrival time of 2nd packet as a function of 2nd packet size x. (This is a generalization of (b).)

Let x = size(packet2). Then, by comparing figures at points (b) and (c) you can generalize to

arrival_time(packet2) = delay(A,R) + size(packet1)/bwdth(A,R) + delay(R,B) + x/bwdth(R,B) + max(x/bwdth(A, R), size(packet1)/bwdth(R,B))

or numerically

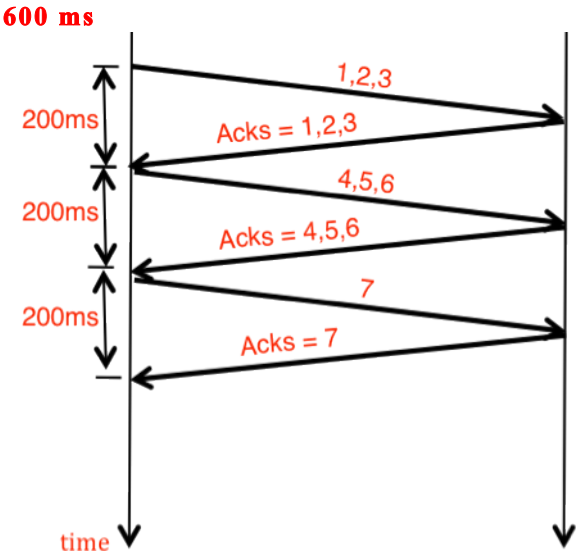
arrival_time(packet2) = 25ms + x/1Mbps + max(x/2Mbps, 20ms)

Question 4. *Flow control (20 points)*

Assume two end-hosts using the sliding window protocol to implement flow control, and Go-back-n to implement reliability. Assume sender sends 7 packets. The window size at the receiver is 3 packets, the round-trip time is 200ms, and the retransmission timeout is 500ms. The transmission time of the packet is negligible, i.e., assume the size of a packet is 0. The time to send all packets is the interval between the time the sender sends the first packet and the time the sender receives the ack from the last packet.

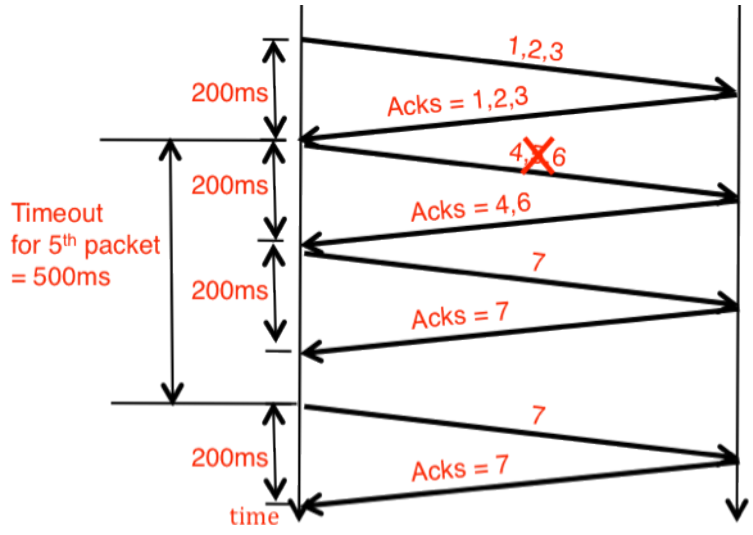
a) How long does it take to send all packets, assuming no losses? Draw the time diagram.

600 ms



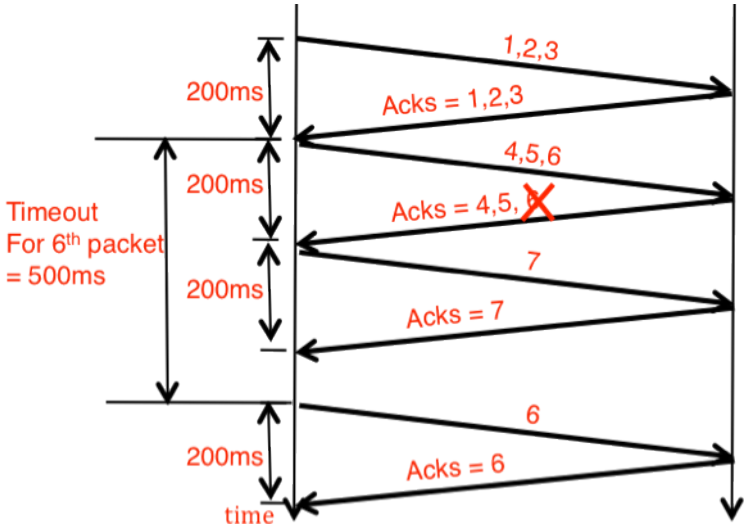
b) How long does it take to send all packets assuming the 5th packet is lost? Draw the time diagram.

900 ms

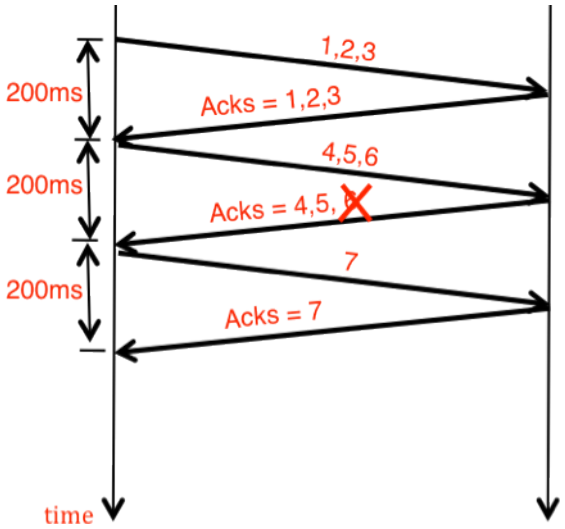


c) How long does it take to send all packets assuming the ack of the 6th packet is lost? Draw the time diagram.

900 ms



If the acks are cumulative then the answer is **600ms** (this answer was also considered correct):



d) Assuming that exactly one packet is lost, what is the *minimum* time it takes to send all packets? Draw the time diagram.

900 ms; see (b) for an example. Or 600ms if assuming cumulative acks; see (b) for an example.

Question 5. *Transactions (10 points)*
Consider the two transactions below. Assume each instruction (i.e., read, write, addition, subtraction) takes one time unit, and acquiring/releasing a lock takes zero time units. Once a transaction acquires a shared lock it cannot upgrade it to an exclusive lock, and once a transaction acquires an exclusive lock it cannot downgrade it to a shared lock..

Transaction1	Transaction2
R(A); A = A + 100; W(A); R(B); B = B – 100; W(B);	R(A); A = A – 50; W(A)

a) What is the minimum possible execution time taken by both transactions when using 2PL (2 phase locking)? Show a schedule that achieves the minimum time. The diagram below shows the first several instructions executed by each transaction for such a schedule. Note that Transaction2 is not getting the lock when requesting it, instead, Transaction2 needs to wait for the lock to be released by Transaction1.

6 time units

Transaction1	Transaction2
Lock_X(A) <granted>	
R(A)	Lock_X(A)
A = A + 100	
W(A)	
Lock_X(B)<granted>	
Unlock(A)	<Lock_X(A)granted>
R(B)	R(A)
B = B-100	A = A – 50;
W(B)	W(A)
Unlock(B)	Unlock(A)

b) What is the minimum possible execution time taken by both transactions when using **strict** 2PL? Show a schedule that achieves the minimum time.

9 time units.

Transaction1	Transaction2
Lock_X(A) <granted>	
R(A)	Lock_X(A)
A = A + 100	
W(A)	
Lock_X(B)<granted>	
R(B)	
B = B-100	
W(B)	
Unlock(A)	<Lock_X(A)granted>
Unlock(B)	R(A)
	A = A – 50;
	W(A)
	Unlock(A)

c) Repeat question (a), assuming Transaction1 is replaced with transaction:

Transaction1'
R(A);
R(B);
A = A + 100;
B = B – 100;
W(A);
W(B);

8 time units

Transaction1'	Transaction2
Lock_X(A) <granted>	
R(A)	Lock_X(A)
Lock_X(B) <granted>	
R(B)	
A = A + 100	
B = B-100	
W(A)	
Unlock(A)	<Lock_X(A)granted>
W(B)	R(A)
Unlock(B)	A = A – 50;
	W(A)
	Unlock(A)

d) Now assume Transaction2 is replaced with transaction:

Transaction2'
R(B);
B = B + 50;
W(B);
R(A);
A = A – 50;
W(A);

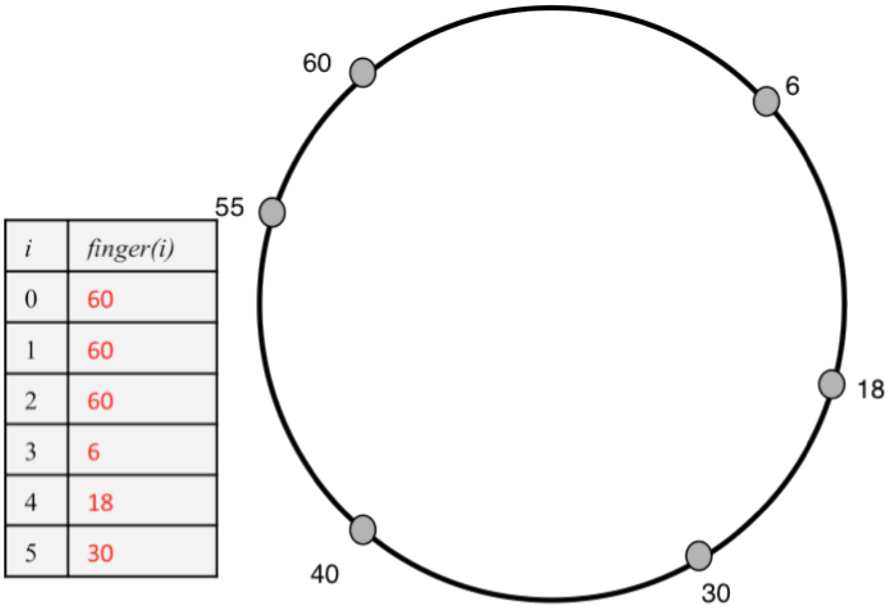
Is it possible to get into deadlock by executing Transaction1 and Transaction2' ? If not, use no more than two sentences to explain why not? If yes, give an example of schedule leading to deadlock.

Yes.

Transaction1	Transaction2'
Lock_X(A) <granted>	Lock_X(B) <granted>
R(A)	R(B)
A = A + 100	B = B + 50
W(A)	W(B)
Lock_X(B)	Lock_X(A)
deadlock	

Question 6. *Chord (10 points)*

Consider the Chord ring in the figure below, where the maximum ID is 63. Each node maintains its successor as well as the finger list.



i	*finger(i)*
0	60
1	60
2	60
3	6
4	18
5	30

i	*finger(i)*
0	30
1	30
2	30
3	30
4	40
5	55

a) Fill in the finger tables for nodes 18 and 55, respectively. Recall that the finger i of node N is the smallest node M that follows $(N + 2^i)$ mod 64 on the ring.

See above.

b) Show the path followed by a lookup for ID=57 starting at node 6. List the nodes touched by this lookup.

6 → 40 → 55

c) How does the finger table of node 55 changes if two new nodes with IDs 58 and 2, respectively, join the ring.

0 → 58
1 → 58
…
3 → 2

d) Assume you use this Chord ring to implement a DHT (distributed hash table). Assume that the following data items are inserted (14, data1), (21, data2), (45, data3). Specify at which nodes are these data items stored.

(14, data1) → 18; (21, data2) → 30; (45, data3) → 55

e) Assume that you want to provide reliability in the face of one node failure, i.e., if a node fails you do not want to lose data. Describe one solution to this problem using no more than three sentences.

Two solutions:

(1) You can replicate data at the successor node. For instance, you can store (14, data1) not only at node 18, but also at node 30.

(2) You can use a function to map the data ID to another ID', and then store the data both under the new ID. For instance, in our case, you can use ID' = ID + 32, i.e., you can store (14, data1) not only at node 18, but also at node 55 (as ID' = 14 + 32 = 46, and 46 maps to node 55).

Question 7. *Page Replacement (15 points)*

Consider four frames and the following page access pattern: A, B, A, C, D, E, F, A, B, C, F, A, E, F

a) Assume we employ the FIFO replacement policy. Show the content of the frames at every access (the first three accesses are filled for you). Give the number of page faults.

Frame	A	B	A	C	D	E	F	A	B	C	F	A	E	F
1	A		A			E				C				
2		B					F				F		E	
3				C				A				A		F
4					D				B					F

Page faults = 11

b) Assume we employ the LRU replacement policy. Show the content of the frames at every access. Give the number of page faults.

Frame	A	B	A	C	D	E	F	A	B	C	F	A	E	F
1	A		A				F				F			F
2		B				E				C				
3				C				A				A		
4					D				B				E	

Page faults = 10

c) Assume we employ the second-chance replacement policy. Show the content of the frames at every access. Give the number of page faults.

Frame	A	B	A	C	D	E	F	A	B	C	F	A	E	F
1	A		A					A				A		
2		B				E				C				
3				C			F				F			F
4					D				B				E	

Page faults = 9