76

جامعة بيرزيت

BIRZEIT UNIVERSITY

Computer Science Department

NAME : ~~Saed Daneyn~~ NUMBER : [redacted]

Comp 431 Exam # 2 11/12/2004

[1] *Fill in blanks:*

- The average internal fragmentation is paging system is

~~external~~ external fragmentation

- The best and fastest implementation of a page table is

with Rigisters

- The advantage of using paging memory management

1- it redyse or There will not be an external ~~fragment~~ fragmentatio

2- Sharing pages

- In MVT, the poorest allocation method is

worst fit

- The solution for external fragmentation in MVT is

The compaction

- If a process most of the time is busy swapping pages in & out of memory, then this is called

The page fault and need a high ~~...~~ utilization of CPU to solve

- Starting execution with zero pages in memory is called

it is not in accutal worke but it a افتراضي

- The best page replacement algorithm which gives the minimum page faults is

optimal Algrithm

- In a paging system, if LA = u , page size = s , then

page No., p = u / s

page offset, d = u % s

if The Page table Entry is full or Empty

[2] (a) Define briefly the following bit types associated with the page table.

(i) legal / illegal Bit

its a bit add to the page table to ~~make~~ indicate if The page hass a Legal or not it hast ~~add~~ data or full page ~~empty~~ ~~region~~ rigon in The page table. 3 hass a page the valid bit is set to $\boxed{1}$. other wise it set to 0

(ii) valid / invalid Bit

it a bit that hass ~~been~~ been add to the page table to ~~inactal~~ indecate if The page is in the ~~memory~~ memory or not

(iii) protection Bit

it is a bit That add to Page table to indicate if The page is read only ~~but~~ That we cant modfied it or read-write page so we can modfay The page

(iv) reference Bit

its a bit that add to page table to indicate if the page hass been refrenced (executed or not)

(b) (i) What is page fault? The persantag ~~There~~ of The time The page is not There in memory

Riyam

The page in this

(ii) What are the steps executed by the page fault service routine?

1 to check if The memory is full or not is not it swap i is not ~~to~~ chose an victom by some Algorithe

2 - ~~swap out~~ to chick if The victom page has been modified or not if it modfied it ~~swapit~~ swop it to af ~~memory~~ memory and restor it if not it replace it or (swap in The needed pages in this page (3) compute The phiscal address of This page and get instraction to make an execute for it

[3] In a computer system with paging memory management, if the page size is $\boxed{8 \text{ KB}}$, memory contains 128000 frames. If the maximum program can be executed in this machine contains 2048 pages, then compute:

- The No. of bits in the LA

The size of program is 2048 pages $= 2^{11}$ = 11 bit = $p = 11$

But Th page size = $8k = 8 \times 2^{10} = 2^{13}$ $= d = 13$

LA = $11 + 13 = 24$

11	13

- The No. of bits in the PA

The mem size contain $128000 = 2^{17}$ $= 17$ $p = 17$ Bit

Th page size $= 2$ ~~048~~ 8 KB $= 2^{13} =$ $d = 13$

PA = $17 + 13 = 30$ bit

- The physical memory size $2^{17} \times 2^{13} = 2^{30}$

Th ph ~~size~~ size $= 2^{17} \times 2^{13} = 2^{30}$

" $128000 \times 2^{12} =$

$2^{17} \times 2^{13} = 2^{30}$

- The page table size

entry 17 Bit need 3 byte so

The page table entry $= 2048 = 2^{11}$

Than Th page size $= 2048 \times 3 = 6144$ Byte

- The maximum program size can be executed

The Maxim program size $= 2048 \times 8$ KB

$= 2048 \times 2^{13}$

$= 2^{11} \times 2^{13}$

$= 2^{24}$ byte

[4] (a) An OS supports a demand paging system, Given:

page fault rate p
memory access m
page fault service routine time t

Compute the EAT?

swap in and swap out

$$(1-p)(m) + p(t + 2t + 2m)$$

mery accesses   swap in only   memory accesses

(b) Given that:

Memory access is 100 micS
Page transfer time 5 milS 5000 micro
40% of the time the page is modified
If the EAT = 1500 micS,

Compute the page fault rate p.

memory access   page not modified   page modified

$$EAT = (1-p)(m) + p[.6\ \text{swap in} + .4 \times \text{swap in and swap out} + \text{memory access}]$$

$$1500 = (1-p)(100) + p[(.6 \times 5000) + (.4 \times 2 \times 5000) + 200]$$

$$1500 = 100 - 100p + p\{3000 + 4000 + 200\}$$

7200

$$1400 = -100p + 7200p$$

$$1600 = 7100p$$

$$p = \frac{1600}{7100} = .225$$

$$\approx .23$$

جامعة بيرزيت
BIRZEIT UNIVERSITY

Computer Science Department

$128 \times 2 = 256$

NAME : [redacted] NUMBER : [redacted]

Comp 431 Exam # 2 17/05/2006

[1] In a computer system with paging memory management, if the page size is 4 KB, memory contains 256000 frames. If the maximum program can be executed in this machine contains 8192 pages, then **compute** by showing your work:

$4 \times 2^{10} = 2^{12}$

- The No. of bits in the LA → 12

LA

p d 12

page size $4 \times 2^{10} = 2^{12}$

Pg num $= 8192 = 2^{13}$

$=$ LA $=$ 24 bit

25

8192

$256 \times 2 = 512$

$512 \times 2 = 1{,}024$

$1{,}024 \times 2 = 2048$

$2048 \times 2 = 4096$

$4096 \times 2 = 8192$

$256000 =$

250×2^{10}

256

- The No. of bits in the PA

Frame num $= 256000$

$\approx 2 \times 2^{10} = 2^{12}$

17 bit

6

- The size physical memory

256000×2^{12} ✓ $= 2$

$=$ # of frame × frame size

8192 / 256000

- The page table size

2^{12}

2^{7}

[2] (a) In executing a certain process, we have the following page references:

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

Compute the **page fault rate** for the following algorithms with **4** frames memory:

LRU replacement algorithm.

11 page fault

(b) In a demand paging system where page table is kept in registers. If servicing a page fault takes 1 millisecond if a free frame is available or the page is not modified and 2 millisecond if page is modified. If memory access is 100 microsecond, and 60% of the time the page is modified. If the **page fault** rate **p** = 30%. Compute the **EAT**.

$(1 - .3) * 100 + .3\left(\frac{60}{100} * 2 + 1 + 100\right)$

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

12 Page fault

[3] *Fill in blanks:*

- The average internal fragmentation ~~is~~ in paging system is ------ ~~[illegible]~~ 1/2 Page

- The advantage of using paging memory management ---Page sharing---

- In MVT, the best allocation algorithm is ------best fit------, and the worst is

 ------worst fit------

- If a process most of the time is busy swapping pages in & out of memory, then this is called ---thrashing---

- Starting execution with zero pages in memory is called ---- ~~Pag~~ pure demand paging

- In a paging system, if LA = u , page size = s , then

  page No., p = u - s

  page offset, d = s

- The bit that indicates if the page in memory ------valid / invalid bit------

- The bit that indicates if the page is changed ----Dirty bit------

- The bit that indicates if the page read or written ----refference bit------

- In a demand paging system where the degree of multiprogramming is 5 and the status of the system is measured, and it is as follows:

  CPU utilization 15%, Disk utilization 95%

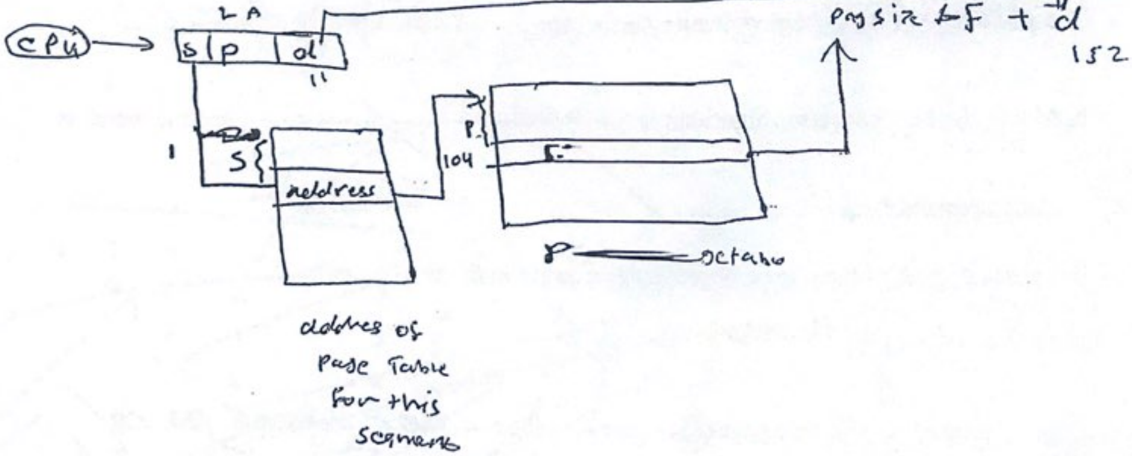
  Can we increase the degree of multiprogramming to improve the performance of the system,

  why ? Yes because ~~[illegible]~~ Disk have utilization 95% no (swapped in or out) ⇒ there exist frame in memory ⇒ we can increase the no of Task in memory

[4] A computer system with 20 bits logical add[redacted]ess and page size = 2048 Byte. A segmentation with paging (paging the segments) is used [redacted] with equal segments size of 128 kb. Given that the LA = $(6206A)_{16}$ in Hex.

Using diagram, Explain how the LA is translated into P[redacted].

Note: Show all numbers involved on the diagram. It is o[redacted]ay if you can't compute the PA precisely.



0.110 0010 0000 0110 1010
1 5 2

0110 0010 0000 0110 1010
1 10

0 1 1 0 0 0 1 0 0 0 0 0 0 1 1 0 1 0 1 0
s p
1 0 4
1 5 2

Byte. As
28 kb;

80/100

جامعة بيرزيت
BIRZEIT UNIVERSITY
Computer Science Department

NAME: Anas Salim NUMBER: 102 3032
Comp 431 Exam #2 17/12/2005

[1] (a) Select the correct answer from the list :

Parity Bit , v/i Bit , R/W Bit , Dirty Bit , Reference Bit , Flag Bit , legal/illegal Bit

R/W Bit indicates if the page is a read only ~~bit~~ page.

legal/illegal bit indicates if the page in the logical address space

v/i Bit indicates if the page in memory

Dirty Bit indicates if the page is changed

Reference Bit indicates if the page is being read or written

(b) In a demand paging system, assume the system status is as shown below with a very poor performance system. Briefly explain what is happening, and what do you suggest to improve it:

CPU utilization is 10% and paging disk utilization is 05%.

because the disk utilization is small so we assigned that no swapping happened mutch may be the # of framing is small and the degree of multiprogramming is small to improve the system; Increesing degree of multiprograming.

(c) Explain briefly the effect of selecting large and small page size

if the page ~~table~~ size Large then:.
~~the external fregmentation is large~~
~~we need more space to save it.~~
- the internal fregmentation is Large ~~and~~ (disadvanteg)
- the page table is small size (advanteg).

if the page ~~table~~ size small then:.
~~the space that we needed is small and~~
- the internal fregmentation small (advanteg)
- the page table size is large (disadvanteg).

[2] (a) In a **demand paging** system , Given that:

Memory access is 10 milS
Page transfer time 100 milS
30% of the time the page is modified
Page fault rate is 20%,

Compute the **EAT** no page fault page fault.

$$EAT = \{ \%80 (10) + 20\% (10 + 10 * \frac{30}{100})$$

$$= \frac{80}{10} + \frac{2}{10} (10 + 3)$$

$$= 8 + 2.6 \approx 10.6 \text{ mils.}$$

4

$$EAT = (1 - 20) * 10 + \frac{20}{100} (\frac{30}{100} * 1000 + 100 + 10)$$

(b) In a **demand paging** system with a **three-levels** page table stored in memory and a set of **associative registers.** Given:

memory access = 5 milS
hit ratio = 95%.

4 (Memory access)

Compute the **EAT** ignoring the lookup time in the associative registers.

$$EAT = (\frac{95}{100} * 5) + (0.05 * 4 * 5)$$

$$= 4.75 + 1 = 5.75 \text{ mils.}$$

$$95 * 5 + 0.05 (4 * 5)$$

PA = 28 bit

2^3 basize

(c) If the **physical address** contains 28 bits with page size is 8KB, compute **memory** size.

$$\text{page size} = 8 KB = 2^3 * 2^{10} = 2^{13}$$

so d = 13 bit

$$28 + 13 = 41 \text{ bit}$$

$$\text{memory size} = 2^{41} \text{ byte.}$$

$2^{28} * 2^3$

$2^{28} \times 2$

2^{28}

8×2^{20}

2^{23}

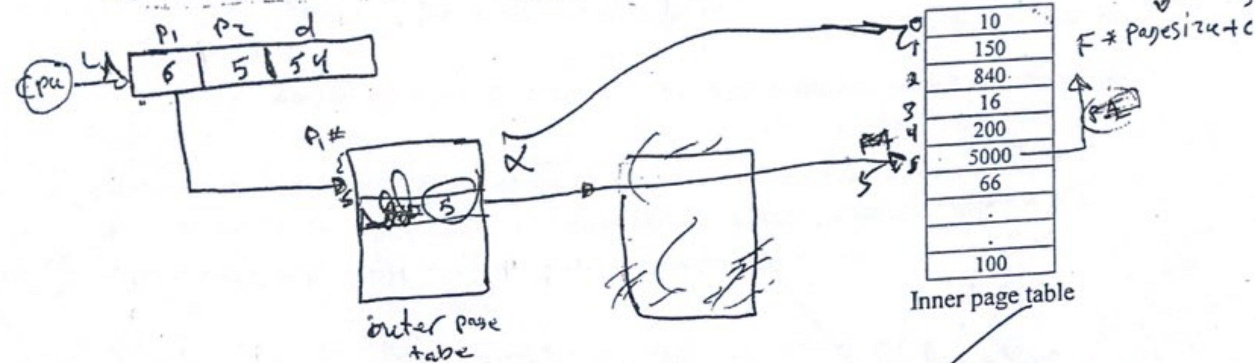
5

28
2^{13}
3 10
8KB
28 - 13
28 (13)

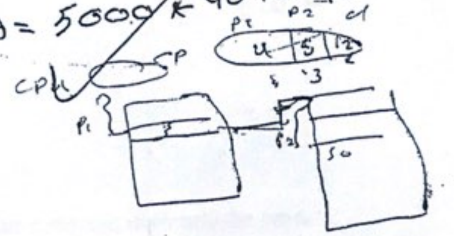
$2^{8} \times$

P_1	P_2	d
4	8	12

[3] (a) In a certain computer, the virtual address is 24 bits with page size is 4096 bytes. The page table is implemented using two-levels with 4 bits for the outer page table and 8 bits for the inner.
Given the virtual address in binary: 0110 0000 0101 0000 0011 0110
Explain using diagram only how the physical address is computed from the virtual address.



$PA = 5000 * 4096 + 54$



(b) For the binary semaphore S defined below.

int S; // S is a binary semaphore

(i) Define the *wait* and *signal* instruction(operation) on S.

```
wait(S)
{ while (S <= 0)
    // do nothing
  S = S - 1;
}

signal(S)
{
  S = S + 1;
}
```

wait(S)
do
{ while S<=0
no op
S = S-1;
}

(ii) As a **condition**, the *wait* and *signal* operations above **must** be executed ***atomically***. Show what will happen if they **don't**. Explain with an example.

in wait and signal $S = S-1$; and $S = S+1$;
its critical section. if it execution not atomically the sum consistence happen in data.
the int S ; ~~is shared in the process~~
if it executed parallel the as example:

wait(S) do
S0: register1 = S;
S1: register1 = register1 - 1;
S2: return S;

signal(S)
S0: register2 = S
S1: register2+1;
S3: return S;

if wats do S0 the signal do S0 and S1;
and wats do S1; that coulan error;

not clear

[4] Given the statements:

S1: int x = 10 , y=5 , z , c=0 ;
S2: c += x ;   c = c + x
S3: y *=2 ;   y = y * 2
S4: cin>>z ;   cin >> z ;
S5: cout<<x<<y<<z<<c ;

c = c + x
y = y * 2
cin >> z
cout << x, y, z, c

(a) Rewrite the statements above using the Dijkstra notation **parbegin** and **parend**;

```
S1 ;
parbegin
  S2 ;
  S3 ;
  S4 ;
parend
S5 ;
```

(b) Rewrite the statements above using **fork** and **join** structure.

```
S1 ;
Fork L1 :
  S4 ;
```

```
S1
~~Fork L1~~
~~Fork L2~~
S2 ;
```

```
count = 3 ;
S1 ;
Fork L1 ;
  S2 ;
  go to L3

L1 : Fork L2
  S3
  go to L3

L2 : S4 ;
L3 : while (count != 0)
       do nothing
Join S2, S3, S4 ;     join count

S5 ;
```

```
S1
count = 3
Fork L1
  Fork L2
  S2
  goto
L2: S3
  goto
L1: S4
L4 : join
```

Exam out of 70
Each question
10 points

BIRZEIT UNIVERSITY

College of Engineering and Technology

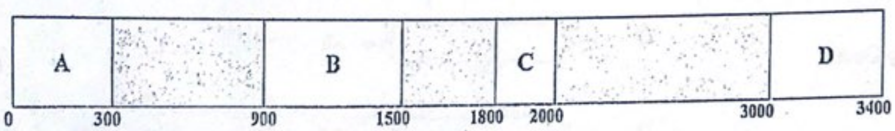
Computer Science Department

Name ------------------------------------ Number ------------------------ Section 10-11 1-2

Comp431 Midterm Exam Fall 14/15

10

[1] In Dynamic(variable) regions memory management (MVT), the memory status looks like:



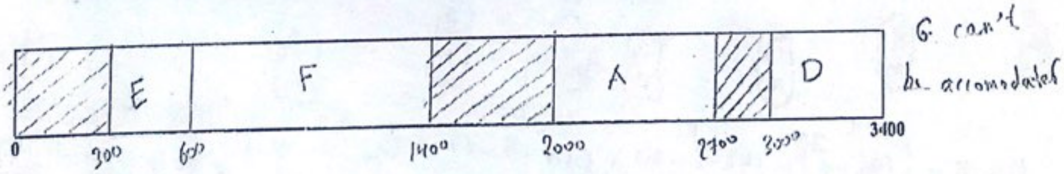
Given the following memory requests in the given order:

- Process E starts and requests 300 memory units.
- Process A requests 400 more memory units.
- Process B exits.
- Process F starts and requests 800 memory units.
- Process C exits.
- Process G starts and requests 900 memory units.

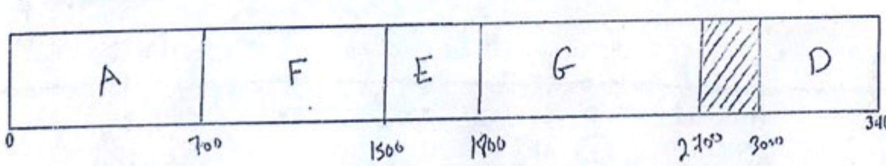
NOTE:

- For a growing process, if the current state cannot accommodate a request, reallocate the process.
- If the request can't be accommodated, just state that.

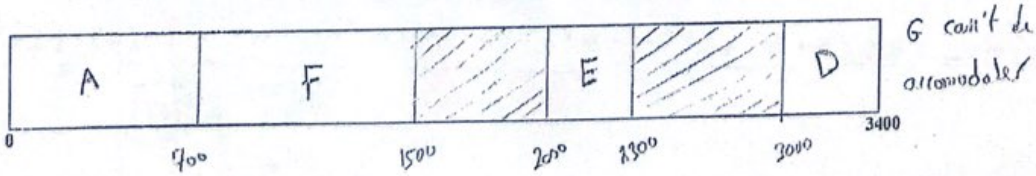
(a) Describe the contents of memory after each request using **<u>first fit</u>**



(b) Describe the contents of memory after each request using **<u>best fit.</u>**



(c) Describe the contents of memory after each request using **<u>worst fit.</u>**



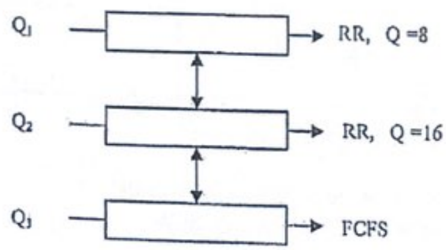
(16) [3] Assume the ready queue looks like:

Process	CPU Burst	Arrival Time
P_1	~~70~~ 64	00
P_2	~~20~~ ~~18~~ 12	06
P_3	~~6~~ 0	11
P_4	~~10~~ 0	20

70 ~~54~~ ~~46~~ 0
20 ~~12~~ 0
6 0
10 ~~2~~ 0

Given a **multilevel Feedback** queues system which looks as:

Q1 is Entry Queue

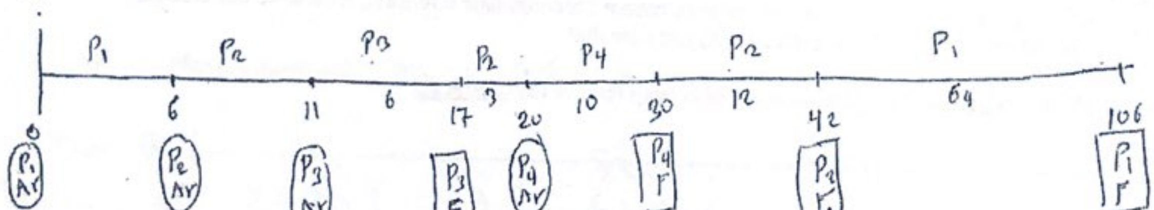


Compare the Average Waiting Time for the scheduling algorithms:

(a) SJF with preemption (SRTF) and

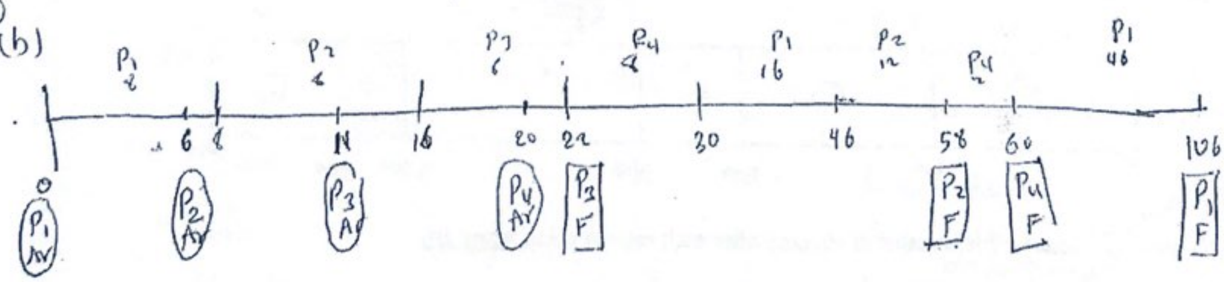
(b) Multilevel Feedback Queues

(5) (a)



$$AWT = \frac{(106-0-70) + (42-6-20) + (17-11-6) + (30-20-10)}{4} = \frac{36+16+0+0}{4} = \boxed{\frac{52}{4}} = \boxed{13}$$

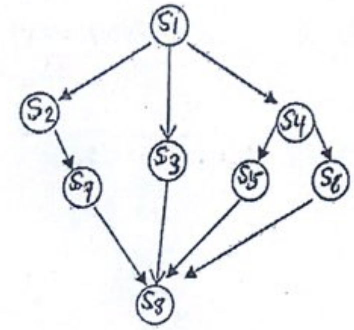
(5) (b)



$$AWT = \frac{(106-0-70) + (58-6-20) + (22-11-6) + (60-20-10)}{4} = \frac{36+32+5+30}{4} = \boxed{\frac{103}{4}}$$

It shows that: SRTF is better algorithm for this case since it gives less AWT.

6 [6] Given the precedence graph:



3 (a) Write an equivalent code using Fork & Join

```
count=4
S1
Fork L1
Fork L2
S3
goto L4
L2: S4
Fork L3
S6
goto L4
L3: S5
goto L4
L1: S2
S7
L4: join count
S8
```

3 (b) Write an equivalent code using parbegin & parend

```
Begin
  S1;
  Parbegin
    Begin
      S2;
      S7;
    END
    S3;
    Begin
      S4;
      Parbegin
        S5;
        S6;
      Parend
    END
  Parend
  S8;
END
```

$\frac{7}{12}$ [7] (a) Is the following system of 4 processes with 2 resource types deadlocked? Show your work.

Current allocation matrix

	R1	R2
P1	1	3
P2	4	1
P3	1	2
P4	2	0

Current request matrix

	R1	R2
P1	1	2
P2	4	3
P3	1	7
P4	5	1

available vector

R1	R2
1	4
2	7
3	9

Let search for a safe sequence

5 <P1, P3, No other process will be satisfied.

∴ There is no safe sequence

∴ may be deadlock

(b) If the available vector is as shown, is the system deadlocked? Show your work.

5 Let search for a safe sequence

<P1, P3, P2, P4>.

There is a safe sequence

No deadlock

2	4
3	7
4	9
8	10
10	10