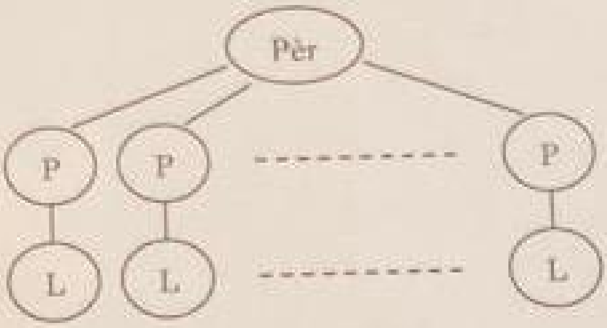
Lebanese University
Faculty of Sciences I

2nd session 2018
Duration: 2 hours

INFO 324-System II

Part I Process : Creation, Communication 20 Points

1. Write a C program under UNIX that creates the process tree on the right. The parent process must create N process (the P_i) which each creates a child process (L_i). In addition, the creation must meet the following two criteria:



(4) a. The parent process must not create the P_i child before the L_{i-1} process (P_{i-1} child) is created.

(3) b. After the creation of all tree processes, all these processes can be running, i.e. the creation of one process should have no relation with the termination of another process.

2. Given the following extract from a C program under Unix: :

```
void f(int i)
    {
            int j;
            for (j=0; j<i; j++)
            if (fork()==0){
                    f(i-1);
                    exit(0);

            }
    // To fill...
    }
```

([illegible]) a. How many processes are created after the call of f(4) ?

(4) b. Generalize the previous result, i.e., how many processes are created by a call to f (n) ?

([illegible]) c. Complete the code (in place of the comment) so that the termination occurs without a zombie process.

Part II Memory management 25 Points

(5) 1. Given free memory partitions of 300k, 125k and 260k (in this order), describe, step by step, how each of the First-fit and Best-fit algorithms would place 100k, 250k and 250k processes (in this order).

2. Let **m** the number of memory cases allocated to a program and W the set of the pages referenced at a given instant. Let **S (m, W)** be the set of pages present in memory after the References **W**. We consider a program with a size of 5 pages. These pages designated by **A, B, C, D, E** are initially in secondary memory (disk).

(4) a. Construct the set S (m, ABCDABEABCDE) with m = 4 in the case of a FIFO replacement algorithm. How many page faults result from this serie of references?

b. Same question with m = 3.

c. Same questions as a) and b) for LRU. 3

d. Same questions as a) and b) for second chance algorithm. 3

3. We consider a paged memory system of 8 frames (frames) of 16 bytes (each), addressable to the byte (i.e. 1 offset = 1 byte), and a process occupying 4 pages in this system. The placement of the pages in memory is represented on the following :

Page 1
Page 0
Page 3
Page 2

4

a. Indicate the contents of the page table of this process.

b. Give the physical address in decimal for the logical address 35. Justify the steps. 4

Part III **File System management** **25 Points**

A) Consider a file with size 1 MB. Assume that the file control block (i.e., FAT, inode, ...) is loaded in memory. The size of the block is 1KB. For the linked strategy the address of the first block is supposed loaded in memory.
Calculate the number of disk I/O operations required for contiguous, linked and indexed (inode) allocation strategies to make the following changes to the file. In the contiguous case, you may assume there is no space to grow in the end. Also assume that the new information to be added to the file is stored in memory. For the linked we assume that only we have one pointer to the first block

5 a. Add 2 blocks at the beginning

3 b. Add 4 blocks at the end

3 c. Remove the 6-middle block

B) Given a FS (File system) where the table topo contains 10 entries, each one corresponds to only one level of indirection (each one points to a map block). Knowing that each block occupies 2 kilobytes, and the number of a block occupies 4 bytes and each block inode contains 16 inodes:

3 a. Which is the maximum size of a file supported by this FS?

3 b. How much, the file of maximum size, does it occupy of effective space on the disc?

5 c. We consider a file containing 5,000,000 bytes. How much blocks (data and map) it is necessary to represent this file on disk? Justify briefly your answer.

3 d. Define the structure file descriptor (fdesc) correspondent to this FS.

Good Luck

Part I

10

1)

```
--- part1-1 ----------------------------------------
#include <stdio.h>
#include <unistd.h>

void main(){
int i,n=5,p[2],x; 1
pipe(p); 1
printf("my pid is %d\n",getpid());
 for(i=0;i<n;i++){
    if(fork()) 1
      read(p[0],&x,sizeof(int)); 2
    else {
      if(fork()) 1
           write(p[1],&x, sizeof(int)); 2
      break; 1
    }
 }
while(1); 1
}
```

2)

```
------- part1-2 ----------------------------
```

(4+4+2)

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>

void f(int);
void main(){
printf("my pid is %d\n",getpid());
f(4);
while(1);
}

void f(int i){
int j;
for(j=0;j<i;j++)
  if (!fork()) {
    f(i-1);
    break;
  }
while(wait(NULL));
}
```

4 a) $f(4) \rightarrow 1 + 4 + (4 \times 3) + (4 \times 3 \times 2) + (4 \times 3 \times 2 \times 1)$ = ~~[illegible]~~

$= 1 + 4 + 12 + 24 + 24$

$= 65$ processe

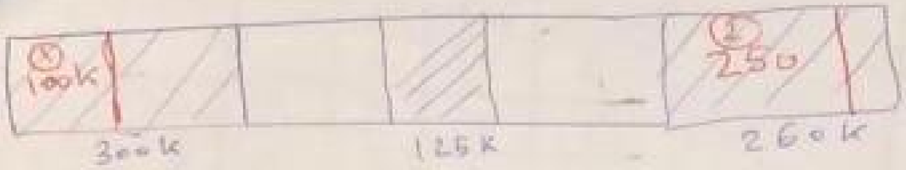
4 b) $f(n) \rightarrow 1 + n + (n-1)n + n(n-1)(n-2) + \dots + n(n-1)(n-2)(n-3)\dots \times 1$

$f(n) \boxed{= \sum_{i=0}^{n} \frac{n!}{i!}}$

?? $\sum_{i=0}^{n} \frac{n!}{(n-i)!}$ (10 sec)

2 c) while (wait (Null));

II : Memory management



1)

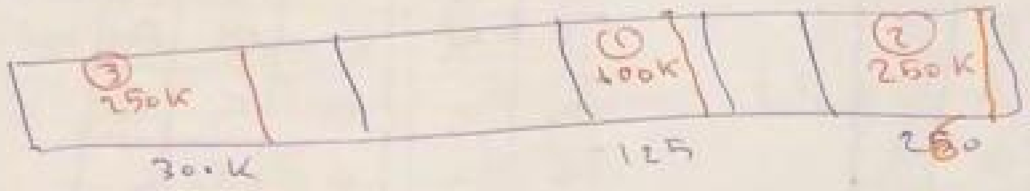
5

First Fit : 100K, 250 K and 250 K

- the 3rd 250 K has no place in first-fit

=> compaction

Best Fit



2) S(m, w)

prog → size = 5 pages

A, B, C, D, E

w = {A,B,C,D,A,B,E,A,B,C,D,E}

a) m=4

FIFO Alg :

	Page fault	Memory			
		C1	C2	C3	C4
A	y	A			
B	y	B	A		
C	y	C	B	A	
D	y	D	C	B	A
A	N	D	C	B	A
B	N	D	C	B	A
E	y	E	D	C	B
A	y	A	E	D	C
B	y	B	A	E	D
C	y	C	B	A	E
D	y	D	C	B	A
E	y	E	D	C	B

page faults = 10 2

4

	Page fault	C1	C2	C3
A	y	A		
B	y	B	A	
C	y	C	B	A
D	y	D	C	B
A	y	A	D	C
B	y	B	A	D
E	y	E	B	A
A	N	E	B	A
B	N	E	B	A
C	y	C	E	B
D	y	D	C	E
E	N	D	C	E

page fault = 9

LRU

8 page faults

Case / Page	Page faults	C1	C2	C3	C4
A	y	A			
B	y	B	A		
C	y	C	B	A	
D	y	D	C	B	A
A	N	A	D	C	B
B	N	B	A	D	C
E	y	E	B	A	D
A	N	A	E	B	D
B	N	B	A	E	D
C	y	C	B	A	E
D	y	D	C	B	A
E	y	E	D	C	B

4

Case / Page	Page fault	C1	C2	C3
A	y	A		
B	y	B	A	
C	y	C	B	A
D	y	D	C	B
A	y	A	D	C
B	y	B	A	D
E	y	E	B	A
A	N	A	E	B
B	N	B	A	E
C	y	C	B	A
D	y	D	C	B
E	y	E	D	C

10 faul

Second chance

Case / Page	Page fault	C1	C2	C3	C4
A	y	B*			
B	y	B*	A*		
C	y	C*	B*	A*	
D	y	D*	C*	B*	A*
A	N	D*	C*	B*	A*
B	N	D*	C*	B*	A*
E	y	E*	D	C	B
A	y	A*	E*	D	C
B	y	B*	A*	E*	D
C	y	C*	B*	A*	E*
D	y	D*	C	B	A
E	y	E*	D*	C	B

10 page faults

4

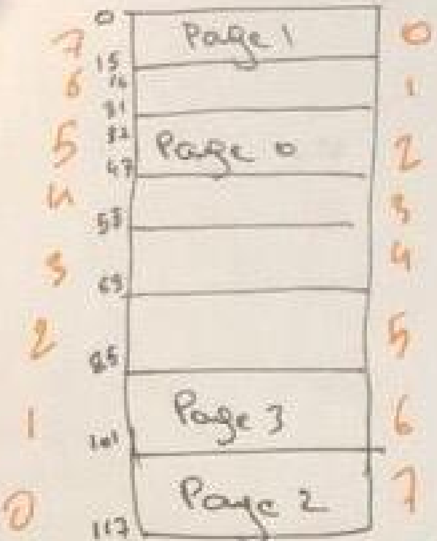
Case / Page	Page fault	C1	C2	C3
A	y	A*		
B	y	B*	A*	
C	y	C*	B*	A*
D	y	D*	C	B
A	y	A*	D*	C
B	y	B*	A*	D*
E	y	E*	B	A
A	N	E*	B	A*
B	N	E*	B*	A*
C	y	C*	E	B
D	y	D*	C*	E
E	N	D*	C*	E*

9 page faults

ory System with pagination (8 pts)

to

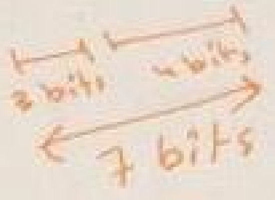
size of frame = 16 bytes ⇒ 2^4



offset : 4 bits

8 pages : 2^3

⇒



a)

Page #	frame #	Presence
0	2	1
1	0	1
2	7	1
3	6	1
⋮	000	0
7	000	0

(4 pts)

5

b) 35 bytes ⇒ addressing in 7 bytes

0	1	0	0	0	1	1

page # 4 bits offset

(4 pts)

$35 = \underline{010}\ \underline{0011}$

⇒ ~~frame~~ page # = 2

3

offset = 3

~~physical address (decimal)~~

Physical : page #2 ⟹ frame # 7

64	32	16	8	4	2	1
1	1	1	0	0	1	1

⇒ ~~115~~ (physical address)

112 + 3 = 115 (physical address)

(16×7) + 3 = 115

3

t II :

ile { size = 1MB = 2^{10} blocks
size (block) = 1 KB

9

- File control block loaded in memory

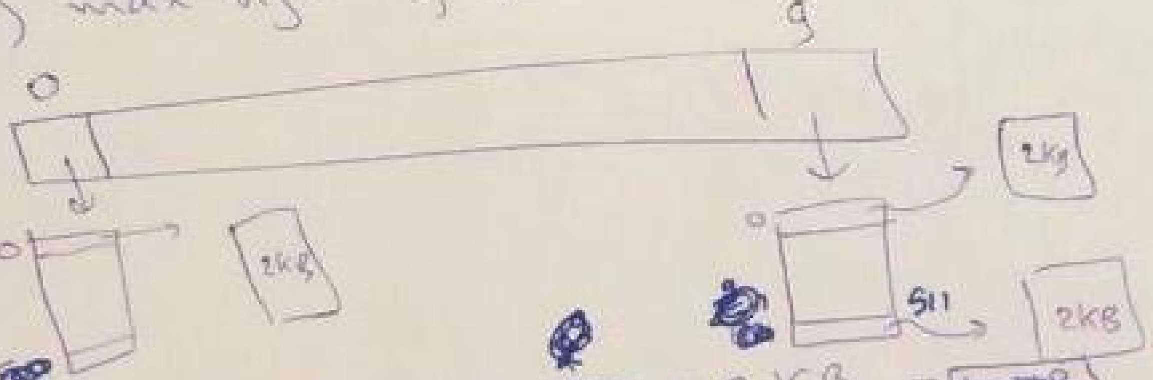
-

	contiguous I/O	linked I/O	Indexed I/O
a) add 2 blocks at beginning	2W + 0R 1	2W + 0R 1	2W + 0R 1
b) add 4 blocks at the end	4W + 2^{10} R + 2^{10} W 1	4W + 2^{10} R 1 + 1W (update pointer)	4W + 0R 1
c) Remove the 6. middle block	508 R + 508 W read all blocks after (515) and write them starting from block # 510 1 delete blocks from 510 → 515	read and follow links to position 510 and update the pointer from 509 → 516 1 509 R + 1 W	0R + 0W 1

B) - 1 block = 2 KB (size) = 2^{11}
- block # = 4 bytes
- block - inode = 16 inodes

nb_entries = $\frac{2^{11}}{2^2} = 2^9$
$= 512$

a) max size of file in FS?



max size = 10 × 512 × 2 KB = long

4

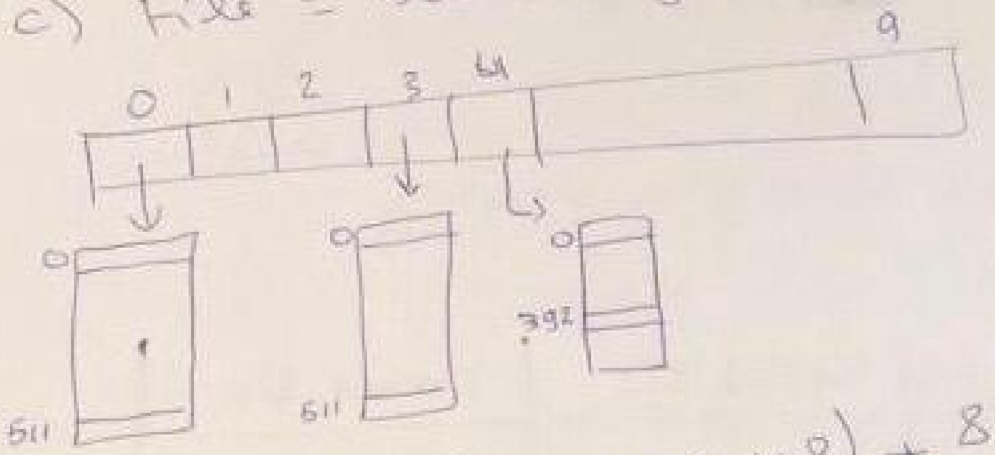
b) effective space = $\underbrace{(10 \times 512 \times 2KB)}_{\text{data blocks}} + \underbrace{(10 \times 2KB)}_{\text{map blocks}}$ + size (inode)

4

4

size (inode) =
$\frac{2KB}{16}$ = 128 Bytes

c) File = 5000000 bytes



5

5000000 = (2441 × 2048) + 832

=> we need 2442 data blocks
- -> 2441 full data block
- -> 1 partial data block

=> we need 5 map blocks
- - 4 full map blocks
- - 1 map block (partially filled)

Summary : { 2442 data blocks
{ 5 map blocks

d)
```
struct fdesc
{
    int topo [10]
    int map [512]
    int buffer [2048]
    :
};
```

3

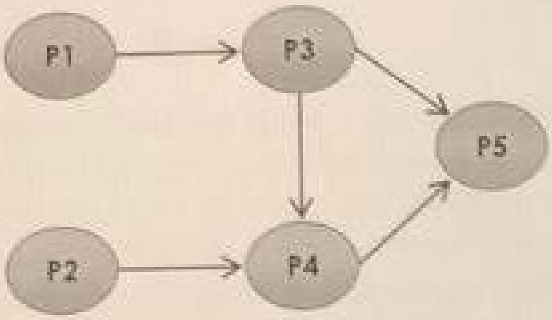
Lebanese University Faculty of Sciences I	INFO324 - E	Final 2016-2017 Duration 2 hours

18

Problem I: Process management (20 points):

1. Write a program C under UNIX where a parent process creates 5 processes P1, P2, P3, P4 and P5. Assume that each Pi process executes the Fi () function (you are not required to write the code). Moreover, using the signals, the execution order of the processes must respect the order presented by the following graph (for example, according to the graph P3 must not be executed before P1):

(12)



2. Taking into account the solution of the preceding question, and assuming that the codes of the functions Fi () are the following: (6)

F1() { i = i-5; }
F2() { i = i*3; }
	F3() { i = i-8; }
	F4() { i = i*3; }
		F5() { i = i+3; }

(i is considered as variable common to all processes and initialized to the value 5 at beginning), provide all possible values of i after the execution of the five processes.

22

Problem II: Memory management (20 points):

A) Consider the following reference set to the following virtual pages:
3, 4, 3, 2, 1, 3, 5, 1, 4, 3, 1, 3, 5, 2, and 3
In a memory system with 3 frames. How many page faults is generated for each of the following replacement algorithms:

(10) 5×5

  a. LRU
  b. Second chance.

B) Below is given a part of the table of segments of a process:

(6)

Segment #	Size of segment	Base address
1	30 KB	.....
2	16 KB	32 KB
3	.....	105 KB
4	8 KB	58 KB

a) Give the physical address of the following virtual address (2, 5703) (2)
b) the physical address of a data in segment 4 is 67502 bytes, find its virtual address (2)
c) complete the missing values in the above table knowing that the virtual address (1, 2453) corresponds to physical address 75157 and the physical address of the last data in segment 3 is 128000 (2)