I3303

جامعة طلاب النفس
كلية العلوم

Lebanese University
Faculty of Sciences I

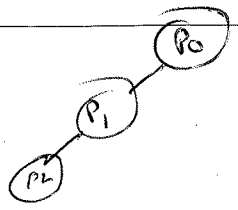
Partial 2018/2019
Duration: 1 hour

I3303 - INFO 324
Operating System II

Problem I — 8 points

Given the following C program under Unix

```
void main(){
    if(fork()){
        printf("A");
        if(fork()){
            wait(0);
            printf("B");
        }
        else
            printf("C");
    }
    else
        if(fork()){
            wait(0);
            printf("D");
        }
        else
            printf("E");
}
```



1. Which are the possible outputs of this program?
2. Add to this program few codes such that the only possible output be EDCBA.

Problem II — 8 points

Given the following program:

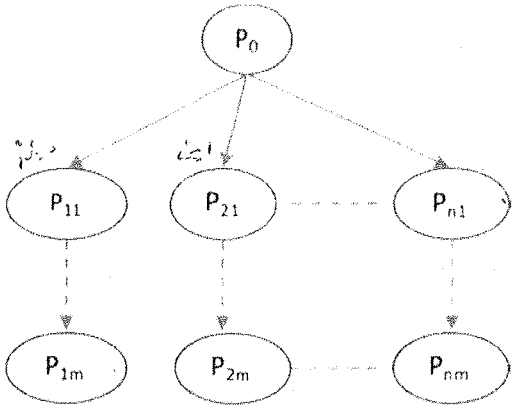
```
int f(int j){
      if (j == 0) return 1;
      return (fork() || fork()) && f(j-1) ;
}

void main(){
      if(f(2))
            printf("test\n");
}
```

Draw the tree of processes generated by the program and show the output results for each process.

Problem III **14 points**

Write a c program under Unix that creates the tree of processes in the figure under the following criteria:

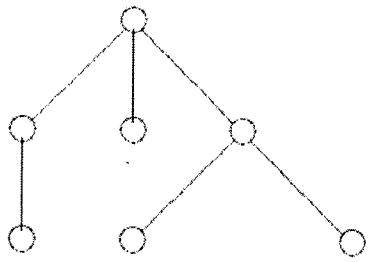
- The number of levels (m) and the number of each process in each level (n) are given to the main process from the command line.
- Process P_0 waits for a time t before the creation of all the processes in the first level. After the creation of level1, process P_{11} should creates it's son P_{12} before giving the turn to P_{21} that creates its son P_{22} and so on.
- The last process for each level (for example P_{n1} in level 1) should give the hand to the first process in the next level (for example P_{12}) and so on.



Lebanese University Faculty of Sciences I	Operating System II I3303/INFO324	Partial 2017-2018 Duration: 1 hour

Problem 1: (20 minutes)

A. Write in one statement the code to create the following tree of processes



P.S: **if** and **else** are considered as one statement (nested if else is not permitted)

B. We consider the following piece of code:

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

void main() {
printf("I'm the main with pid=%d\n",getpid());
if (fork())
 fork() && (fork() || fork());
else
 fork()*fork();
while(1);

}
```

1. How many processes are created after the execution of the above program?
2. Draw the tree of processes generated.

Problem 2: (parts A and B are independent) (40 minutes)

A. A parent process creates N parallel child processes that have a point of appointment (RDV). The two processes should execute two functions: **Pre_RDV()** and **RDV()** consecutively. A process arriving at the RDV point (i.e., before executing the function **RDV()**) should waits if there is at least one other process that did

not arrived. The last arriving process wakes up all the blocked processes. Write a solution that solves this problem.

B. In this part, we suppose that we have a shared data segment that contains a shared variable between **N <u>separated processes</u>** (each process is an independent C program). These processes should access the shared variable and increment it by 1. The shared variable is created by the first process.

a. Write the code of the first process that creates the shared data segment and the shared variable and initialize it to 1. This process should maintain the **<u>way of communicating</u>** the (ID) of the shared data segment between the different processes.

b. Write the code of the other processes (all have similar behavior) that should access the shared data segment via its ID and increment the shared variable and print out on the screen the value of the shared variable along with its pid.

kel child bado ynaffez l fnen

shmget (IPC_PRIVATE,

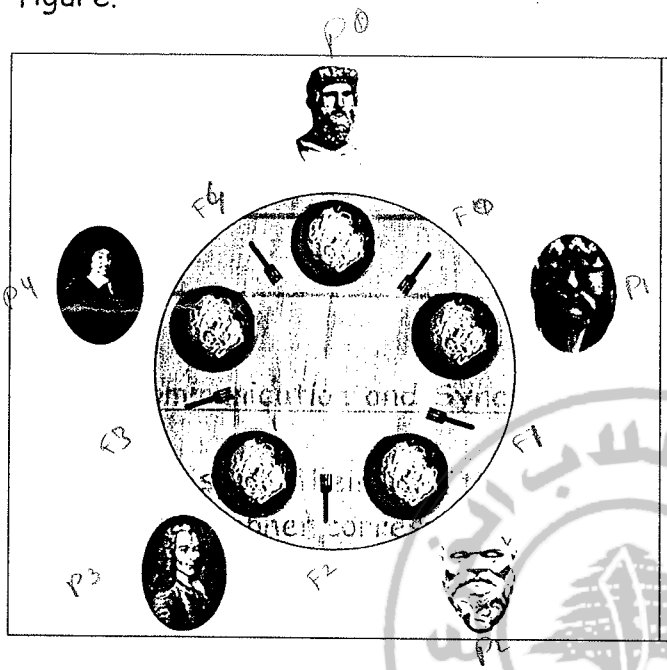
shmat (shmid, NULL, 0) ; IPC_CREAT)

shmdt

shmdt

Problem II: Communication and Synchronization (15 points):

Five philosophers spend their lives thinking and eating around a table (see the left figure). Suppose each philosopher corresponds to a process, then its pseudo code will be as shown in the right figure.



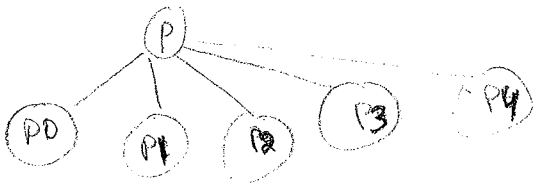

```
While (1) {
    thinking();
    eating();
}
```

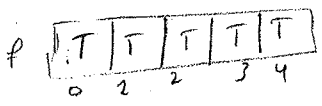
Each philosopher has two states: "I think", "I eat" by which he always passes in this order. To eat, they need two forks, but there are only five forks. Before beginning to eat, he must ensure that both left and right forks are free.

One of the proposed solutions is that each philosopher, when he is hungry, acts in order as follows: (a) takes the left fork, (b) the right one, (c) eats, (d) rests them

A) Write a program C that creates 5 processes representing the five philosophers, to implement this proposed solution using the tools of communication and synchronization already studied in the course.

B) An in-depth study of the previous solution shows that it is a bad solution because it cannot avoid blocking as if each process takes the left fork in the same time. In this situation, each process must wait the other and all of them are blocked. Update the code in (A) to avoid this kind of blocking by ensuring for example that at max two processes can eat in the same time or at max 4 processes can take the left fork in the same time.





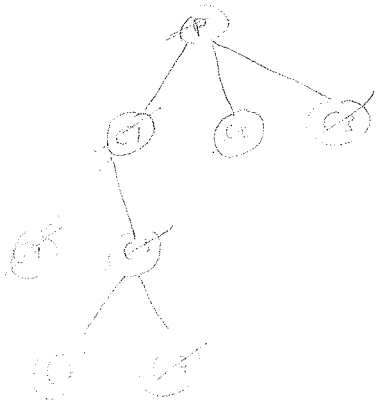
hungry = 1

0 1

Problem I (15 points):

a) Draw the family tree of the processes generated by this program.

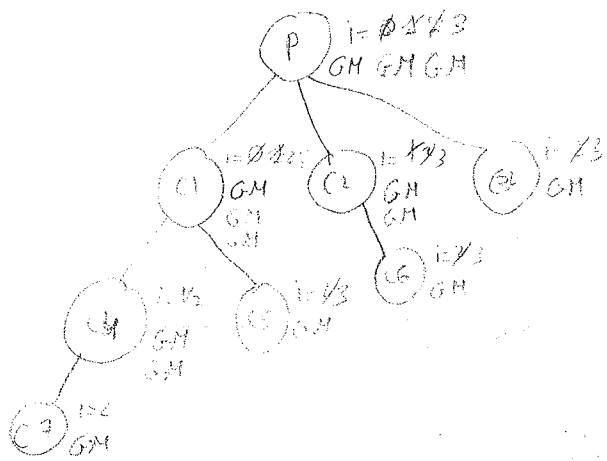
```
#include <unistd.h>
int main(void)
{
    if (!fork())
        fork () && (fork() || (fork() && fork()));
    else
        fork () && fork();
    sleep (2);
    return 0;
}
```

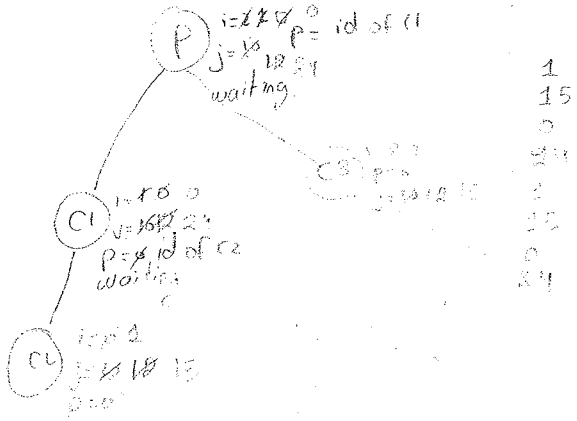


b) Display the results printed by the two following programs. Justify your answers.

Program 1	Program 2

void main{
int i;
for(i=0;i<3;i++){
fork();
printf("Good Morning\n");
}
}
	void main{
	int i=2, j=10, p;
	while(i){
	i--; p=fork(); wait(0);
	}
	j += 2;
	if(p==0) {j += 3; i ++;}
	else {j *= 2; i *= 2;}
	printf("i=%d, j=%d\n",i,j);
	}





Lebanese University
Faculty of Sciences I

6 April 2016
Duration : 1 hour

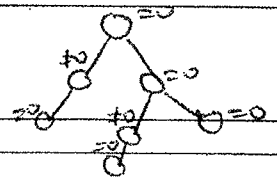
INFO 324
Operating System II

Dr. Ahmad Faour

Problem I — 6 points

Draw the graph generated by the following code:

```
void main(){
      if( (fork() || fork()) && !(fork()))
            fork();
}
```



Problem II — 12 points

Giving the following program:

```
1.   void main(){
2.         int p[2], x = 1;
3.         pipe(p);
4.         if(fork()){
5.               printf("A\n");
6.               if(fork(){
7.                     // Ligne 7
8.                     // Ligne 8
9.                     printf("B\n");
10.              }
11.              else{
12.                    // Ligne 12
13.                    printf("C\n");
14.              }
15.        }
16.        else{
17.              // Ligne 17
18.              printf("D\n");
19.              // Ligne 19
20.        }
     }
```

The following questions are independent

a) What are the possible outputs?

b) We add the following statements :

```
Line 7 : wait(0) ;
Line 8 : wait(0) ;
```

a) What are the possible outputs?
Justify your answer.

b) Now, we add the following statements :

```
Line 7 : write(p[1],&x,sizeof(int));
Line 8 : write(p[1],&x,sizeof(int));
Line 12 : read(p[0],&x,sizeof(int));
Line 17 : read(p[0],&x,sizeof(int));
```

c) What are the possible outputs?
Justify your answer.

d) We add also the following :

```
Line 7 : write(p[1],&x,sizeof(int));
Line 12 : read(p[0],&x,sizeof(int));
Line 17 : read(p[0],&x,sizeof(int));
Line 19 : write(p[1],&x,sizeof(int));
```

e) What are the possible outputs? Justify your answer.

Problem III — 12 points

1. Write a program that creates N child processes. The child processes shared a pipe of communication, and behave according to the following :
   - The child processes with even id (i.e., 0th process, 2th process...) are producers of messages: each one wait a time t before writing in the pipe the character 'a' and do the same indefinitely.
   - The child processes with odd order (1th process, 3th process...) are consumers of messages: each one wait a time t before reading from the pipe character by character and in indefinitely manner.
2. Now, suppose that the writers are faster than readers. Rewrite the program such that the pipe does not contain more than M characters (i.e., if the pipe contains M characters, no any process can write in it). The readers can read from pipe as usual.

P.S: think of the way to communicate the variant number of characters inside the pipe between processes (i.e., the reader and writer processes must be up to date regarding the contents of the pipe).

Good Luck

جامعة طلاب النفس
كلية العلوم

Lebanese University
Faculty of Sciences I

18 April 2015
Duration : 1 hour

INFO 324
Operating System II

Dr. Ahmad Faour

Problem I

15 points

A) What is the possible output of the following two programs? Justify your answer while taking into account all possible scenarios and identifying if a possible deadlock can occur.

Program 1

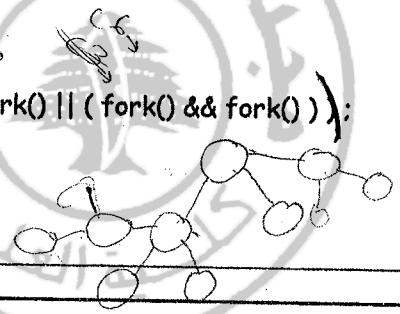
```
void main()
{
        int i, j = 1, p[2];
        pipe(p);
        write(p[1],&j,sizeof(int));
        for(i=1;i<5;i++)
                if(!fork()){
                        close(p[1]);
                        break;
                }
        read(p[0],&j,sizeof(int));
        printf("%d\n",i);
}
```

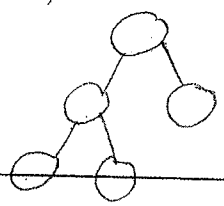
Program 2

```
void main()
{
        int i, j = 1, p[2];
        pipe(p);
        write(p[1],&j,sizeof(int));
        for(i=1;i<5;i++)
                if(!fork()){
                        close(p[1]);
                        break;
                }
        wait(0);
        read(p[0],&j,sizeof(int));
        printf("%d\n",i);
}
```

B) Draw the tree of processes generated by the following code :

```
int main(void)
{
        (fork() || fork()) && fork() && (fork() || ( fork() && fork() ));
        sleep(2);
        Return 0;
}
```



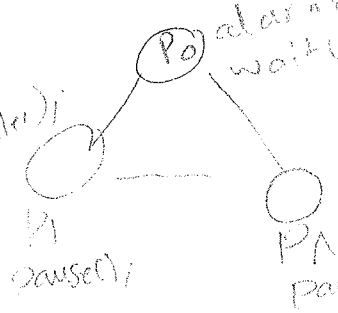


Problem II

15 points

Write a C program under UNIX where a parent process P0 creates N child processes (P1, P2... PN). The index of processes is in the order of creation, i.e., P1 is created before P2, etc.
All the processes are waiting of a signal using the "pause" statement.
The parent process P0, turn on an alarm for 5 seconds and wait. In the reception of the SIGALARM signal, it activates the process P1 by sending a signal SIGUSR1. P1 in its turn, activate P2 in the same way and exit. Then each process must activate the next process and exit. The last process PN must send to the parent P0 a signal of type SIGUSR2 and exit. In the reception of SIGUSR2, the parent P0 must print the message "End of execution" and exit.

N.B: think about the way to communicate the pid of processes between them such that the process can know the pid of the next process and so on.



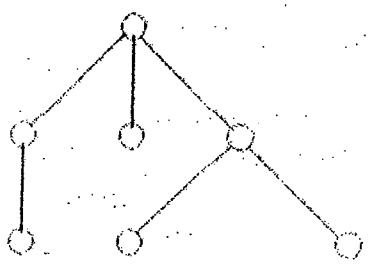
Bon courage


جامعة طلاب النفس
كلية العلوم

Problem I

8 poi

1) Write a few lines of code to generate the following tree of processes:



2) Recreate the same tree above using just one C statement.

(fork() || fork()) && fork && (fork() || (fork() || (fork() && fork();)))

Problem II

12 poir

Write a C program that creates 3 processes H, M, S that increment the 3 "hands" of a clock. S receive signal SIGALRM each second and send a signal to M when its counter pass from 59 à 0. M increme its counter when it receives a signal. When its counter pass from 59 to 0, M send a signal to H. T parameters correspond to the initialization values of the counters.

Problem III

10 poin

Consider the following program where a process father creates a process child. The process father writ on the screen the characters 'A' and 'B'. The process child writes the characters 'C' and 'D'. We suppo that the process creation is done with success.

```
#include <stdio.h>
#include <unistd.h>
void main()
{
     if(fork()){     /* father */
          printf("A");
          printf("B");
     }
     else{           /* child */
          printf("C");
          printf("D");
     }
}
```

1. Which is (are) the result(s) printed by executing this program?
2. Change, the least possible, the program for that the only result will be "CDAB".
3. Change, the least possible, the program for that the only result will be "ACBD"

Lebanese University
Faculty of Sciences I

April 14, 201
Duration : 1 ho

INFO 324
OPERATING SYSTEM II

Dr. Ahmad FAOUR

Problem I — 30%

Which are the results printed by the two following programs. Justify your answers.

Program 1

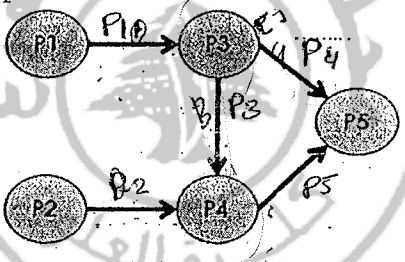
```
void main{
      int i;
      for(i=0;i<3;i++){
            fork();
            printf("Good Morning\n");
      }
}
```

Program 2

```
void main{
      int i=2, j=10, p;
      while(i){
            i--; p=fork(); wait(0);
      }
      j += 2;
      if(p==0)    {j += 3; i ++;}
      else        {j *= 2; i *= 2;}
      printf("i=%d, j=%d\n",i,j);
}
```

Problem II — 40%

1. Write a C program under UNIX where a parent process creates 5 processes P1, P2, P3, P4 and P5. Suppose that each process Pi execute the function Fi( ) (you are not asked to write the code). Moreover, by using pipes of communications, the order of process execution must follow the order presented by the following graph (for example, as the process P3 should not be executed before P1):



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

F1() { i = i+5; }	F3() { i = i+8; }	F5() { i = i+1; }
F2() { i = i*3; }	F4() { i = i*2; }	

(i supposed to be a common variable shared by all processes initialized to the value 10), provide all possible values of i after the execution of the five processes.

Problem III — 30%

The idea of this problem is to sort four numbers in ascending order, in a distributed manner. For that a parent process reads four numbers from the user, let a, b, c and d. It sends the two values (a and b) to the first child process and the other two (c and d) to a second child process. Each child must return the two values in ascending order (the minimum then the maximum), or (min1, max1) from the first process and (min2, max2) from the second. To find the maximum and minimum of four numbers, the parent process then resends the couple (max1, max2) to the first and (min1, min2) to the second. The two Childs return (min, A) and (B max) to their father who finally compares the values A and B, and displays the final result that matches the four numbers sorted in ascending order. Write a C program on UNIX that implements the scenario above.

Example: If the four numbers read are (12, 8, 4, 7) then the father sends (12 and 8) to the first child and (4 and 7) to the second child. The father receives (8 and 12) from the first and (4 and 7) from the second. As a result, the father resends (8 and 4) to the first and (12 and 7) to the second and receives (4 and 8) and (7 and 12). After comparing the two extremities (8 and 7), the father shows the final result: 4, 7, 8, and 12.


جمعية طلاب النفس
كلية العلوم

LEBANESE UNIVERSITY
Faculty of Sciences – Section I

Second Semester
Partial 2009-2010

Course Code : INFO 324 (English)
Instructor: Dr. Ahmad FAOUR

Duration: 1 hour

Problem I: (8 Points)

1. How many processes are created after the execution of the following program :

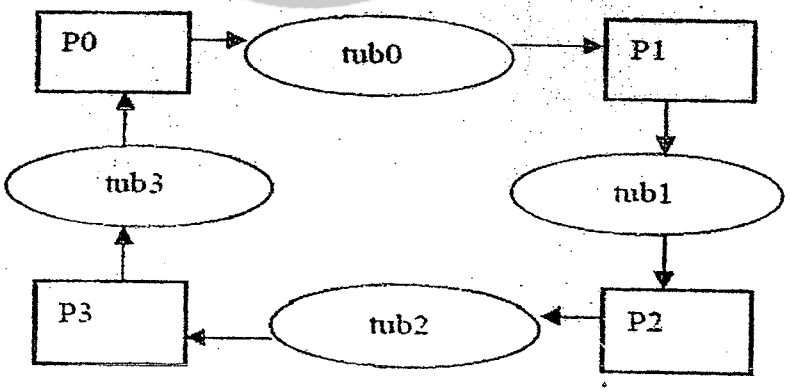
```
void main() {
      int i;
      for(i=1;i<=3;i++)
            fork();
      fork();
}
```

2. Express in the simplest possible function of N (in terms of numerical series) the number of processes created by the following program:

```
void main() {
      int i;
      for(i=1;i<=N;i++)
            fork();
      fork();
}
```

Problem II (12 points)

Consider N processes that communicate through pipes. Each process shares two pipe (one with the right process and another process with the left). For example for N = 4, proc communicate according to the following:



1) Complete the code below in order to implement this communication architectu of the N processes created. The standard input and standard output of each proc Pi are redirected to the appropriate pipes. For example, the process P0, the input and output respectively become the pipes tub3 and tub0.

```
#define N 4
void proc( int ) ;
int main ( )
{       int i ;
        for( i=0 ; i <N; i++)
                if ( fork() ==0)
                {       proc(i); exit(0);
                }
        exit(0) ;
}
```

Attention: You should not write the code for the proc.

oblem III (10 points)

e primitive system **(cat)** displays the contents of a file whose name is passed as paramete rite a program where the parent process sends the contents of a file **(toto.txt)** to its child usin ; original cat. The child must display on the screen the contents of the file in capital letters.

Content of the file read by the father

Content displayed by the child

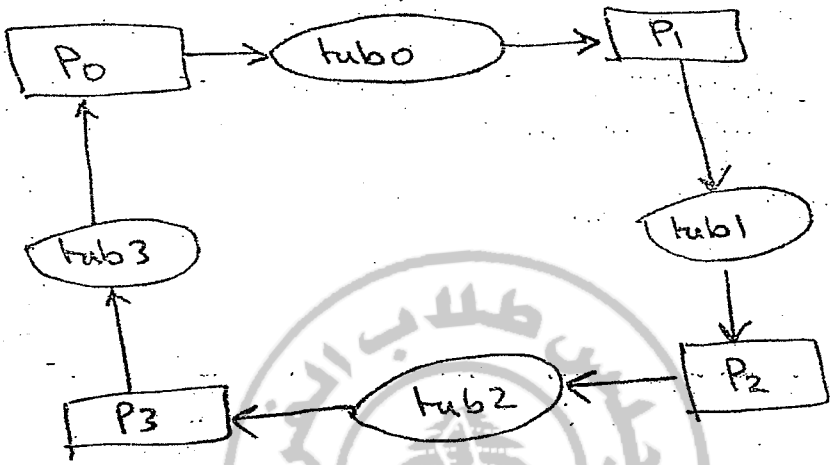
Good Luck

Problem I :

1) 16 processes (4 pts)

2) 2^{N+1} (4 pts)

Problem II (12 points)



1)
```
#define N 4
void proc(int)
int main()
{
  int i; int fd[N][2];                          (1 pt)
  for(i=0; i<N; i++)                            } (2 pts)
    pipe(fd[i]);
  for(i=0; i<N; i++)
    if (fork()==0)
      {
        dup2(fd[i][1], 1);                      }(4 pts)
        dup2(fd[(N+i-1)%N][0], 0);              }
        for(int j=0; j<N; j++)                  }
          { close(fd[j][0]);                    } (2 pts)
            close(fd[j][1]);                    }
      }
```

```
        fvoc(i);
        exit(0);
      }
   exit(0);
}
```

Problem III : (10 pts)

```
#include <stdio.h>
#include <string.h>
#define READ 0
#define write 1
int main()
{
  int pid, fd[2]; char *sentence; char *[illegible];
  if (pipe(fd) == -1)
    { perror("pipe failed");
      exit(1);
    }                                   (2 pts)
  if (((pid=fork()) < 0)
    { perror("fork failed");            (2 pts)
      exit(2);
    }
  if (pid != 0)
    { close(fd[Read]);                  (3 pts)
      dup2(fd[write],1);
      exec("cat", "cat", "toto.txt");
    }
  else { close(fd[write]);              (3 pts)
         while (read(fd[read], [illegible], 1) != E
```

```
    if (c >= 'a' && c <= 'z' && c != EOF)
        Sentence = Sentence & (c - 32);
    }
    printf("%s\n", Sentence);
}
```


إتحاد طلاب النفس
كلية العلوم

LEBANESE UNIVERSITY
Faculty of Sciences – Section V

Second Seme
Partial 2008-

Course Code : INFO 324 (English)
Instructors: Dr. Ahmad FAOUR/ Dr. Ihab SBEITY

Duration: 1 h

Problem I: (5 Points)

How many processes result from the execution of the following C program? Draw a that represents relations between the created processes.

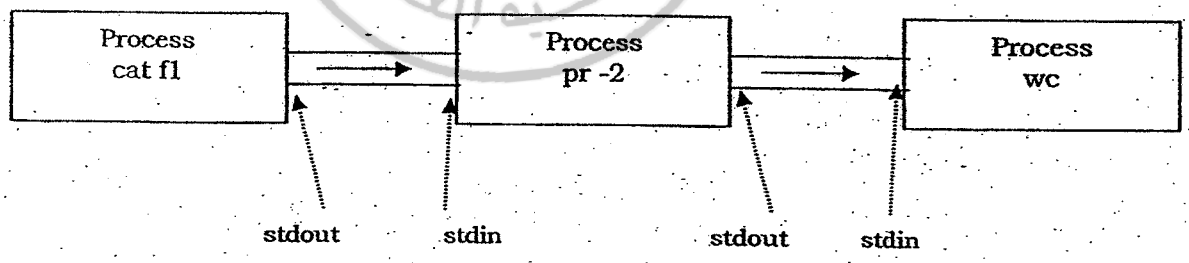
```
/* ========== prog.c ==========*/
void main() {
      int pid;
      fork();
      if (fork())
      {
            pid=fork();
           if (pid)
                 fork();
      }
}
```

Problem II: (13 Points)

Write a C program that executes the shell command

cat filename1 | pr -2 | wc

where *cat* allows to display the contents of a file given in parameter; *pr -2* prints the res on 2 columns and *wc* counts the number of lines, words and characters printed.



Problem III: (12 Points)

The problem consists in calculating the sum of a matrix elements. We decide to make parallel treatment of a matrix (L rows, C columns), where each row is treated by a differ process.

Write a C program where a father process creates L processes where each one calculate the sum of a row and send the result to its father. The father must calculate and display th global sum.

Good W