1. Sa se scrie un program care cauta in n fisiere text,ale caror nume sunt date ca argumente in linia de comanda,liniile care contin stringul “Cautare”.Pentru fiecare fisier se va creea un proces separat, care va afisa numarul de linii in care apare stringul.

[sourcecode lang="c"]  
#include   
#include   
#include   
#include   
int main(int argc, char\* argv[]) {  
int i,pid;  
for (i=1; i  
pid = fork();  
if (pid < 0) {  
printf("error");  
return 1;  
}  
if (pid==0) {  
break;  
}  
}  
int number =0;  
if (pid==0) {  
FILE\* fisier = fopen(argv[i],"r");  
if (fisier < 0) {  
printf("Error");  
}  
char linie[100];  
while ( fgets(linie,100,fisier) != 0) {  
if (strstr(linie,"Cautare") != 0){  
number+=1;  
}  
}  
printf("Procesul %d a gasit de %d ori Cautare in fisierul %s \n",getpid(),number,argv[i]);  
}  
return 0;  
}  
[/sourcecode]  
  
2. Sa se scrie 2 programe care comunica prin memorie partajata. La intervale aleatoare de timp, primul program citeste cate o linie de la intrarea lui standard si trimite celuilalt continutul acestei linii impreuna cu un semnal. La receptionarea mesajului, al doilea program ia linia din memoria partajata, elimina cifrele din ea si o tipareste pe iesirea lui standard.

Aici avem două fișiere, unul client:

[sourcecode lang="c"]  
//c2c.c

#include   
#include   
#include   
#include   
#include   
#include   
#include   
#include

int main() {  
int shmid = shmget((key\_t) 11180,1024,0644);  
if (shmid == -1 ) {  
perror(“eroare shmget”);  
return 1;  
}  
struct shmid\_ds shmid\_ds,\*info;  
info = &shmid\_ds;  
if (shmctl(shmid,IPC\_STAT,info) == -1 ) {  
perror(“eroare shmctl “);  
return 1;  
}  
char\* data = shmat(shmid,(void \*)0,0);  
if (data == (char \*)(-1)) {  
printf(“Eroare shmat”);  
return 1;  
}  
int oPid;  
oPid = info->shm\_cpid;  
printf(“Pidul creator este %d, pidul meu este %d, shmid este %d \n”,oPid,getpid(),shmid);  
char linie[100];  
int i=0;  
while (i<3) {  
printf("Dati o linie: ");  
gets(data);  
printf("Ati dat: %s",data);  
kill(oPid,SIGUSR1);  
int rand = 1;  
//rand = rand();  
sleep(rand);  
i++;  
}  
shmdt(data);  
}  
[/sourcecode]

Și server:

[sourcecode lang="c"]  
#include   
#include   
#include   
#include   
#include   
#include   
#include   
#include   
#include   
//asa trebuie declarate variabilele ca sa le poti modifica din handler  
volatile sig\_atomic\_t got\_usr1;  
void semnal\_handler(int sig) {  
write(1,”Handler apelat \n”,40);  
got\_usr1 =1;  
}

int main() {  
int shmid = shmget((key\_t) 11180,1024,0644|IPC\_CREAT);  
if (shmid == -1) {  
perror(“eroare shmget”);  
return 1;  
}  
char\* data = shmat(shmid,(void \*)0, 0);  
if (data == (char \*)(-1)) {  
perror(“Eroare shmat”);  
return 1;  
}  
signal(SIGUSR1,semnal\_handler);  
while(1==1) {  
if (got\_usr1) {  
printf(“server: %s \n”,data);  
int i=0, j=0;  
char rez[100];  
for (i; i< strlen(data)+1; i++) {  
while (data[i] == '0' || data[i] == '1' ||  
data[i] == '2' || data[i] == '3' ||  
data[i] == '4' || data[i] == '5' ||  
data[i] == '6' || data[i] == '7' ||  
data[i] == '8' || data[i] == '9')  
i++;  
rez[j]=data[i];  
j++;  
}  
got\_usr1=0;  
printf("server: %s \n",rez);  
}  
sleep(1);  
}  
shmdt(data);  
printf("server closing %d\n",shmid);  
shmctl(shmid,IPC\_RMID,NULL);  
}  
[/sourcecode]

3. Sa se scrie o pereche de programe care comunica prin coada de mesaje. Primul program citeste, la intervale aleatoare de timp, linii de la intrarea lui standard. Liniile cu lungimea intre 20 si 30 de octeti vor fi depuse in coada. Al doilea program citeste liniile din coada si le afiseaza la iesirea standard doar pe cele care incep cu o litara.

Și aici avem două fișiere: c.c  
[sourcecode lang="c"]  
#include   
#include   
#include   
struct msgbuff {  
long type;  
char string[30];

};

int main() {  
int msgid = msgget((key\_t) 11180, 0666|IPC\_CREAT);  
char linie[30];  
int i=5;  
struct msgbuff msg;  
msg.type = 1;  
while (i–) {  
printf(“Dati o linie:”);  
scanf(“%s”,linie);  
strcpy(msg.string,linie);  
msgsnd(msgid, &msg,strlen(linie),0);  
int rand =1;//rand=rand();  
sleep(rand);  
}

}

[/sourcecode]  
Și s.c  
[sourcecode lang="c"]  
#include   
#include   
#include   
#include   
struct msgbuff {  
long type;  
char string[30];

};

int main() {  
int msgid = msgget((key\_t) 11180, 0666);  
char linie[30];  
int i=5;  
struct msgbuff msg;  
while (i–) {  
printf(“Server\n”);  
if (msgrcv(msgid, &msg,30,1,0) == -1 ) {  
perror(“Msgrcv”);  
return 1;  
}  
if ((msg.string[0] >= 97 && msg.string[0] <= 122) ||  
(msg.string[0] >= 65 && msg.string[0] <=97 ))  
printf(“server: %s \n”,msg.string);  
else  
printf(“Nu incepe cu litera”);  
}

}  
[/sourcecode]