

Gebze Technical University
Computer Engineering

CSE 244
2017 Spring

MIDTERM REPORT

STUDENT NAME
LÜTFULLAH TÜRKER

STUDENT NUMBER
141044050

1. Problem Çözümüne Yaklaşım

2. Test Cases

3. Çalıştırma Ve Ekran Görüntüleri

1. Problem Çözümüne Yaklaşım

timerServer için :

void createNewMatrix(int n,struct timeval tStart,pid_t client):Yeni matrix oluşturmak ,determinantını almak ,FIFO ile matrisi client e göndermek,matrisin oluşturulma süresini,client pid sini ve determinantını log dosyasına yazmak için bu fonksiyonu kullanıyoruz.

void writeLog(double det,double time,pid_t client): Log dosyasına gelen 3 parametreyi yazıyor.Bu fonksiyon createNewMatrix() fonksiyonunda çağrılıyor.

double findDet(double a[20][20] , double k): gelen kxk lık matris'in determinantını alıp return eden fonksiyonumuz.Bu fonksiyon da createNewMatrix() fonksiyonu içerisinde ve mainde (det == 0) kontrolü için çağrılıyor.

int searchInPids(pid_t arr[200],pid_t search,int size): Gelen Pid arrayi içerisinde sıradan bir search işlemi uyguluyor.Bulunursa 1 ,bulunamazsa 0 return ediyor.

static void sigHandler (int sigNo, siginfo_t *siginfo, void *context): Sinyal handler fonksiyonu.SIGINT ve SIGUSR1 sinyallerini handle ediyor.İçinde if ile hangi sinyal geldiyse ona göre işlemler yapıyor.SIGUSR1 sinyali geldiğinde static tanımlı olan pid arrayine sinyali gönderen prosesin pid sini yazıyor.Tabi yazmadan önce o proses daha önceden zaten handle edilmeyen sinyal göndermiş mi diye bakıyoruz (searchInPids fonksiyonu ile).Eğer ilk defa geliyorsa pid arrayine gelen pid yi ekliyor ve kullanılan array boyutunu 1 artırıyoruz.SIGINT sinyali geldiğinde ise ekrana sinyal geldiğine dair mesaj basıp static olan sinyal flag değişkenini 1 yapıyoruz ve aşağıdaki işlemlerde sinyal değişkeni if(!sigFlag) şeklinde kullanıldığı için işlemler için alınan yerler açılan dosyalar kapatılıp ve diğer proseslere (seeWhat,showResults) haber verilip programdan çıkılıyor.Haber verme işlemini client pid si elimizde olduğu için sinyal gelince client pid sine de kill fonksiyonu ile sinyal göndererek çıkabiliyoruz.clientde de showResult ile bir fifo bağlantısı olduğu için client sonlanırken showResults u da sonlandırıyor.Böylece 3 ü birden sonlanmış oluyor.Yukarıda bahsedilen static değişkenler :

static volatile sig_atomic_t sigFlag = 0; ➔ SIGINT sinyal kontrol flag'ı

static volatile sig_atomic_t sigClientFlag = 0; ➔ SIGUSR1 sinyal kontrol flag'ı

pid_t client_arr[200]; ➔ client Pid arrayi

static int used = 0; ➔ Arrayin kullanılan boyutu (size)

main();

Matrislerin oluşturulma süresini programın başlama süresinden itibaren alacağımız için programın başında o anki zamanı `gettimeofday` ile alıyoruz. Usage kontrolü ve matrix order kontrolü yapıyoruz. Sinyal handler oluşturuyoruz. fork yaparak bir child oluşturuyoruz. Bu proses yeni client eklenmesi durumlarında sürekli pid verebilmek için program süresince server in (parent ın) pid sini `mainFifo (argv[3])` ya yazıp siliyor. Her yazdığında silme işlemini bir süre sonra fifo çok fazla dolmasın diye yapıyorum. Bu sırada ana proses `SIGINT` 'in gelmeme şartının olduğu bir while döngüsünde gelen sinyalleri maskeleyip bloklayıp `argv[1]` de verilen milisaniye kadar beklettikten sonra `unblock` yapıyor ve biriken client Array ini for döngüsünde `used` a kadar giderek her dönüşte fork yaparak matris oluşturmaya gönderiyor. Ana döngünün sonunda server, client lerinden birine `SIGINT` gelmiş mi diye arrayi geziyor ve birine bile geldiyse `exit` yapıyor.

seeWhat için :

Convolution için kernel matris i derste verilen 2. Matris olarak seçtim.

Inverse ve convolution almak için kullanılan cofactor, transpose, convolution ve determinant fonksiyonları mevcuttur. Bu fonksiyonlarda internette fazlasıyla yararlanılmıştır. Yararlandığım kaynakların linkleri kodun o bölgesinde yorum olarak yazıyor. Sinyal handler fonksiyonu var. serverdaki gibi `SIGINT` gelince static flag i 1 yapıyor.

`void fillNxNMatrix(double matrix[20][20], double matInv[10][20], int whichRegion, int n);` Gelen $2n \times 2n$ matrisi `whichRegion` parametresine göre $n \times n$ lik bir matrisini oluşturup `matInv` matrisine atayan fonksiyon.

`void fillLog(double matrix[20][20], int n, FILE* matrixLog);` gelen matrisi matlab formatına uygun şekilde log dosyasına yazan fonksiyon.

`void sendShowResult(int tmpFifo, double result, double timedifInverse);` `showResults` a gönderilen fifo ya gelen diğer 2 parametreyi yazan fonksiyon.

main();

En başta while döngüsünde `mainFifo` yu (`argv[1]`) open yapıyoruz. Fifodan server in pid sini alıp kapatıyoruz. **Matrixleri serverdan almak için fifo oluşturuyoruz ve bu fifonun adını (Client%d, getpid()) koyuyoruz Fifonun ismine clientin pid sini vermemin sebebi birden fazla client aynı anda matris istediğinde aynı fifo ya yazmaması gerekiyor. (Çakışma olmaması için) ve server ın da client sinyal gönderdiğinde clientin pid sini alıp kaydettiği ve iki taraf da fifonun adına sahip olduğu için.**

`showResults` un da clientten bilgi ve onun pid sini alabilmesi için 2 tane dosya oluşturuyoruz. Birisi oluşan tüm clientlerin pid sini tutacak. (Her client en başta kendi pid sini bu dosyaya yazacak.) Diğer ise fifo ile log için gerekli time ve result bilgilerini `showResults` a aktaracak.

Ana döngü yine sinyal flagı 1 olmadığı sürece dönecek ve en başta servera matris göndermesi için istekte bulunacak (`SIGUSR1` sinyali gönderecek). sinyali alan server `createNewMatrix` fonksiyonu ile random matrix oluşturup fifo ile bize gönderecek. **Her matris için gerekli bilgiler hesaplanıp bir log dosyası oluşturmamız gerekiyor. Bu log dosyasının adı her matris de ve de her clientde çakışmavacak şekilde farklı olması gerekiyor.**

Bu problemi çözüm yöntemim → (“Client%dMatrix%d.log”,getpid(),++logCount)

Log un adında pid var .Çünkü her clientle karşılaşması için.

Logun adında global bir count var .Her matrisde adının farklı olması için.

Matrix serverdan alındıktan sonra elimizde arraye dolduruyoruz ve matris işlemlerine başlıyoruz.

Inverse alma işlemini fork yaparak bir child prosese yaptırıyorum ve içeride ayrılan her nxn lik matrisin determinant=0 kontrolü gibi kontrolleri yapıyor.fork un en başında hesaplama süresini bulmak için start time alıyoruz ve bitince de end alıp inverse alma süresini elde ediyoruz.child prosesden çıkmadan önce elde ettiğimiz result1 ve timeElapsed sayılarını fifo ile showResults a gönderiyoruz ve child a exit yapıyoruz.

Daha sonra bu işlemler yapılırken convolution için de fork yapıyor ve işlemleri child prosese yaptırıyoruz.convolution sonucunda da süre ve result2 hesaplanıyor ve fifo ile showResults a aktarılıyor.bu aktarma işlemini inverse ile convolution yani farklı prosesler yaptığı için ortaya senkronizasyon problemi çıkıyor.Önce hangisinin yazacağı tesadüfi olacağı için ikisinin arasında parent wait yapıyor ve biri bitmeden diğeri fifoya yazmamış oluyor.İşlemler bittiğinde gerekli yerlere gerekli bilgiler fifo ile gönderildi ve matrixin log dosyası oluşturuldu ,matlab formatında 3 matris de yazıldı.hepsi bitince yeni bir matrix istemek için döngü başa dönüyor ve server a sinyal gönderiliyor.SIGINT sinyali yani ctrl+c gelmediği sürece bu döngü devam ediyor.Bu prosesde ctrl+c yaparsanız showResuts da kapanır çünkü showResults bilgi aldığı pid ye sürekli çalışan aktif bir proses mi diye kontrol ederek hareket ediyor ve hareket durduğunda kendisi de duruyor.

showResults :

void writeLog(int cPid,double result1,double result2,double time1,double time2); Aldığı parametrelerdeki bilgileri showResults.log dosyasına yazar.

Bu program da bir sinyal handlera sahip ve diğerleri gibi SIGINT sinyali geldiğinde static flag ini 1 yapıyor.

main();

mainde önce client lerden pid lerini aldığımız dosyayı açıyoruz.Açtıktan sonra şartı dosya açık olduğu,sinyal gelmediği ve clientin prosesi durmadığı sürece devam etmek olan bir döngü içinde işlemlere başlıyoruz.Okuma sonucunda dosyadan her yeni pid çekildiğinde fork yaparak child oluşturuyoruz.

Her Child bir client i temsil ediyor.Dolayısıyla her child da client kapanana veya sinyal gelene kadar oluşan her matrisin bilgilerini clientin açmış olduğu fifo ile alıyor ve writeLog fonksiyonunu çağırarak her matrisi log dosyasına yazıyor.Bu aktarımın yapıldığı fifonun adı da yine her client için ayrı olması için pid içeriyor (“temp%d”,client_pid) .Her matris alındıktan sonra bu fifo boşalıp yenisi gelmesi için client de silinip tekrar oluşturuluyor ve yeni matris geliyor.

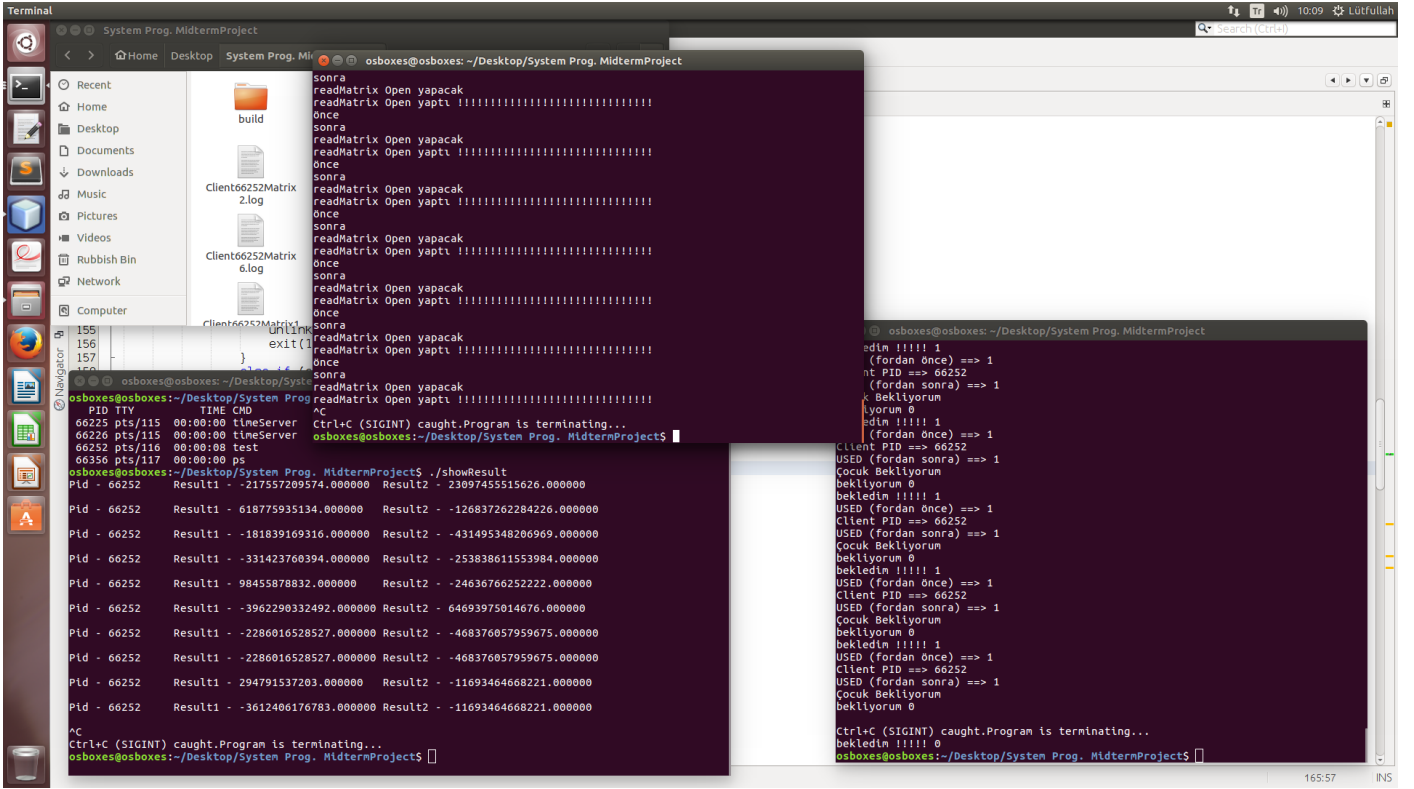
CONVOLUTION

Convolution en çok kullanılan Görüntü İşleme tekniklerinden biridir. Fotoğraflarda gördüğümüz filtreler soldurma, blurlaştırma, keskinleştirme, kenar bulma işlemleri convolution matrisleri ile yapılır. Convolution işlemi $n \times n$ bir resim üzerinde $n \times n$ bir maskenin sol üst köşeden başlanarak, maskenin merkezi her bir pixel üzerinden geçecek şekilde bütün resmi tarama işlemidir. Bu tarama işlemi sırasında maske içerisinde kalan her pixel maskenin katsayıları ile çarpılıp bu çarpımların toplamı, yeni resimde maskenin merkezinin geldiği konuma yazılır.

2. Test Cases

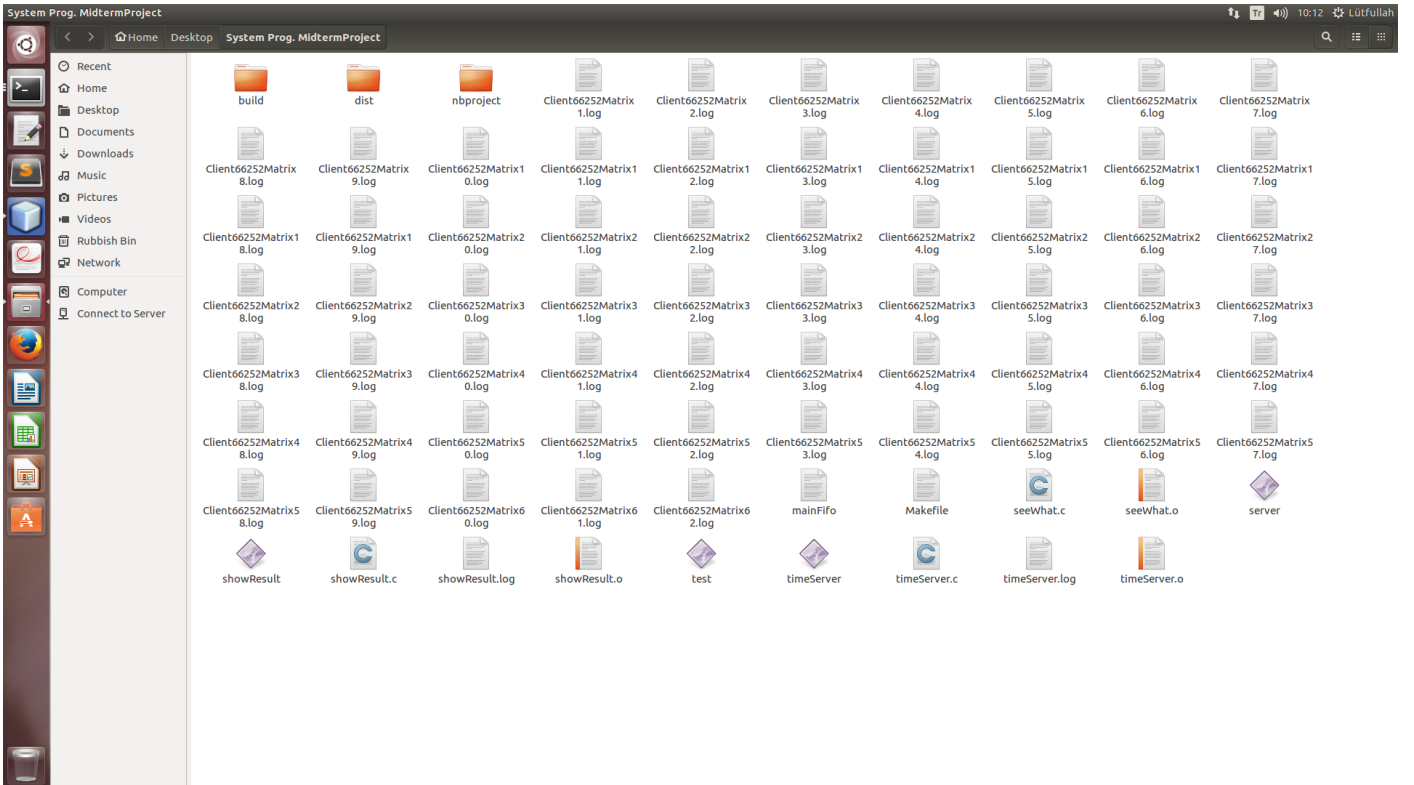
Senaryo	Test	Beklenen Sonuç	Gerçek Sonuç
3 program aynı anda çalışır. Her biri kendilerine ait olan loglara belli bilgiler yazarlar.	3 programı aynı anda çalıştırmak yerine sadece seeWhat ve timerServer 1 çalıştırınca ne olur ?	İki program sürekli çalışır. 3. program eğer showResult ise açık olması gerekmez.	İki program da normal olarak çalışır. SeeWhat showResult un alması için bilgileri fifo'ya yazar ama okuyacak showResultun olmaması bir sorun oluşturmaz. showResult.log dosyası olmadan iki programda sorunsuz çalışır.
seeWhat çalışmak için timerServer a ihtiyaç duyar showResults da çalışmak için seeWhat a ihtiyaç duyar.	seeWhat ın şartı olan timerServer çalıştırılmadan seeWhat tek başına çalıştırılırsa veya aynı olay showResults ve seeWhat arasında olsa ne olur ?	Birinin çalışması için diğerinin de çalışması şart olduğu için çalıştırılınca hata verip çıkar.	timerServer çalıştırılmadan seeWhat çalıştırılırsa seeWhat sonsuzda timerServer in çalıştırılmasını bekler. Aynı şekilde showResults da seeWhat ı bekler. Yani hata verip çıkmaz. Gerekli şartın sağlanmasını bekler.
3 program da sinyal gelmediği sürece sonsuza kadar çalışır.	3 ü de çalışırken sadece 1 ine sinyal gönderilirse ne olur ?	Sinyal gönderilmiş olan işlem sonlanır. Diğerleri için sonlanan işlem ön şart ise onlar da hata verip sonlanır.	Birine sinyal gönderilince sinyal gönderilen işlem diğerleriyle iletişime geçerek onlara da sonlanması için sinyal gönderir veya sonlanması için diğerinin ihtiyacı olan dosyaları kapatır ve hepsi birlikte hatasız şekilde tüm açılan dosyalar alınan yerler geri verilerek sonlanır.

3. Çalıştırma ve Ekran Görüntüleri



```
osboxes@osboxes: ~/Desktop/System Prog. MidtermProject
$ ./showResult
Pid - 66252 Result1 - 618775935134.000000 Result2 - -126837262284226.000000
Pid - 66252 Result1 - -181839169316.000000 Result2 - -431495348206969.000000
Pid - 66252 Result1 - -331423760394.000000 Result2 - -253838611553984.000000
Pid - 66252 Result1 - 98455878832.000000 Result2 - -24636766252222.000000
Pid - 66252 Result1 - -3962290332492.000000 Result2 - 64693975014676.000000
Pid - 66252 Result1 - -2286016528527.000000 Result2 - -468376057959675.000000
Pid - 66252 Result1 - -2286016528527.000000 Result2 - -468376057959675.000000
Pid - 66252 Result1 - 294791537203.000000 Result2 - -11693464668221.000000
Pid - 66252 Result1 - -3612406176783.000000 Result2 - -11693464668221.000000
$ Ctrl-C (SIGINT) caught.Program is terminating...
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$
```

Test etmek için printler olan bir çalışma



Kısa süreli bir çalışma sırasında program klasörünün akışı

Files

```
osboxes@osboxes: ~/Desktop/System Prog. MidtermProject
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall seeWhat.c
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o test seeWhat.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./test mainFifo
```

System Prog. MidtermProject

Client68489Matrix1 7.log 8.log 9.log
Client68489Matrix1 00.log
Client68489Matrix1 01.log
Client68489Matrix1 02.log
Client68489Matrix1 03.log
Client68489Matrix1 04.log
Client68489Matrix1 05.log
Client68489Matrix1 06.log
mainFifo
Makefile
pidTemp
seeWhat.c
seeWhat.o
server
showResult
showResult.c

```
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall timeServer.c
timeServer.c: In function 'main':
timeServer.c:148:21: warning: 'mainFifo' may be used uninitialized in this function [-Wmaybe-uninitial
zed]
    close(mainFifo);
    ^
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o timeServer timeServer.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./timeServer 300 3 mainFifo
```

PID	TTY	TIME	CMD
66226	pts/115	00:00:00	timeServer
68471	pts/115	00:00:00	timeServer
68472	pts/115	00:00:00	timeServer
68489	pts/117	00:00:03	test
68538	pts/116	00:00:03	showResult
68539	pts/116	00:00:00	showResult
68583	pts/80	00:00:00	ps

```
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ps -a
PID TTY TIME CMD
66226 pts/115 00:00:00 timeServer
68471 pts/115 00:00:00 timeServer
68472 pts/115 00:00:00 timeServer
68489 pts/117 00:00:11 test
68538 pts/116 00:00:16 showResult
68539 pts/116 00:00:01 showResult
68753 pts/80 00:00:00 ps
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$
```

Result1	Result2
5915870229192.000	541146680552175.000
1412889572474.000	355004528346170.000
4133004608064.000	75207393986004.000
1978924064264.000	125203815375421.000
25057153413.000	17914794560481.000
245538579852.000	135681352384477.000
906904736340.000	38888856909750.000
297772487707.000	5437343516309.000
594977145898.000	129578691693804.000
655639161993.000	68442069462005.000
499536634926.000	28643507075658.000
499536634926.000	28643507075658.000
342834201588.000	331417601583050.000
2092994999135.000	210141772763904.000

Files

```
osboxes@osboxes: ~/Desktop/System Prog. MidtermProject
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall seeWhat.c
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o test seeWhat.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./test mainFifo
```

System Prog. MidtermProject

Client68489Matrix1 8.log 9.log 00.log
Client68489Matrix1 01.log
Client68489Matrix1 02.log
Client68489Matrix1 03.log
Client68489Matrix1 04.log
Client68489Matrix1 05.log
Client68489Matrix1 06.log
Client68489Matrix1 07.log
Client68489Matrix1 08.log
Client68489Matrix1 09.log
Client68489Matrix1 10.log
Client68489Matrix1 11.log
Client68489Matrix1 12.log

```
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall timeServer.c
timeServer.c: In function 'main':
timeServer.c:148:21: warning: 'mainFifo' may be used uninitialized in this function [-Wmaybe-uninitial
zed]
    close(mainFifo);
    ^
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o timeServer timeServer.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./timeServer 300 3 mainFifo
```

PID	TTY	TIME	CMD
66226	pts/115	00:00:00	timeServer
68471	pts/115	00:00:00	timeServer
68472	pts/115	00:00:00	timeServer
68489	pts/117	00:00:11	test
68538	pts/116	00:00:03	showResult
68539	pts/116	00:00:00	showResult
68583	pts/80	00:00:00	ps

```
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ps -a
PID TTY TIME CMD
66226 pts/115 00:00:00 timeServer
68471 pts/115 00:00:00 timeServer
68472 pts/115 00:00:00 timeServer
68489 pts/117 00:00:12 test
68538 pts/116 00:00:17 showResult
68539 pts/116 00:00:01 showResult
68760 pts/80 00:00:00 ps
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$
```

Result1	Result2
6342608094402.000	337353539305504.000
2221469398973.000	337353539305504.000
818492441892.000	1255876234804170.000
78397840719.000	1255876234804170.000
236421239649.000	40152389964971.000
191314603322.000	74223516444886.000
2941202364753.000	74223516444886.000
2941202364753.000	74223516444886.000
446302607759.000	15767912587427.000
446302607759.000	15767912587427.000
4698177566876.000	15767912587427.000
1260356207794.000	31499509677749.000
2948552592160.000	10645804233514.000
2321750388708.000	153189102352338.000


```
osboxes@osboxes: ~/Desktop/System Prog. MidtermProject
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall seeWhat.c
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o test seeWhat.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./test mainFifo
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ^C
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$

osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -c -pedantic -errors -Wall timeServer.c
timeServer.c: In function 'main':
timeServer.c:148:21: warning: 'mainFifo' may be used uninitialized in this function [-Wmaybe-uninitial
ized]
    close(mainFifo);
    ^
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ gcc -o timeServer timeServer.o -ln
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ./timeServer 300 3 mainFifo
Ctrl+C (SIGINT) caught. Program is terminating...
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$

System Prog. MidtermProject
9.log 00.log 01.log
Client68489Matrix1 02.log Client68489Matrix1 03.log Client68489Matrix1 04.log
Client68489Matrix1 05.log Client68489Matrix1 06.log Client68489Matrix1 07.log
Client68489Matrix1 08.log Client68489Matrix1 09.log Client68489Matrix1 10.log
Client68489Matrix1 11.log Client68489Matrix1 12.log Client68489Matrix1 13.log
Client68489Matrix1 Client68489Matrix1 Client68489Matrix1

osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ps -a
PID TTY TIME CMD
66226 pts/115 00:00:00 timeServer
68471 pts/115 00:00:00 timeServer
68472 pts/115 00:00:00 timeServer
68489 pts/117 00:00:03 test
68538 pts/116 00:00:03 showResult
68539 pts/116 00:00:00 showResult
68583 pts/80 00:00:00 ps
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ps -a
PID TTY TIME CMD
66226 pts/115 00:00:00 timeServer
68471 pts/115 00:00:00 timeServer
68472 pts/115 00:00:00 timeServer
68489 pts/117 00:00:11 test
68538 pts/116 00:00:16 showResult
68539 pts/116 00:00:01 showResult
68753 pts/80 00:00:00 ps
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$ ps -a
PID TTY TIME CMD
66226 pts/115 00:00:00 timeServer
68471 pts/115 00:00:00 timeServer
68472 pts/115 00:00:12 test
68489 pts/117 00:00:12 test
68538 pts/116 00:00:17 showResult
68539 pts/116 00:00:01 showResult
68760 pts/80 00:00:00 ps
osboxes@osboxes:~/Desktop/System Prog. MidtermProject$
```

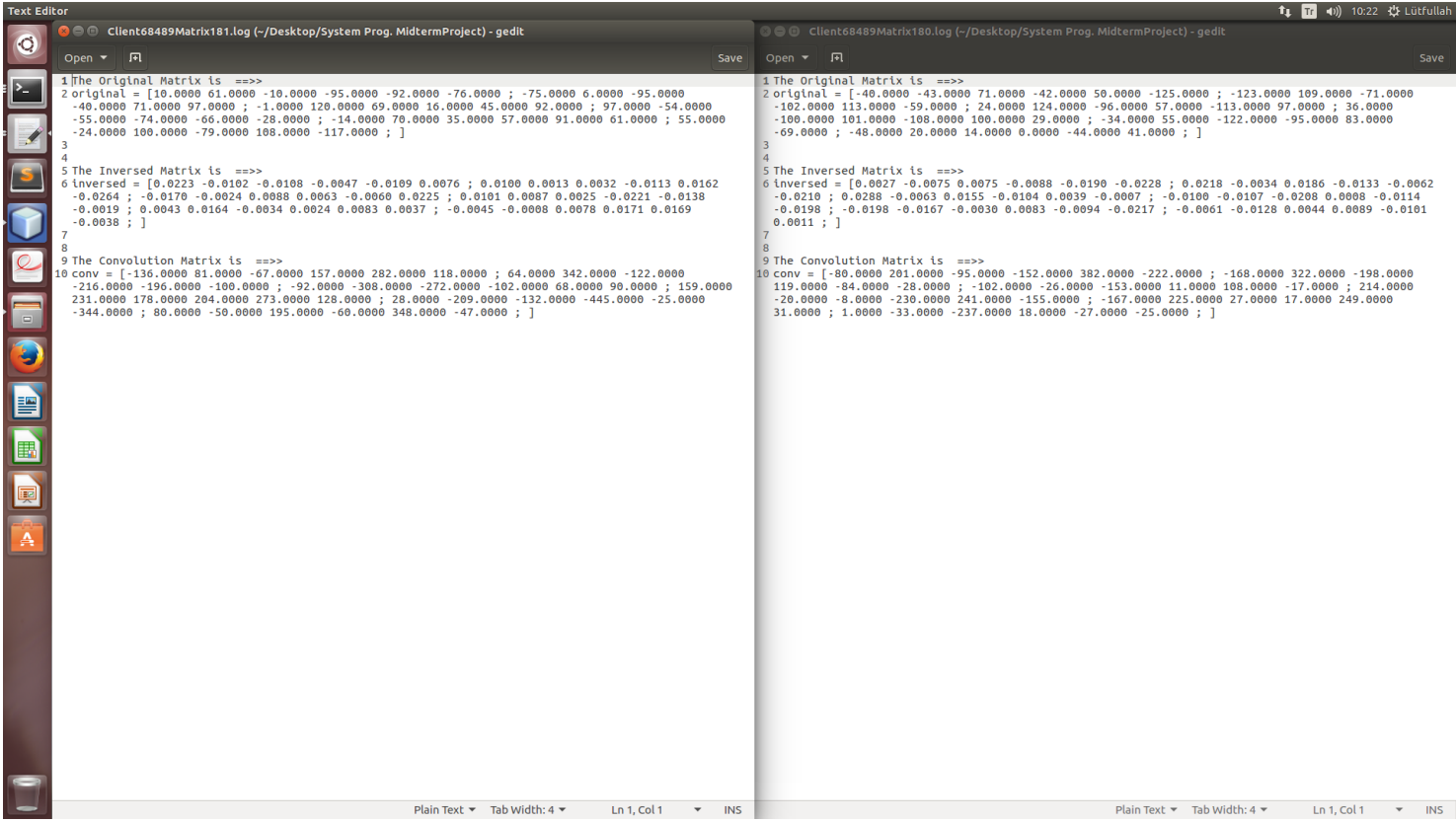
Normal bir çalışma sırasında 3 programın akışı

Görüldüğü gibi programlar akarken ps -a komutuyla çalıştırılan işlemlere bakınca sadece program süresince fofoları kontrol ettiğimiz childlar ile birlikte 2 şer proses gözüküyor. Açılan her dosya ,oluşturulan her child kapatılıyor.

```
timeServer.log (-/Desktop/System Prog. MidtermProject) - gedit
200 42798.509 68489 483664369839
201 43091.651 68489 -140137103600
202 43394.075 68489 -818492441892
203 43096.281 68489 -2695034173888
204 43998.169 68489 -3067672617627
205 44299.174 68489 -78397840715
206 44600.078 68489 -428597930482
207 44901.180 68489 -382491273444
208 45204.192 68489 236421239649
209 45505.630 68489 259374158343
210 45808.870 68489 1977616967492
211 46110.059 68489 191314603322
212 46411.725 68489 -2204148160782
213 46713.211 68489 4589235675080
214 47015.904 68489 2941202364753
215 47316.861 68489 6932937129410
216 47618.121 68489 -680393745681
217 47918.860 68489 -1211327626305
218 48219.586 68489 1162042452844
219 48520.525 68489 8116199539290
220 48821.723 68489 -446302607759
221 49122.915 68489 358260528848
222 49424.359 68489 -2582435123496
223 49725.071 68489 1765841307579
224 50028.363 68489 -1487408251407
225 50329.649 68489 3493310494898
226 50631.386 68489 4698177560876
227 50932.427 68489 4673693835566
228 51239.585 68489 -382392127682
229 51542.142 68489 1260356207794
230 51843.137 68489 2575266301792
231 52143.873 68489 4334506155594
232 52446.854 68489 -2948525292160
233 52747.800 68489 416590390538
234 53048.775 68489 -4336735558888
235 53350.145 68489 2321750388708
236 53651.126 68489 1473034490507
237 53952.100 68489 -4173994103426
238 54253.255 68489 -1540717950488
239 54554.351 68489 -744170911542
240 54856.151 68489 -16790954587
241 55157.748 68489 1739908556619
242 55459.119 68489 1616993170792
243 55759.938 68489 -2992974195576
244 56061.442 68489 413478924906
245 56362.580 68489 -2819183852456
246 56665.016 68489 165674441183
247 56965.924 68489 1511170561388
248 57266.795 68489 -2442531711936
249 57568.436 68489 -2088372987962
250 57869.744 68489 -4283030997414
251 58171.163 68489 962645364656
252 58472.702 68489 10831588017498
253 58773.814 68489 -1573908609934
254 59074.793 68489 3394187682952
255 59377.579 68489 -854391412028

showResults.log (-/Desktop/System Prog. MidtermProject) - gedit
150 Result 1 = -6342680894402.0000 Elapsed Time = 0.1730
151 Result 2 = -337353539305504.000000 Elapsed Time = 0.0790
152
153 Pid of Client ==> 68489
154 Result 1 = -2221469398973.0000 Elapsed Time = 0.1160
155 Result 2 = -337353539305504.000000 Elapsed Time = 0.0790
156
157 Pid of Client ==> 68489
158 Result 1 = -818492441892.0000 Elapsed Time = 0.1610
159 Result 2 = -1255876234804170.000000 Elapsed Time = 0.1120
160
161 Pid of Client ==> 68489
162 Result 1 = -78397840719.0000 Elapsed Time = 0.1610
163 Result 2 = -1255876234804170.000000 Elapsed Time = 0.1120
164
165 Pid of Client ==> 68489
166 Result 1 = 236421239649.0000 Elapsed Time = 0.1380
167 Result 2 = 40152389964971.000000 Elapsed Time = 0.1350
168
169 Pid of Client ==> 68489
170 Result 1 = 191314603322.0000 Elapsed Time = 0.1610
171 Result 2 = -74223516444886.000000 Elapsed Time = 0.1100
172
173 Pid of Client ==> 68489
174 Result 1 = 2941202364753.0000 Elapsed Time = 0.1560
175 Result 2 = -74223516444886.000000 Elapsed Time = 0.1100
176
177 Pid of Client ==> 68489
178 Result 1 = 2941202364753.0000 Elapsed Time = 0.1560
179 Result 2 = -74223516444886.000000 Elapsed Time = 0.1100
180
181 Pid of Client ==> 68489
182 Result 1 = -446302607759.0000 Elapsed Time = 0.1310
183 Result 2 = -15767912587427.000000 Elapsed Time = 0.1430
184
185 Pid of Client ==> 68489
186 Result 1 = -446302607759.0000 Elapsed Time = 0.1310
187 Result 2 = -15767912587427.000000 Elapsed Time = 0.1430
188
189 Pid of Client ==> 68489
190 Result 1 = 4698177560876.0000 Elapsed Time = 0.7050
191 Result 2 = -15767912587427.000000 Elapsed Time = 0.1430
192
193 Pid of Client ==> 68489
194 Result 1 = 1260356207794.0000 Elapsed Time = 0.1290
195 Result 2 = 31499509677749.000000 Elapsed Time = 0.0950
196
197 Pid of Client ==> 68489
198 Result 1 = -2948552592160.0000 Elapsed Time = 0.1460
199 Result 2 = -10645804233514.000000 Elapsed Time = 0.0780
200
201 Pid of Client ==> 68489
202 Result 1 = 2321750388708.0000 Elapsed Time = 0.1510
203 Result 2 = 153189102352338.000000 Elapsed Time = 0.1100
204
205 Pid of Client ==> 68489
```

Çalıştırma sonucu oluşan timerServer.log ve showResults.log dosyaları



Çalıştırma sonucu oluşan 100lerce matris arasından 2 matrisin matlab formatında yazılmış log dosyaları