ED 5
Système multi-tâches sous UNIX

On se propose de voir comment les principes d'un système multi-tâches vus aux Exercices Dirigés précédents sont mis en œuvre par UNIX. On dispose pour cela de deux programmes⊡

`calcul`: qui effectue uniquement des calculs trigonométriques

`disque`: qui écrit uniquement un gros fichier sur le disque

Le source de ces deux programmes est donné ci-dessous. Les constantes numériques ont été choisies afin de rendre la durée de ces programme assez proche (30s environ sur un PC).

```
/*--- calcul.c ----*/
#include <math.h>
main(){
double s,x;
int i,j;

for (j=1;j<=2000;j++){
 s=0;
 for(i=1;i<=50000;i++){
   x=sin(3.141592654*s); x=cos(3.141592654*s); x=tan(3.141592654*s); s+=0.1;
 }
}
}

/*--- disque.c ----*/
#include <fcntl.h>
#define BSIZE 8192
#define FSIZE 15000
char buf[BSIZE];
int fdes,i,j;

main(){
 for (j=0;j<6;j++){
   fdes=open("./fichiertest",O_WRONLY|O_CREAT,0777);
   for (i=0;i<FSIZE;i++) write(fdes,buf,BSIZE);
   close(fdes);
 }
}
```

Exercice 1⊡Gestion des processus

La commande `ps` permet de connaître quels sont les processus en cours de traitement. L'option ps -u affiche (entre autre) les informations suivantes⊡

- USER : l'utilisateur qui a lancé le processus

- PID : le Process IDentification number. Chaque processus a son propre PID.

- %CPU, %MEM : Le taux d'utilisation de la CPU et de la mémoire pour ce processus

- STAT: Le statut du processus.

- TIME : la quantité totale de temps CPU consommé

Les cas les plus courants de STAT sont⊡

R En cours d'exécution (ie, dans la "run queue" - pas en attente)

S Le processus est dormant ("sleeping") depuis moins de 20s

I Le processus est dormant ("idle") depuis plus de 20s

(S et I sont souvent en attente de saisie au clavier.)

Question 1

L'interpréteur de commandes permet de lancer des commandes en "arrière plan", ce qui permet à l'utilisateur de lancer une nouvelle commande sans attendre la fin de la première. Il suffit pour cela de terminer la commande par le caractère `&`. On peut également lancer plusieurs commandes en séquence en les séparant par un point-virgule.

Lancer en parallèle 3 processus `calcul` par la commande⊡

```
calcul&;calcul&;calcul&
```

puis lancer `ps -u` régulièrement (la commande !! permet de répéter la dernière commande sans la retaper)

Interpréter les résultats affichés

Question 2

On souhaite voir l'évolution du statut d'un processus (avec le champs `STAT` de `ps`).

Lancer un processus calcul, puis l'interrompre avec `CTRL-Z`.

Que devient son statut ?

Lancer la commande `bg`

Que se passe-t'il ?

Détruisez le processus avec la commande `kill -9` suivie du PID du processus

Question 3

L'option -`aux` de `ps` permet de visualiser tous les processus en cours de traitement sur la machine.

Commenter cette liste en identifiant les processus système

Exercice 2 : Parallélisme calcul et E/S

La commande `time` permet de connaître le temps d'exécution d'un programme. La version de `time` utilisée ici affiche dans l'ordre les résultats suivants⊡

- U: (user) temps CPU pour le processus (précision 0.5s)

- S: (system) temps CPU que le noyau a consacré au processus (précision 0.5s)

- E: (elapsed) temps total d'exécution du processus (précision 1s)

- P: rapport (U+S)/E en pourcentage (parfois >100% du fait de l'imprécision)

Question 1

Lancer `time calcul`

Interpréter les résultats affichés par `time`*. Répéter une ou deux fois l'opération. Que constate-t'on ?*

Question 2

Lancer `calcul&;time calcul`

Interpréter les résultats. Calculer le taux d'occupation de la CPU.

Question 3

Lancer `time disque`

Calculer le temps passé en E/S

Question 4

Effacer le fichier produit et lancer `time calcul&;time disque &`

Interpréter les résultats

Exercice 3 L'écroulement (thrashing)

Les programmes `gentil et mechant` ci-dessous utilisent une matrice de 24000*4096 caractères (codés sur un octet), soit 96 Mo.

```
/*--- gentil.c ---- */
#include <math.h>
#define NBRE 24000
#define PAGE 4096
char t[NBRE][PAGE];
main(){
double s,x;
int i,j,k;
  for (j=0;j<NBRE;j++)
     for (k=0;k<PAGE;k++) t[j][k]=(int)sin(3.141592654);
}
/*--- mechant.c ---- */
#include <math.h>
#define NBRE 24000
#define PAGE 4096
char t[NBRE][PAGE];
main(){
double s,x;
int i,j,k;
 for (k=0;k<PAGE;k++)
  for (j=0;j<NBRE;j++) t[j][k]=(int)sin(3.141592654);
}
```

Si la mémoire physique libre est limitée à 256 Mo, le système ne pourra pas gérer plus de 2 ou 3 processus sans faire fréquemment appel à la mémoire virtuelle. Quand toute la mémoire physique est occupée, le gestionnaire de mémoire virtuelle doit choisir des pages (ici, des morceaux de 4096 octets) à retirer, qu'il recopie sur le disque. En gros, c'est la page la moins récemment utilisée qui est choisie (algorithme LRU).

Question 1

Déterminer à l'aide de la commande `free`*, la quantité de mémoire physique disponible et la taille de l'espace disque disponible pour la mémoire virtuelle. (variable selon le type de PC utilisé)*

En déduire le nombre de processus MAX pouvant s'exécuter sans appels fréquents à la mémoire virtuelle.

Question 2

La commande `vmstat` permet de voir l'évolution des processus et du système dans le temps. Elle affiche à l'intervalle de temps choisi divers paramètres utiles pour cet exercice□

- r : nombre de processus "runnable" (`vmstat` lui-même n'est pas compté)
- b: nombre de processus bloqués en attente disque
- free : mémoire libre disponible (en Ko)
- si: "swap in" lecture de pages sur le disque (en Ko/s)
- so: "swap out" écriture de pages sur le disque (en Ko/s)
- id: "idle" pourcentage de temps pendant lequel la CPU est inutilisée

Lancer MAX+1 processus `gentil` *en parallèle avec la commande vmstat 1 (le 1 signfie 1 affichage par seconde)*

Commenter l'évolution des champs r, b, free, si, so, id

Calculer le taux d'occupation du CPU (cf Exercice 2, question 2)

Question 3

Idem avec MAX+1 processus `mechant`

Pourquoi y a t-il de telles différences ?

Question 4

Idem avec MAX+2 processus `gentil` puis `mechant`

Que constate-t'on?

ED 5
Complément

Exercice 1⊡Gestion des processus

Question 1

```
> ps -u
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
jb         4020  0.0  0.6   4400  1916 tty1     Ss   14:14   0:00 -bash
jb         5190  0.0  0.3   2480   848 tty1     R+   14:22   0:00 ps -u
```

On lance 3 calculs en parallèle⊡

```
> ca&;ca&;ca&
[1] 5629
[2] 5630
[3] 5631
> ps -u
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
jb         5590  0.1  0.6   4400  1892 tty2     Ss   14:29   0:00 -bash
jb         5629 32.7  0.1   1588   296 tty2     R    14:29   0:13 ca
jb         5630 31.0  0.1   1588   296 tty2     R    14:29   0:12 ca
jb         5631 29.3  0.1   1584   292 tty2     R    14:29   0:11 ca
jb         5677  0.0  0.3   2484   848 tty2     R+   14:30   0:00 ps -u

> !!
ps -u
USER        PID %CPU %MEM SIZE  RSS TTY STAT START   TIME COMMAND
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
jb         5590  0.1  0.6   4400  1892 tty2     Ss   14:29   0:00 -bash
jb         5629 33.2  0.1   1588   296 tty2     R    14:29   0:16 ca
jb         5630 31.7  0.1   1588   296 tty2     R    14:29   0:16 ca
jb         5631 30.4  0.1   1584   292 tty2     R    14:29   0:15 ca
jb         5678  0.0  0.3   2480   852 tty2     R+   14:30   0:00 ps -u
```

Question 2

```
> ca
**** taper ici CTRL Z *****
Suspended
> ps -u
jb         5590  0.1  0.6   4404  1908 tty2     Ss   14:29   0:00 -bash
jb         5715  8.0  0.1   1588   292 tty2     T    14:31   0:01 ca
jb         5716  0.0  0.3   2480   848 tty2     R+   14:31   0:00 ps -u

> bg
[1]    ca &
```

bg a pour effet de reprendre la dernière commande suspendue en arrière plan. C'est une astuce utile si on a lancé une commande en oubliant de la faire suivre par un &.

```
> ps -u
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
jb         5590  0.1  0.6   4404  1916 tty2     Ss   14:29   0:00 -bash
jb         5715 30.1  0.1   1588   292 tty2     R    14:31   0:09 ca
jb         5735  0.0  0.3   2484   852 tty2     R+   14:31   0:00 ps -u
> kill -9 5715
```

Question 3

```
> ps -aux
USER        PID %CPU %MEM    VSZ   RSS TTY   STAT START   TIME COMMAND
root          1  0.2  0.1    740   288 ?     Ss   14:11   0:02 init [3]
root       2398  0.0  0.2   2032   624 ?     Ss   14:12   0:00 /sbin/syslog-ng
nobody     3178  0.0  0.1   1628   428 ?     Ss   14:12   0:00 /sbin/portmap
root       3567  0.0  0.2   1972   564 ?     Ss   14:12   0:00 /usr/sbin/cron
root       3590  0.0  0.7   5344  2028 ?     Ss   14:12   0:00 /usr/sbin/cupsd
root       3623  0.0  0.4   5808  1292 ?     Ss   14:12   0:00 /usr/sbin/sshd …
root       3659  0.0  0.8   6616  2364 ?     Ss   14:12   0:00 login -- jb
root       3665  0.0  0.2   2060   652 tty3  Ss+  14:12   0:00 mingetty tty3
root       3666  0.0  0.2   2056   632 tty4  Ss+  14:12   0:00 mingetty tty4
root       3667  0.0  0.2   2056   632 tty5  Ss+  14:12   0:00 mingetty tty5
root       3685  0.0  0.2   2060   652 tty6  Ss+  14:12   0:00 mingetty tty6
root       5580  0.0  0.7   6408  2120 ?     Ss   14:29   0:00 login -- jb
jb         5590  0.1  0.6   4404  1916 tty2  Ss   14:29   0:00 -bash
jb         5755  0.0  0.3   2480   852 tty2  R+   14:32   0:00 ps -aux
```

Exercice 2 : Parallélisme calcul et E/S

Question 1

```
> time ca
20.80user 0.02system 0:21.86elapsed 95%CPU …

> !!
time ca
20.60user 0.02system 0:21.62elapsed 95%CPU …

> !!
time ca
20.82user 0.08system 0:21.77elapsed 95%CPU …
```

Question 2

```
> ca &;time ca
[1] 1012
20.76user 0.01system 0:42.57elapsed 48%CPU …
[1]  + Exit 32                ca
>
```

Question 3

```
> time di
0.03user 3.26system 0:11.87elapsed 27%CPU..

> rm fich*□; time di
0.02user 3.31system 0:13.88elapsed 24%CPU..

> rm fich*□; time di
0.02user 3.28system 0:12.98elapsed 25%CPU...
```

Question 4

```
> time ca&;time di&
[1] 1203
[2] 1204
>
[2]  + Done                   di
0.04user 2.66system 0:16.90elapsed 15%CPU …
[1]  + Exit 32                ca
21.98user 1.71system 0:29.57elapsed 80%CPU …
```

Exercice 3□L'écroulement (thrashing)

Question 1

```
> free
             total       used       free     shared    buffers     cached
Mem:        276448      33284     243164          0       2488      11788
-/+ buffers/cache:      19008     257440
Swap:      1534168      12780    1521388
```

Question 2

```
> ge&;ge&;ge&;vmstat 1
*** je retire ici l'affichage résultant ***
> !!
> ge&;ge&;ge&;vmstat 1
[1] 1405
[2] 1406
[3] 1407
```

procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu---

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa
5	0	10460	251984	468	5644	128	177	190	191	313	126	2	4	91	2
2	0	10460	251620	468	5644	0	0	4	0	257	95	1	12	87	0
3	0	10460	238180	468	5664	0	0	0	0	270	62	16	84	0	0
3	0	10460	224440	468	5664	0	0	0	0	252	39	14	86	0	0
3	0	10460	210820	468	5664	0	0	0	0	270	68	12	88	0	0
3	0	10460	197260	476	5656	0	0	0	100	254	187	19	81	0	0
4	0	10460	184360	476	5664	0	0	0	0	270	62	14	86	0	0
3	0	10460	170740	476	5664	0	0	0	0	251	44	19	81	0	0
3	0	10460	157196	476	5664	0	0	0	0	270	63	14	86	0	0
3	0	10460	143576	476	5664	0	0	0	0	252	48	16	84	0	0
3	0	10460	129896	484	5656	0	0	0	16	263	69	20	80	0	0
3	0	10460	116156	484	5656	0	0	0	0	261	53	14	86	0	0
3	0	10460	102492	484	5664	0	0	0	0	252	48	14	86	0	0
3	0	10460	88812	484	5664	0	0	0	0	269	62	16	84	0	0
3	0	10460	75852	484	5664	0	0	0	0	252	53	20	80	0	0
3	0	10460	62292	492	5656	0	0	0	28	273	208	15	85	0	0
3	0	10460	49348	492	5664	0	0	0	0	251	46	16	84	0	0
3	0	10460	35668	492	5664	0	0	0	0	270	58	15	85	0	0
3	0	10460	21988	492	5664	0	0	0	0	252	48	19	81	0	0
3	0	10460	8336	492	5664	0	0	0	0	269	60	15	85	0	0
0	7	58228	3480	120	840	892	54792	3044	54896	1652	779	14	68	0	19
0	0	10900	256036	196	3728	1256	0	2212	28	446	494	0	8	14	78
1	0	10900	256036	196	3720	0	0	0	0	271	53	0	1	99	0
0	0	10900	256036	196	3720	0	0	0	0	259	35	0	0	100	0
0	0	10900	255800	204	3952	0	0	264	0	282	65	0	1	95	4
0	0	10900	255800	204	3952	0	0	0	0	260	44	0	0	100	0

***** taper CTRL-C pour arrêter vmstat quand tout est fini (r=0) *****

Question 3

```
> me&;me&;me&;vmstat 1
```

[1] 1420
[2] 1421
[3] 1422

procs ----------memory---------- ---swap-- -----io---- -system-- ----cpu----

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa
0	0	10496	253240	420	5332	160	247	225	261	319	125	3	5	90	3
3	1	10496	240984	428	5324	0	0	12	0	264	63	0	7	91	2
3	0	10496	14960	428	5340	0	0	0	0	271	65	0	100	0	0
1	5	77380	3648	116	988	11496	79004	13312	79064	3015	1010	0	17	0	83
1	5	88992	4096	172	3280	12676	0	13408	0	4346	587	0	33	0	67
1	5	88992	3648	180	3760	9016	0	9504	0	2105	312	0	33	0	67
0	4	145520	5296	176	3416	812	56528	812	56572	454	135	0	24	0	76
0	4	145520	5884	176	3428	324	0	324	0	375	46	0	4	0	96
0	6	145520	3712	188	3620	9908	0	10100	16	2444	225	1	34	0	65
0	5	145520	4120	192	3672	9956	0	10024	4	2346	327	0	30	0	70
0	5	145520	4492	196	3868	13808	0	14000	0	3253	334	1	53	0	46
2	7	145520	3220	204	3892	9060	0	9104	8	2115	365	14	35	0	51
1	4	151600	4076	212	3780	4140	6080	4192	6084	1002	299	84	16	0	0
1	4	167956	4304	176	2468	6544	16356	6704	16388	1755	328	58	42	0	0
2	2	167956	4048	176	2456	10244	0	10244	0	2645	181	56	44	0	0
2	3	184184	3132	156	1600	9052	16228	9052	16248	2355	250	52	48	0	0
2	2	185760	3964	112	1624	2552	1576	2736	1580	827	122	67	33	0	0
7	5	188688	3660	108	1596	12008	7336	12008	7340	3181	279	42	58	0	0
3	4	210176	3372	72	992	5200	23572	5212	23584	1443	275	65	35	0	0
1	4	210176	4640	68	940	464	3028	584	3036	465	108	85	15	0	0
1	3	210176	5084	68	940	320	0	320	0	369	45	98	2	0	0
1	4	210176	3380	68	952	11472	0	11472	0	2882	293	43	57	0	0
1	5	210176	3440	76	1140	9136	560	9340	564	2242	301	54	46	0	0
1	5	210176	4956	88	1188	6148	17140	6340	17156	1725	215	65	35	0	0
0	6	210176	4912	88	1296	92	0	212	0	340	31	41	5	0	54
0	5	179948	95964	116	2112	9568	0	10408	0	2373	248	1	46	0	53
3	4	179948	90540	140	2784	4464	0	5164	0	1182	196	48	19	0	33
4	3	179820	81512	176	3732	8004	0	8984	0	2109	158	61	39	0	0
2	4	179820	70900	200	3780	10448	0	10504	40	2521	297	57	43	0	0
3	1	179820	63284	216	3776	7436	0	7476	4	1978	170	75	25	0	0
3	0	179820	63248	220	3852	0	0	52	0	270	52	99	1	0	0
2	0	179820	63476	256	4048	48	0	276	0	273	303	95	5	0	0
2	0	179820	62136	272	5028	524	0	1516	0	322	134	98	2	0	0
2	0	179820	62152	280	5016	0	0	0	60	255	61	100	0	0	0
2	0	179820	62152	280	5016	0	0	0	0	269	65	100	0	0	0
2	0	179820	62116	284	5056	84	0	132	0	262	64	99	1	0	0
2	0	179820	62116	284	5056	0	0	0	0	270	63	100	0	0	0
2	0	179820	62124	284	5068	0	0	0	0	251	48	100	0	0	0
2	0	179820	62124	292	5060	0	0	0	16	273	69	100	0	0	0
1	0	101520	158064	292	5096	128	0	128	0	258	57	95	5	0	0
0	0	10880	254152	292	5060	136	0	136	0	279	69	90	5	3	2

Question 4

```
> ge&;ge&;ge&;ge&;vmstat 1
[1] 1425
[2] 1426
[3] 1427
[4] 1428
```

procs -----------memory---------- ---swap-- -----io---- -system-- ----cpu----

r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa
0	0	10528	250620	424	6396	127	182	190	196	312	126	2	4	91	2
6	0	10528	249816	424	6384	0	0	4	0	271	72	1	7	92	0
5	0	10528	236196	424	6384	0	0	0	0	251	192	16	84	0	0
4	0	10528	222456	424	6388	0	0	0	0	270	65	18	82	0	0
4	0	10528	209496	424	6388	0	0	0	0	252	41	15	85	0	0
4	0	10528	195756	432	6380	0	0	0	68	272	68	15	85	0	0
4	0	10528	182076	432	6380	0	0	0	0	252	50	20	80	0	0
4	0	10528	168352	432	6388	0	0	0	0	270	61	17	83	0	0
4	0	10528	154612	432	6388	0	0	0	0	251	44	21	79	0	0
4	0	10528	140932	432	6388	0	0	0	0	270	64	15	85	0	0
4	0	10528	127132	440	6380	0	0	0	16	256	54	16	84	0	0
4	0	10528	114180	440	6388	0	0	0	0	269	68	16	84	0	0
4	0	10528	100620	440	6388	0	0	0	0	252	193	16	84	0	0
4	0	10528	86880	440	6388	0	0	0	0	271	62	15	85	0	0
4	0	10528	73140	440	6388	0	0	0	0	250	44	15	85	0	0
4	0	10528	59400	448	6380	0	0	0	28	273	77	23	77	0	0
4	0	10528	45720	448	6380	0	0	0	0	251	52	17	83	0	0
4	0	10528	31980	448	6388	0	0	0	0	269	59	16	84	0	0
5	0	10528	18928	448	6388	0	0	0	0	251	40	13	88	0	0
4	0	10528	6820	448	6388	0	0	0	0	269	92	20	80	0	0
5	3	57080	4136	120	1572	360	47208	1836	47312	1444	574	15	73	0	12
4	3	118500	3776	156	2732	424	61420	1048	61492	1225	502	15	78	0	7
4	1	118756	4084	164	3220	1512	256	2028	256	404	490	17	83	0	0
4	1	123748	3856	176	3656	116	4992	564	4992	285	162	20	80	0	0
3	1	145128	4296	188	3896	32	21380	252	21388	333	155	16	82	0	1
0	0	10924	253752	216	4788	912	0	1828	24	349	176	10	68	22	0

La différence de comportement de mechant est maintenant assez spectaculaire. Le thrashing est proche !!

```
> me&;me&;me&;me&;vmstat 1
[1] 1434
[2] 1435
[3] 1436
[4] 1437
```

procs		memory				swap		io		system		cpu			
r	b	swpd	free	buff	cache	si	so	bi	bo	in	cs	us	sy	id	wa
0	0	10528	246848	756	9772	222	315	288	329	333	126	3	5	89	3
6	1	10528	235560	756	9812	0	0	8	0	258	200	0	8	90	2
4	0	10528	10208	756	9804	0	0	0	0	269	65	2	98	0	0
2	8	106536	3372	104	1796	532	152972	1996	153084	2527	2230	0	17	0	83
0	9	185740	4404	124	1988	9484	23044	10660	23044	3279	1051	0	12	0	88
0	10	185740	3280	144	2600	9508	0	10168	24	2321	265	0	34	0	66
0	8	185740	3288	156	2828	11840	0	12068	4	2857	312	0	38	0	62
1	7	185948	3252	176	2908	12804	208	12904	208	3064	313	0	49	0	51
1	6	227876	3752	100	1176	5724	41928	6168	41940	1507	308	0	31	0	69
0	6	229508	7744	100	1300	1040	1632	1224	1636	528	128	0	14	0	86
0	7	229508	7636	108	1700	672	0	1076	4	378	56	0	5	0	95
1	5	229508	4336	120	2768	9212	0	10264	28	2287	258	0	38	0	63
0	6	229508	3852	148	3176	13676	0	14116	0	3406	242	0	60	0	40
0	6	231772	4444	156	3176	13156	2264	13176	2264	3205	306	1	53	0	46
9	9	269124	3536	140	2892	5368	37396	5376	37408	1490	223	0	39	0	61
0	7	286328	7480	144	2700	424	17704	440	17708	361	136	0	14	0	86
0	8	286328	8000	156	2856	224	0	368	36	362	44	0	7	0	93
1	6	286328	3832	160	2976	9064	0	9196	0	2329	191	0	42	0	58
1	6	286328	3656	176	2976	14736	0	14764	0	3644	244	2	66	0	32
1	6	286328	3992	192	3000	13896	0	13928	0	3317	314	0	64	0	36
0	8	288056	4480	172	2208	15184	1728	15468	1732	3692	296	1	67	0	32
2	6	308232	4044	172	2944	7300	20764	8060	20812	1942	227	0	44	0	56
7	9	333660	3272	176	3004	4772	36096	4888	36116	1334	307	0	29	0	71
2	6	333660	6468	180	3324	1856	224	2184	224	618	136	9	9	0	81
2	5	333660	4264	200	4136	4748	0	5568	0	1180	203	79	21	0	0
1	8	336488	4524	180	3860	11648	8712	11656	8760	2918	269	42	58	0	0
1	7	336488	4540	180	3876	96	0	96	0	309	26	92	8	0	0
1	6	336488	5288	180	3876	0	0	0	0	316	34	99	1	0	0
3	7	336488	3280	200	3908	12560	0	12600	0	3083	247	48	52	0	0
2	7	336488	3252	204	4076	10668	128	10836	164	2431	374	48	52	0	0
1	7	336488	3288	216	4340	11508	836	11804	836	2502	445	53	47	0	0
1	6	337940	4436	188	3312	2276	12000	2276	12032	841	199	87	13	0	0
2	5	337940	3328	188	3316	7964	0	7964	0	1989	190	70	30	0	0
2	7	337940	3496	140	2148	8968	11592	8968	11600	2152	355	59	41	0	0
1	8	337940	4608	92	1596	4180	33100	4180	33100	1114	296	72	28	0	0
1	9	299400	98956	100	1916	6452	5120	6792	5120	1574	355	34	38	0	29
0	5	299268	6136	156	2952	97696	1304	98784	1332	22199	3474	0	44	0	56
0	6	299268	6500	160	2956	0	0	4	0	315	49	0	2	0	98
0	6	299268	4188	164	2956	7084	0	7084	20	1828	403	0	32	0	68
4	5	299268	3212	168	2944	12036	0	12036	16	2892	359	2	49	0	49
3	3	299268	3316	168	2952	3256	0	3256	0	881	158	88	13	0	0
2	6	299268	3152	168	2964	9684	744	9684	748	2307	329	58	42	0	0
2	4	299268	3200	160	3016	5868	13780	5972	13784	1615	228	58	42	0	0
1	4	299268	4236	140	2020	2928	9324	2932	9352	920	170	76	24	0	0
2	3	299268	3304	144	2036	6816	0	6820	0	1840	156	72	28	0	0
1	3	299268	5696	60	1364	6676	6896	6936	6912	1747	236	58	42	0	0
1	5	299268	3108	68	1444	3632	0	3716	0	1112	120	81	19	0	0
1	4	299268	4096	64	1704	7948	7716	8328	7724	2154	197	63	37	0	0
1	6	299228	3960	100	2152	3624	20956	4092	20956	1133	194	71	29	0	0

1	5	299228	4304	128	2704	7372	10296	8144	10332	2377	294	79	21	0	0
0	6	299228	4308	128	2788	0	0	104	0	306	26	23	5	0	72
0	5	203260	97980	136	3352	4640	0	5228	0	1313	164	0	28	0	72