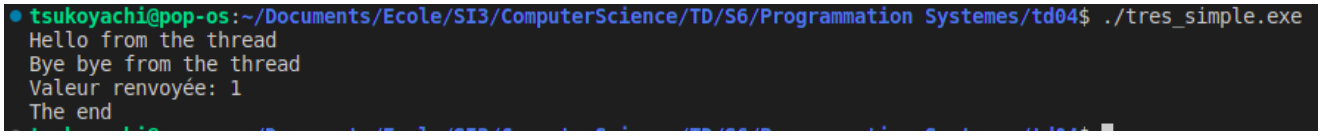
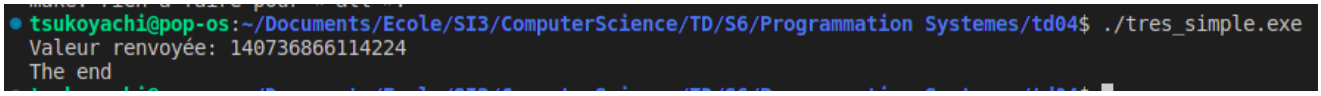
Compte Rendu TD04

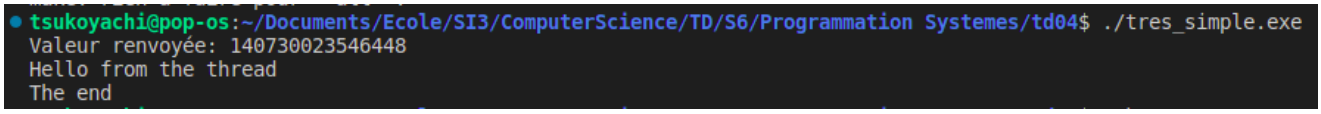
Exercice 1 :

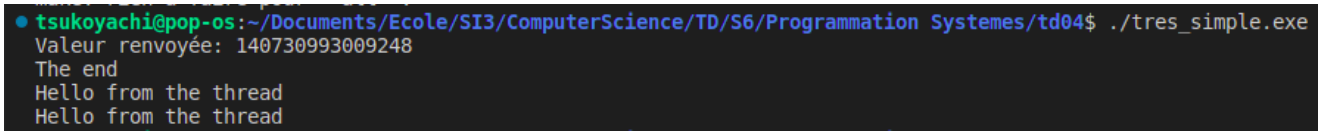
Sans rien modifier au code "tres_simple.c" on obtient le résultat suivant :



En retirant l'appel à pthread_join() me donne aléatoirement trois résultats distincts :

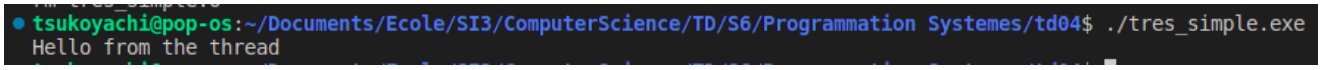






Dans les trois cas ma conclusion est la même, la durée de vie d'un thread est la même que celle du programme principal.

Dans le cas où l'on ajoute un exit(0) après le printf("Hello from the thread"), on obtient ceci :



Dans ce cas-ci on constate que l'exécution du programme et de ses threads est stoppée par un exit(0) dans le thread. Pour quitter un thread sans couper le programme il faut utiliser pthread_exit().

Enfin, dans le cas où on déplace le exit(0) juste avant le pthread_join(), on obtient ceci :

```
tsukoyachi@pop-os:~/Documents/Ecole/SI3/ComputerScience/TD/S6/Programmation Systemes/td04$ ./tres_simple.exe
tsukoyachi@pop-os:~/Documents/Ecole/SI3/ComputerScience/TD/S6/Programmation Systemes/td04$
```

Dans ce cas, le exit(0) a également coupé l'exécution du code principal et celle de son thread.

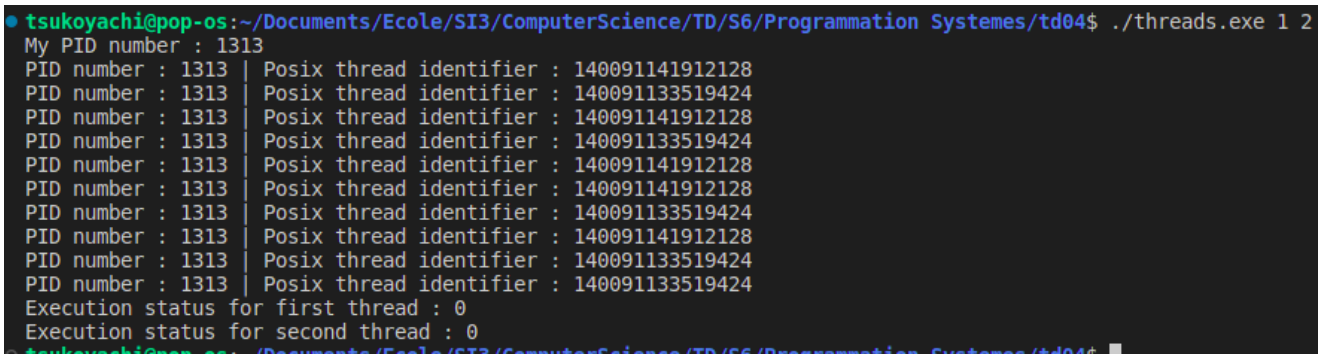
On comprends donc que exit(0) coupe tous les threads alors que pthread_exit() ne coupe que le thread courant, cela s'applique également à l'exécution du code principal.

Exercice 2 :

Voici mon code pour thread.c :

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

void *thread_print(void *arg) {
    int sleepTime = atoi(arg);
    for(int i = 0; i<5;i++){
        printf("PID number : %d | Posix thread identifier : %ld\n",getpid(),pthread_self());
        sleep(sleepTime);
    }
    pthread_exit((void *) 0);
}

int main(int argc, char *argv[]) {
    if(argc < 3){
        printf("Not enough parameter, we need 2 integer...\n");
        exit(1);
    }

    pthread_t t1, t2;
    long return1,return2;
    printf("My PID number : %d\n",getpid());
    pthread_create(&t1,NULL,&thread_print,argv[1]);
    pthread_create(&t2,NULL,&thread_print,argv[2]);
    pthread_join(t1,(void *) &return1);
    pthread_join(t2,(void *) &return2);
    printf("Execution status for first thread : %ld\n",return1);
    printf("Execution status for second thread : %ld\n",return2);
    return 0;
}
```

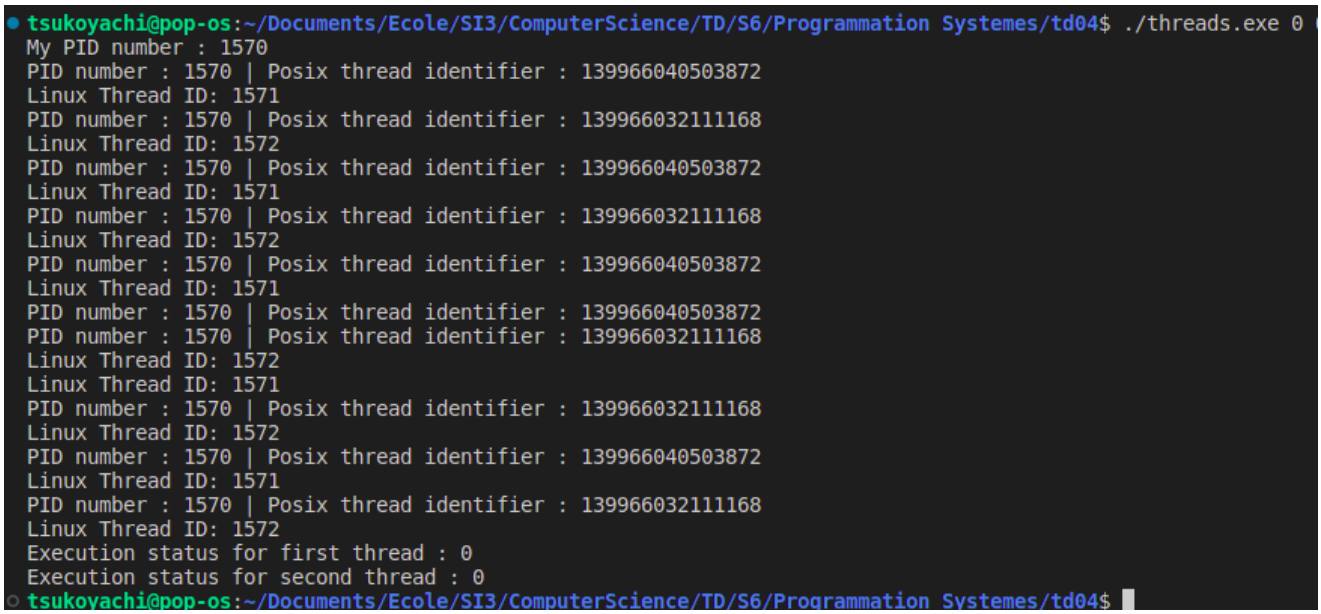
Voici le résultat de son exécution :



Avant passage dans les pthread_create n1 et n2 sont des chaînes de caractères contenant des nombres, on utilise la fonction atoi afin de les reconvertir en int, cela traîte également les cas où je ne passe pas des nombres à ma fonction puisque dans ce cas 0 sera mis à la place.

Exercice 3 :

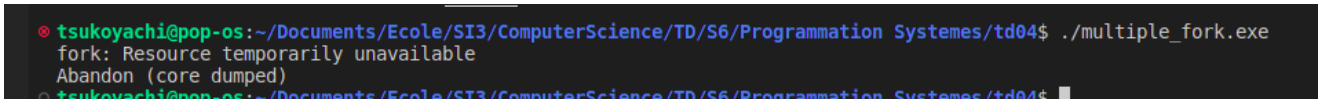
Après ajout de la ligne permettant d'afficher l'Id du thread via sycall(SYS_gettid) pour les thread on obtient ceci :



On constate effectivement que l'id obtenu pour chaque thread est différent de l'identifiant du thread Posix.

Exercice 4 :

En modifiant la variable NPROCESS de multiple_fork à 100000 on obtient :



Donc non, il n'est pas possible de créer autant de processus qu'on le souhaite.

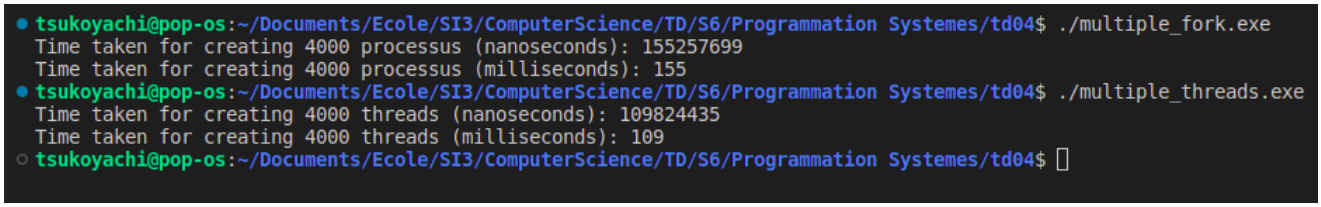
Exercice 5 :

Voici mon programme multiple_thread.c :

```c
#define NTHREADS 4000

void *wait_threads(void *arg){
    sleep(10);
    pthread_exit((void *) 0);
}

long create_thread(long n) {

    pthread_t *t = (pthread_t *) malloc(sizeof(pthread_t)*n);
    if(t == NULL){
        printf("Allocation error...");
        exit(1);
    }
    struct timespec vartime = timer_start(); /* Démarrage de la mesure temporelle */
    /* Création de n thread s'exécutant en parallèle */
    for (int i = 0; i < n; ++i) {
        pthread_create(&t[i],NULL,&wait_threads,NULL);
    }

    /* On mesure le temps écoulé pour la création des n processus */
    long time = timer_end(vartime);

     for (int i = 0; i < n; ++i) {
        pthread_join(t[i],NULL);
    }

    return time;
}

int main(int argc, char *argv[]) {
    int n = NTHREADS;

    long time_processus = create_thread(n);

    printf("Time taken for creating %d threads (nanoseconds): %ld\n", n, time_processus);
    printf("Time taken for creating %d threads (milliseconds): %ld\n", n, time_processus / NANO_TO_MILLI);

    /* On flushe la sortie standard */
    fflush(stdout);

    /* Fin du père */
    exit(0);
}
```

On va donc appeler ce programme et multiple_fork.c pour créer 4000

threads et 4000 processus puis on va comparer les temps :



On constate que la création des threads est plus rapide que la création des processus.

Exercice 6 :

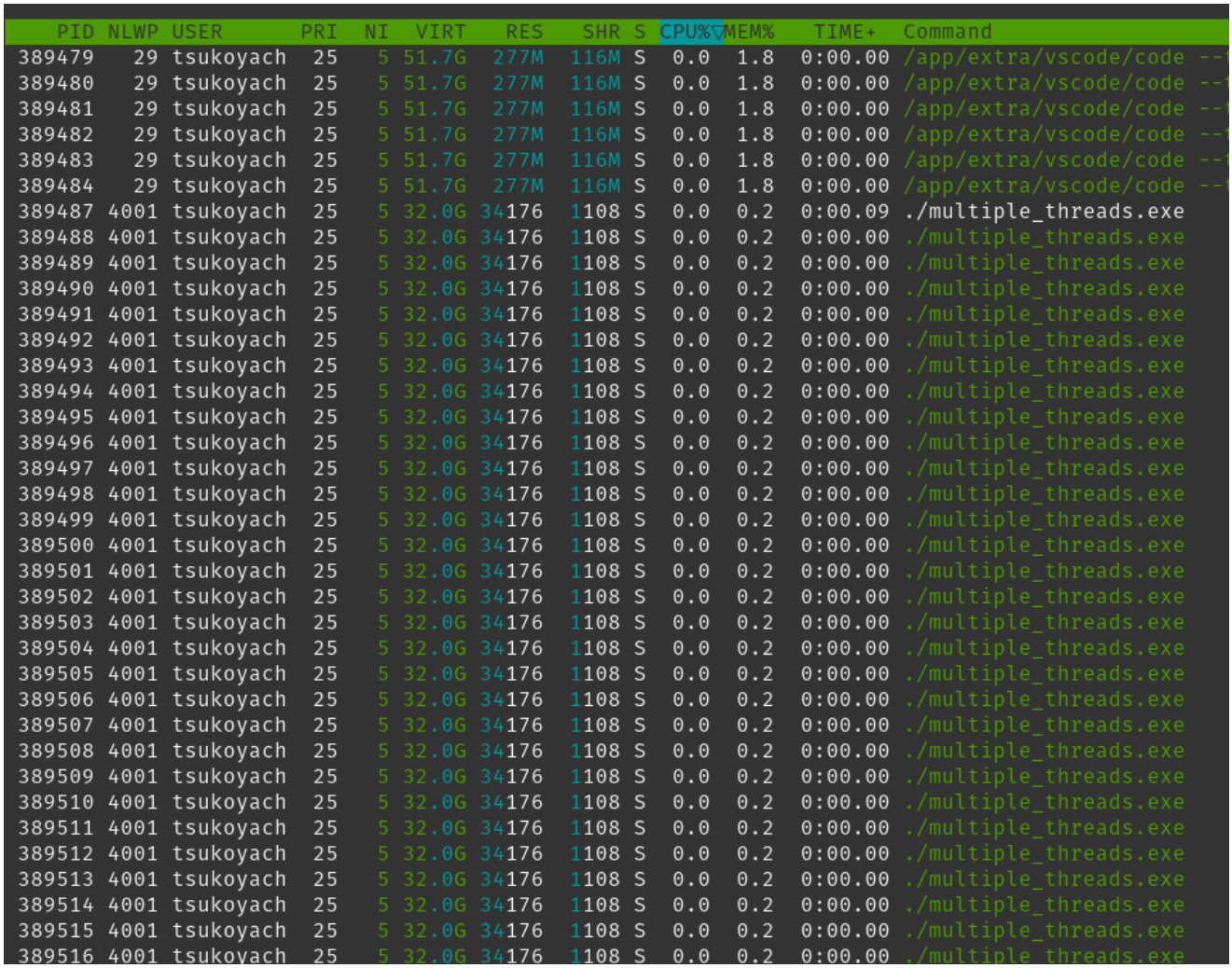
Avec htop on obtient ceci pour multiple_fork.c :

```
tsukoyachi@pop-os:~$ ps aux | grep multiple_fork.exe
tsukoya+  393993  7.0  0.0   2528  1204 pts/7    SN+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393994  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393995  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393996  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393997  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393998  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  393999  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394000  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394001  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394002  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394003  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394004  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394005  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394006  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394007  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394008  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394009  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394010  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394011  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394012  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394013  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394014  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394015  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394016  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394017  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394018  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394019  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394020  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394021  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394022  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394023  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394024  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394025  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394026  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394027  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394028  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394029  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394030  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394031  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394032  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394033  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394034  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394035  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394036  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394037  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394038  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
tsukoya+  394039  0.0  0.0   2528    92 pts/7    S<+  11:21   0:00 ./multiple_fork.exe
```

```
LWP     PID USER      PRI  NI  VIRT   RES   SHR S  CPU%  MEM%   TIME+  Command
  1  360822 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360823 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360824 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360825 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360826 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360827 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360828 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360829 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360830 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360831 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360833 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360835 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360836 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360837 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360838 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360839 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360840 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360841 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360842 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360843 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360844 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360845 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360846 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360847 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360849 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360850 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360851 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360852 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360853 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360854 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360855 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360856 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360858 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360859 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360860 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360861 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360862 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
  1  360863 tsukoyach  15  -5  2528    92     0 S   0.0   0.0  0:00.00 ./multiple_fork.exe
```

Et ceci pour multiple_thread.c :

```
tsukoyachi@pop-os:~$ ps aux | grep multiple_threads.exe
tsukoya+  420431  2.5  0.2 32787716 34180 pts/7  SNl+ 11:23   0:00 ./multiple_threads.exe
tsukoya+  424458  0.0  0.0  19036  2416 pts/4    S<+  11:23   0:00 grep --color=auto multiple_threads.exe
```



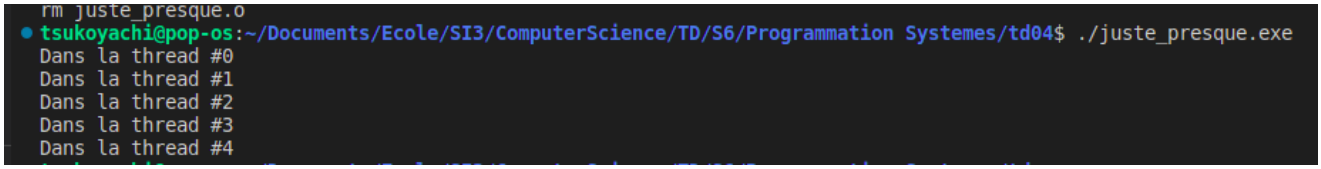
PID	NLWP	USER	PRI	NI	VIRT	RES	SHR	S	CPU%	MEM%	TIME+	Command
389479	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389480	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389481	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389482	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389483	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389484	29	tsukoyach	25	5	51.7G	277M	116M	S	0.0	1.8	0:00.00	/app/extra/vscode/code --t
389487	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.09	./multiple_threads.exe
389488	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389489	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389490	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389491	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389492	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389493	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389494	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389495	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389496	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389497	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389498	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389499	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389500	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389501	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389502	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389503	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389504	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389505	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389506	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389507	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389508	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389509	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389510	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389511	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389512	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389513	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389514	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389515	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe
389516	4001	tsukoyach	25	5	32.0G	34176	1108	S	0.0	0.2	0:00.00	./multiple_threads.exe

Dans le cas du multiple_fork.c apparaît plusieurs fois sous htop, une ligne par processus comme sur ps aux, on voit également que le nlwp est à 1 à chaque fois, donc que chaque processus ne possède qu'un seul thread avec un PID qui est différent à chaque fois.

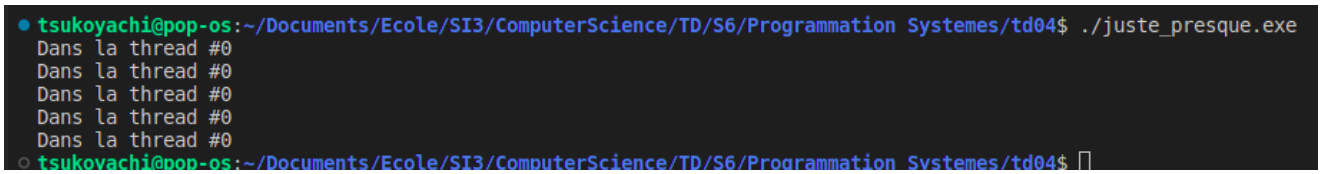
Dans le cas du multiple_thread.c, le programme apparait plusieurs fois sous htop mais qu'une seule fois sous ps aux, mais c'est dû à l'implémentation des threads sous linux qui sont implémenté comme les processus, d'où le fait qu'il est des pid différent à chaque fois, cependant on note le NLWP à 4001 (4000 thread + exécution principale).

Exercice 7 :

Première exécution de juste_presque.c sans rien enlever (en augmentant le sleep à 7):



Maintenant supprimons le sleep() et voyons l'exécution :



Le problème ici est un accès concurrant à la variable i à chaque fois, les threads sont bien créer comme il faut, mais le paramètre passé est l'adresse de i et non sa valeur, d'une executieton du for à l'autre la valeur a cette adresse est modifiée.

Pour résoudre cela on peut faire un tableau pour stocker les valeurs de i et mettre l'adresse d'une case de ce tableau en paramètre, comme ceci :

```c
/* Creation des threads */
for (i = 0; i < MAX; i++) {
    /*On stock l'argument dans un tableau pour la case du tableau ait une adresse distincte
    de celle des autres arguments de la boucle pour que la valeur ne change pas d'un passage de boucle à l'autre */
    arg[i] = i;
    pthread_create(&threads[i], NULL, func, (void *)&arg[i]);
}
```

Cela corrige le soucis sans modifier l'autre méthode.

Exercice 8 :

Malgré de nombreuses exécution du jeu, je n'ai pas réussi à constater le bug où les joueurs prenaient des allumettes alors que le plateau n'en possédait plus.

En retirant l'appel à sleep par contre, on constate l'apparition du bug cité précédemment ainsi qu'une désynchronisation des valeurs du nombre d'allumette :

```
Joueur 0 prend 1 allumettes, reste 18 allumettes
Joueur 0 prend 2 allumettes, reste 16 allumettes
Joueur 0 prend 1 allumettes, reste 15 allumettes
Réinitialisation du jeu avec 19 allumettes.
Joueur 1 prend 3 allumettes, reste 11 allumettes
Joueur 1 prend 3 allumettes, reste 10 allumettes
Joueur 0 prend 2 allumettes, reste 13 allumettes
Joueur 0 prend 3 allumettes, reste 6 allumettes
Joueur 0 prend 2 allumettes, reste 4 allumettes
Joueur 0 prend 3 allumettes, reste 1 allumettes
Joueur 0 prend 1 allumettes, reste 0 allumettes
Joueur 0 prend 3 allumettes, reste 16 allumettes
Joueur 0 prend 1 allumettes, reste 15 allumettes
Joueur 0 prend 1 allumettes, reste 14 allumettes
Joueur 0 prend 2 allumettes, reste 12 allumettes
Joueur 0 prend 1 allumettes, reste 11 allumettes
Joueur 0 prend 2 allumettes, reste 9 allumettes
Joueur 0 prend 3 allumettes, reste 6 allumettes
Joueur 0 prend 1 allumettes, reste 5 allumettes
Joueur 0 prend 1 allumettes, reste 4 allumettes
Joueur 0 prend 2 allumettes, reste 2 allumettes
Joueur 0 prend 1 allumettes, reste 1 allumettes
Joueur 0 prend 1 allumettes, reste 0 allumettes
Joueur 0 prend 1 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
Joueur 0 prend 0 allumettes, reste -1 allumettes
```

Après avoir remis l'appel à sleep on constate également une utilisation de 100% du CPU avec htop :