

```
// programme fils.c
#include <stdio.h>
#include <unistd.h>

int glob=10; // variable partage

void Processus_P1() {
    int x;
    x=glob;
    x=x+7;
    glob = x;
    printf("ici processus P1, glob = %d\n", glob);
}

void Processus_P2(){
    int x;
    x=glob;
    x=x-2;
    glob = x;
    printf("ici processus P2 , glob = %d\n",glob);
}

int main ( )
{
    printf(" Befor call of P1 & P2 --> glob = %d \n",glob);
    Processus_P1();
    Processus_P2();
    printf(" After call of P1 & P2 --> glob = %d \n",glob);
    return 0;
}
```

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

int glob=10; // variable partage

void* Processus_P1 ( ) {

    int x;
    x=glob;
    sleep(5);
    x=x+7;
    glob = x;

    printf("ici P1 [%d], glob = %d\n", (int)pthread_self(), glob);
    pthread_exit(NULL);
}

void* Processus_P2 ( )
{

    int x;
    x=glob;
    sleep(5);
    x=x-2;
    glob = x;

    printf("ici P2 [%d], glob = %d\n", (int)pthread_self(), glob);
    pthread_exit(NULL);
}

int main( )
{
    pthread_t P1, P2;

    printf("valeur intiale de vraibale partage glob = %d\n",glob);

    //creation d'un thread pour P1
    if ( pthread_create(&P1, NULL, Processus_P1 , NULL) != 0) return -1;
    printf("creation du Processus P1 : thread[%d] avec succes\n", (int)P1);

    // creation d'un thread pour P2
    if ( pthread_create(&P2, NULL, Processus_P2 , NULL) != 0) return -1;
    printf("creation du Processus P2 :thread [%d] avec succes\n", (int)P2);

    // attendre la fin des Processus (threads)
    pthread_join(P1, NULL);
    pthread_join(P2, NULL);

    printf("Fin d'execution de Processus P1 et P2\n");
    printf("Valeur finale variable partage de glob = %d \n",glob);

    printf("\n\n");
    return 0;
}
```

```

// programme fils.c
#include <stdio.h>
#include <unistd.h>

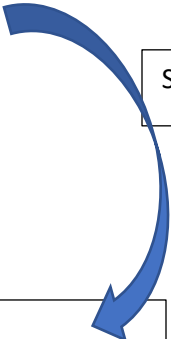
void P1() {
    int i=1;
    for(i=1; i < 1000; i++) printf("%c",'A');
}

void P2() {
    int i=1;
    for(i=1; i < 1000; i++) printf("%c",'B');
}

int main () {
    P1();
    printf("\n\n\n");
    P2();
    return 0;
}

```

Séquentielle vers parallèle



```

// programme fils.c
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>

void* Processus_P1() {
    int i=1;
    for(i=1; i < 1000; i++) printf("%c",'A');
    pthread_exit(NULL);
}

void* Processus_P2() {
    int i=1;
    for(i=1; i < 1000; i++) printf("%c",'B');
    pthread_exit(NULL);
}

int main ()
{
    pthread_t P1, P2;

    printf(" \n lancer execution en parallèle de Processus P1 et P2\n");

    //creation d'un thread pour P1
    pthread_create(&P1, NULL, Processus_P1 , NULL);

    // creation d'un thread pour P2
    pthread_create(&P2, NULL, Processus_P2 , NULL);
    pthread_join(P1,NULL);
    pthread_join(P2,NULL);

    printf(" \n Fin d'execution de Processus P1 et P2\n");
    return 0;
}

```