

```
/*  
Ejemplo simple de procesos livianos (threads) hilos  
Secuencia de hilos ABAB... sin datos compartidos
```

Este ejemplo escribe el abecedario en pantalla

Utiliza sincronizacion con semforos POSIX

```
Hilo A escribe la letra 'a'  
Hilo B escribe la letra 'B'  
Hilo A escribe la letra 'c'  
Hilo B escribe la letra 'D'  
Hasta la 'Z'  
*/
```

```
#include <stdio.h>  
#include <sys/types.h>  
#include <sys/stat.h>  
#include <fcntl.h>  
#include <stdlib.h>  

```

```
void hiloA(void);  
void hiloB(void);
```

```
sem_t sema, semb ;
```

```
int main(int argc, char **argv)  
{
```

```
    sem_init(&sema,0,1);  
    sem_init(&semb,0,0);
```

```
    pthread_t hA,hB;  
    printf("main(): lanzo hiloA!\n");  
    pthread_create(&hA,NULL,(void*)&hiloA,NULL);
```

```
    printf("main(): lanzo hiloB!\n");  
    pthread_create(&hB,NULL,(void*)&hiloB,NULL);
```

```
    printf("main(): espera por fin de hiloA!\n");  
    pthread_join(hA,NULL);
```

```
    printf("main(): espera por fin de hiloB!\n");  
    pthread_join(hB,NULL);
```

```

        printf("main(): fin!\n");
        return 0;
    }

void hiloA(void)
{
    char letra = 'a' ;
    while(letra <= 'y')
    {
        sem_wait(&sema) ;
        printf("hiloA escribe %c \n",letra);
        letra = letra + 2 ;
        sem_post(&semb) ;
    }
}

void hiloB(void)
{
    char letra = 'B' ;
    while(letra <= 'Z')
    {
        sem_wait(&semb) ;
        printf("-----hiloB escribe %c \n",letra);
        letra = letra + 2 ;
        sem_post(&sema) ;
    }
}

```

.....

```

/*
Ejemplo simple de procesos livianos (threads) hilos
Secuencia de hilos ABAB... con datos compartidos

```

Este ejemplo escribe el abecedario en pantalla

Utiliza sincronizacion con semforos POSIX

```

Hilo A escribe la letra 'a'
Hilo B escribe la letra 'B'
Hilo A escribe la letra 'c'
Hilo B escribe la letra 'D'
Hasta la 'Z'
*/

```

```

#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdlib.h>
#include <unistd.h>

```

```

#include <pthread.h>
#include <semaphore.h>

void hiloA(void);
void hiloB(void);

sem_t sema, semb ;

char letra = 'A' ;

int main(int argc, char **argv)
{

    sem_init(&sema,0,1);
    sem_init(&semb,0,0);

    pthread_t hA,hB;
    printf("main(): lanzo hiloA!\n");
    pthread_create(&hA,NULL,(void*)&hiloA,NULL);

    printf("main(): lanzo hiloB!\n");
    pthread_create(&hB,NULL,(void*)&hiloB,NULL);

    printf("main(): espera por fin de hiloA!\n");
    pthread_join(hA,NULL);

    printf("main(): espera por fin de hiloB!\n");
    pthread_join(hB,NULL);

    printf("main(): fin!\n");
    return 0;
}

void hiloA(void)
{
    while(letra <= 'y' )
    {
        sem_wait(&sema) ;
        letra = letra + 32 ;
        printf("hiloA escribe %c \n",letra);
        letra = letra + 1 ;
        sem_post(&semb) ;
    }
    pthread_exit(0);
}

void hiloB(void)
{
    do
    {

```

```

sem_wait(&semb) ;
letra = letra - 32 ;
printf("-----hiloB escribe %c \n",letra);
letra = letra + 1 ;
sem_post(&sema) ;
}while(letra != 'Z'+1 );
pthread_exit(0);
}

```

.....

/*

Este ejemplo crea dos hilos cada uno tiene asociado el código de una función se sincronizan para mostrar en pantalla el abecedario ordenado, un hilo escribe en minúsculas y el otro hilo escribe en mayúsculas

Para la creación de las letras mayúsculas y minúsculas se utilizan los códigos ASCII

CON SINCRONIZACION Y UNA VARIABLE GLOBAL para la comunicación entre los hilos (dato compartido)

línea de compilación
cc hilos02.c -lpthread

*/

```

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

```

```

void hilo1(void);
void hilo2(void);

```

```

pthread_mutex_t m1 = PTHREAD_MUTEX_INITIALIZER ;
pthread_mutex_t m2 = PTHREAD_MUTEX_INITIALIZER ;

```

```

char letra = 'A';

```

```

int main(void)

```

```

{
pthread_t h1 , h2 ;

```

```

pthread_mutex_lock(&m2);

```

```

pthread_create(&h1,NULL,(void *)hilo1,NULL);
printf("hilo h1 creado \n");

```

```

pthread_create(&h2,NULL,(void *)hilo2,NULL);
printf("hilo h2 creado \n");

```

```
pthread_join(h1,NULL); // bloque al proceso padre de los hilos por la finalizacion del
hilo h1
printf("hilo h1 termino \n");
```

```
pthread_join(h2,NULL); // bloque al proceso padre de los hilos por la finalizacion del
hilo h2
printf("hilo h2 termino \n");
```

```
exit(0);
```

```
}
```

```
void hilo1(void)
```

```
{
while(letra <= 'y')
{
pthread_mutex_lock(&m1);
letra = letra + 32 ;
printf("hilo %d escribe la letra %c \n",pthread_self() , letra);
letra = letra + 1 ;
pthread_mutex_unlock(&m2);
// sleep(1);
}
}
```

```
void hilo2(void)
```

```
{
do
{
pthread_mutex_lock(&m2);
letra = letra - 32;
printf("-----hilo %d escribe la letra %c \n",pthread_self() , letra);
letra = letra + 1 ;
pthread_mutex_unlock(&m1);
// sleep(1);
} while(letra != 'Z'+1 );
}
```