线程编程实例

使用pthread_create()创建线程，并使原线程与新线程分别打印自己的线程id

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <unistd.h>
void *tfn(void *arg)
{
	printf("tfn--pid=%d,tid=%lu\n", getpid(), pthread_self());
	return (void*)0;
}
int main()
{
	pthread_t tid;
	printf("main--pid=%d,tid=%lu\n", getpid(), pthread_self());
	int ret = pthread_create(&tid, NULL, tfn, NULL);
	if (ret != 0){
		fprintf(stderr, "pthread_create error:%s\n", strerror(ret));
		exit(1);
	}
	sleep(1);
	return 0;
}
```

创建新线程，在新线程中修改原线程中定义在全局区的变量，并在原线程中打印该数据。

```
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>
#include <unistd.h>
int var = 100;
void *tfn(void *arg)
{
```

```
        var = 200;
        printf("thread\n");
        return NULL;
}
int main(void)
{
        printf("At first var = %d\n", var);
        pthread_t tid;
        pthread_create(&tid, NULL, tfn, NULL);
        sleep(1);
        printf("after pthread_create, var = %d\n", var);
        return 0;
}
```

使用pthread_exit()退出线程，为线程设置退出状态，并将线程的退出状态输出。

```
#include <stdio.h>
#include <unistd.h>
#include <pthread.h>
#include <stdlib.h>
typedef struct{
        int a;
        int b;
} exit_t;
void *tfn(void *arg)
{
        exit_t *ret;
        ret = malloc(sizeof(exit_t));
        ret->a = 100;
        ret->b = 300;
        pthread_exit((void *)ret); //Ïß³ÌÖÕÖ¹
        return NULL;                        //Ïß³Ì·µ»Ø
}
int main(void)
{
        pthread_t tid;
```

```
        exit_t *retval;
        pthread_create(&tid, NULL, tfn, NULL);
        //µ÷ÓÃpthread_join¿ÉÒÔ»ñÈ¡Ïß³ÌµÄÍË³ö×´Ì¬
        pthread_join(tid, (void **)&retval);
        printf("a = %d, b = %d \n", retval->a, retval->b);
        return 0;
}
```