

课程名称	操作系统	课程编号	A2130330
实验地点	C410/411		
实验名称	Assignment4 在多线程中使用 pthread 和 semaphores		

一、实验内容

使用任何 Linux 操作系统的发行版来创建一个 C 程序，该程序以交错的方式使用两个线程。解决方案必须包含 pthread 库和 semaphores，并且必须基于文档中提供的 shopping.c 程序。解决方案必须修改 print_produce()和 print_dairy()函数，使 Salad 总是在 Butter 之前被打印，Milk 总是在 apple 之前被打印。下面的语法用于表达此需求，其中符号<用于表示“之前打印”。

Salad < Butter and Milk < Apples.

二、实验步骤及方案

1. 在全局变量中创建两个信号量来控制后续的打印顺序。
2. 主函数首先中对两个信号量初始化。初始化值均为 0。其中 sem1 作用为控制打印 produce 的线程，sem2 为控制打印 dairy 的线程。
3. 为保证首先打印 Salad，再打印 Butter 和 Milk，故在 print_dairy()，print_produece()进行修改，使打印 dairy 前先进行 sem_wait(&sem2)，且 producece 中打印完 salad 后进行 sem_post(&sem2)。因 sem2 初始化值为 0，故 dairy 中打印 Butter 前会一直等待,直到 produce 中打印完 Salad 后对 sem2 加 1。
4. 为保证 Apples 在 Butter 后 Mike 后打印，故 produce 中在打印 Apples 前进行 sem_wait(&sem1)，dairy 中打印完 Butter 和 Mike 后进行 sem_post(&sem1)。因 sem1 初始值为 1，故 prodeuce 中打印完 Salad 后，在打印 Apples 前会一直 wait，知道 diary 中打印完全部后对 sem 加一。
5. 线程结束后在主函数中摧毁 sem1，sem2。

三、结果及分析

实验源码:

```
* shopping.c */
```

```

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

sem_t sem1;
sem_t sem2;

void *print_produce( void * );
void *print_dairy( void * );
void *print_dairy(void *items)
{
    int i = 0;
    char** array = (void*)items;
    {
        sem_wait(&sem2);
        printf("got %s\n", (array[i++]) );
        printf("got %s\n", (array[i++]) );
        sem_post(&sem1);
    }
    return( NULL );
}

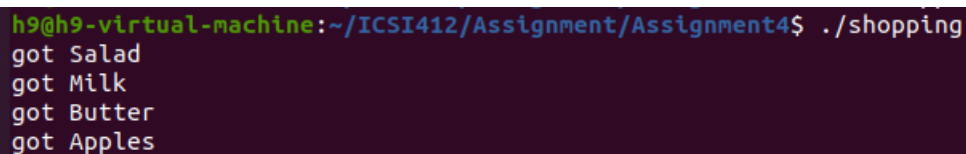
void *print_produce(void *items)
{
    int i = 0;
    char** array = (void*)items;
    {
        printf("got %s\n", (array[i++]) );
        sem_post(&sem2);
        sem_wait(&sem1);
        printf("got %s\n", (array[i++]) );
    }
    return( NULL );
}

int main()
{
    char *produce[] = { "Salad", "Apples", NULL };
    char *dairy[] = { "Milk", "Butter", NULL };
    if (sem_init(&sem1, 0, 0) == -1)
    {
        exit(1);
    }
    if (sem_init(&sem2, 0, 0) == -1)

```

```
{  
    exit(1);  
}  
  
pthread_t th1, th2;  
pthread_create( &th1, NULL, print_produce, (void*)produce);  
pthread_create( &th2, NULL, print_dairy, (void*)dairy);  
pthread_join(th1, NULL);  
pthread_join(th2, NULL);  
sem_destroy(&sem1);  
sem_destroy(&sem2);  
}
```

实验结果如下图所示：



```
h9@h9-virtual-machine:~/ICSI412/Assignment/Assignment4$ ./shopping  
got Salad  
got Milk  
got Butter  
got Apples
```

图 1 输出结果

结果分析：

成功按照要求打印的顺序完成输出。

四、心得体会

1. 熟悉并掌握了在 c 语言中使用 pthread 进行多线程编程。
2. 熟悉并掌握了在多线程中二元 Semaphore 的工作原理，掌握了相关函数如 sum_init, sum_wait, sum_post, sem_destroy。并使用其来保证两个或多个关键代码段不被并发调用。
3. 熟悉并掌握了信号量的使用。