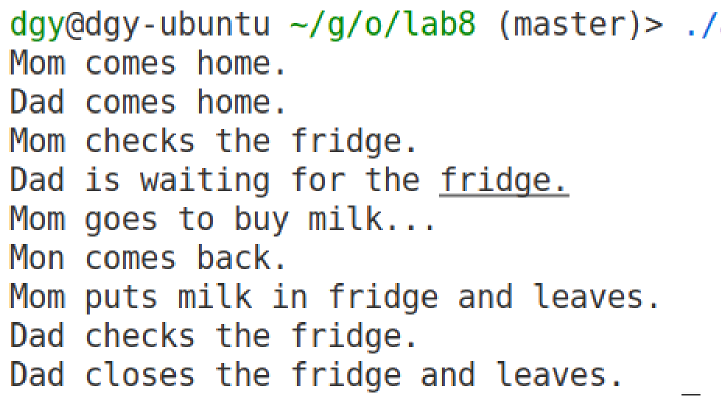
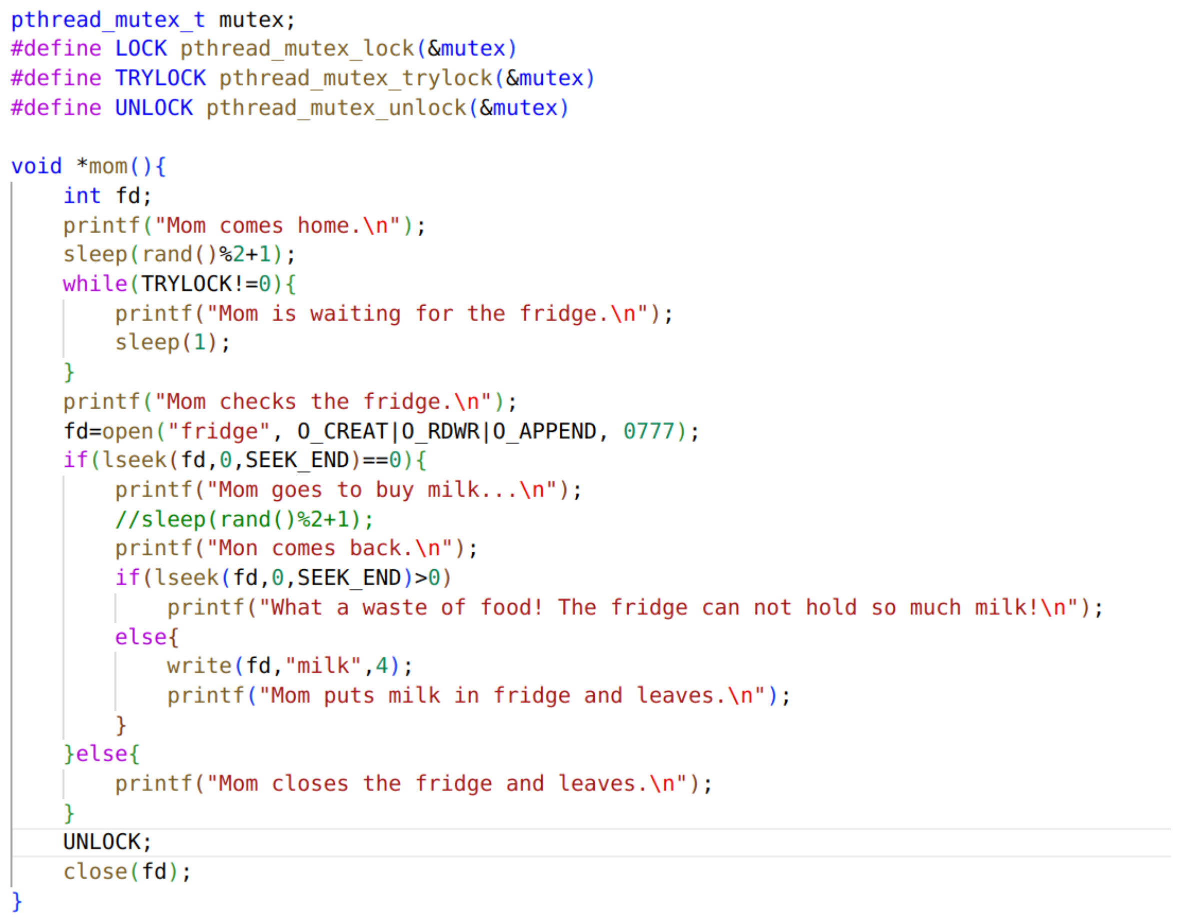
1.



```
dgy@dgy-ubuntu ~/g/o/lab8 (master)> ./a.out
Mom comes home.
Dad comes home.
Mom checks the fridge.
Dad is waiting for the fridge.
Mom goes to buy milk...
Mon comes back.
Mom puts milk in fridge and leaves.
Dad checks the fridge.
Dad closes the fridge and leaves.
```



```c
pthread_mutex_t mutex;
#define LOCK pthread_mutex_lock(&mutex)
#define TRYLOCK pthread_mutex_trylock(&mutex)
#define UNLOCK pthread_mutex_unlock(&mutex)

void *mom(){
    int fd;
    printf("Mom comes home.\n");
    sleep(rand()%2+1);
    while(TRYLOCK!=0){
        printf("Mom is waiting for the fridge.\n");
        sleep(1);
    }
    printf("Mom checks the fridge.\n");
    fd=open("fridge", O_CREAT|O_RDWR|O_APPEND, 0777);
    if(lseek(fd,0,SEEK_END)==0){
        printf("Mom goes to buy milk...\n");
        //sleep(rand()%2+1);
        printf("Mon comes back.\n");
        if(lseek(fd,0,SEEK_END)>0)
            printf("What a waste of food! The fridge can not hold so much milk!\n");
        else{
            write(fd,"milk",4);
            printf("Mom puts milk in fridge and leaves.\n");
        }
    }else{
        printf("Mom closes the fridge and leaves.\n");
    }
    UNLOCK;
    close(fd);
}
```

```c
void *dad(){
    int fd;
    printf("Dad comes home.\n");
    sleep(rand()%2+1);
    while(TRYLOCK!=0){
        printf("Dad is waiting for the fridge.\n");
        sleep(1);
    }
    printf("Dad checks the fridge.\n");
    fd=open("fridge", O_CREAT|O_RDWR|O_APPEND, 0777);
    if(lseek(fd,0,SEEK_END)==0){
        printf("Dad goes to buy milk...\n");
        //sleep(rand()%2+1);
        printf("Dad comes back.\n");
        if(lseek(fd,0,SEEK_END)>0)
            printf("What a waste of food! The fridge can not hold so much milk!\n");
        else{
            write(fd,"milk",4);
            printf("Dad puts milk in fridge and leaves.\n");
        }
    } else{
        printf("Dad closes the fridge and leaves.\n");
    }
    UNLOCK;
    close(fd);
}
```

2.

```
Dad removes lock.
me adds lock.
me closes the fridge and leaves.
me removes lock.
Sister puts milk in fridge and leaves, and remaining milk is 1.
Sister adds lock.
Sister is waiting for the fridge.
Dad adds lock.
Dad takes milk from fridge and leaves, and remaining milk is 0.
Dad removes lock.
Dad adds lock.
Dad closes the fridge and leaves.
Dad removes lock.
Mom puts milk in fridge and leaves, and remaining milk is 1.
Mon adds lock.
Mom is waiting for the fridge.
me adds lock.
me takes milk from fridge and leaves, and remaining milk is 0.
me removes lock.
Dad adds lock.
Dad closes the fridge and leaves.
Dad removes lock.
Sister puts milk in fridge and leaves, and remaining milk is 1.
Sister adds lock.
Sister is waiting for the fridge.
me adds lock.
me takes milk from fridge and leaves, and remaining milk is 0.
me removes lock.
Dad adds lock.
Dad closes the fridge and leaves.
Dad removes lock.
Mom puts milk in fridge and leaves, and remaining milk is 1.
Mon adds lock.
Mom is waiting for the fridge.
me adds lock.
me takes milk from fridge and leaves, and remaining milk is 0.
me removes lock.
```

```c
#include <time.h>
#include <sys/stat.h>

int milk = 0;
pthread_mutex_t mutex;
pthread_cond_t cond;
#define LOCK pthread_mutex_lock(&mutex)
#define TRYLOCK pthread_mutex_trylock(&mutex)
#define UNLOCK pthread_mutex_unlock(&mutex)
#define WAIT pthread_cond_wait(&cond, &mutex)
#define SIGNAL pthread_cond_signal(&cond)

void *mom(){
    while (1){
        TRYLOCK;
        printf("Mon adds lock.\n");
        while(milk>0){
            printf("Mom is waiting for the fridge.\n");
            WAIT;
        }
        milk++;
        printf("Mom puts milk in fridge and leaves, and remaining milk is %d.\n", milk);
        UNLOCK;
    }
}


void *sister(){
    while (1){
        TRYLOCK;
        printf("Sister adds lock.\n");
        while(milk>0){
            printf("Sister is waiting for the fridge.\n");
            WAIT;
        }
        milk++;
        printf("Sister puts milk in fridge and leaves, and remaining milk is %d.\n", milk);
        UNLOCK;
    }
ilk-new.c
        UNLOCK;
    }
}


void *dad(){
    while (1){
        TRYLOCK;
        printf("Dad adds lock.\n");
        if (milk>0){
            milk--;
            printf("Dad takes milk from fridge and leaves, and remaining milk is %d.\n", milk);
        }else{
            printf("Dad closes the fridge and leaves.\n");
            SIGNAL;
        }
        printf("Dad removes lock.\n");
        UNLOCK;
        sleep(rand()%2+1);
    }
}

void *me(){
    while (1){
        TRYLOCK;
        printf("me adds lock.\n");
        if (milk>0){
            milk--;
            printf("me takes milk from fridge and leaves, and remaining milk is %d.\n", milk);
        }else{
            printf("me closes the fridge and leaves.\n");
            SIGNAL;
        }
        printf("me removes lock.\n");
        UNLOCK;
        sleep(rand()%2+1);
    }
}
```

```c
            printf("me takes milk from fridge and leaves, and remaining milk is %d.\n", milk);
        }else{
            printf("me closes the fridge and leaves.\n");
            SIGNAL;
        }
        printf("me removes lock.\n");
        UNLOCK;
        sleep(rand()%2+1);
    }
}


int main(int argc, char * argv[]) {
    srand(time(0));
    pthread_t p1, p2, p3, p4;
    int fd = open("fridge", O_CREAT|O_RDWR|O_TRUNC , 0777);  //empty the fridge
    close(fd);
    // Create two threads (both run func)
    pthread_create(&p1, NULL, mom, NULL);
    pthread_create(&p2, NULL, dad, NULL);
    pthread_create(&p3, NULL, sister, NULL);
    pthread_create(&p4, NULL, me, NULL);

    // Wait for the threads to end.
    pthread_join(p1, NULL);
    pthread_join(p2, NULL);
    pthread_join(p3, NULL);
    pthread_join(p4, NULL);
}
```