(2,0)

1. [2,0 pontos] ***SO Kid*** também se dedicou a outro exercício apresentado em aula/laboratório. Semelhante ao exercício anterior, agora há um conjunto de *threads* tentando manipular uma variável global mas sem nenhuma ordem preestabelecida. No entanto, **apenas exige-se que se garanta a exclusão mútua ao se atualizar a variável global**. Após inicializar a execução do programa de *SO Kid*, observa-se que o mesmo não produz o resultado esperado e sequer finaliza. **Utilizando-se do fragmento principal do código de SO Kid abaixo, identifique o(s) problema(s) e apresente a(s) respectiva(s) correção(ões).**
→ **Caso deseje, realize a(s) correção(ões) diretamente no código abaixo.**

```
void *mythread(void *data);
pthread_mutex_t count_mutex = PTHREAD_MUTEX_INITIALIZER;

#define N 3 // number of threads
#define MAX 10
int x = 0;

int main(void) {
   pthread_t tids[N];
   int i;

    for(i=0; i<N; i++) {
        pthread_create(&tids[i], NULL, mythread, NULL);
    }

    for(i=0; i<N; i++) {
        pthread_join(tids[i], NULL);
        printf("Thread id %ld returned\n", tids[i]);
    }
    return(1);
}

void *mythread(void *data) {


 while(x < MAX) {
   pthread_mutex_lock(&count_mutex);
   x++;
   printf("Thread ID%ld: x is now %d.\n", pthread_self(), x);
   sleep(2);
}

pthread_exit(NULL);
```

→ pthread_mutex_unlock(&cont_mutex); ✓