1.

Case 1:

- Các hàm enroll() và disenroll() cùng thay đổi giá trị của biến chỉ số sinh viên đang có trong lớp trong cùng một thời điểm dẫn tới có thể sẽ bị miss sự cập nhật của 1 trong 2 hàm.

- Problem này gọi là: **Race condition**

Case 2:

- Tại thời điểm số lượng sinh viên đã đầy nhưng cố gắng đăng ký hoặc số lượng sinh viên đang rỗng nhưng cố gắng xóa đăng ký

- Problem này gọi là: **Producer – Consumer**

2.

Pseudo-code to solve those above problems

Method: Using semaphore

- Khai báo hàm

#define class\_size 50

- sem\_t empty;

- sem\_t full;

- pthread\_mutex\_t mutex;

- int volatile curr\_student;

- enroll( ){

sem\_wait(&empty); /\*sovle case 2\*/

pthread\_mutex\_lock(&mutex); /\* solve case 1/\*

curr\_student++;

pthread\_mutex\_unlock(&mutex);

sem\_post(&full);

}

- disenroll( ){

sem\_wait(&full); /\*sovle case 2\*/

pthread\_mutex\_lock(&mutex); /\* solve case 1/\*

curr\_student--;

pthread\_mutex\_unlock(&mutex);

sem\_post(&empty);

}