```
#include <iostream>

struct process {
    pid_t pid, trun;
    bool flag[2];
    process(pid_t pid_): pid(pid_) {
        flag[0] = flag[1] = false;
    }
}p[2] = {0, 1};

pid_t CF; // Control Flow
void context_switch(const process& caller, process& callee) {
    // synchronize
    callee.trun = caller.trun;
    callee.flag[0] = caller.flag[0];
    callee.flag[1] = caller.flag[1];
    // context switch
    CF = (~CF & 1);
}

int main()
{
    CF = 0;
    // concurrnet execution
    if (CF == 0) {
        p[CF].trun = (~CF & 1);
        context_switch(p[CF], p[~CF & 1]);
    }
    if (CF == 1) {
        p[CF].trun = (~CF & 1);
        p[CF].flag[CF] = true;
        while (p[CF].flag[~CF & 1] && p[CF].trun == (~CF & 1))
            ;
        // critical section
        printf("p%d enters critical section!\n", p[CF].pid);
        context_switch(p[CF], p[~CF & 1]);
    }
    if (CF == 0) {
        p[CF].flag[CF] = true;
        while (p[CF].flag[~CF & 1] && p[CF].trun == (~CF & 1))
            ;
        // critical section
        printf("p%d enters critical section!\n", p[CF].pid);
        return -1;
        printf("p%d exit critical section\n", p[CF].pid);
        p[CF].flag[CF] = false;
        // reminder
    }
    if (CF == 1) {
        printf("p%d exit critical section\n", p[CF].pid);
        p[CF].flag[CF] = false;
        // reminder
    }
    return 0;
}
```