pseudo3.cpp

```cpp

enum SERVICE
{
    DELIVERY,
    CONSULT,
    _TOT_SERVICES
};

#define NONE (-1)

class SantaClausV3 = Monitor
{
    int TOT[_TOT_SERVICES];
    int MIN[_TOT_SERVICES];

    int id_santa_selected;
    int n_santa;
    condition_variable await_someone[];
    condition_variable wait_all_passed[];
    condition_variable wait_greetings[];

    condition_variable wait_service[_TOT_SERVICES];
    int turnstile[_TOT_SERVICES];
    condition_variable wait_end_of_service[];
    bool end_of_service[];

    SantaClausV3(int n_reindeer, int n_elves, int n_santa_, int min_reindeer, int min_elves)
    {
        TOT[DELIVERY] = n_reindeer;
        TOT[CONSULT] = n_elves;
        MIN[DELIVERY] = min_reindeer;
        MIN[CONSULT] = min_elves;

        id_santa_selected = NONE;
        n_santa = n_santa_;

        await_someone = array(n_santa);
        wait_all_passed = array(n_santa);
        wait_greetings = array(n_santa);
        turnstile = {0};
        wait_end_of_service = array(n_santa);
        end_of_service = {false};
    }

    entry void new_service(SERVICE s)
    {
        if (id_santa_selected != NONE)
            if (await_someone[id_santa_selected].any()) // selected Santa is free
                await_someone[id_santa_selected].notify_one();
        while (turnstile[s] == 0)
            wait_service[s].wait(lock);

        turnstile[s]--;
        if (turnstile[s] > 0)
            wait_service[s].notify_one();
        else
            wait_all_passed[id_santa_selected].notify_one();
        while (!end_of_service[id_santa_selected])
            wait_end_of_service[id_santa_selected].wait(lock);

        if (wait_end_of_service[id_santa_selected].any())
            wait_end_of_service[id_santa_selected].notify_one();
        else
            wait_greetings[id_santa_selected].notify_one();
    }

    entry void start_service(SERVICE& s, int id)
    {
        if (id_santa_selected == NONE)
            id_santa_selected = id;
        while (!((wait_service[DELIVERY].getCnt() >= MIN[DELIVERY] || wait_service[CONSULT].getCnt() >= MIN[CONSULT]) && id_santa_selected == id))
            await_someone[id].wait(lock);

        if (wait_service[DELIVERY].getCnt() >= MIN[DELIVERY]) // serving the reindeer
            s = DELIVERY;
        else // serving the elves
            s = CONSULT;
        turnstile[s] = MIN[s];
        end_of_service[id] = false;
        wait_service[s].notify_one(); // first reindeer/elf awakening
        while (turnstile[s] > 0)
            wait_all_passed[id].wait(lock);

        id_santa_selected = NONE;
        for (int i = 0; i < n_santa; i++) // selecting new santa
            if (await_someone[i].any())
            {
                id_santa_selected = i;
                break;
            }
    }

    entry void end_service(int id)
    {
        end_of_service[id] = true;
        wait_end_of_service[id].notify_one();
        while (wait_end_of_service[id].any())
            wait_greetings[id].wait(lock);
    }
};

SantaClausV3 sc = SantaClausV3(20, 10, 5, 9, 3);

void reindeer(SantaClausV3& sc)
```

```cpp
{
    while (true)
        sc.new_service(DELIVERY);
}

void elf(SantaClausV3& sc)
{
    while (true)
        sc.new_service(CONSULT);
}

void santa(SantaClausV3& sc, int id)
{
    SERVICE s;

    while (true)
    {
        sc.start_service(s, id);
        if(s == DELIVERY)
            <delivering toys>
        else
            <arguing with a bunch of stupid elves>
        sc.end_service(id);
    }
}

```