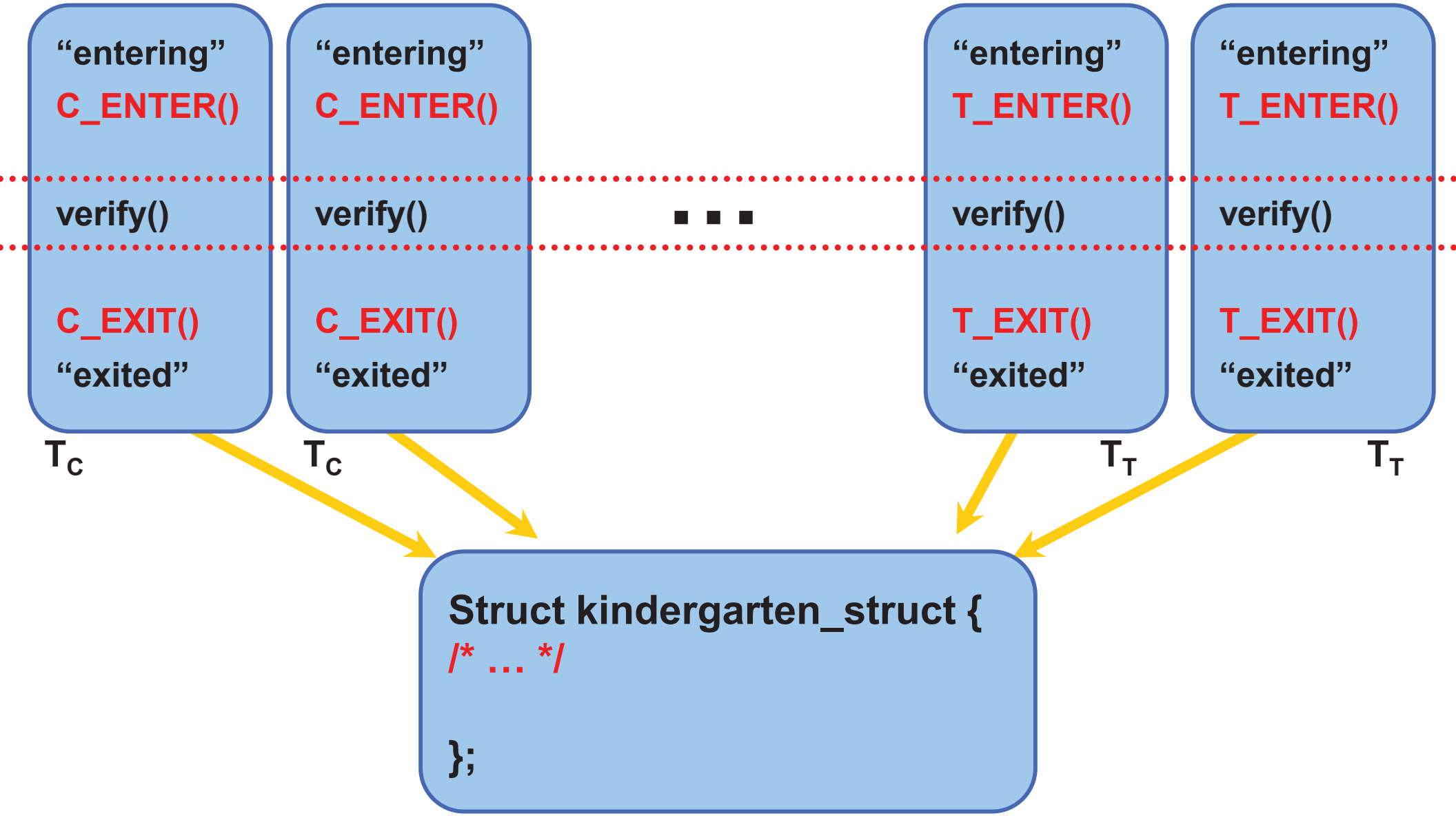
Z3: Επίλυση προβλήματος συγχρονισμού

- Ένα **νηπιαγωγείο (Kindergarten)**
- **Δάσκαλοι** και **παιδιά**.
- Καθορισμένη μέγιστη αναλογία παιδιών ανά δάσκαλο: **R** παιδιά ανά δάσκαλο, π.χ. 3:1.

- Δεδομένη υλοποίηση
- **N** νήματα: **C** νήματα προσομοιώνουν παιδιά, τα υπόλοιπα **N - C** δασκάλους.
- Σας δίνεται κώδικας, που αποτυγχάνει.

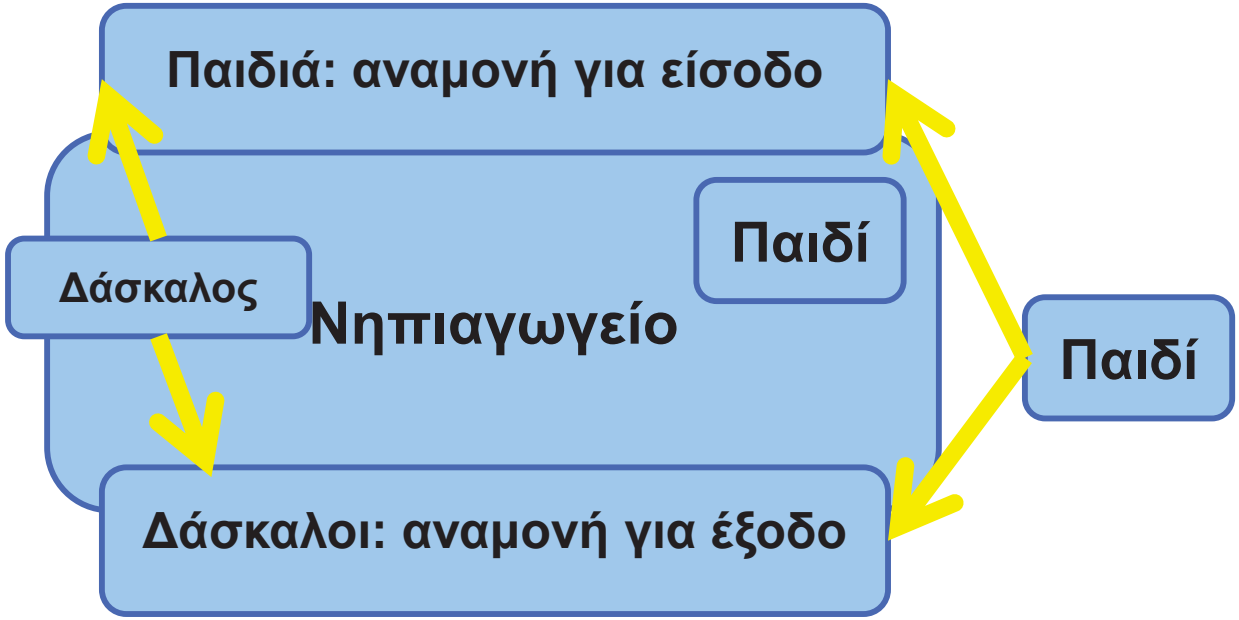
Z3: Επίλυση προβλήματος συγχρονισμού



Ζ3: Επίλυση προβλήματος συγχρονισμού

- Συνθήκες αλλαγής κατάστασης:
  - Παιδί:
    - Μπαίνει -> υπάρχουν τουλάχιστον (**C+1)/R** δάσκαλοι για να με υποστηρίξουν;
    - Βγαίνει -> άνευ όρων (ενημερώνει αν θέλει κάποιος δάσκαλος να βγει αν **(N - C - 1) * R >= C**)
  - Δάσκαλος:
    - Μπαίνει -> αν περιμένουν παιδιά, μπορούν να μπούν μέχρι **R**
    - Βγαίνει -> υπάρχουν αρκετοί δάσκαλοι για να υποστηρίξουν τα παιδιά; **(N - C - 1) * R >= C**.

Z3: Επίλυση προβλήματος συγχρονισμού



Ζ3: Επίλυση προβλήματος συγχρονισμού (condition variables)

```
pthread_mutex_t Lock;
pthread_cond_t cond;
int counter = 0;

/* Thread A */
pthread_mutex_lock(&Lock);
while (counter < 10)
    pthread_cond_wait(&cond,
                      &Lock);
pthread_mutex_unlock(&Lock);
```

```
/* Thread B */
pthread_mutex_lock(&Lock);
counter++;
If (counter >= 10)
    pthread_cond_signal(&cond);
pthread_mutex_unlock(&Lock);
```

```
shared int teacher = 0, child = 0;
mutex mutex(1);
semaphore door(0);

void Teacher()
{
    for (;;) {
        lock(mutex);
        teacher++;
        room += R;
        if (room == R)
            sem_post(door);
        unlock(mutex);

        for (;;) {
            teach();
            lock(mutex);
            if (room >= R) {
                teacher--;
                room -= R;
                unlock(mutex);
                break;
            }
            unlock(mutex);
        }
        go_home();
    }
}
```

```
void Child()
{
    for (;;) {
        for (;;) {
            sem_wait(door);
            lock(mutex);
            if (room) {
                child++;
                room--;
                if (room)
                    sem_post(door);
                unlock(mutex);
                break;
            }
            unlock(mutex);
        }

        learn();
        lock(mutex);
        child--;
        room++;
        unlock(mutex);

        go_home();
    }
}
```

```
void Parent()
{
    for (;;) {
        lock(mutex);
        if (teacher * R >= child)
            printf("Regulation respected\n");
        else
            printf("Regulation violated\n");
        unlock(mutex);
        go_home();
    }
}
```