# АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
# МУТЕКСИ

Задачата се свежда до създаването на емулатор, който да интерпретира последователно по една команда от нишка, като редува нишките. Съществена роля играе правилното структуриране на данните. Предлагам следното решение:

1. Всяка команда се записва в инстанс на структурата:

```
typedef struct{
    int cmd;   // Command Code 1 - HLD; 2 - RLS
    int mutNm; // Mutex Number: A – 0, B – 1, … Z - 25
}command;

#define MAX_THREADS 10 // Максимален брой нишки
#define MAX_CMD 30  // Максимален брой команди в една нишка
#define MAX_MUT 27  // Максимален брой мутекси
```

В двумерния масив **cms[MAX_CMD][MAX_THREADS]** от тип **command** въвеждаме командите на нишките по колони.

В масива **ncmd[MAX_THREADS]** се съхранява броя на командите на всяка нишка (по реда на въвеждането).

В масива **ip[MAX_THREADS]** се съхраняват указателите към текущата команда на всяка нишка (по реда на въвеждането). В началото всички сочат 0. Ако дадена нишка приключи нормално, съответния указател получава стойност (-1).

Масивът **mut[MAX_MUT]** съдържа стойностите на мутексите, последователно от ‘A’ до ’Z’. За мутекс ’A’ е елемента с индекс 0, за мутекс ’B’ е елемента с индекс 1, …, за мутекс ’Z’ е елемента с индекс 25. Масивът **mut** е инициализиран със стойности (-1), което означава, че нито един мутекс не се използва в нито една от командите. За всеки срещнат различен мутекс при въвеждане на данните в съответния елемент на масива се записва стойност 0 (този мутекс се използва в командите и в момента е свободен). В началото всички мутекси са свободни. Ако бъде изпълнена команда HLD, която успява да заключи мутекс – в съответния елемент на масива се записва номера на нишката, на която принадлежи командата. Нишките са номерирани от 1 до M. Обратно → ако дадена команда се опита да заключи мутекс, трябва да провери дали съответния елемент на масива е 0 (мутексът е свободен). Ако стойността е >0 – получаваме номера на нишката, която го е заключила.

Във вложени цикли се изпълняват командите по редове. Ако след последователното изпълнение на по една команда от нишка, „прогрес маркерът“ се е променил, това означава, че не е наличен блокаж – изпълнението продължава. Ако „прогрес маркерът“ е непроменен, е постигнат блокаж – следва печат на резултатите.

Тази задача е много подходяща за използване на генерични структури данни (list, queue, vector).

2.05.2019 г П. Панов