

12. laboratorijas darbs

1. daļa

Ir klase `MatrFun` ar statisko metodi `printRow(int Row, int M[][])`. Metode izvada ekrānā matricas rindiņas `Row` elementus. Pēc kārtējā elementa izvades notiek fiktīva „nulles” aizture.

Galvenajā programmā ir deklarēta matrica `M` un izveidoti divi pavedieni: `Even` un `Odd`. Abi pavedieni izsauc vienu un to pašu augstāk minēto metodi `printRow(...)` no klases `MatrFun`.

Pavediens `Even` apstrādā matricas rindiņas ar indeksiem `0, 2, 4...`

Pavediens `Odd` apstrādā matricas rindiņas ar indeksiem `1, 3, 5...`

Pavedienu izveidošana galvenajā programmā:

```
new Matr(M, "Even", 0, 5);
```

```
new Matr(M, "Odd", 1, 5);
```

Konstruktorā parametri:

1. Matrica `M`.
2. Pavediena vārds `Name`.
3. Pirmās apstrādājamās rindiņas indekss `StartRow(0 vai 1)`.
4. Pavediena prioritāte `Priority`.

Iespējamie programmas rezultāti:

```
//  
// Bez sinhronizācijas  
//  
Row: 0. Elements: 1 Row: 1. Elements: 5 2 6 3 7 4 8  
Row: 2. Elements: 9  
Row: 3. Elements: 13 10 14 11 15 12 16  
Row: 4. Elements: 17  
Row: 5. Elements: 21 18 22 19 23 20 24
```

```
-----  
//  
// Ar sinhronizāciju  
//
```

```
Row: 0. Elements: 1 2 3 4  
Row: 1. Elements: 5 6 7 8  
Row: 2. Elements: 9 10 11 12  
Row: 3. Elements: 13 14 15 16  
Row: 4. Elements: 17 18 19 20  
Row: 5. Elements: 21 22 23 24
```

2. daļa.

Iepriekš uzrakstītajā programmā noformēt metodi `printRow(...)` kā **nestatisko** metodi.

Galvenajā programmā paredzēt `MatrFun` objekta radīšanu un nodošanu pavedienam kā konstruktora parametru (līdzīgi matricai). Paredzēt resursa aizsardzību *tieši* `run()` metodē:

```
public void run() {  
    ... (<Resurss>) {  
        <apstrāde>  
    }  
}
```

Iegūtie rezultāti *būs līdzīgi* 1. daļas rezultātiem.