## Esempio di uso del costrutto monitor

L'algoritmo scan

1

## Algorimo *scan*: scheduling di un disco a teste mobili

- In un ambiente multiprogrammato, il sistema operativo deve servire le richieste di processi che competono per l'accesso a dischi.
- Disco a testina mobile: la testina si sposta radialmente lungo N tracce [0,..N-1]:
  - dalla traccia piu` esterna (0) verso l'interno (direzione SU)
  - nella direzione opposta (GIU)
- L'algoritmo SCAN e` una possibile politica di scheduling delle richieste di accesso al disco, che ha come obiettivo la minimizzazione del numero dei cambiamenti di direzione del braccio del disco.

2

## SCAN

**Politica**:

- ogni richiesta e` caratterizzata da una traccia T, che individua la destinazione della testina, quando la richiesta verra` servita:
  - se la testina si muove verso l'interno (SU), verranno servite tutte le richieste con T raggiungibile in quella direzione (T>posizione corrente)
  - se la testina si muove verso l'esterno (GIU), verranno servite tutte le richieste con T raggiungibile in quella direzione (T<posizione corrente) (le richieste sono servite secondo l'ordine di vicinanza alla richiesta corrente).
- le richieste sono servite secondo l'ordine di vicinanza alla richiesta corrente.
- Quando nella direzione scelta non ci sono più richieste da servire, la direzione del braccio viene invertita ed il procedimento è ripetuto.

3

## Soluzione: monitor

- Definiamo un monitor, il cui compito e` realizzare la politica di servizio mediante le due procedure entry:
  - **Richiesta(T)**: viene invocata dai processi per ottenere l'accesso al disco sulla traccia T
    - se il disco è ***occupato***, la richiesta del processo viene accodata in funzione dello stato corrente del disco (direzione del braccio e numero di traccia interessata) in una coda associata ad una direzione in modo da rispettare la politica di risveglio scelta.
  - **Rilascio***:* viene invocata dai processi per rilasciare il disco:
    - se vi sono richieste in attesa sulla stessa direzione, verra` risvegliata la prima (la piu` vicina alla posizione corrente della testina); altrimenti, la direzione della testina verra` invertita e verra` risvegliato il primo processo in attesa nella nuova direzione.

4

## Monitor

- Il monitor e` caratterizzato dalle seguenti variabili:
  - **occupato**: indica se vi e` un processo che sta accedendo al disco;
  - **direzione**: indica la direzione corrente della testina (SU, GIU)
  - **posizione**: indica la destinazione corrente
  - **dir_SU**, **dir_GIU**: condition associate alle due direzioni, sulle quali si sospenderanno i processi in attesa di servizio.

5

```
typedef int traccia;
typedef enum{SU,GIU}dir;
monitor movimento_braccio
{  traccia posizione=0;
   boolean occupato=false;
   dir direzione=SU;
   condition dir_SU,dir_GIU;

   public void Richiesta(traccia dest)
   { if (occupato==true)
        if ((direzione==SU)&& (dest<posizione))
               dir_GIU.wait(dest);
        else dir_SU.wait(N-dest);
        occupato=true;
        posizione=dest;
   }
```

6

```
public void Rilascio
{ occupato=false;
   if (direzione==SU)
        if (dir_SU.queue)
               dir_SU.signal;
        else{   direzione=GIU;
               dir_GIU.signal; }

   else  if (dir_GIU.queue)
               dir_GIU.signal;
        else{ direzione=SU;
               dir_SU.signal; }
}
} /* fine monitor*/
```

7