Sistemi Operativi (modulo II / B) CON SOLUZIONE 28 maggio 2019

Esercizio 1

Utilizzare i semafori per sincronizzare i seguenti thread in modo che stampino RADAR. (Inserire le P e le V direttamente nel codice qui sotto):

```
thread R {
      print "R";
      V(SemR1);
      P(SemA2);
      print "R";
}
```

```
thread A {
      P(SemR1);
      print "A";
      V(SemA1);
      P(SemD1);
      print "A";
      V(SemA2);
}
```

```
thread D {
      P(SemA1);
      print "D";
      V(SemD1);
}
```

Semafori e valori di inizializzazione: I semafori `SemR1`, `SemA1`, `SemA2`, `SemD1` sono inizializzati a 0 (rosso) e sono utilizzati per attendere la stampa della rispettiva lettera (es. `SemA1` attende la stampa della prima "`A`")

Breve spiegazione della soluzione proposta: L'unico thread che può stampare è `R`. Dopo ogni stampa viene reso verde il rispettivo semaforo. Ad esempio dopo la stampa della prima "`R`" viene reso verde `SemR1`. Prima di ogni print viene attesa la stampa del carattere precedente. Ad esempio, per stampare la seconda "`R`" si attende la stampa della seconda "`A`" (`SemA2`). Ogni print è un'istanza del *produttore-consumatore* in cui si attende che il thread precedente abbia stampato la sua lettera e si segnala la stampa della propria.

Esercizio 2

Si consideri una **variante del problema dei filosofi** in cui i filosofi raccolgono **due bacchette qualsiasi** tra le 5 disponibili. Progettare il monitor `tavola` con i due metodi `raccogli` e `deposita` che raccolgono e depositano **una singola bacchetta** (completare il codice qui sotto, commentandolo opportunamente):

```
Monitor tavola {
  int bacchette = 5; // Ci sono 5 bacchette

  void raccogli() {
    while (bacchette == 0)
      wait(); // non ci sono bacchette, attende
    bacchette--; // raccoglie una bacchetta
  }
}
```

```
void deposita() {
  bacchette++;// deposita una bacchetta
  // notifica solo un thread in quanto
  // le bacchette sono tutte equivalenti
  notify();
}
```

Discutere la possibilità di stallo e progettare due metodi `raccogliAtomico` e `depositaAtomico` che permettano ai filosofi di raccogliere le **due bacchette in modo atomico**, scrivendo il relativo codice:

Se tutti i filosofi si allocano una bacchetta avremo che `bacchette==0` e i 5 filosofi si bloccheranno tutti in attesa della seconda bacchetta. E' una situazione di attesa circolare irrisolvibile, ovvero uno stallo. La raccolta atomica previene lo stallo perché evita il *possesso e attesa*, condizione necessaria alla stallo:

```
  void raccogliAtomico() {
    while (bacchette < 2)
      wait(); // servono 2 bacchette, attende
    // raccoglie due bacchette
    bacchette=bacchette-2;
}
```

```
void depositaAtomico() {
  // deposita due bacchette
  bacchette=bacchette+2;
  // notifica un thread in quanto le
  // bacchette sono tutte equivalenti
  notify();
}
```