

- Naam, adres, collegekaartnummer op eerste vel.	Op ieder volgend vel minstens collegekaartnummer
- Collegekaart zichtbaar op tafel leggen	Iedere vraag duidelijk nummeren.
- Mobiele telefoons uitzetten en niet op tafel	

Met dit tentamen kan je tot 4,5 van de 10 punten voor het decembertentamen verdienen.

### 1. Interrupts en Scheduling

U heeft de hand weten te leggen op een grote partij goedkope PCs. Tijdens de fabricage van deze machines is een fout geslopen in de interrupt hardware, waardoor hardware interrupts niet meer bruikbaar zijn (ze zijn als het ware permanent *disabled*). Software interrupts e.d. werken wel. U gaat voor deze machines een multiprogrammering systeem implementeren, d.w.z. een bedrijfssysteem dat het concurrent executeren van processen mogelijk maakt.

- In een multiprogrammering systeem als Linux wordt een proces switch i.h.a. alleen uitgevoerd na een interrupt. Op welk moment kan in uw systeem, dat immers zonder HW interrupts werkt, een proces switch plaats vinden?
- Is het mogelijk om message passing te implementeren? Licht toe.
- Is het mogelijk om preëemptieve proces scheduling toe te passen? Licht toe.  
(2 punten)

### 2. Processen en scheduling

- Geef in een diagram weer in welke toestanden een (single threaded) proces zich kan bevinden. Geef aan wat elk van die toestanden betekent en hoe de overgangen plaats vinden (geef minstens 5 toestanden).
- Geef een korte beschrijving (o.a. met voor- en nadelen) van de volgende scheduling algoritmen zoals gebruikt voor CPU-scheduling:
  - First-Come-First-Serve
  - Round Robin
  - Shortest Job First; hoe bepaal je wat de "shortest job" is?
- Bij client-server systemen wordt de server vaak als multi-threaded proces geïmplementeerd. Wat zijn de voordelen hiervan?  
(1,5 punten)

### 3. Het "Bakery algorithm"

Peterson en Galvin beschrijven in hoofdstuk 6 een oplossing voor het kritieke-sectie probleem voor meerdere processen, het "bakkerswinkel algoritme". De deelnemende processen hebben elk een uniek procesnummer van 0 t/m (n-1) en delen de volgende variabelen met beginwaarden:

```
int kiezend[n] = {0}, nummer[n] = {0};
```

De toegang tot de kritieke sectie wordt als volgt geregeld door proces i:

```
while(1) {
    /* Niet kritieke deel van het proces */
    k = 0;
    kiezend[i] = 1;
    for (j = 0; j < n; j++) k = (nummer[j] > k) ? nummer[j] : k;
    nummer[i] = k + 1;
    kiezend[i] = 0;
    for (j = 0; j < i; j++) {
        while (kiezend[j]);
        while ((nummer[j]) && (nummer[j] <= nummer[i]));
    }
    for (j = i + 1; j < n; j++) {
        while (kiezend[j]);
        while ((nummer[j]) && (nummer[j] < nummer[i]));
    }
    /* Hier komt de kritieke sectie */
    nummer[i] = 0;
}
```

- Ieder proces trekt een nummer; zijn deze nummers uniek? Waarom?
- In welke volgorde worden de processen toegelaten tot de kritieke sectie? Hoe wordt dat afgedwongen?
- Is de oplossing fair? Waarom?
- Wat is de functie van de `while (kiezend[j])` statements?  
(3 punten)

- |  |  |
|--|--|
| - Naam, adres, collegekaartnummer op eerste vel. | Op ieder volgend vel minstens collegekaartnummer |
| - Collegekaart zichtbaar op tafel leggen         | Iedere vraag duidelijk nummeren.                 |
| - Mobiele telefoons uitzetten en niet op tafel   |  |

#### 4. Semaforen en monitors

- Wat zijn “monitors” en waarvoor worden ze gebruikt?
- Geef in pseudo-code aan hoe je met behulp van een monitor een tellende semafoor zou implementeren (functies takeSem, giveSem en evt. initSem).

(1,5 punten)

#### 5. Resource allocatie

- Drie processen beschikken samen over vier gelijksoortige resources die slechts één voor één kunnen worden aangevraagd of vrijgegeven. Elk proces heeft maximaal twee resources tegelijk nodig, die dan natuurlijk wel één voor één moeten worden aangevraagd. Bewijs dat er geen deadlock kan optreden.
- $N$  processen beschikken samen over  $M$  gelijksoortige hulpbronnen die slechts één voor één kunnen worden aangevraagd of vrijgegeven. Elk proces  $p_i$  heeft maximaal gelijktijdig  $m_i$  hulpbronnen nodig ( $i = 1..N$ ). De waarde van  $m_i$  is minimaal één en maximaal  $M$ ; de som van de maximaal benodigde aantallen hulpbronnen  $m_i$  is kleiner dan  $N+M$ . Bewijs dat er geen deadlock kan optreden. (Hint: probeer de resources zo slecht mogelijk te verdelen.)

(2 punten)