Analyzing Critical Section Solutions

This handout presents several ***proposed*** solutions to the 2 process critical section problem, and analyzes them. In these solutions, one process is numbered 0 and the other is numbered 1. The variable *i* holds the number corresponding to the process executing the code, and the variable *j* holds the number corresponding to the other process. All the code shown is shared by both processes, but the variables *i* and *j* hold different values.

First Proposed Solution

Here, *turn* contains the number of the process whose turn it is to execute the critical section.

```
1 int turn;
                                            /* entry section */
2 while (turn != i)
3             /* do nothing * / ;
                                            /* critical section */
           ...
                                            /* exit section */
4 turn = j;
```

Second Proposed Solution

Here, *inCS*[0] is **true** when process 0 is in the critical section, and **false** otherwise. A similar statement holds for *inCS*[1].

```
1 int inCS[2] = { 0, 0 };
                                            /* entry section */
2 while (inCS[j])
3             /* do nothing * / ;
4 inCS[i] = 1;
                                            /* critical section */
           ...
                                            /* exit section */
5 inCS[i] = 0;
```

Third Proposed Solution

Here, *interested*[0] is **true** when process 0 wants to enter the critical section, and **false** otherwise. A similar statement holds for *interested*[1].

```
1 int interested[2] = { 0, 0 };
                                            /* entry section */
2 interested[i] = 1;
3 while (interested[j])
4             /* do nothing * / ;
                                            /* critical section */
           ...
                                            /* exit section */
5 interested[i] = 0;
```

Fourth Proposed Solution

This combines the first and third proposed solutions.

```
1 int interested[2]; = { 0, 0 };
2 int turn;
                                            /* entry section */
```

```
3 interested[i] = 1;
4 turn = j;
5 while (interested[j] && turn == j)
6            /* do nothing * / ;
                                                  /* critical section */
          ...
                                                  /* exit section */
7 interested[i] = 0;
```