Lista zagadnień nr 15

Zadania domowe

Zadanie 15

Problem obiadujących filozofów to klasyczny problem z dziedziny synchronizacji współbieżnych procesów. Wyobraźmy sobie pięciu filozofów siedzących przy okrągłym stole. Przy stole jest pięć talerzy (po jednym dla każdego filozofa) i pięć widelców (pomiędzy talerzami – każdy filozof ma z lewej i prawej strony po jednym widelcu). Kiedy filozof robi się głodny, musi podnieść dwa widelce (po swojej lewej i prawej stronie), aby zjeść posiłek. Po skończonym posiłku odkłada widelce, aby jego koledzy też mogli ich użyć.

Zaimplementuj procedurę `philosopher`, mającą realizować zjedzenie posiłku przez filozofa. (Procedura ta powinna być oczywiście udostępniona skryptowi sprawdzającemu przy pomocy formy `provide`). Procedura ta będzie otrzymywać dwa parametry. Pierwszym będzie reprezentacja stołu `dining-table`, a drugim – numer filozofa `k` (z zakresu od 0 do 4). Filozof może podnieść `i`-ty widelec, używając wywołania:

```
((dining-table 'pick-fork) i)
```

Odłożenie widelca natomiast jest realizowane wywołaniem:

```
((dining-table 'put-fork) i)
```

Filozof o numerze `k` może sięgnąć tylko do widelców o numerach `k` i $k+1$ (modulo 5). W ramach procedury należy podnieść dwa widelce (co oznacza skonsumowanie posiłku przez filozofa), a następnie je odłożyć.

Każdy filozof będzie posiadać własny, współbieżnie działający do pozostałych wątek. Każdy z tych pięciu wątków będzie wywoływać procedurę `philosopher` pewną liczbę razy (w zależności od tego, jak bardzo głodny jest dany filozof). Należy zadbać o to, aby nigdy nie nastąpiło zakleszczenie (i filozofowie nie umarli z głodu, czekając na swoje widelce).

Nie ma potrzeby synchronizować samodzielnie wywołań `pick-fork` i `put-fork` – operacje te są zaimplementowane w taki sposób, że współbieżne wywołania tych procedur są bezpieczne. Próba podniesienia widelca, który już został

podniesiony przez innego filozofa, spowoduje, że wątek będzie oczekiwał na odłożenie widelca.

Reprezentacja stołu używana do sprawdzenia rozwiązania nie jest udostępniona. Przetestowanie rozwiązania lokalnie wymaga zaimplementowania własnej reprezentacji. Taka implementacja nie jest jednak częścią zadania i nie trzeba jej załączać do rozwiązania.