Конкурентно и дистрибуирано програмирање *04. јул 2016.*

1. (20) К За *readers/writers* проблем направите решење прослеђивањем штафете/жетона (*passing the baton*) уз помоћ семафора, тако да сигнал кôд задовољава следеће услове: Процеси који читају могу да читају када већ има активних читача, само ако нема два или више закаснелих процеса који пишу, у супротном се закашњавају. Ако су у неком тренутку постојала бар два закаснела процеса који пишу, тада процеси који пишу имају приоритет у односу на процесе који читају, све док не нестану сви закаснели процеси који пишу. У осталим ситуацијама важи логика уобичајеног решења *readers/writers*. проблема Напишите општи сигнал кôд (без специфичности везаних за место у кôду). Упутство: користите променљиву *dwflag* која је *true* ако су се појавила два закаснела процеса који пишу.

2. (20) К Проблем путовања лифтом. Путник позива лифт са произвољног спрата. Када лифт стигне на неки спрат сви путници који су изразили жељу да сиђу на том спрату обавезно изађу. Након изласка путника сви путници који су чекали на улазак уђу у лифт и кажу на који спрат желе да пређу. Тек када се сви изјасне лифт прелази даље. Није потребно оптимизовати пут лифта и путника. Написати код за лифт, путнике и мониторе са *Signal and Wait* дисциплином.

3. (20) Симулација монитора помоћу *message passing*-а - активни монитори са више операција. Прикажите аналогије са мониторима и опишите разлике.

4. (20) Аутомобили који долазе са севера и југа морају да пређу реку преко моста (*One lane bridge problem*). На мосту, на жалост, постоји само једна возна трака. Значи, у било ком тренутку мостом може да прође један или више аутомобила који долазе из истог смера (али не и из супротног смера). Користећи CSP написати програм који решава дати проблем. Спречити изгладњивање.

*Испит траје **3** сата.*

Конкурентно и дистрибуирано програмирање *04. јул 2016.*

1. (20) К За *readers/writers* проблем направите решење прослеђивањем штафете/жетона (*passing the baton*) уз помоћ семафора, тако да сигнал кôд задовољава следеће услове: Процеси који читају могу да читају када већ има активних читача, само ако нема два или више закаснелих процеса који пишу, у супротном се закашњавају. Ако су у неком тренутку постојала бар два закаснела процеса који пишу, тада процеси који пишу имају приоритет у односу на процесе који читају, све док не нестану сви закаснели процеси који пишу. У осталим ситуацијама важи логика уобичајеног решења *readers/writers*. проблема Напишите општи сигнал кôд (без специфичности везаних за место у кôду). Упутство: користите променљиву *dwflag* која је *true* ако су се појавила два закаснела процеса који пишу.

2. (20) К Проблем путовања лифтом. Путник позива лифт са произвољног спрата. Када лифт стигне на неки спрат сви путници који су изразили жељу да сиђу на том спрату обавезно изађу. Након изласка путника сви путници који су чекали на улазак уђу у лифт и кажу на који спрат желе да пређу. Тек када се сви изјасне лифт прелази даље. Није потребно оптимизовати пут лифта и путника. Написати код за лифт, путнике и мониторе са *Signal and Wait* дисциплином.

3. (20) Симулација монитора помоћу *message passing*-а - активни монитори са више операција. Прикажите аналогије са мониторима и опишите разлике.

4. (20) Аутомобили који долазе са севера и југа морају да пређу реку преко моста (*One lane bridge problem*). На мосту, на жалост, постоји само једна возна трака. Значи, у било ком тренутку мостом може да прође један или више аутомобила који долазе из истог смера (али не и из супротног смера). Користећи CSP написати програм који решава дати проблем. Спречити изгладњивање.

*Испит траје **3** сата.*