Конкурентно и дистрибуирано програмирање *16. јун 2018.*

1. (12.5) К1 Синхронизација на баријери коришћењем координаторског процеса. Да ли су процеси који се синхронизују на баријери могли да бришу (ресетују) *flag* за стизање на баријеру, а да кôд буде исправан? Образложите.

2. (12.5) К1 Посматра се један узак и релативно кратак пут кроз кањон, којим пролазе каубоји и Индијанци. Ако њиме пролазе само каубоји или само Индијанци, они ће бити фини и проћи једни поред других. Ако кроз клисуру желе да прођу и каубоји и Индијанци, примењује се „закон јачег“, тј. оних којих има више имају првенство пролаза кроз кањон. Треба обезбедити да једном добијено првенство пролаза не важи баш заувек – када се однос снага промени, треба забранити долазак нових особа које су до тада биле бројније, како би они други (сада бројнији) могли да добију право проласка кроз кањон. Коришћењем семафора, написати програм који решава овај проблем.

3. (12.5) К2 Монитор *kasa_radnja* треба да управља редовима свих каса и бројем каса код наплаћивања робе у великој трговинској радњи. Радња има MAXKASA каса, али се зависно од броја купаца динамички мења број каса које раде. Логика рада је следећа: ако број купаца у реду за било коју касу пређе MAXRED, повећава се број каса које раде за 1 (било која каса која није радила може да почне да ради). Ако било која каса постане празна, смањује се број каса које раде за 1, изузев ако ради само једна каса. Купци бирају касу са најкраћим редом. Купци позивају процедуре *biram_kasu* и *odlazim*, а касирке процедуру *spremna_za_rad* и функцију *sledeci_kupac*. Број касирки је увек једнак или већи од броја MAXKASA. Једини процеси који позивају мониторске процедуре су процеси купци и процеси касирке. Интеракција касирке и купца око плаћања и робе није део задатка и у процесима треба написати *roba_za_naplacivanje* и *placam* код процеса купца и *racun_je* код процеса касирки. Не сме се задржавати монитор *kasa_radnja* док траје укуцавање артикала и плаћање. Напишите монитор *kasa_radnja*.

4. (12.5) К2 Постоје два типа атома, водоник и кисеоник, који долазе до баријере (*The H_2O problem*). Да би се формирао молекул воде потребно је да се на баријери у истом тренутку нађу два атома водоника и један атом кисеоника. Уколико атом кисеоника дође до баријере на којој не чекају два атома водоника онда он чека да се они сакупе. Уколико атом водоника дође до баријере на којој се не налазе један кисеоник и један водоник он чека на њих. Баријеру треба да напусте два атома водоника и један атом кисеоника. Користећи условне критичне регионе написати програм који симулира понашање водоника и кисеоника, тако да сваки атом сазна идентификацију атома са којим чини молекул воде, као и да они атоми који су дошли раније, раније и ступе у интеракцију.

5. (15) К3 Претпоставите да је систем за куповину карата реализован у *Lindi* тако што се у простор торки убацују торке за свако седиште сваког лета. Та торка садржи број лета, аеродром узлетања, аеродром слетања, датум узлетања, време узлетања, датум слетања, време слетања и редни број седишта. Написати процесе за креирање директног лета, који поред информација о сваком седишту специфицира и укупан број седишта, и куповину карте. Такође, написати процес који покушава да укине лет, ако се путници који су купили карте за директан лет могу распоредити на друге летове између истих аеродрома узлетања и слетања, са највише једним преседањем. Притом, мора да буде испуњено да узлетање других летова буде у симетричном опсегу око предвиђеног тренутка узлетања лета који се укида од ± 3 сата и слетање на одредишту буде унутар истог дана када је било предвиђено слетање на лету који се укида. Ако то није могуће, не укида се лет и потребно је вратити све на стање пре покушаја укидања лета. Постоји само један процес који покушава укидање летова. Није потребно реализовати део у коме се путници обавештавају о промени лета. Систем може да садржи и друге торке.

6. (15) К3 У свемиру постоји N небеских тела која међусобно интерагују (*N Body Gravitational Problem*), по Њутновом закону гравитације. Свако тело интерагује са сваким, при чему размењују информације о позицији, брзини и вектору силе. Користећи поштанске сандучиће потребно је решити овај проблем користећи торбу послова за дохватање посла. Потребно је реализовати следеће: *Worker* (обавља израчунавање), *Bag* (обавља поделу посла) и *Collector* (обавља прикупљање резултата).

*Испит траје **3** сата.*

Напомена: На вежбанци назначити који део или делови се раде.
Дозвољено је користити готове структуре података (листе, редове, стек, хеш мапе, стабла, ...).