CS 2SD3

Assignment #3. Due April 10 (Monday), 2023, 23:59 via Avenue. Do not hesitate to discuss with TA or instructor all the problems as soon as you discover them. This assignment labour consuming. Start early! This assignment cannot practically be extended as the course ends on Wednesday, April 12.

Instructions: For all assignments, the students must submit their solution to Avenue → Assessments → Assignment #

Students can simply solve the exercises on a paper and use a smartphone app called Microsoft Lens - PDF Scanner **and convert their entire solution into a single PDF file and submit it to avenue. The maximum upload file size is 2Gb in avenue for each submission. Please also attach your LTSA and JAVA files separately.**

Please make sure that the final PDF file is readable.

Students, who wish to use Microsoft word and do not have Microsoft Word on their computer, are suggested to use google document editor (Google Docs**). This online software allows you to convert your final file into PDF file.**

There will be a mark deduction for not following the submission instruction.

Please first finish the assignment on your local computer and at the end, **and attach your solution as a PDF file.**

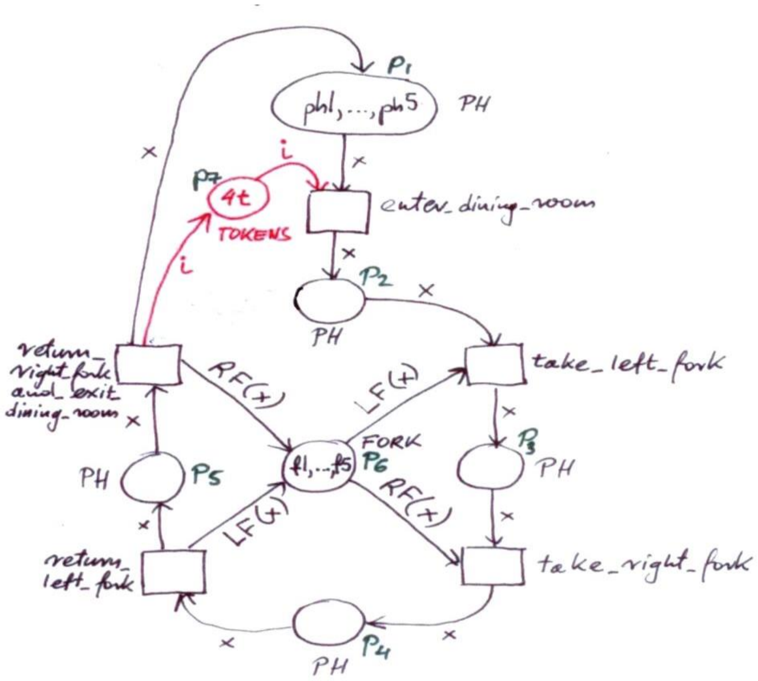
You will have unlimited number of submissions until the deadline.

Students must submit their assignments to Avenue**. *Any problem with Avenue, please discuss with Mahdee Jodayree <mahdijaf@yahoo.com>, the lead TA for this course.***

Total: 123 pts

1.[10] Consider the Coloured Petri Net solution to Dining Philosophers with a butler, presented below.

colour PH = with ph1 | ph2 | ph3 | ph4 | ph
colour FORK = with f1 | f2 | f3 | f4 | f5
colour TOKENS = with t
var x : PH
var i: TOKENS
fun LF x = case of ph1 ⇒ f2 | ph2 ⇒ f3 | ph3 ⇒ f4 | ph4 ⇒ f5 | ph5 ⇒ f1
fun RF x = case of ph1 ⇒ f1 | ph2 ⇒ f2 | ph3 ⇒ f3 | ph4 ⇒ f4 | ph5 ⇒ f5



Interpretation of places:
p_1 - thinking room

p_2 - philosophers without forks in the dining room
p_3 - philosophers with left forks in the dining room
p_4 - philosophers that are eating
p_5 - philosophers that finished eating and still with right forks in the dining room
p_6 - unused forks
p_7 - butler or counter

Prove that this solution is deadlock-free by mimicking the proof of Proposition from page 33 of Lecture Notes 12.

2.[15] A self-service gas station has several pumps for delivering gas to customers for their vehicles. Customers are expected to prepay a cashier for their gas. The cashier activates the pump to deliver gas.

a.[5] Provide a model for the gas station with *N* customers and *M* pumps. Include in the model a range for different amounts of payment and that customer is not satisfied (ERROR) if incorrect amount of gas is delivered.

b.[5] Specify and check (with *N*=2, *M*=3) a safety property FIFO (First In First Out), which ensures that customers are served in the order in which they pay.

c.[5] Provide a simple Java implementation for the gas station system with *N*=2, *M*=3.

3.[15] The cheese counter in a supermarket is continuously mobbed by hungry customers. There are two sorts of customer: bold customers who push their way to the front of the mob and demand services; and meek customers who wait patiently for service. Request for service is denoted by the action *getcheese* and service completion is signalled by the action *cheese*.

(a)[5] Assuming that there is always cheese available, model the system with FSP for a fixed population of two bold customers and two meek customers.

(b)[5] Assuming that there is always cheese available, model the system with Petri nets (any kind).

(c)[5] For the FSP model, show that meek customers may never be served when their requests to get cheese have lower priority than those of bold customers.

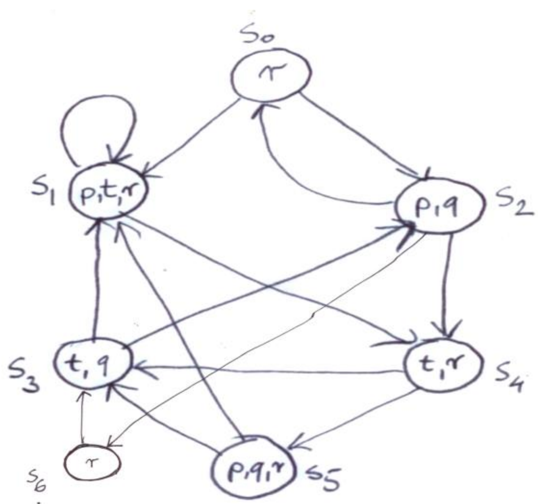
4.[10] To restore order, the management installs a ticket machine that issues tickets to customers. Tickets are numbered in the range 1..*MT*. When ticket *MT* has been issued, the next ticket to be issued is ticket number 1, i.e. the management install a new ticket roll. The cheese counter has a display that indicates the ticket number of the customer currently being served. The customer with the ticket with the same number as the counter display then goes to the counter and is served. When the service is finished, the number is incremented (modulo *MT*). Model this system (with FSP) and show that, even when their requests have low priority, meek customers are now served.

5.[10] Translate the model of the cheese counter from Question 4 into a Java program. Each customer should be implemented by a dynamically created thread that obtains a ticket, is served cheese and then terminates.

6.[10] Design (with FSP) a message-passing protocol which allows a producer process communicating with a consumer process by *asynchronous* messaging to send only a bounded number of messages, *N*, before it is blocked waiting for the consumer to receive a message. Construct a model which can be used to verify that your protocol prevents queue overflow if ports are correctly dimensioned.

7.[38] This question deals with Model Checking.

(a) Consider the system *M* defined below:



Determine whether $M, s_0 \models \varphi$ and $M, s_2 \models \varphi$ hold and justify your answer, where φ is the LTL or CTL formula:

(i)[2] $\neg p \Rightarrow r$

(ii)[2] $\neg$ EG r

(iii)[2] E(t U q)

(iv)[2] F q

b.[6] Express in LTL and CTL: 'Event *p* precedes *s* and *t* on all computational paths' (You may find it easier to code the negation of that specification first).

c.[6] Express in LTL and CTL: 'Between the events *q* and *r*, *p* is never true but *t* is always true'.

d.[6] Express in LTL and CTL: ‘Φ is true infinitely often along every paths starting at s’. What about LTL for this statement?

e.[6] Express in LTL and CTL: ‘Whenever p is followed by q (after some finite amount of steps), then the system enters an ‘interval’ in which no r occurs until t’.

f.[6] Express in LTL and CTL: ‘Between the events q and r, p is never true’.

8.[15] Consider *Readers-Writers* as described in the first part of LN12 and Chapter 7 of the textbook. Take the case of two readers and two writers and provide a model in LTL or CTL (your choice). You have to provide a state machine that defines the model as figures on pages 30 and 33 of LN15 for *Mutual Exclusion*, appropriate atomic predicates as n_1, n_2, t_1, t_2, c_1, c_2 for Mutual Exclusion, and appropriate safety and liveness properties.