Notes:
- The exam is open-book and open-laptop. Web browsing is allowed.
- You are not allowed to communicate with anyone during the exam in any way (except with the lecturer).
- You may submit your exam on paper or electronically. In the latter case, include all files of your submission in a zip file and submit it using the “Submit” button in the course Web page. Allowed file formats are: PDF, PNG and mdzip (MagicDraw).
- If you find that there is not enough information in the text below and you need to make additional assumptions, please write down your assumptions.

Part 1. Pente

*Pente* is a strategy board game created in 1977 by Gary Gabrel, based on the Japanese game *ninuki-renju*. The game is played on a board with 19 vertical and 19 horizontal lines. The description of the game below is taken from Wikipedia and [www.pente.net](http://www.pente.net).

*How to play*: The game starts with the board completely clear of stones. The first player (black) begins the game by playing one stone on the center point. Thereafter the players take turns placing their stones, one at a time, on any empty intersection. The stones are placed on the intersections of the lines (including the very edge of the board), rather than in the squares. Once played, a stone cannot be moved again, except
when removed by a capture. Players alternate turns adding new stones to the board, building up their positions, until one player wins.

Captures: Whenever your opponent has two stones (and only two) which are adjacent, those stones are vulnerable to capture. The pair can be captured by bracketing its two ends with your own stones. The captured stones are removed from the board. Captures can be made vertically, horizontally, or diagonally, and multiple captures can occur on a single move.

To capture stones

bracket your opponent

and remove the pair.

Winning the game: The game ends immediately when one player places five stones in a row or captures five pairs of opponent’s stones. The opposing player has no "last chance" to make a final move. When a player obtains an unblocked row of four stones, called a tessera, a win is imminent. Therefore, an unblocked row of three stones, called a tria, is a serious threat that should be blocked unless a stronger offensive move exists. An unblocked row of three stones, if it contains one gap, is still considered a tria. In the example below, black has formed a tria, while white has formed a tessera and will win with the next move. In a Pente game, the player who does the last move is always the winner (we assume that there are no draws).
Task 1. [8 points]
Design a domain model (UML class diagram) to store information regarding *Pente* tournaments. A tournament is identified by the date and the place where it is held. A tournament consists of a set of played games. Every game has two players one playing whites and the other blacks. A game should contain just enough information to replay the sequence of moves performed by the players during the game.

Task 2. [8 points]
Write a sequence diagram that describes how the classes in the domain model (and other classes if needed) interact in order to get the number of opponent’s stones captured by the winner of a certain game.

*Hint:* to count the number of stones it may be useful to replay the moves of the game and track how many stones are captured after each move. To do this, it is possible to add a class to keep the game state after every move as a distribution of stones. In the game state the total number of white stones captured and the total number of black stones captured can be stored. A method (that you can suppose to be a black-box) should update collection of stones after every move and return if the current move was a capture or not.

Task 3. [5 points]
Suppose to have drawn a sequence diagram to get the number of opponent’s stones captured by the winner of a certain game (the one described in task 2). Write a sequence diagram to get the number of opponent’s stones captured by a player in the entire tournament whenever he or she is the winner of a game of the tournament.

Task 4. [4 points]
Design an application model for the above scenario(s) *(the application must contain only operations derived from the sequence diagram(s) and the additional application classes (outside the domain) that are needed to implement the application)*.

### Part 2. Coffee machine

Task 5. [10 points]
Specify the controller of a coffee machine as a statechart diagram.
The key components of the coffee machine are:

1. A single power on/off button,
2. A water reservoir with level sensor (with capacity of up to 2 litters),
3. A warming plate with a weight sensor and a presence sensor (to detect if the carafe is placed on it and the weight inside the carafe),
4. An ergonomic glass carafe with capacity of 1500 ml, and
5. A programmable auto-off timer (for the sake of brevity, we will assume that the function “auto-off timer programming” is specified elsewhere – and not by you. Therefore, we will assume that the auto-off timer is set to 2 hours).

The operation of the coffee machine is as follows. The coffee machine starts operating when the user presses the power button. If the water reservoir is not empty and the carafe is on the warming plate, the coffee machine starts making coffee (i.e. warming up the water in the reservoir). The coffee machine stops making coffee whenever one of the following happens:

1. The carafe contains approx. 1500 ml of coffee, as sensed by the weight sensor (we assume that the weight sensor reports the weight in millilitres based on an approximate conversion from grams to millilitres).
2. the water reservoir is empty (sensed by the level sensor), or
3. the carafe is removed from the warming plate (sensed by the presence sensor).

The warming plate will operate as long as the carafe is on it. However, for safety reasons the warming plate must be turned off when:

1) the water reservoir is empty (sensed by the level sensor),
2) the carafe contains less than 100 ml (sensed by the weight sensor), or
3) the carafe is removed from the warming plate (sensed by the presence sensor).

When the user returns the carafe to the warming plate, the coffee machine resumes its operation in the state it was before removing the carafe (e.g. either coffee making or...
warming-only). Since the status of the components of the coffee machine might have been changed while the carafe was out of the warming plate (e.g. the carafe is empty when it is put back into the warming plate), it is important to periodically monitor the state of the components of the coffee machine (e.g. amount of water in the reservoir, amount of water in the carafe, etc.)

The coffee machine is automatically turned off after 2 hours of operation or when the carafe has been removed from the warming plate for more than 5 minutes.

Finally, the user can manually turn the coffee machine off by keeping pressed the power button for at least 2 seconds. (Hint: you can consider that the power button generates two events: buttonPressed and buttonReleased).

Use the following statechart as the starting point for your solution:

For this task, you should submit a statechart diagram (on PDF, PNG or MagicDraw format).

**Part 3. Petri nets**

We consider a segment of a factory with two conveyor belts, two machines, one robot and one buffer (see Figure 3). Raw parts arrive through a first conveyor belt, called the raw line. The robot moves each part from the raw line into machine M1, then into the buffer, then into machine M2 and finally to the second conveyor belt (called the finished line). If machine M2 is free, the robot is also allowed to move a part directly from M1 to M2.

M1 can hold at most one part at a time, and the same applies for M2. The robot can only move one part a time. The buffer can hold at most 7 parts. The conveyor belts can hold any number of parts. The following figure shows the case where the raw line holds 2 parts, M1 and M2 hold one part each, the buffer holds two parts, and the finished line holds one part.
Task 6. [2 points] In the Petri net in Figure 1, if machine M2 is empty, the robot can move a part from machine M1 to machine M2 directly, but it can also choose to move the part from machine M1 to the buffer (even if M2 is empty). Modify the above Petri net so that if the robot can move a part from M1 directly to M2, it will do so (it will not move that part to the buffer).

Task 7. [5 points] Modify the Petri net in Figure 1 so that a new part can be added to the raw conveyor belt if and only if at least one of the following conditions is true:

- Machine M1 is empty
- Machine M2 is empty
- The buffer is not full
Note: Tasks 5 and 6 can be submitted as a single Petri net integrating both modifications. They can be submitted in PDF, PNG or PNML format.

Task 8 [8 points]
We consider the following Petri net.

- There is no reachable marking, in which both $s_3$ and $s_7$ contain a token. Explain in your own words how the place $s_4$ and the transitions $t_2$, $t_3$, $t_5$ and $t_6$ make sure, that this cannot happen.
- Modify the Petri net in such a way, that a marking in which both $s_2$ and $s_6$ contain a token is also not reachable.

Submit your answer to this question in PDF, PNG or PNML format.

Acknowledgment: Exercise originally written by Prof. Barbara König, University of Duisburg-Essen