Concordia University

Department of Computer Science and Software Engineering

SOEN 331 - Sections S and U: Formal Methods for Software Engineering

Assignment 2 on Relational calculus and Z specification

Dr. Constantinos Constantinides, P.Eng.

constantinos.constantinides@concordia.ca

14 March, 2024

1 General information

Date posted: Thursday, 14 March, 2024.

Date due: Thursday, 28 March, 2024, by 23:59.

Weight: 10% of the overall grade.

2 Introduction

You must find a partner and between the two of you should designate a team leader who will submit the assignment electronically. There are three (**3**) problems in this assignment. The total weight of the assignment is **100** points.

3 Ground rules

1. This is an assessment exercise. You may not seek any assistance while expecting to receive credit. **You must work strictly within your team and seek no assistance for this assignment (e.g. from the teaching assistants, fellow classmates and other teams or external help)**. You should **not** discuss the assignment during tutorials. I am available to discuss clarifications in case you need any.

2. **Both partners are expected to work relatively equally on each problem**. Accommodating a partner who did not contribute will result in a penalty to **both**. You cannot give a "free pass" to your partner, with the promise that they will make up by putting more effort in a later assignment.

3. **You must prepare this assignment in LaTeX**, by using a compiler on your local machine (see Resources on course website). Submissions not prepared using LaTeXwill **not** be accepted.

4. If there is any problem in the team (such as lack of contribution, etc.), you must contact me as soon as the problem appears.

5. Make sure you have an up-to-date VPN client on your local machine and you are able to access our electronic submissions system.

6. Late submissions will **not** be accepted, and no submissions will be accepted by email.

PROBLEM 1 (40 pts)

Consider a system that handles railway connections as it associates (source) cities to all their corresponding destinations. The requirements of the system are as follows:

1. A source city may have connections to possibly multiple destination cities.
2. Multiple source cities may share the same group of destination cities.

We introduce type $CITY$. A possible state of the system is shown below as it is captured by variable *connections*:

$$
\begin{array}{l}
connections = \\
\quad \{ \\
\quad\quad Montreal \mapsto \{Ottawa, Kingston, Quebec, Halifax\}, \\
\quad\quad Ottawa \mapsto \{Montreal, Toronto\}, \\
\quad\quad Toronto \mapsto \{Montreal, Ottawa\}, \\
\quad\quad Halifax \mapsto \{Montreal, Quebec\}, \\
\quad\quad Quebec \mapsto \{Montreal, Halifax\}, \\
\quad\quad Kingston \mapsto \{Montreal\} \\
\quad \}
\end{array}
$$

1. (2 pts) Is *connections* a binary relation? Explain why, and if so express this formally.
2. (2 pts) In the expression $connections \in (...)$, what would the RHS be?
3. (2 pts) Is *connections* a function? If so, define the function formally, and reason about the properties of injectivity, surjectivity and bijectivity.
4. (2 pts) Describe the meaning and evaluate the following expression:

   $$\{Montreal, Halifax\} \lhd connections$$

5. (2 pts) Describe the meaning and evaluate the following expression:

   $$connections \rhd \{\{Montreal, Halifax\}\}$$

6. (2 pts) Describe the meaning and evaluate the following expression:

$$\{Montreal, Quebec, Halifax\} \lhd connections$$

7. (2 pts) Describe the meaning and evaluate the following expression:

$$connections \rhd \{\{Ottawa, Kingston, Quebec, Halifax\}, \{Montreal, Ottawa\}, \{Montreal\}\}$$

8. (2 pts) Describe the meaning and evaluate the following expression that forms a post-condition to some operation:

$$\begin{aligned} connections' = connections \oplus \{ & \\ & Halifax \mapsto \{Montreal, Charlottetown, Quebec\}, \\ & Charlottetown \mapsto \{Halifax\} \\ & \} \end{aligned}$$

9. (6 pts) Assume that we need to add a new entry into the database table represented by *connections*. We have decided <u>not</u> to deploy a precondition. What could be the consequences to the system if we deployed a) **set union** and b) **relational overriding**?

10. (2 pts) Consider operation *AddConnection* to **add a new entry** to the table, defined by the following pair of assertions:

$$\begin{aligned} & city? \notin \mathrm{dom}\ connections \\ & connections' = connections \cup \{city? \mapsto destinations?\} \end{aligned}$$

What would be result of the call `AddConnection(Montreal, (Boston, NYC))`, and in the case of failure, whom should we blame and why?

11. (2 pts) Consider the following modification to the postcondition of *AddConnection*:

$$connections' = connections \oplus \{city? \mapsto destinations?\}$$

What would be result of the call `AddConnection(Kingston, (Boston, NYC))`? In the absence of a precondition, can relational overriding unconditionally capture the intent of the operation?

12. (3 pts) Consider the following state schema in the Z specification language:

$$
\begin{array}{|l}
\hline
\textit{RailwayManagement} \\
\textit{cities} : \mathcal{P}\, CITY \\
\textit{connections} : CITY \pfun \mathcal{P}\, CITY \\
\hline
\textit{cities} = \mathrm{dom}\ \textit{connections} \\
\hline
\end{array}
$$

Define the schema for operation *GetDestinations* which returns all destinations for a given city.

13. (1 pt) (**PROGRAMMING**) Define global variable *connections* in Common LISP and populate it with some sample data. Demonstrate that the variable indeed contains the ordered pairs as shown above.

14. (3 pts) Describe how you would validate variable *connections*, i.e. how to show that it holds a function.

15. (3 pts) (**PROGRAMMING**) Define a predicate function, `isfunctionp`, in Common LISP that reads a variable like *connections* and indicates if the variable corresponds to a function or not.

16. (2 pts) (**PROGRAMMING**) Define function *GetDestinations* in Common LISP.

PROBLEM 2 (50 pts)

Consider an airport management system. Each airport has a unique id (e.g. Montreal:YUL). Let us introduce the types $AIRPORT$ and $CITY$. We also introduce variable *airports* that contains associations between airport id's and their corresponding host cities. A possible state of the system is shown below:

$$
\begin{array}{l}
airports = \\
\quad \{ \\
\quad\quad YUL \mapsto Montreal, \\
\quad\quad LCY \mapsto London\_UK \\
\quad\quad LHR \mapsto London\_UK, \\
\quad\quad MIL \mapsto Milan, \\
\quad\quad SFO \mapsto San\_Francisco, \\
\quad\quad SDQ \mapsto Santo\_Domingo \\
\quad \}
\end{array}
$$

1. (1 pt) (**PROGRAMMING**) Define global variable *airports* in Common LISP and populate it with the above data. Demonstrate that the variable indeed contains the ordered pairs as shown above.

2. (2 pts) Provide a declaration of variable *monitored* that holds all airport id's.

3. (3 pts) What kind of variable is *airports*? Provide a formal definition together with any and all applicable properties.

4. (3 pts) Describe what *data structure* you would deploy to model variable *airports*. Note that you may not use a Dictionary. Should this be an ordered or an unordered structure? Discuss.

5. (4 pts) Provide a formal specification of the state of the system in terms of a **Z specification schema**.

6. (6 pts) Provide a schema for operation `AddAirportOK` that adds a new airport to the system. With the aid of success and error schema(s), provide a definition for operation `AddAirport` that the system will place in its exposed interface.

7. (3 pts) (**PROGRAMMING**) Define function *AddAirport* in Common LISP.

8. (6 pts) Provide a schema for operation `UpdateAirportOK` that updates the host city of a given airport. With the aid of success and error schema(s), provide a definition for operation `UpdateAirport` that the system will place in its exposed interface.

9. (1 pt) What would be the value of variable *airports* upon calling *UpdateAirport*(*YUL*, *Dorval*)?

10. (3 pts) (**PROGRAMMING**) Define function *UpdateAirport* in Common LISP.

11. (6 pts) Provide a schema for operation `DeleteAirportOK` that removes an airport from the system. With the aid of success and error schema(s), provide a definition for operation `DeleteAirport` that the system will place in its exposed interface.

12. (3 pts) (**PROGRAMMING**) Define function *DeleteAirport* in Common LISP.

13. (6 pts) Provide a schema for operation `GetAllAirportsOK` that returns all airports hosted by a given city. With the aid of success and error schema(s), provide a definition for operation `GetAllAirports` that the system will place in its exposed interface.

14. (3 pts) (**PROGRAMMING**) Define function *GetAllAirports* in Common LISP.

PROBLEM 3 (10 pts)

Consider the temperature monitoring system from our lectures notes. In this part, we will extend the specification with a new requirement:

Operation ReplaceSensor Sensors are physical devices, subjected to deterioration, or other type of damage. We need to provide the ability of the system to replace a sensor with another. For simplicity, let us not plan to ("fix" and) reuse them, but permanently remove them from the system.

4 What to submit

Your main submission consists of a `.tex` file and its corresponding `.pdf` compilation. All code should (1) be embedded in your main submission file and (2) be included as supporting documentation in separate files, and placed in a folder called `/code`. All demonstrations of interactions with a language environment should be embedded in your main submission file.

Place all the above files (`.tex`, `.pdf`, and the `/code` folder) into a folder that is named after you and your partner, where the name of the person to submit goes first, e.g. if Lisa Gerrard and Nick Cave were partners and Lisa were the one to submit, then the folder is called `/gerrard-cave`. Zip your folder and submit it at the Electronic Assignment Submission portal at

(`https://fis.encs.concordia.ca/eas`)

under **Assignment 2**.

END OF ASSIGNMENT.