COMP 3004 – *SCAPES* Deliverable #2

<u>Due:</u> Thursday, December 5, 2019 at 4:00 PM (afternoon)

<u>Collaboration:</u> This deliverable must be completed by **registered teams**

<u>Feature Implementation</u>

You will start from your D1 code, and you will implement the following *SCAPES* features, as specified in the project description posted in *cuLearn*:

- Compile an existing program that may contain **all** the instructions in the SCAPL instruction set, as specified in Table 1 of the SCAPES project description
- Execute an existing program that has been compiled by SCAPES

Design requirements:

- The output of the compilation step must use an internal format designed by your team, consisting of **objects** representing the SCAPL instructions and variables of the program, as well as the associations between these objects. Your object design should closely match the object model found in the Assignment #2 solution diagram.
- The compilation output, in the form of internal format objects, **must** be saved to persistent storage in serialized form. Regardless of your choice of serialization techniques, the output must be human-readable. Submissions where the compilation output (the serialized objects) cannot be viewed in persistent storage, or where the serialized objects are not human-readable, will not be graded.
- The input of the execution step must be a compiled program that is stored in the internal format that is output from the compilation step, and that is retrieved from persistent storage.
- Both of the two features in this deliverable **must** be implemented to use polymorphism, as indicated in the Assignment #2 solution diagram.
- Your code must include the implementation of a Gang of Four design pattern, to be documented briefly in your README file.
- Both features must support the usage of array elements where indicated in the SCAPL instruction set.

Implementation requirements:

- The Linux Ubuntu platform, as provided in the official course virtual machine (VM), will be used as the test bed for evaluating your code.
- All source code must be written in C++, and it must be designed and implemented at the level of a 3rd year undergraduate Computer Science student, following standard UNIX programming conventions, and using advanced OO programming techniques such as polymorphism and design patterns.
- Only libraries already provided as part of the course VM may be used.
- The *SCAPES* user interface (UI) will preferably be graphical in nature. A console-based UI will be an acceptable alternative, but will not earn full marks for any of the coding deliverables. In general, user features should be easily navigable, either as menu items and/or pop-up menus. The look-and-feel of the *SCAPES* system should be professional and consistent with commercially available UIs.
- The *SCAPES* system will run on a single host, with all its data stored on that same host.
- Data storage, both in memory and in persistent storage, must be organized for ease of retrieval and efficient use of space. There should be no duplication of information anywhere.
- Persistent data may be stored in flat files, or any other mechanism already available in the VM provided, including the Qt SqlLite library.

Submission requirements:

- Delivery and deployment of your code must be *turnkey*. This means that the user must be able to install all the software for the project with one (1) command, build it with one (1) command, and then launch the project with one (1) command. Specific instructions for building and launching the project must be included in a README file. Projects that do not conform to these requirements will not be graded.
- Your submission must contain a minimum of two (2) complete SCAPL-language programs, already stored in persistent storage. One of the programs can be basic with at least one (1) loop, and the other program must be more complex with at least two (2) loops (including at least one (1) nested loop), and at least one (1) compound condition. Each program:
  - must have a useful purpose (not just a random set of instructions)
  - must use **all** the instructions in the SCAPL instruction set
  - must be fully documented using inline comments
  - must print control flow information during execution, for example after every jump
  - must print out the contents of all variables at the end of the program

<u>Grading</u>

Grading breakdown:

• *Feature implementation:*	*100%*
Compile program	50%
Run program	50%

<u>Format</u>

Coding deliverables must be delivered as a single **`tar`** or **`zip`** file consisting of all source code, data files, and configuration scripts, as well as installation, build, and launch instructions in a readme file. Do **not** provide object files and project executables as part of your submission.