Concordia University
Department of Computer Science and Software Engineering
Advanced Program Design with C++
COMP 345 --- W2016

Instructor: Dr. Nora Houari

Project Final Build

Demonstrating full capacity of the *POWER GRID* system

Due April 13th, 2016(strict)

Team Name:	TEAM U

Team members	Signature
Audrey-Ann Jean-Weisz	ID # 26980104
Catalin Popescu	ID # 29678999
Marco Lyi	ID # 29815341
Andre Marques Manata	ID # 27148225

1. Final Incremental Build General Information

You must deliver an operational version demonstrating the full capacity of your system. This is about demonstrating that the code build is effectively aimed at solving specific project problems and completely implementing specific system features. The code build must not be just separated portions of the final project, but a fully operationally integrated software that can be demonstrated by its operational usage.
The presentation should be organized as follows:

1. Brief presentation of the Problem Analysis, Design, and Use of Engineering Tools as listed below under "Graduate attributes—skills**"**
2. Demonstration of the functional requirements as listed below under "Functional Requirements".

You are graded according to how effectively you can demonstrate that the features are implemented. If you cannot really demonstrate the integrated features through execution, you will have to prove that the features are implemented by explaining how your code implements the features and what are the expected integration problems, in which case you may lose some marks, even if your explanations are satisfactory.
During your presentation, you have to demonstrate that you are well prepared for the presentation, and that you can easily provide clear explanations as questions are asked about your understanding of the problem being solved, the structure and functioning of your code, as well as your use of tools.

The final Build of the project shall include the following essential game components and features.

Game components:

In addition to the essential game components required for build#1, this final build shall include

- the "Step 3" card

Required features:

a) Game setup
b) Basic dynamics

Both features are detailed in the following two subsections.

a) Game Setup:

In addition to the game setup requirements for build 1:

- Load (from file) "Step 3" card.

b) Game Dynamics

<u>Basic play</u>

- Game play for three-players must be supported.
- Perform the rounds play of: (1) select the player order, (2) auction power plants, (3) buying resources, (4) building, and (5) Bureaucracy.

- Players have their cities connected on the map and their resources updated (as per the gaming rules)

Instructions:

Once players have finished the previous rounds (phase 1 to 5) of the game; the connected cities on the map are updated and the resources are saved.

2. Evaluation Criteria

The following criteria will be used to evaluate build#1 of the project:

- Final Build is executable.
- Implementation and operationalization of required game components and features.
- Quality of design, code and documentation.
- Your system architecture and use of design patterns
- Quality of presentation / demo.

3. Grading

Functional requirements			**70**
Map creation and editing			**32**
User-driven creation of map (board) elements, such as province, country, and connectivity between cities			9
Saving a map to a file exactly as edited.			4
Loading a map from an existing file, then editing the map(e.g. adding connections between cities)			3
Load at the bottom of the board the resource market			4
Verification of map correctness before saving			6
Load (from file) a player possession: overview card, houses, and money, and place it on the board.			3
Load from the file(s) o The remaining resources as a supply on the game area. o The power plant cards near the board. o "Step 3" card			3
Game Play			**38**
Implementation of game driver implementing the game phases(1- 5)			10
Game starts by user saved map file, then load the map as a connected graph			5
Phase 1: User chooses two players game, and each takes possession of an overview card, houses, and money.			4
Each player places one of his houses left of the 1 on scoring track, to begin the game			3
Place the remaining resources as a supply near the board			3
Take the power plant cards and place them near the board			3
Basic playing: **Phase 2:** Auction power plant **Phase 3:** Buying resources **Phase 4:** Building **Phase 5:** Bureaucracy			10
Graduate attributes- skills			**30**
Knowledge-base	Indicator 1.3: Knowledge-base in a specific domain: demonstrated knowledge of programming principles used in the implementation.		2
Design	Indicator 4.1: Problem identification and information gathering: knowledge and correct understanding of the functional requirements and the game rules.		4
	Indicator 4.3: Architectural and detailed design: Rationale for overall project architectural structure.		6
	Indicator 4.4: Implementation and validation: Correct use of C++ features leading to stable execution that has been properly tested in various situations.		4
Use of tools	Indicator 5.1: Ability to use appropriate tools, techniques and resources: proficient use of particular		4

	tools (C++ language, libraries, project management 2tools, etc.) for the implementation.		
	I4ndicator 5.2: Ability to select appropriate tools, techniques, and resources: justified adoption of tools in the project (e.g. compiler, IDE, libraries, project management tools, etc.).		2
Communication	Indicator 7.3: Documentation: Code readability: layout, naming. Consistent use of comments		4
	Indicator 7.4: Oral presentation: Structure and demonstrated preparation of presentation, using appropriate presentation techniques. Demonstrated knowledge of code base/clarity of explanations.		4
Total			**100**

Appendix A: Team's check list

Each member of the team and the team as whole need to provide a good design and programming style, as indicated below.

Design and Programming style
A good architectural and modular design (modules, design patterns, .h and cpp, files)
Proper use of comments
Proper use of significant names for identifiers, code readability
Proper structure of base class, derive class, arrays, pointers, vectors.
Good memory management (free of the heap after usage)

Appendix B: Individual Student Contribution

Each member of the team needs to complete the peer evaluation form found with the project on Moodle prior their presentation.