N.McFadyen 2022

Board Game Analysis with Computational Thinking

Group Members:

Game:

As you play your chosen game, keep track of your discussions and experiences in the concept map below. Follow the instructions in class and on this sheet. Complete the attached handouts as you play. Do this as a group.

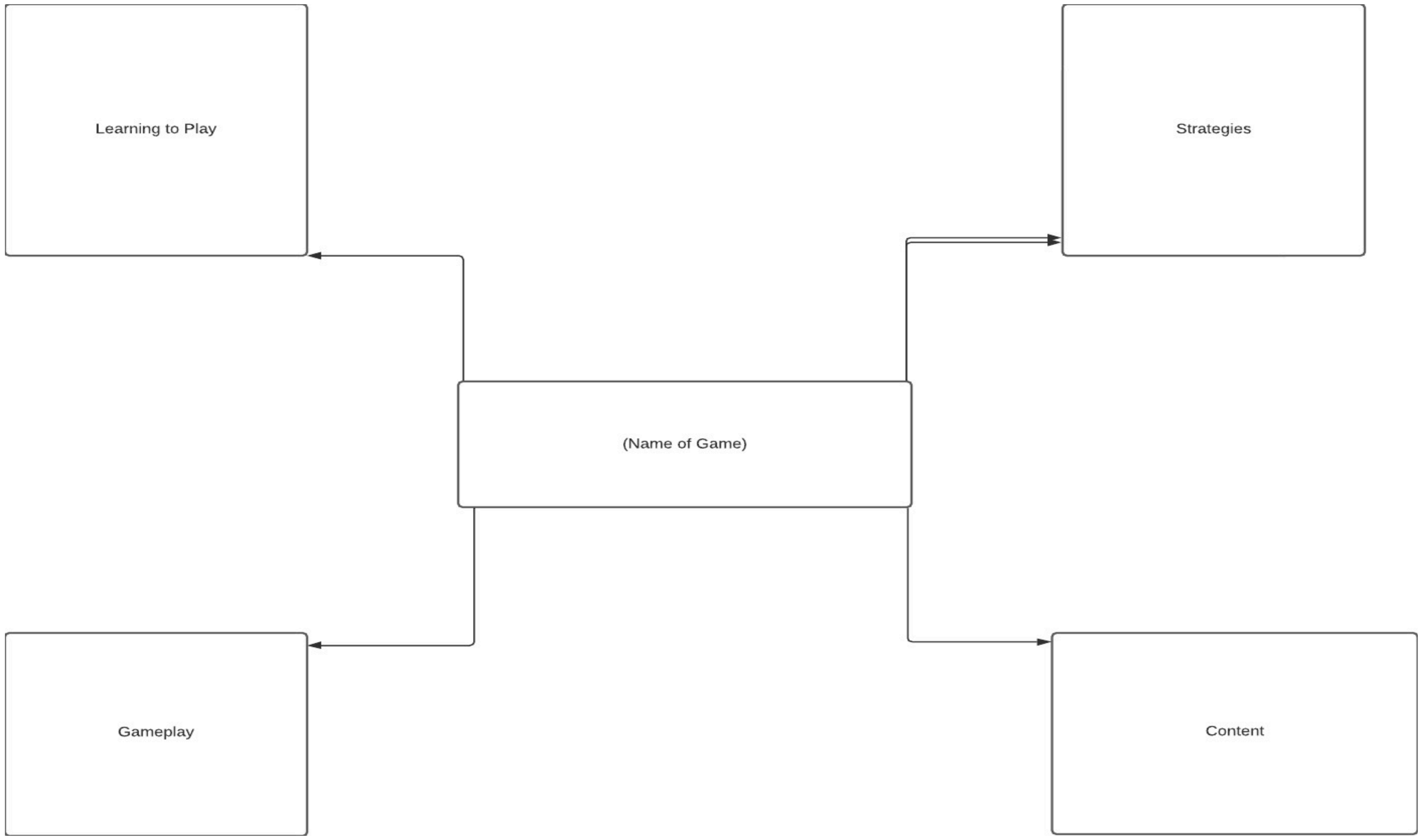
1. ***Choose a board game to play and read over this assignment as a group.***
2. ***Learn to play the game by watching a "How it's played video on YouTube" and have at least one member of your group read the rules***
3. ***Play a practice round of the game.***
4. ***Discuss with your group if you interpreted the rules correctly. Revise what you need to.***
5. ***Play the game and take turns completing the sheet. You should have an open discussion as you play regarding the questions on the sheet.***
6. ***After the game is complete, review your answers as a group. There will be a debrief discussion, submit your work.***

Elements of Computational Thinking

- *Decomposition*
- *Pattern Recognition*
- *Abstraction*
- *Debugging*
- *Simulations*
- *Algorithms*

N.McFadyen 2022

Complete this concept map as you play the game. Write down your thoughts, ideas, and discussions. Think about if the game was easy to learn, what strategies did you use, what kinds of content (story, art, theme) does the game have, what is the gameplay like?



N.McFadyen 2022

Board Game Computational Thinking Analysis

Name:
Date:

1. Decompose the game, and indicate your tasks, goals, actions, and win conditions.
    a. Task:

    b. Goals:

    c. Actions you can take:

    d. Win Conditions

2. What patterns or similarities repeat themselves throughout the game?
    a. Pattern 1
    b. Pattern 2

3. **Abstraction:** What details in the game are irrelevant to the goals? What actions are necessary to be successful?

4. **Algorithms**
   a. What strategies did you employ? What was successful? Did you revise this?

   b. As you learned to play did you have to **debug** your understanding of the rules or instructions while you played?

   c. Provide step-by-step instructions as to how to be successful when you play the game?

   1.

   2.

   3.

   4.

The four cornerstones of computational thinking

There are four key techniques (cornerstones) to computational thinking:

- **decomposition** - breaking down a complex problem or system into smaller, more manageable parts
- **pattern recognition** – looking for similarities among and within problems
- **abstraction** – focusing on the important information only, ignoring irrelevant detail
- **algorithms** - developing a step-by-step solution to the problem, or the rules to follow to solve the problem

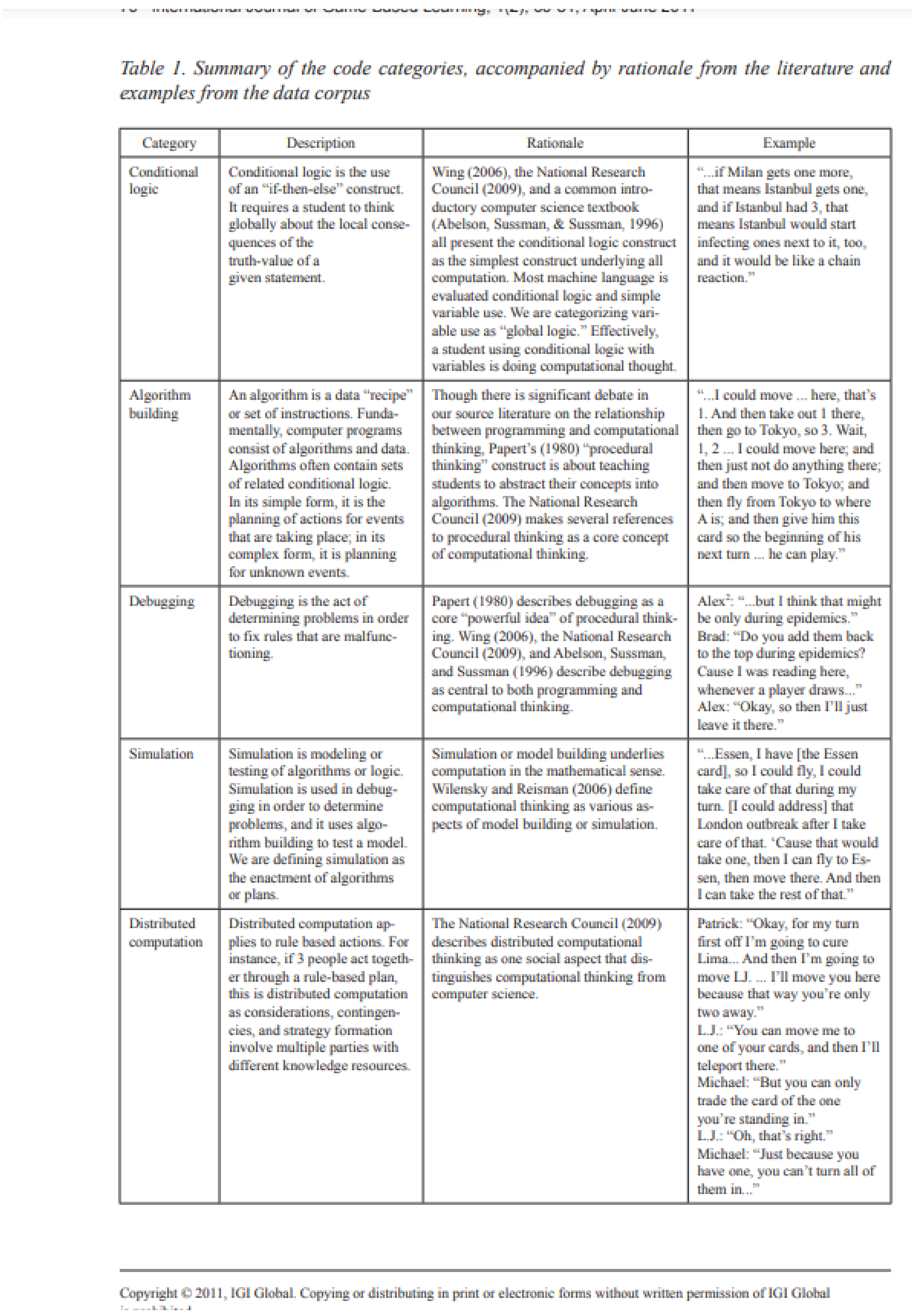


Table 1. Summary of the code categories, accompanied by rationale from the literature and examples from the data corpus

Category	Description	Rationale	Example
Conditional logic	Conditional logic is the use of an "if-then-else" construct. It requires a student to think globally about the local consequences of the truth-value of a given statement.	Wing (2006), the National Research Council (2009), and a common introductory computer science textbook (Abelson, Sussman, & Sussman, 1996) all present the conditional logic construct as the simplest construct underlying all computation. Most machine language is evaluated conditional logic and simple variable use. We are categorizing variable use as "global logic." Effectively, a student using conditional logic with variables is doing computational thought.	"...if Milan gets one more, that means Istanbul gets one, and if Istanbul had 3, that means Istanbul would start infecting ones next to it, too, and it would be like a chain reaction."
Algorithm building	An algorithm is a data "recipe" or set of instructions. Fundamentally, computer programs consist of algorithms and data. Algorithms often contain sets of related conditional logic. In its simple form, it is the planning of actions for events that are taking place; in its complex form, it is planning for unknown events.	Though there is significant debate in our source literature on the relationship between programming and computational thinking, Papert's (1980) "procedural thinking" construct is about teaching students to abstract their concepts into algorithms. The National Research Council (2009) makes several references to procedural thinking as a core concept of computational thinking.	"...I could move ... here, that's 1. And then take out 1 there, then go to Tokyo, so 3. Wait, 1, 2 ... I could move here; and then just not do anything there; and then move to Tokyo; and then fly from Tokyo to where A is; and then give him this card so the beginning of his next turn ... he can play."
Debugging	Debugging is the act of determining problems in order to fix rules that are malfunctioning.	Papert (1980) describes debugging as a core "powerful idea" of procedural thinking. Wing (2006), the National Research Council (2009), and Abelson, Sussman, and Sussman (1996) describe debugging as central to both programming and computational thinking.	Alex[2]: "...but I think that might be only during epidemics." Brad: "Do you add them back to the top during epidemics? Cause I was reading here, whenever a player draws..." Alex: "Okay, so then I'll just leave it there."
Simulation	Simulation is modeling or testing of algorithms or logic. Simulation is used in debugging in order to determine problems, and it uses algorithm building to test a model. We are defining simulation as the enactment of algorithms or plans.	Simulation or model building underlies computation in the mathematical sense. Wilensky and Reisman (2006) define computational thinking as various aspects of model building or simulation.	"...Essen, I have [the Essen card], so I could fly, I could take care of that during my turn. [I could address] that London outbreak after I take care of that. 'Cause that would take one, then I can fly to Essen, then move there. And then I can take the rest of that."
Distributed computation	Distributed computation applies to rule based actions. For instance, if 3 people act together through a rule-based plan, this is distributed computation as considerations, contingencies, and strategy formation involve multiple parties with different knowledge resources.	The National Research Council (2009) describes distributed computational thinking as one social aspect that distinguishes computational thinking from computer science.	Patrick: "Okay, for my turn first off I'm going to cure Lima... And then I'm going to move LJ. ... I'll move you here because that way you're only two away." L.J.: "You can move me to one of your cards, and then I'll teleport there." Michael: "But you can only trade the card of the one you're standing in." L.J.: "Oh, that's right." Michael: "Just because you have one, you can't turn all of them in..."