**Common Themes and Question Patterns in CS2800 Exams (2017-2024)**

**Question 1: Concept Pairs (All Years)**

Every exam contains a Question 1 asking about pairs of related software engineering concepts where you must:

* Describe each concept (typically 3-6 lines per concept)
* Explain how the concepts are connected

Common pairs that appear across multiple years:

* **Code Release and Incremental Business Value** (2018, 2019)
* **Code Smell and Refactoring** (2018, 2019)
* **Reintegration Merge and Sync Merge** (2022, 2023)
* **Literate Programming and Programming Standards** (2023, 2024)
* **Aggregation Relationship and Composition Relationship** (2022, 2024)
* **Good Code and Test Driven Development** (2022, 2024)

This question consistently accounts for 40 marks (10 marks per pair in earlier years, sometimes 8-12 marks per pair in later years).

**Question 2: Code Analysis and Testing**

Question 2 typically focuses on code analysis, metrics, and testing concepts:

1. **Control Flow Graphs**:
   * Drawing control flow graphs from provided code (2017, 2019, 2022, 2023, 2024)
   * Using flow graphs to calculate cyclomatic complexity (2019, 2023)
2. **Code Metrics and Complexity**:
   * Calculating cyclomatic complexity (2017, 2019, 2023)
   * Explaining complexity metrics (2017, 2022)
3. **Testing Concepts**:
   * Test coverage and path testing (2017, 2019, 2024)
   * Testing types like system testing, configuration testing (2022)
   * TDD principles (2019, 2022)
4. **Code Smells and Standards**:
   * Identifying and fixing code smells (2023, 2024)
   * Java programming standards (2017, 2018, 2023)
   * Checkstyle and automated checking of standards (2023)
5. **Software Tools**:
   * Eclipse IDE features (2018)
   * Maven (2022)
   * Debugging tools (2021)

**Question 3: Version Control Systems**

Question 3 consistently focuses on version control systems, with earlier years focusing on SVN and later years shifting to Git:

1. **Core Version Control Concepts**:
   * Delta and version tracking (2017, 2019, 2021)
   * Branches and merging (all years)
   * Working copies and repositories (2017, 2021)
2. **SVN-specific Topics** (2017-2022):
   * SVN commands like update, commit, merge (2019, 2021)
   * Locking and the lock-modify-unlock problem (2017)
   * Branch management and sync merges (2019, 2021)
   * Cherry pick merges (2019, 2021)
3. **Git-specific Topics** (2023-2024):
   * Git commands (checkout, branch, commit, push) (2023, 2024)
   * Git branch management (2023)
   * Git merge conflicts (2023)
4. **Software Engineering Process**:
   * Team collaboration in version control (2019, 2022)
   * Release management (2021)
   * Feature branches (2019, 2022, 2023)

**Question 4: Design Patterns and UML**

Question 4 typically focuses on design patterns and UML diagrams:

1. **UML Class Diagrams**:
   * Drawing UML diagrams from descriptions (2017, 2021, 2022, 2024)
   * Adding associations, multiplicities (2021, 2022, 2024)
2. **Design Patterns**:
   * Singleton pattern (2019, 2021, 2023)
   * Factory pattern (2021, 2022, 2023)
   * State pattern (2019, 2024)
   * Adapter and Facade patterns (2023)
   * Visitor pattern (2021)
   * Flyweight pattern (2022)
   * Observer pattern (2022)
3. **Refactoring**:
   * Using patterns to refactor problematic designs (2017, 2018, 2021)
   * Primitive obsession smell and how to fix it (2021, 2022)
   * Improving cohesion (2022)

**Observation of Question Evolution**

1. The basic structure of the exam has remained remarkably consistent from 2017 to 2024.
2. There's been a shift from SVN to Git in later years (2023-2024), reflecting industry trends.
3. Questions have become more structured and detailed in later years, with clearer mark allocations.
4. The 2024 exam introduces a puzzle game scenario in Question 4, which is a more complex system design problem than seen in some earlier years.
5. More emphasis on smells and specific refactoring techniques in recent exams (2021-2024).
6. The marks distribution across questions has become more balanced in later years.

**Key Topics That Appear Almost Every Year**

1. Software development methodologies (TDD, Agile, Waterfall)
2. Code quality (metrics, standards, smells)
3. Version control systems and branching strategies
4. Design patterns (especially Singleton, Factory, State)
5. UML class diagrams
6. Testing concepts and test coverage

This analysis should help you focus your revision on the core recurring topics while also preparing for the slight variations in how questions are asked from year to year.