

Supplementary Material A

Concept Mapping Task & Scoring Rubric

Purpose. Materials and procedures for the pre-/post-course concept mapping task used to describe changes in the structure and complexity of students' conceptual knowledge of Object-Oriented Programming (OOP).

Part 1. Concept Mapping Task — Instructions for Participants

(administered in weeks 1 and 16)

Task: Map your understanding of Java/OOP concepts.

1. What is a concept map?

A diagram of concepts and their relationships. It contains:

- **Nodes:** boxes or circles, each with one concept.
- **Links:** directed lines/arrows between two nodes.
- **Linking phrases:** short phrases on links that, together with the two nodes, form a complete, meaningful sentence (e.g., “*Class is a blueprint for an Object*”).

2. Your task.

- Place **all 15** concepts (list below) on the provided digital canvas.
- Arrange them to reflect your organization (e.g., general at top, specific below).
- Draw **directed** links to show relationships.
- **Label every link** with a linking phrase that makes the sentence true and specific.
- **Do not add new nodes** beyond the 15 provided; you may add words in linking phrases only.

3. Concept list (15).

Object; Class; Inheritance; Polymorphism; Encapsulation; Abstraction; Interface; Method; Variable; Constructor; ArrayList; LinkedList; Data Structure; Algorithm; Exception.

4. Time limit and conditions.

- **25 minutes**, individual work.

- **No external resources** (notes, internet, AI tools).
- Work in the course's digital canvas; submit as **PDF** (export) with your anonymized ID.

Part 2. Scoring Rubric (Rater Manual)

Total Score = A (Propositions) + B (Hierarchy) + C (Cross-links)

A. Propositions (link accuracy and specificity)

Score each **directed, labeled** link.

SCORE	DESCRIPTION	EXAMPLE
2	Valid & specific proposition (factually correct; precise relation).	<i>Class is a blueprint for Object; Constructor initializes Object state</i>
1	Partially valid (association is broadly true but vague/trivial).	<i>Method relates to Algorithm</i>
0	Invalid or unlabeled (incorrect relation; missing/ambiguous phrase; wrong direction).	<i>Inheritance is inside an ArrayList; unlabeled arrow</i>

Scoring procedure: sum scores over all links. **Unlabeled** or **undirected** links receive **0**. Multiple links between the same pair are permitted but must express **distinct** relationships to earn independent credit.

B. Hierarchical structure (0–5)

Assess overall vertical organization from general to specific.

SCORE	DESCRIPTION
5	Clear, multi-level hierarchy with a valid superordinate concept (e.g., <i>Data Structure / Abstraction</i>) and coherent subordinate groupings.
4	Clear hierarchy with multiple levels; minor placement errors.
3	Basic hierarchy (2–3 levels); overly simple or notable inaccuracies.
2	Weak hierarchy; mostly linear/web-like; little vertical structure.
1	Minimal hierarchy; one or two parent–child pairs without an overarching structure.
0	No discernible hierarchy; chaotic layout.

C. Cross-links (0–3 per link)

Cross-links connect **different conceptual branches** (e.g., OOP principle ↔ collection type).

SCORE	DESCRIPTION	EXAMPLE
3	Significant cross-link showing a deep, non-obvious synthesis.	<i>ArrayList enables Polymorphism by storing heterogeneous subclasses</i>
2	Valid cross-link across branches; common but correct.	<i>Method can throw an Exception</i>
1	Trivial cross-link; technically cross-branch but superficial.	<i>Class uses an Algorithm</i>
0	Invalid cross-link; connects within one branch only or is incorrect.	<i>Encapsulation is a type of Data Structure</i>

Cross-link identification rule: first, segment the map into branches (e.g., *OOP principles*, *Collections*, *Program units*). A cross-link must connect nodes from **different** branches and satisfy the proposition validity criteria.

Edge-case rules (apply consistently):

- **Synonyms** in linking phrases are fine; **node names must match** the provided list (minor variants accepted if unambiguous; e.g., “List (ArrayList)” scores for *ArrayList* when clearly indicated).
- **Direction matters.** If direction reverses meaning, score **0**.
- **Unclear/general linking phrases** (e.g., “is related to,” “works with”) usually earn **1** unless a more specific relationship is also provided elsewhere.

Part 3. Rater Training, Reliability, and Disagreement Resolution

1. **Training.** Two raters scored 5 practice maps (not in the dataset), discussed discrepancies, and refined examples in this rubric.
2. **Independent scoring.** Both raters independently scored all pre/post maps (anonymized).
3. **Reliability.**
 - For the **Total Score**, compute **two-way random-effects ICC(2,1)**.
 - For **Hierarchy** and **Cross-link** subscales (ordinal), compute **quadratic-weighted κ** (and report 95% CIs).

- For **Propositions**, optionally report per-link agreement/ κ on a stratified sample of links.
4. **Consensus.** Raters met to resolve all discrepancies; consensus scores were used in analyses.

Part 4. Reliability Statistics (Concept Maps)

- **Total Score:** two-way random-effects ICC(2,1).
- **Hierarchy and Cross-links:** quadratic-weighted Cohen's κ (with 95% CIs).
- **Propositions (optional):** per-link agreement/ κ on a stratified sample.

Table A 4.1 – Inter-rater Reliability (Concept Maps)

METRIC	STATISTIC	95% CI (LOWER, UPPER)	N MAPS	NOTES
ICC(2,1) – Total score	0.87	0.81, 0.91	82	Two-way random effects
κ (quadratic-weighted) – Hierarchy	0.78	0.70, 0.85	82	Ordinal scale 0–5
κ (quadratic-weighted) – Cross-links	0.74	0.66, 0.82	82	Ordinal weights 0–3
κ / % agreement – Propositions (opt.)	0.72 / 84%	0.63, 0.80	200	Stratified link sample ($\approx 10\%$)

Supplementary Material B

Complete Semi-Structured Interview Protocol (Dyadic)

Overview & logistics

ITEM	DETAILS
Format	Dyadic, semi-structured (two participants from the same section).
Duration	60 minutes (± 10).
Language	Mandarin Chinese (primary coding from Mandarin; English used for reporting).
Setting	Quiet seminar room; screen sharing enabled (stimulus recall).
Recording	Audio–video recorded; screen capture during stimulus recall.

ITEM	DETAILS
Materials	Consent form; Interview guide; 3 pre-selected IDE log excerpts per participant; whiteboard/markers.
Ethical note	Participation is voluntary and does not affect grades. Participants may skip questions or stop at any time.

Dyad composition. Dyads were formed within section by **comfort/ability band** to support open discussion; cross-section pairing was not used.

1. Opening (5 min)

- Welcome; explain goals: we're interested in experiences, not correctness.
- Ground rules: confidentiality; speak to each other as well as to the interviewer.
- Confirm consent (verbal).

2. Warm-up & context (5 min)

- Q1. "Briefly describe how you approached programming assignments *before* this course."
Probe: prior pair-programming or tools.
- Q2. "What did you expect the AI assistant would do for you?"
Probe: difficulty, learning, grades.

3. Core A — Interaction strategies (12 min)

(RQ1: Strategic Dance—Socratic Inquiry vs. Transactional Delegation)

- Q3. "Walk us through a typical AI-assisted coding session: how did it start; what did you ask; how did it end?"
Probe A: conceptual questions (*Socratic Inquiry*).
Probe B: direct code requests (*Transactional Delegation*).
- Q4. "How did looming deadlines influence how you talked to the AI?"
- Q5. "With unlimited time, how would you *prefer* to use the AI?"

4. Core B — Critical evaluation (12 min)

(RQ2: *verification*; *Novice's Dilemma*; *Boilerplate Blindspot*)

- Q6. “Tell me about a time the AI gave code that didn’t work or wasn’t optimal. How did you notice?”
Probes: diagnostic steps; emotions.
- Q7. “What personal rules did you develop for deciding whether to trust a suggestion?”
- Q8. “Have you accepted code without fully understanding it? What happened later?”

5. Core C — Metacognition & self-efficacy (10 min)

(RQ3: *perceived learning*; *mastery conflation*; ***attenuated metacognitive calibration***)

- Q9. “How has using the AI changed how you judge your programming ability?”
Probe: prompt craft vs. conceptual mastery.
- Q10. “Imagine tomorrow you must solve a similar assignment **without** AI. How confident do you feel, and why?”
- Q11. “Did your sense of what *counts* as learning change over the semester?”

6. Stimulus-recall segment (15 min)

Purpose. Evoke in-the-moment cognitions; triangulate logs with self-report.

Selection (pre-interview, per participant).

Three anonymized log excerpts:

1. **Conceptual inquiry** prompt (e.g., “explain the difference...”).
2. **Delegation near deadline** (within 24–36 hours of due time).
3. **Large acceptance** later linked to an error (Boilerplate Blindspot or Novice’s Dilemma).
Each excerpt shows: timestamp; user prompt/context; first ~20 lines of AI reply; anonymized commit hash; days-to-deadline.

Procedure (repeat for each excerpt).

- Prompt A (reconstruction): “What were you thinking right before you wrote this prompt?”

- Prompt B (decision-making): “What options did you consider after seeing the suggestion?”
- Prompt C (reflection): “Looking back, would you do anything differently?”
- Pair-mate follow-up: “What do you remember observing or feeling?”
Interviewer tags codes live (e.g., ILLU, DELE, TRUST).

7. Closing & member-check (6 min)

- Q12. “If you were designing next semester’s course, how would you integrate the AI to maximize learning?”
- Q13. “What did we miss?”
- Mini member-check: summarize two or three emergent points; ask for confirmation.
- Thank participants; remind them of transcript-verification opportunity; honorarium; end recording.

8. Interviewer prompts for neutrality & depth

SITUATION	NEUTRALITY PROMPT	DEPTH PROBE
Participant gives evaluative answer about AI (“It’s amazing”)	“Could you give a concrete example of that?”	“What specifically made it feel amazing in that moment?”
One participant dominates	“I’d like to pause and hear [other’s name] view as well.”	—
Vague timeline	“What week of the project was this?”	“What was the deadline pressure like at that point?”
Emotional recall (frustration)	“Take your time—can you describe what thoughts were running through your mind?”	“How did that feeling affect your next actions?”

Translation and transcription. Interviews were conducted in Mandarin by a bilingual researcher, transcribed, translated to English, and back-translated by a second bilingual researcher. Discrepancies were resolved by consensus. Coding was conducted primarily on Mandarin transcripts; English translations were used for reporting quotations.

9. Dyadic Deference/Overshadowing Audit Log (Negative Cases)

Markers used. Hedging; topic withdrawal; interruptions; dominance across turns; self-deprecation before yielding.

Dyadic Deference/Overshadowing Instances (Audit Log)

DYAD ID	ECOLOGY	MARKER OBSERVED	EVIDENCE SNIPPET (ANONYMIZED)	HOW HANDLED DURING INTERVIEW	HOW REPRESENTED IN ANALYSIS
D-07	AI-enabled	Hedging + topic withdrawal	“Maybe it’s not important... you go first.”	Interviewer paused dominant speaker; invited quieter peer	Paired memo; coded as negative case (S3)
D-12	Pair-programming	Interruption + dominance across turns	“[P1] begins explaining; [P2] interrupts 3× in ~30s.”	Turn-taking reminder; direct question to quieter peer	Both accounts retained; tension noted (S1/S3)
D-15	AI-enabled	Self-deprecation before yielding	“I don’t know this well; your approach was probably right.”	Validation + probe for concrete example	Discrepant evidence preserved (S3: TRUST)
D-19	Pair-programming	Long silence after disagreement	“(10s pause)... ‘Let’s move on.’”	Neutral prompt to revisit a specific code line	Logged as silenced case; not overwritten

Supplementary Material C

Prompt-Type Coding Rubric & Examples

Purpose. Define what counts as a prompt and how prompts were categorized into **Transactional Generation** or **Conceptual Inquiry**.

C1. Definitions and inclusion/exclusion

- **Prompt (eligible unit).**
 - (a) Any user-initiated **natural-language instruction** in the IDE chat panel, or
 - (b) an explicit **acceptance of a generated completion** that was **immediately preceded** by a typed intent comment (e.g., `// complete the file parser`).
- **Excluded.**
Pure autocompletions without explicit user instruction; background hints; linting suggestions.

C2. Categories and decision rules

1. Transactional Generation (TG)

- **Intent:** obtain code/fixes with minimal rationale.
- **Linguistic cues:** imperatives (“write/implement/fix/add”), enumerated requirements, file/class names.
- **Verification stance:** absent or superficial (“just give me...”).
- **Examples:**
 - “Write a Java method to parse CSV and handle IO exceptions.”
 - “Fix the NullPointerException in this snippet.”
- **Boundary rule:** TG can include brief constraints (“use streams”), but no request for reasoning/test cases.

2. Conceptual Inquiry (CI)

- **Intent:** understand principles, trade-offs, explanations, or tests.
- **Linguistic cues:** “why/when/what are the implications,” “explain,” “compare,” “give me two designs with pros/cons,” “write a failing test first.”
- **Verification stance:** explicit reasoning/test orientation.
- **Examples:**
 - “Explain when to prefer `final` for a class in Java.”
 - “Give two alternatives for this loop, with pros/cons and a failing test for each.”

Mixed prompts. Code request **plus** explicit explanation/tests → **CI**. Code request **without** explanation/tests → **TG**.

C3. Coder training & reliability

- **Training set:** 10% of prompts (stratified by student and week) double-coded; rules refined.
- **Independent coding:** full corpus labeled by coder A; 20% random sample double-coded by coder B.
- **Agreement:** report **percent agreement** and **Cohen's κ** (binary TG vs. CI; 95% CI).
- **Consensus:** disagreements resolved; final labels used for descriptive summaries.

C4. Verification-Intensity Ratings (0–2) for the Stratified Prompt Subset

Scale.

0 = None/Shallow; 1 = Local test/edit; 2 = Test-first/refactor.

Table C4.1 — Verification-Intensity × Prompt Type (n = 220)

PROMPT TYPE	0 = NONE/SHALLOW	1 = LOCAL TEST/EDIT	2 = TEST-FIRST/REFACTOR	TOTAL	PERCENTAGE OF SUBSET
Transactional Delegation	98	50	12	160	72.7%
Socratic Inquiry	10	28	22	60	27.3%
Total	108	78	34	220	100%

Descriptive notes.

- Most “2 = test-first/refactor” entries occurred early in assignments or when encountering novel APIs.
- A minority of Transactional prompts showed non-trivial verification (1–2), informing S3 (Evaluation) coding.

C5. Ecology-Specific Theoretical Sufficiency Checkpoints

Table C5.1 — Sufficiency Monitoring by Ecology

ECOLOGY	LAST NEW FIRST-ORDER CODE (DATE)	CODE LABEL(S) AT LAST EMERGENCE	SUBSEQUENT INTERVIEWS (N)	ROLE OF SUBSEQUENT DATA (DENSIFICATION FOCUS)		NOTES (MEMO IDS)
AI-enabled	2024-11-15	BOIL; AMC	5	S1/S2/S3/S4		M-2024-11-15-AI-BOIL-AMC
Pair-programming	2024-11-08	TRUST; SOCR	4	S1/S3/S4		M-2024-11-08-PP-TRUST-SOCR

C6. Codebook Version History and Boundary Rule Changes

Table C6.1 — Versioned Codebook Changes

VERSION TAG/DATE	TRIGGERING EVIDENCE (BRIEF)	CHANGE MADE (BOUNDARY RULE OR CATEGORY)	AFFECTED CODES	BACK-APPLICATION PERFORMED?
v0.1 / 2024-10-10	Dyad claimed “working together” + one-shot acceptance in logs	Added sub-theme Illusion of Dialogue under Partnership Framing	PART/ILLU	Yes
v0.2 / 2024-10-24	Mixed “explain-then-generate” prompts with no intervening test/edit	Rule: classify as Inquiry only if a local test/refactor occurs before acceptance; else TG	SOCR/TRAN	Yes
v0.3 / 2024-11-12	Late-semester faults traced to “trivial” glue code	Elevated Boilerplate Blindspot as a distinct evaluation pattern	BOIL	Yes
v0.4 / 2024-11-25	Prompts requesting variants + failing tests before code	Added verification-intensity axis (0–2) to stratified subset; used to interpret TG/CI heterogeneity	S3 (Eval)	N/A (auxiliary rating)

C7. Prompt Labeling Reliability (TG vs. CI)

Table C7.1 — Prompt Labeling Reliability (Stratified Subset)

METRIC	VALUE	95% CI (LOWER, UPPER)	SUBSET SIZE	NOTES
Percent agreement	87.3%	—	220	Stratified by week and student
Cohen's κ (TG vs. CI)	0.79	0.73, 0.85	220	Binary labels; disagreements resolved by consensus

Disagreement handling. All disagreements were resolved by consensus; refined rules were back-applied to the corpus.