Publishing production: study guide

Media, communications, language, and creative operations · suite apt-338-publishing-production · generated 2026-09-15T15:27:48.653Z

Detail	Value
Title	Publishing production: study guide
Generated	2026-09-15T15:27:48.653Z
Fixture/version	apt-338-publishing-production
Sector	Media, communications, language, and creative operations
Guide version	v2

> Private practice result. Not an official exam certificate, employer decision, hiring signal, admissions decision, or guaranteed outcome. Scores stay on this device unless you export them.

> Answer keys and scoring logic stay server-side and are never included in any download or export.

How to use this guide

This file contains the whole study outline for this suite: every skill it draws on, the full lesson for each of those skills, worked examples, practice tips, a glossary, and where each piece of material comes from. Nothing here is a summary of a page you still have to visit.

1. Read the skills section end to end once, without timing yourself.
2. Work the guided practice mode for the suite, using the practice tips as a checklist.
3. Move to the mini-test only when guided practice feels unhurried.
4. Sit the full simulation last, once, in the conditions you expect on the day.

Practice attempts are stored on the device you used, never on an account. Exporting a result is the only way anything leaves that device.

Practice modes and durations

Practice modes for Publishing production

Mode	Duration	What it is for
Guided practice	15 minutes	Untimed, with feedback after every item.
Mini-test	18 minutes	A short timed set for checking pace.
Full simulation	45 minutes	Full length and full time, in one sitting.

- Start guided practice: https://learn.novusstreamsolutions.com/aptitude/practice/new?bank=apt-338-publishing-production&mode=guided
- Start mini-test: https://learn.novusstreamsolutions.com/aptitude/practice/new?bank=apt-338-publishing-production&mode=mini
- Start full simulation: https://learn.novusstreamsolutions.com/aptitude/practice/new?bank=apt-338-publishing-production&mode=full

Skills covered, in full

This suite draws on 5 skill constructs. Each one below carries its complete lesson.

Planning and scheduling exercises

Resources, dependencies, deadlines, constraints, and contingency planning.

Learn to build a feasible schedule from dependencies, resource limits, deadlines, and uncertainty. A strong plan makes the critical sequence visible and includes triggers for replanning.

What you should be able to do after this lesson:

1. Convert a task list into durations, predecessors, resource needs, milestones, and constraints.
2. Identify the chain of dependent tasks most likely to control the finish date.
3. Add proportionate buffers, checkpoints, and contingencies without hiding an impossible deadline.

Worked examples and pitfalls

Worked scenario: a four-hour shutdown: Inspection takes 30 minutes before either repair can start. Two repairs take 90 and 120 minutes, but both need the same technician; testing takes 45 minutes after both finish. The minimum sequence is 30 + 90 + 120 + 45 = 285 minutes, longer than the shutdown. The correct response is to surface the conflict and change scope, staffing, or window, not to place overlapping bars on a chart and call the plan feasible.

Dependency and contingency check: Draw arrows between tasks, then mark scarce resources and approval gates. For the highest-risk dependency, define an early warning and a response: for example, if a permit is not approved by noon, defer nonessential work and notify the owner rather than discovering the conflict at start time.

How to practise this skill

- Check resource conflicts as well as task order; parallel work is impossible when both tasks need the same person or asset.
- Protect the completion milestone with visible review points, not an unexplained buffer at the end.
- When a constraint makes the plan impossible, state it early and propose the smallest viable change.

Glossary

Dependency: A rule that one task or milestone must precede or enable another.

Critical path: The dependent task sequence that determines the earliest possible finish under the stated plan.

Slack: The time a task may move without delaying a dependent milestone or final completion.

Study this next

- Scheduling: https://learn.novusstreamsolutions.com/search?q=Scheduling
- Critical path method: https://learn.novusstreamsolutions.com/search?q=Critical%20path%20method
- Resource allocation: https://learn.novusstreamsolutions.com/search?q=Resource%20allocation
- Planning and scheduling exercises lesson on Novus Learn: https://learn.novusstreamsolutions.com/aptitude/skills/planning-scheduling/lesson

Where this material comes from

- Novus Learn original scheduling, project-coordination, dispatch, and operations suite scenarios.
- Novus educational framework: tasks, durations, dependencies, resources, constraints, milestones, and contingencies.

> Educational preparation only. Novus Learn does not administer official exams and does not guarantee scores or hiring outcomes.

Attention and concentration

Sustained focus, selective attention, and accuracy under time pressure.

Attention and concentration is the skill of still noticing on minute forty of a checking block as reliably as you did on minute two, and of pointing the noticing at the right thing when two things compete. It has three distinguishable parts - selective attention, sustained attention or vigilance, and divided attention - and assessments load them differently: cancellation and checking tasks load vigilance, conflict tasks load selection, and dispatch-style monitoring loads division. It is the construct that decides whether a records clerk, a dispatcher, an air-side controller or a quality inspector catches the one wrong digit in a shift. Practice here is device-local, with no account and nothing uploaded.

What you should be able to do after this lesson:

1. Count occurrences of a target in a dense string without double-counting or losing your place, using a fixed scan path rather than free looking.
2. Separate hits, misses and false alarms in your own results and work out whether you are set too eager or too cautious for the scoring rule in force.
3. Predict where in a long block your error rate will rise, and schedule micro-resets against that curve instead of pushing through it.
4. Recognise a transposition error, the class that survives a same-characters gist check and is therefore missed most often.
5. Explain why an automatic process such as reading interferes with a controlled one, and use that to anticipate which distractors will actually cost you time.
6. Run a two-stream monitoring task by alternating on a fixed cadence rather than attempting genuine simultaneity.

Worked examples and pitfalls

A cancellation count you can actually check: Count every 7 in this row: 4 7 1 7 7 3 9 7 2 8 7 5 7 6 0 7. Working left to right and tapping once per hit: hits at the second, fourth, fifth, eighth, eleventh, thirteenth and sixteenth positions. That is seven sevens. Two error modes produce nearly all the wrong answers. The first is the adjacent pair - the 7 7 at positions four and five gets counted once, because the eye takes a repeated character as one perceptual object. The second is losing the place after the 9, where the visually similar 9 and 7 force a moment of re-checking and the scan restarts a character early, producing an over-count of eight. The defence for both is a fixed scan path with a physical anchor: a fingertip or cursor moving one character at a time, never jumping back. Re-scanning to check is the thing that creates the double-count, so if you must verify, verify by counting a second time from the RIGHT-hand end and comparing totals, never by re-reading part of the row.

Hits, misses and false alarms: the eager candidate loses: A checking batch contains 300 records, 40 of which are genuinely faulty. Candidate A flags 46 records and 36 of them are truly faulty. Hits 36, misses 40 minus 36 equals 4, false alarms 46 minus 36 equals 10. Candidate B flags 38 and 34 are truly faulty: hits 34, misses 6, false alarms 4. On hit rate alone A wins, 36 out of 40 which is 90 percent against B's 34 out of 40 which is 85 percent. Now apply the scoring rule that most checking tasks actually use, where a false alarm

cancels a hit: A scores 36 minus 10 equals 26, B scores 34 minus 4 equals 30. B wins by four despite catching two fewer faults. This is why 'flag anything that looks odd' is bad advice on a scored checking task, and why the first thing to read on a checking item is whether wrong flags are penalised. Your response criterion - how much evidence you demand before flagging - is adjustable, and it should be set from the scoring rule, not from your temperament.

The transposition that passes every gist check: Compare these two lines and decide whether they match. Invoice 4820-7391-06. Invoice 4820-7931-06. They do not: the middle group reads 7391 in the first and 7931 in the second, with the 3 and the 9 swapped. Transpositions are the most-missed error class in record checking for a structural reason - the character SET is identical, the length is identical, the first and last characters of the group are identical, so every fast check the visual system runs comes back clean. Substitutions and omissions change the character inventory and get caught; transpositions do not. Two habits raise the catch rate. Read digits in fixed groups of two rather than as a whole number, so 73-91 against 79-31 becomes a mismatch at the first group instead of a subtle difference somewhere in a four-digit blur. And check groups in a deliberately non-natural order - last group, first group, middle group - because reading left to right lets the confirmation you built at the start carry you through the middle, which is exactly where the swap is usually planted.

Conflict: why reading fights you: The classic demonstration is Stroop's, published in 1935: the word RED printed in blue ink, with the instruction to name the ink colour. Naming takes measurably longer, and errors go up, because reading a familiar word is automatic and cannot be switched off, so the automatic response has to be suppressed before the controlled one can be produced. The same conflict has a numeric version you can test on yourself in a second: how many characters are in the string 4 4 4? The answer is three, and the digit 4 pulls at you the entire way. In an assessment this appears wherever the salient feature and the asked-for feature come apart - a chart where the tallest bar is not the answer to the question, a form where the highlighted field is not the one being verified, a row where the bold total is not what the stem requested. The practical move is to name the target feature out loud before you look - 'ink colour', 'character count', 'the value for March' - because pre-loading the target biases selection before the automatic reading response gets a chance to win.

Where the errors actually appear in a 45-minute block: Errors in a long checking block are not spread evenly. Mackworth's 1948 clock-watching study established the pattern that gives the effect its name: detection declines over a prolonged watch, with the sharpest deterioration early rather than at the very end. Practically, on a 45-minute self-timed checking block, expect your per-minute error rate in minutes 20 to 45 to run visibly above minutes 1 to 20 even though nothing about the material changed, and expect the subjective sense of effort to lag the actual decline, so it will not feel like you are getting worse. Two things work against it. Break the block into three fifteen-minute segments with a ten-second reset between them - look away, unfocus, breathe out - which costs thirty seconds of a 45-minute block, about one percent of the time, and buys back more than that in caught errors. And score your practice by segment rather than as one number, because a single overall accuracy figure hides exactly the information you need: whether your problem is skill, which shows as flat error rate, or endurance, which shows as a rising one.

Two streams: alternate on a cadence, do not try to merge: A dispatch-style monitoring task: keep a running total of the numbers announced on channel one while watching channel two for the code word AMBER. Genuine simultaneity is not available - the two tasks compete for the same control resource - so the choice is not whether to alternate but whether to alternate deliberately or accidentally. Deliberate looks like this: fix the arithmetic to a rhythm, updating the total only at each announcement and holding a single number between updates, which frees the gaps for channel two. If the total is 34 and the next announcement is 7, you spend under a second reaching 41 and then you are free again. Accidental looks like re-deriving the running total from the beginning because you did not trust it, which locks up the whole window and is when AMBER goes past unheard. The measurable failure of divided attention is almost never a failure to hear the target; it is a failure to have any spare capacity at the moment it arrived. The related phenomenon worth knowing is

inattentional blindness, illustrated by Simons and Chabris in 1999: an unexpected and perfectly visible event is missed entirely when attention is committed to a counting task.

How to practise this skill

- Fix your scan path before you start, and never re-scan backwards to double-check. Backward re-scanning is what produces both double-counts and lost places; if you need to verify a count, run it again from the other end and compare.
- Read the scoring rule before the first item. Whether false alarms are penalised changes the correct strategy completely, and a criterion set for the wrong rule costs more marks than slow reading does.
- Check numeric strings in fixed groups of two or three and in a deliberately shuffled group order. Transpositions are the errors that survive whole-string comparison, and they are the ones planted on purpose.
- Time your practice in segments and log accuracy per segment, not per session. A flat error rate means you need speed; a rising one means you need endurance and breaks, and the two problems have opposite fixes.
- Name the target feature out loud before each conflict item. Pre-loading the relevant dimension is the cheapest available defence against the salient-but-wrong answer.
- Practise with an actual distractor present rather than in silence. A test hall has movement, coughing and page-turning, and attention practice in perfect quiet trains a condition you will not meet.

Glossary

Selective attention: Prioritising one source or feature while suppressing competing ones. It is what fails when the visually salient element of a display captures the response instead of the requested one.

Sustained attention (vigilance): Maintaining detection performance over a long, low-event period. It is the capacity that checking, monitoring and inspection tasks are really sampling.

Vigilance decrement: The decline in detection performance over a prolonged watch, typically steepest early in the block. Mackworth's 1948 clock test is the standard public reference.

Divided attention: Handling two streams that compete for control. Performance is best modelled as fast alternation with a switch cost, not as genuine parallelism.

Hit, miss, false alarm, correct rejection: The four outcomes of any detection decision: flagging a real fault, missing one, flagging a clean record, and correctly leaving a clean record alone. Any honest accuracy claim needs at least the first three.

Response criterion: How much evidence you require before flagging. Shifting it trades misses against false alarms without changing your underlying sensitivity, which is why it must be set from the scoring rule.

Stroop interference: The slowing that occurs when an automatic response, such as reading a word, conflicts with the requested response, such as naming its ink colour. Named for Stroop's 1935 experiments.

Inattentional blindness: Failing to notice a fully visible, unexpected event because attention is committed elsewhere - the effect Simons and Chabris demonstrated in 1999 with a counting task.

Study this next

- Attention and concentration skill hub: https://learn.novusstreamsolutions.com/aptitude/skills/attention-concentration
- Cognitive Skills Lab: attention practice: https://learn.novusstreamsolutions.com/cognitive-skills/attention-concentration
- Emergency dispatch and 911 communications suite: https://learn.novusstreamsolutions.com/aptitude/tests/emergency-dispatch-and-911-communications

- Error detection lesson: https://learn.novusstreamsolutions.com/aptitude/skills/error-detection/lesson
- Perceptual speed lesson: https://learn.novusstreamsolutions.com/aptitude/skills/perceptual-speed/lesson
- Attention and concentration lesson on Novus Learn: https://learn.novusstreamsolutions.com/aptitude/skills/attention-concentration/lesson

Where this material comes from

- Character strings, invoice comparisons, batch counts and monitoring scenarios above are written for Novus Learn. All figures are original and no published test item is reproduced.
- Terminology and directions of effect follow widely published work: Mackworth (1948) on the vigilance decrement, Stroop (1935) on response conflict, Simons and Chabris (1999) on inattentional blindness, and the standard hit/miss/false-alarm framing from signal detection theory. Concepts only; no result figures are attributed to those studies.
- Novus Learn aptitude construct registry (catalog seed) for construct scope and suite mapping.
- Public educational framing only - not affiliated with any official exam board, publisher or employer, and not a clinical, diagnostic or attention-disorder screening measure.

> Educational preparation only. Novus Learn does not administer official exams and does not guarantee scores or hiring outcomes.

Data checking and clerical accuracy

Matching, filing, coding, record checking, and identifying discrepancies.

Clerical accuracy is the ability to handle a large volume of records correctly and consistently: filing them in the right order, coding them against a key, spotting duplicates that do not look like duplicates, and holding that standard on the two-hundredth record as well as the second. Where error detection asks whether two things match, clerical accuracy asks whether you can apply a rule reliably at volume. It is assessed in administrative, records, clinical admin, school office, evidence-handling and insurance operations selection, and it is exactly the skill an employer is buying when they hire for a data-heavy back-office role. Practice stays on this device; there is no account and nothing is uploaded unless you export it.

What you should be able to do after this lesson:

1. Apply an alphabetical filing rule consistently, and state whether a given system files letter-by-letter or word-by-word. The two produce different, equally correct orders.
2. Sort alphanumeric references correctly under both string ordering and natural numeric ordering, and explain why record systems zero-pad.
3. Code records against a written key, handling every boundary condition (under, up to, and over, inclusive ranges) exactly as written rather than as intuited.
4. Identify duplicate records that differ only in formatting (case, whitespace, date order, punctuation inside a reference), and route genuinely ambiguous ones to exceptions instead of resolving them by guess.
5. Measure your own accuracy decay across a long batch and use the measurement to set a realistic attempt rate.
6. Convert a stated scoring rule into a target items-per-minute figure, and show why the optimum rate moves when the penalty multiplier changes.

Worked examples and pitfalls

Word-by-word or letter-by-letter: two correct answers, one rule: File these four names: Van Dyke, Vandenberg, Van Horn, Vance. Under word-by-word filing, each space-separated unit is compared in turn and 'nothing files before something', so the first unit 'Van' sorts ahead of both 'Vance' and 'Vandenberg'. The order is Van Dyke, Van Horn, Vance, Vandenberg. Under letter-by-letter filing, spaces are ignored entirely,

so the keys are VANCE, VANDENBERG, VANDYKE, VANHORN, and the order is Vance, Vandenberg, Van Dyke, Van Horn. Both orders are correct filing; only one is correct for the system you are working in. Test items state the convention in the instructions, and the candidates who lose marks are almost always the ones applying whichever convention their previous employer used. The same fork appears with prefixes and punctuation: whether Mc and Mac interfile, whether St is treated as Saint, whether a hyphen counts as a space. Read the rule, restate it to yourself in one sentence, then apply it mechanically and do not let a name that 'obviously' belongs somewhere override it.

Alphanumeric codes: A-102 at the front or the back of the drawer: Sort the references A-7, A-12, A-70, A-102, B-3. Under natural numeric ordering, the way a person reads them, the answer is A-7, A-12, A-70, A-102, B-3. Under plain string ordering, the way most software sorts by default, each character is compared in turn, so '1' comes before '7' and the answer is A-102, A-12, A-7, A-70, B-3, with A-102 first rather than last. Both are defensible; a filing item is testing which one the stated system uses. This is also why serious record systems zero-pad their references: rewrite the set as A-007, A-012, A-070, A-102 and the string sort and the numeric sort agree, permanently. If you are ever asked to design or clean a reference scheme, pad to a fixed width and the whole class of problem disappears. In a test, the giveaway is a set that deliberately mixes one-, two- and three-digit suffixes; that mixture exists only to separate candidates who apply the stated rule from candidates who apply the intuitive one.

Coding to a key: every mark is at a boundary: A claims routing key: under 500 pounds with no injury reported goes to CS1; under 500 with injury goes to CI2; 500 to 4,999 with no injury goes to CS3; 500 to 4,999 with injury goes to CI4; 5,000 and over goes to CE5 regardless of injury. Now code five records. R-118 at 499.99, no injury: CS1, since 499.99 is under 500. R-119 at exactly 500.00, no injury: CS3, because 'under 500' excludes 500 itself and the second band starts there. R-120 at 4,999.00 with injury: CI4, since the band is inclusive at its top. R-121 at exactly 5,000.00 with no injury: CE5, because the top band is 'and over' and its 'regardless of injury' clause overrides the injury split that governs the lower bands. R-122 at 86.40 with injury: CI2. Four of those five decisions turn on a boundary, and that is not an accident, coding items are written so that the interior cases are trivial and every discriminating mark sits on an edge. Before coding anything, underline the boundary words: under, up to, and over, between, inclusive. Then decide, once, what each one does to the endpoint, and apply that decision to every record in the batch.

Duplicates that are not textually identical: Three rows arrive in a merge. Row 1: SMITH, JANE | 07/04/1988 | AB123456C. Row 2: Smith, Jane | 1988-04-07 | AB 123456 C. Row 3: SMITH, JANE | 04/07/1988 | AB123456C. Rows 1 and 2 are the same person: normalise case, strip the spaces from the reference, and convert the ISO date and they match exactly. Row 3 is the genuinely hard one. If the file is in day/month order it is a different date of birth and possibly a different person; if that row came from a system using month/day order it is the same record again. You cannot tell from the row itself, so the correct action is to send it to the exception queue with the ambiguity noted, not to merge it and not to discard it. This is the judgement that separates competent records work from the appearance of it: the goal is not to make every row disappear, it is to make every decision defensible. In test form the item usually asks 'how many distinct individuals are represented', and the answer is often given as a range or accompanied by a 'cannot determine' option for exactly this reason.

Where accuracy actually decays on a long batch: Run a self-measurement rather than trusting a number from anyone: take 200 record pairs, split them into four blocks of 50, and log errors per block. Most people find block 1 slightly worse than block 2, a warm-up cost, and then a rise across blocks 3 and 4 as vigilance falls. The reason to measure it is that the arithmetic of small percentages is brutal at volume. Checking 200 pairs at 98 percent accuracy passes 4 bad records; at 99.5 percent it passes 1. Scale that to a realistic month of 20,000 records and the same two accuracy rates mean 400 defects against 100: a four-fold difference in downstream rework from a 1.5 point difference that would look like noise on a single test. Once you know where your own curve turns, the intervention is cheap: a deliberate twenty-second break at that point, or splitting the batch so the hardest records fall in your strongest block. Practice sessions on this site

are recorded on your own device, so building a block-by-block picture across several sessions costs nothing but the logging.

Setting an attempt rate from the scoring rule: A 120-item checking test with a 10-minute limit, scored as correct minus incorrect. Suppose practice has told you that you hold 92 percent at ten items a minute and 96 percent at seven and a half. Fast: 10 x 10 = 100 attempted, 92 correct and 8 wrong, score 84. Careful: 10 x 7.5 = 75 attempted, 72 correct and 3 wrong, score 69. Fast wins by 15. Now change the rule to correct minus three times incorrect: fast scores 92 - 24 = 68, careful scores 72 - 9 = 63, and the 15-point gap has shrunk to 5. Now suppose your real accuracy at ten a minute is 80 percent rather than 92. A gap most people do not discover until they measure it. Fast now gets 80 right and 20 wrong: under simple correct-minus-incorrect that is 60, already behind the careful strategy's 69, and under the triple penalty it is 80 - 60 = 20 against 63. Same test, same person, opposite advice, and the two deciding variables are the penalty multiplier, which the instructions hand you, and your own accuracy-at-speed, which only measurement gives you. Do not pick a pace from temperament. Measure two rates in practice, write both accuracy figures down, and do this arithmetic before the test rather than during it.

How to practise this skill

- Before the first record of any batch, write the filing or coding rule at the top of the page in your own words. The single largest source of clerical error is applying a remembered rule from a previous system.
- Underline the boundary words in a coding key (under, up to, and over, inclusive), and resolve each endpoint once. Interior cases carry almost no marks; the edges carry nearly all of them.
- Normalise before you compare: mentally strip case, spaces and punctuation from references, and restate dates in one fixed order. Half of apparent duplicates are formatting, and half of apparent non-duplicates are too.
- When a record is genuinely ambiguous, mark it and move on. Time spent resolving one unresolvable row is taken from thirty rows you could have done correctly, and a guessed merge is worse than a flagged one.
- Measure your accuracy at two different speeds in practice and write both numbers down. You cannot choose an attempt rate rationally without them, and the fast rate is almost never as accurate as it feels.
- Practise on batches long enough to hit your own fatigue point, not on ten-item samples. The construct is specifically about sustained accuracy, and a ten-item drill measures the part of the curve that was never in doubt.

Glossary

Word-by-word filing: An alphabetical convention that compares space-separated units in turn, with a shorter first unit filing before a longer one. Under it, Van Horn files before Vance.

Letter-by-letter filing: An alphabetical convention that ignores spaces and punctuation entirely, comparing the run of letters. Under it, Vance files before Van Dyke.

Natural sort: Ordering that reads embedded digit runs as numbers, so A-7 precedes A-12. Contrasted with string ordering, which compares characters one at a time and puts A-102 first.

Zero padding: Writing references to a fixed width by adding leading zeros (A-007 rather than A-7) so that string ordering and numeric ordering produce the same result. The standard structural fix for alphanumeric filing errors.

Primary and secondary sort key: The field sorted on first and the field used to break ties within it. A batch sorted by site then by surname will look wrong if the two keys are applied in the opposite order.

Canonicalisation: Converting records to a single standard form (one case, no stray whitespace, one date order, punctuation stripped from references) before any comparison, so that formatting differences stop masquerading as data differences.

Exception queue: The destination for records that cannot be resolved from the information available. Routing an ambiguous record here is a correct outcome; guessing it into a merge is not.

Boundary condition: The endpoint of a coded band, where wording such as under, up to or and over decides which side a value falls. Coding tests concentrate their discriminating items here.

Study this next

- Data checking and clerical accuracy skill hub: https://learn.novusstreamsolutions.com/aptitude/skills/clerical-accuracy
- Clerical accuracy practice in the Cognitive Skills Lab: https://learn.novusstreamsolutions.com/cognitive-skills/clerical-accuracy
- Error detection lesson: https://learn.novusstreamsolutions.com/aptitude/skills/error-detection/lesson
- Administrative and clerical entry suite: https://learn.novusstreamsolutions.com/aptitude/tests/administrative-and-clerical-entry
- Clerical checking suite: https://learn.novusstreamsolutions.com/aptitude/tests/clerical-checking
- Data checking and clerical accuracy lesson on Novus Learn: https://learn.novusstreamsolutions.com/aptitude/skills/clerical-accuracy/lesson

Where this material comes from

- All filing sets, reference codes, routing keys and records above were invented for this lesson. The two filing orders, the two sort orders and every score calculation in the attempt-rate example were worked through and checked by recomputation.
- Filing and sorting conventions cross-checked against standard public descriptions such as the Wikipedia articles 'Alphabetical order', 'Natural sort order' and 'Collation'. Terminology only; the examples are original.
- Novus Learn aptitude construct registry (catalog seed) for construct scope and suite mapping.
- Public educational framing only: not affiliated with any official exam board, publisher or employer, and no copyrighted test item is reproduced.

> Educational preparation only. Novus Learn does not administer official exams and does not guarantee scores or hiring outcomes.

Problem solving

Selecting methods, combining information, troubleshooting, and reaching practical solutions.

Problem solving is the construct that asks you to choose a method, not just execute one: to combine a table with a rule, decide whether an estimate settles the question or an exact calculation is needed, narrow a fault by halving the search space, and check the answer against the constraint that actually binds. It shows up across operations, logistics, manufacturing, technician and analyst selection, and in the situational sections of public-safety batteries. It is also the construct where the marking rewards a defensible route as much as a number. Everything you practise stays on this device unless you export it.

What you should be able to do after this lesson:

1. Combine a pricing or specification table with a written brief to reach an answer neither source contains on its own, and find the break-even point where the better option changes.
2. Narrow a fault on a chain of components by bisection, and state the worst-case number of tests before you begin.
3. Solve a working-backwards problem from a known end state, then verify by running the sequence forwards.
4. Decide whether an order-of-magnitude estimate settles a feasibility question or whether the margin is

close enough to require exact arithmetic.
5. Identify the binding constraint in a resource-allocation problem and explain why a per-unit heuristic on the non-binding resource gives the wrong answer.
6. Separate a root cause from a symptom, and describe the test that would confirm the cause rather than merely correlate with it.

Worked examples and pitfalls

Two price structures and the distance where they cross: A delivery of 55 km. Courier A charges 8.00 base plus 0.45 per km. Courier B charges 20.00 flat for the first 40 km, then 0.90 per km beyond that. Which is cheaper? Courier A costs 8 plus 0.45 times 55, which is 8 plus 24.75, giving 32.75. Courier B costs 20 plus 0.90 times 15, which is 20 plus 13.50, giving 33.50. Courier A wins by 0.75. Now the question the item is really testing: is A always cheaper? At 45 km, A costs 8 plus 20.25 which is 28.25, while B costs 20 plus 4.50 which is 24.50. B wins comfortably. So the answer flips somewhere between, and finding where is one line of algebra. Above 40 km, B costs 20 plus 0.9 times distance minus 40, which simplifies to 0.9d minus 16. Set that equal to A's 8 plus 0.45d: 8 plus 0.45d equals 0.9d minus 16, so 24 equals 0.45d, so d is about 53.3 km. Below 53.3 km B is cheaper; above it A is. Check at the crossover: A is 8 plus 24 which is 32.00, and B is 48 minus 16 which is also 32.00. The general lesson is that whichever option has the lower per-unit rate always wins eventually, regardless of the base charges, and the base charges only decide where eventually starts. A single quoted distance never answers a which-is-cheaper question for a fleet.

Halving the search space beats walking the chain: Forty sensors sit on a single daisy-chained cable and exactly one connection is broken, cutting off everything downstream. How many tests do you need in the worst case? Walking the chain from one end and testing each sensor in turn takes up to 40 tests and 20 on average. Test the midpoint instead. If sensor 20 responds, the break is in the upper half and you have eliminated 20 candidates in one measurement; if it does not, the break is below and you have eliminated the other 20. Repeat on the surviving half: 40 becomes 20, then 10, then 5, then 3, then 2, then 1. Six tests, worst case, because each test halves what is left and two to the power six is 64, comfortably more than 40. The saving grows as the problem grows: 1,000 candidates need only ten tests. Two conditions have to hold for bisection to be valid, and stating them is part of the answer. The fault must be monotone, meaning everything on one side of the break behaves differently from everything on the other; and a single test at any point must tell you which side you are on. When there are two independent breaks, or when a test is only meaningful at the ends, bisection does not apply and a different strategy is needed. Candidates who reach for bisection reflexively on a problem that fails those conditions lose more than they save.

Working backwards from the number you were given: A team started a quarter with an unknown budget. In week one it spent half of it. In week two it spent 400 of what remained. In week three it spent a third of what remained after that. It finished with 1,200. How much did it start with? Attempting this forwards means carrying an unknown through three operations. Backwards it is arithmetic. After week three, 1,200 is what is left having spent a third, so 1,200 represents two thirds of the week-three opening balance, which was therefore 1,800. Before week two's spend of 400, the balance was 1,800 plus 400, which is 2,200. That 2,200 is what remained after spending half, so the starting budget was 4,400. Verify forwards, always: 4,400 less half is 2,200; less 400 is 1,800; less a third of 1,800, which is 600, leaves 1,200. Correct. The mechanical rule is to invert each operation and apply the inversions in reverse order. The inverse of spending a third is dividing by two thirds, not multiplying by three, and that specific slip is the most common wrong answer in this family. Working backwards is the right tool whenever the end state is known exactly and the operations are individually invertible, which covers most budget, mixture and journey problems phrased as how much did it start with.

When an estimate is enough, and when it is not: A maintenance window is three hours. Two technicians must service 1,850 units, each taking about 4.5 minutes, plus a shared 20-minute setup and 15-minute teardown. Feasible? Estimate first: 1,850 units at 4.5 minutes is 8,325 technician-minutes; split between two

people that is about 4,163 minutes, which is roughly 69 hours. The window is 3 hours. The answer is no by a factor of more than twenty, and no refinement of the setup and teardown figures could change it. Precision here would be wasted effort. Now the same problem with 90 units. Ninety at 4.5 minutes is 405 technician-minutes, halved to 202.5 minutes, plus 35 minutes of setup and teardown, giving 237.5 minutes. That is 3 hours and 57 and a half minutes against a 3-hour window. Still no, but only just, and now every assumption matters: whether the technicians can genuinely work in parallel, whether setup is shared or duplicated, whether 4.5 minutes is a mean or a best case. The judgement being assessed is knowing which regime you are in. If your rough figure misses the target by an order of magnitude, stop and answer. If it lands within a factor of two, the estimate has not settled anything and you must do the exact arithmetic and name your assumptions.

The binding constraint decides, and it is rarely the obvious one: A van carries at most 900 kg and at most 6 cubic metres. Pallet type X weighs 150 kg, occupies 0.8 cubic metres and is worth 200. Pallet type Y weighs 90 kg, occupies 1.4 cubic metres and is worth 260. What mix maximises value? The instinctive heuristic is value per kilogram: X gives 200 over 150, about 1.33, while Y gives 260 over 90, about 2.89, so load Y. That is wrong, and the reason is that weight is not what runs out. Value per cubic metre tells the opposite story: X gives 250 and Y gives about 186. Enumerate the feasible whole-pallet mixes. Six X uses 900 kg and 4.8 cubic metres, worth 1,200. Five X and one Y uses 840 kg and 5.4 cubic metres, worth 1,260. Four X and two Y uses 780 kg and exactly 6.0 cubic metres, worth 1,320. Three X and three Y needs 6.6 cubic metres and does not fit. Two X and three Y uses 570 kg and 5.8 cubic metres, worth 1,180. Four Y alone uses 5.6 cubic metres and is worth 1,040. The best mix is four X and two Y at 1,320. Look at what binds it: volume is used to the last cubic metre while 120 kg of payload goes unused. Once volume is identified as the binding constraint, X's superior value per cubic metre explains the X-heavy answer, and the spare weight explains why two Y still get on board. The transferable move is to compute the usage of every constraint at your proposed answer and see which one is exhausted; a per-unit ratio computed against a non-binding resource is worse than no heuristic at all.

Cause, symptom, and the test that tells them apart: A pump trips out twice a shift. The obvious fix is to reset it, which works for about four hours. Ask why once and you find the thermal overload relay is tripping. Ask again and the motor is running hot. Again and the bearing is running hot. Again and the grease has dried out. Again and the lubrication interval was set for a duty cycle far lighter than the pump has actually been running since the line was rebalanced last year. Each level supports a different fix: resetting the trip costs nothing and lasts four hours; replacing the relay costs a part and lasts until the bearing seizes; regreasing lasts weeks; changing the lubrication schedule to match the real duty cycle is the one that stops the fault recurring. Stop asking why when the next answer stops being something you can act on, or when it leaves the boundary of the system you can change. There is one discipline that separates this from storytelling. A correlation is a lead, not a cause: the fact that the trips started after the line was rebalanced is suggestive, not proof. The confirming test is to intervene and observe: restore the original duty cycle, or apply the corrected greasing interval to this pump and not to the identical one on the parallel line, and see which one trips. If a proposed cause cannot be tested by changing it, treat it as a hypothesis and say so rather than closing the investigation.

How to practise this skill

- Write the question you are actually being asked at the top of your working, in your own words. Problem-solving items usually supply more information than the question needs, and the commonest failure is a correct calculation of something nobody asked for.
- Before any comparison of two options, ask where they cross. One quoted quantity never settles which is cheaper or faster, and the break-even is usually a single line of algebra away.
- State the worst-case number of steps before starting a search or a fault trace. If bisection applies, say so and say why; if the fault is not monotone or a test only works at the ends, say that instead and pick a different method.

- Estimate before you compute, then decide explicitly which regime you are in. Missing the target by an order of magnitude ends the question; landing within a factor of two means the estimate proved nothing and the exact numbers are now compulsory.
- Finish every allocation or capacity problem by listing how much of each constraint your answer consumes. The constraint you exhaust is the one that explains the answer, and the one you do not is where a per-unit heuristic will mislead you.
- Always verify a backwards solution by running it forwards. Attempts stay on this device, so the extra minute costs nothing and catches the inverted-fraction error that makes this family's most attractive wrong answer.

Glossary

Break-even point: The value at which two options cost or take the same. Below it one option wins and above it the other does, which is why a single quoted case never answers a comparison question.

Binding constraint: The limit that is fully consumed by the chosen solution. Identifying it explains the answer and reveals which per-unit ratios are relevant and which are noise.

Slack: The unused portion of a constraint at a proposed solution. Spare capacity in one resource is what lets a mix include items that look inefficient on the binding resource.

Bisection: Halving a search space with each test. It reduces a search of n candidates to about log base two of n tests, but only where the fault is monotone along the chain.

Working backwards: Solving from a known end state by inverting each operation in reverse order. The reliable method whenever the final value is given and every step is individually invertible.

Order-of-magnitude estimate: A rough calculation intended to settle feasibility rather than produce a figure. It is decisive when the answer is off by a factor of ten and useless when the margin is within a factor of two.

Root cause: The condition whose removal stops the fault recurring, as opposed to a symptom whose treatment suppresses it temporarily. A candidate cause that cannot be tested by changing it remains a hypothesis.

Monotone fault: A fault where everything on one side of a break behaves differently from everything on the other. The precondition that makes bisection valid, and the assumption most often left unstated.

Study this next

- Problem solving skill hub: https://learn.novusstreamsolutions.com/aptitude/skills/problem-solving
- Problem solving practice bank: https://learn.novusstreamsolutions.com/cognitive-skills/problem-solving
- Diagrammatic reasoning lesson: https://learn.novusstreamsolutions.com/aptitude/skills/diagrammatic-reasoning/lesson
- Maintenance technician suite: https://learn.novusstreamsolutions.com/aptitude/tests/maintenance-technician
- Supply chain planner suite: https://learn.novusstreamsolutions.com/aptitude/tests/supply-chain-planner
- Problem solving lesson on Novus Learn: https://learn.novusstreamsolutions.com/aptitude/skills/problem-solving/lesson

Where this material comes from

- Every figure above was computed and checked for Novus Learn: the courier break-even at about 53.3 km, the six-test bisection bound for 40 candidates, the 4,400 starting budget verified forwards, and the four-X-two-Y loading enumerated across all feasible whole-pallet mixes.
- Terminology checked against the public Wikipedia articles 'Binary search algorithm', 'Break-even', 'Shadow price' and 'Root cause analysis'. Definitions only; every scenario above is original.

- Novus Learn aptitude construct registry (catalog seed) for the construct scope and related suite mapping.
- Public educational framing only: not affiliated with any official exam board, publisher or employer, and no copyrighted test item is reproduced.

> Educational preparation only. Novus Learn does not administer official exams and does not guarantee scores or hiring outcomes.

Situational judgment

Evaluating workplace responses against role-relevant principles.

Learn a repeatable way to compare workplace responses: establish the facts, identify duties and risks, respect role boundaries, then choose a proportionate first action. This is educational preparation, not an official scoring guide.

What you should be able to do after this lesson:

1. Separate facts stated in a scenario from assumptions that the scenario does not support.
2. Rank response options by immediate risk, policy or role obligations, proportionality, and follow-through.
3. Explain why a strong first action is better than passive, punitive, or unauthorized alternatives.

Worked examples and pitfalls

Worked scenario: an unverified safety concern: A colleague reports a possible equipment fault while a deadline is approaching. First distinguish the known fact, the report, from the unverified cause. A strong response protects people and affected work, checks the concern through the right channel, tells the relevant lead, and records what was done. Ignoring the report underreacts; shutting down unrelated work or accusing someone before checking the facts overreacts.

Method: facts, duties, risks, response: Write four short notes before ranking options: what is known, who may be affected, which duty or boundary applies, and what safe next step is available now. Prefer an action that addresses the immediate issue and creates useful follow-through. Do not reward an option merely because it sounds decisive.

How to practise this skill

- Answer the question asked: best first action, worst action, or complete response are different tasks.
- Check whether an option acts within the person's authority and escalates only as far as the risk requires.
- When two options look reasonable, prefer the one that gathers missing facts and communicates ownership.

Glossary

Proportionality: Matching the urgency and scope of a response to the evidence, likely impact, and authority available.

Role boundary: The limit of what a person may decide or do without approval, specialist help, or escalation.

Follow-through: Confirming ownership, recording the decision, and checking that the issue was actually resolved.

Study this next

- Situational judgement test: https://learn.novusstreamsolutions.com/search?q=Situational%20judgement%20test
- Workplace ethics: https://learn.novusstreamsolutions.com/search?q=Workplace%20ethics

- Decision-making: https://learn.novusstreamsolutions.com/search?q=Decision-making
- Situational judgment lesson on Novus Learn: https://learn.novusstreamsolutions.com/aptitude/skills/situational-judgement/lesson

Where this material comes from

- Novus Learn original situational-judgment suite scenarios and published construct mapping.
- Novus educational framework: facts, duties, risks, role boundaries, proportional action, and follow-through.

> Educational preparation only. Novus Learn does not administer official exams and does not guarantee scores or hiring outcomes.

Recommended preparation

- Practice planning and scheduling: https://learn.novusstreamsolutions.com/aptitude/skills/planning-scheduling/lesson
- Strengthen attention and concentration: https://learn.novusstreamsolutions.com/aptitude/skills/attention-concentration/lesson
- Strengthen clerical accuracy: https://learn.novusstreamsolutions.com/aptitude/skills/clerical-accuracy/lesson

Answer keys, scoring and privacy

> Answer keys and scoring logic stay server-side and are never included in any download or export.

Formal suite answer keys and scoring logic stay on the server and are not part of any download, in any format. Downloadable keys exist only for the open, untimed practice material (the practice packs on the puzzles, cognitive-skills and reasoning practice lab pages), where the answers are already public teaching content.

Novus Learn needs no account. Your practice history lives in this browser's storage on this device, and is sent to a server only if you create an account and switch on backup. This file contains no attempt, session or result link, so it is safe to share.
