Question Bank (answers marked)

ICTGAM533 — Create complex 3-D interactive games

RTO	[RTO name]
Training package	Information and Communications Technology
Unit release	1 (current)
Delivery mode	online
Document	Question Bank · version 1.0 · review by [review date]

The complete self-marking question bank generated for this unit (119 questions), with correct answers marked and each question's mapped performance criteria / knowledge evidence shown. The same bank ships as a Moodle-importable file; this document is the human-readable record for validation and audit.

Answers are shown — assessor/validation copy. Do not distribute to learners.

E1 — Prepare to create game

Q1 · Multiple choice (multiple answers) · E1-Q1 Obtaining the production baseline

PC 1.1, KE KE3, KE KE5

A small studio is about to estimate a beta prototype for a complex 3-D interactive game. Which items should be obtained before the team treats the requirements as production-ready? Select all that apply.

- **✔ The latest authorised project brief or statement of work**
- **✔ The Game Design Document covering mechanics, rules, progression and player feedback**
- **✔ Technical specifications, including engine version, target platform and build pipeline requirements**
- **✔ Organisational procedures for version control, naming, reviews, bug reporting and security**
- A personal folder structure used successfully on a previous unrelated project
- Post-release marketing analytics from a different game with a similar genre

Feedback: Preparation begins with an authorised brief, supporting design and technical documents, and organisational procedures. These sources define scope, platform, workflow, constraints and evidence requirements.

Q2 · Select missing words · E1-Q2 Clarifying conflicting requirements PC 1.1, KE KE3, KE KE6

A producer finds that the art bible asks for cinematic realistic lighting, while the target hardware list includes low-spec school laptops. In a requirements clarification log, record the source document and section, describe the [[1]], identify the [[2]], assign the [[3]], and capture the [[4]] with date and approval evidence.

Words: **unclear or conflicting requirement** (group 1) · **production impact** (group 1) · **person responsible for answering** (group 1) · **agreed decision** (group 1) · **personal preference** (group 1) · **final marketing tagline** (group 1) · **unapproved assumption** (group 1) · **player high score** (group 1)

Feedback: A clarification log turns unclear, conflicting or assumed requirements into traceable production decisions before they become rework or schedule risk.

Q3 · Multiple choice · E1-Q3 Technical specifications shaping assets PC 1.2, KE KE2, KE KE5, KE KE6

A brief calls for a visually rich 3-D exploration game, but the beta must run reliably on older laptops. What is the best asset-planning response?

- **✓ Set budgets for LODs, textures, collision, shaders and target-hardware performance.**
- Create all art at maximum fidelity first, then decide near beta which assets should be optimised, merged, replaced or removed when performance problems become visible.
- Use only placeholder cubes for the beta because target hardware constraints mean creative asset planning and representative visual direction are unnecessary until release.
- Allow each artist to choose export settings independently so the team can compare visual styles, file sizes and import behaviours at the end of production.

Feedback: Technical specifications shape the asset list. Performance is an asset requirement, so asset planning should include budgets, LODs, collision, shader choices and representative hardware testing.

Q4 · Matching · E1-Q4 Asset register fields and their purpose PC 1.2, KE KE3, KE KE4, KE KE6

Match each asset register field with the production-control purpose it supports.

Owner or responsible role	→ Shows who is accountable for creating, integrating or approving the asset
Dependencies	→ Identifies other work that must exist first, such as a rig, shader, code system or audio trigger
Beta priority	→ Helps separate assets needed for testable gameplay from lower-priority final-game content
Acceptance criteria	→ Defines how the asset will be judged against creative and technical requirements
Required format and engine destination	→ Controls how the asset should be exported, imported and located in the engine pipeline

Feedback: An asset register connects creative and technical requirements to production control. It should show responsibility, priority, dependencies, status, formats, risks and acceptance criteria.

Q5 · Multiple choice (multiple answers) · E1-Q5 Asset integration issues to discuss

PC 1.3, KE KE2, KE KE3, KE KE5

A technical artist is preparing the first test import of character and environment assets into the game engine. Which issues should be discussed with relevant personnel before large-scale asset production continues? Select all that apply.

- **✔ Scale, units, pivot points and orientation between the 3-D tool and the engine**
- **✔ Rig hierarchy, skeleton rules, animation clips and controller requirements**
- **✔ Texture formats, material setup, shader compatibility and memory budget**
- **✔ Collision, LODs, prefab setup and in-game placement requirements**
- **✔ Audio looping, compression, loudness and whether sounds should stream or load into memory**
- Postponing all test imports until every final asset is complete
- Increasing every texture size because the engine can import high-resolution files

Feedback: Integration is where assets meet the engine and gameplay systems. Early discussion and test imports help expose scale, rig, material, collision, audio and data issues before they become expensive rework.

Q6 · True / False · E1-Q6 Engine capabilities and technical budgets

PC 1.3, KE KE2, KE KE5, KE KE6

Powerful engine features such as advanced lighting, virtualised geometry, complex shaders or high-end physics still need technical budgets because they can affect GPU time, memory, build size and target-platform performance.

Answer: **True**

Feedback: Engine capabilities are not free. A feature may be available in the engine but still unsuitable for the beta target hardware or production constraints unless budgeted and tested.

Q7 · Drag the words · E1-Q7 Game development milestone terminology

PC 1.4, KE KE3, KE KE4, KE KE6

Complete the milestone descriptions. Simple geometry used to test layout, scale and interaction is a [[1]]. A small section demonstrating intended final quality across art, audio, code and gameplay is a [[2]]. When major features are implemented but content may be incomplete and bugs remain, the build is typically [[3]]. The quality gate where features and required assets are complete or controlled exceptions are recorded, with testing focused on defects, balance, performance and polish, is [[4]].

Words: **greybox/blockout** (group 1) · **vertical slice** (group 1) · **alpha** (group 1) · **beta** (group 1) · **concept** (group 1) · **release candidate** (group 1) · **sprint backlog** (group 1) · **asset register** (group 1)

Feedback: Milestone terms help structure the sequence toward a beta prototype, but definitions should still be confirmed for the project or studio.

Q8 · Numerical · E1-Q8 Calculating critical path slip PC 1.4, PC 1.5, KE KE1, KE KE4, KE KE6

A beta prototype critical path is: rig setup 2 days, player controller scripting 4 days, combat integration 3 days, and beta smoke test 1 day. The tasks cannot overlap. Rig setup finishes 1 day late and player controller scripting takes 6 days instead of 4. Other task durations are unchanged. What is the forecast slip in the beta date, in days?

Answer: **3** ± 0

Feedback: The planned critical path is 2 + 4 + 3 + 1 = 10 days. The forecast critical path is 3 + 6 + 3 + 1 = 13 days. The beta date slips by 3 days unless scope, resources or sequencing change.

Q9 · Multiple choice · E1-Q9 Sequencing a beta prototype PC 1.4, KE KE3, KE KE4, KE KE6

A team must determine the development sequence for a beta prototype that proves a new 3-D combat loop. Which sequence best supports reliable gameplay testing while reducing downstream blockers?

- **✔ Define beta criteria, greybox the core loop, test risks early, integrate progressively and gate readiness.**
- Finish all final character and environment art first, then begin gameplay scripting once the visual style, material library and lighting direction are completely locked.
- Create a polished promotional trailer first, then use stakeholder reactions to decide whether a playable beta prototype should still be built and tested.
- Assign each discipline to work independently until the final week, then combine all systems for the first time in a single beta integration push.

Feedback: The beta sequence should be guided by what must be proved, dependencies, high-risk systems, progressive integration and readiness gates.

Q10 · Matching · E1-Q10 Scheduling test activities PC 1.5, KE KE4, KE KE5, KE KE6

Match each testing activity with the most suitable point or purpose in the beta production schedule.

Unit or component testing of scripts	→ Checks individual behaviours or code components early as they are developed
Smoke testing	→ Confirms a build launches and basic paths work before deeper testing continues
Integration testing	→ Checks that imported assets and connected systems work together in the engine
Playtesting	→ Evaluates game feel, usability, readability and whether the core experience works for players
Regression testing	→ Verifies that fixes have not broken previously working features

Feedback: Testing should be scheduled throughout production, not treated as a final clean-up activity. Different test types serve different purposes at different stages.

Q11 · Multiple choice (multiple answers) · E1-Q11 Scheduling the asset pipeline

PC 1.5, KE KE3, KE KE4, KE KE6

A production schedule currently allocates three days for “make environment props” and nothing else. Which additions would make the schedule more realistic for beta asset production and integration? Select all that apply.

- **✔ Requirement confirmation and source or GDD reference checks**
- **✔ Internal creative review and technical art review before export**
- **✔ LOD, collision, export, engine import, prefab or setup work**
- **✔ In-game QA against acceptance criteria and time for revisions**
- **✔ Separate visibility for beta-critical or high-risk assets rather than hiding them in a broad task**
- Only modelling time, because importing and testing assets can safely wait until the final beta build
- Removing dependency information so the schedule is easier to read

Feedback: Asset scheduling should cover the full pipeline: confirmation, creation, review, technical preparation, export, import, integration, QA and revision. Beta-critical and high-risk assets usually need separate visibility.

Q12 · True / False · E1-Q12 Schedule, scope and quality trade-offs

PC 1.5, KE KE4, KE KE6

If the beta date is fixed and the team has limited capacity, the safest schedule-control response is to quietly reduce acceptance criteria and remove testing time so the task list still appears on schedule.

Answer: **False**

Feedback: This is false. Production scheduling involves trade-offs between scope, time, resources and quality, but quality gates and testing should not be hidden or quietly removed. Better responses include visible scope tiers, buffers, placeholders, decision points, escalation and protected beta acceptance criteria.

Q13 · Multiple choice · E1-Q13 Defining a monitoring strategy

PC 1.6, KE KE3, KE KE4, KE KE6

A producer asks for a monitoring strategy for the beta prototype schedule. Which response is strongest?

- **✔ Track schedule, quality, risks, bugs, build health and critical path movement with agreed review and escalation rules.**
- Ask team members whether they feel on track at the end of each month, then avoid changing the schedule unless the beta date has already passed and the client has complained.
- Monitor only the percentage of tasks marked complete because build stability, asset acceptance, risk exposure and bug severity can be checked after the beta milestone.
- Use a private spreadsheet that is not shared with artists, programmers, QA or stakeholders so reporting remains simple and fewer people can question the schedule.

Feedback: Monitoring compares actual work with the schedule and triggers decisions. It should include measurable indicators, reporting rhythm, responsibility, quality checks, risk review and escalation actions.

Q14 · Select missing words · E1-Q14 Monitoring variance, risk and done criteria

PC 1.6, KE KE3, KE KE4, KE KE6

Monitoring a beta prototype should compare actual progress with planned progress. A [[1]] is the gap between plan and reality. A [[2]] records likelihood, impact, owner and response for threats. The [[3]] is the dependent task chain controlling the earliest beta date. A Definition of Done prevents false progress by requiring work to be [[4]].

Words: **schedule variance** (group 1) · **risk register** (group 1) · **critical path** (group 1) · **accepted against agreed criteria** (group 1) · **asset bible** (group 1) · **source format** (group 1) · **product trailer** (group 1) · **announced to players** (group 1)

Feedback: Effective monitoring looks at schedule variance, risks, critical path movement and quality acceptance, not just whether tasks have been ticked off.

Q15 · Short answer · E1-Q15 Identifying the risk-control artefact

PC 1.6, KE KE6

A build engineer records that an unfamiliar engine plug-in might block beta builds. The record includes likelihood, impact, owner and response actions. What production-control artefact is being used?

Accepted answer: **risk register**

Feedback: A risk register records potential events that could affect production, along with likelihood, impact, owner and response. It should be reviewed regularly and used to trigger action.

Q16 · Multiple choice · E1-Q16 Discussing data and scripting formats

PC 1.3, KE KE1, KE KE2, KE KE3

A gameplay programmer needs spawn-wave data for enemies, while designers need to tune values without editing compiled code. What is the best preparation action before asset and scripting work scales up?

- **✔ Agree and document a supported data format with import and validation rules.**
- Hard-code all enemy spawn values into gameplay scripts so designers cannot accidentally change balance while preparing beta builds, test waves or tuning passes.
- Let each designer create enemy waves in any spreadsheet or text format, then ask the programmer to manually convert and troubleshoot them at the end of production.
- Choose the format that looks most readable in a presentation because spawn-wave data is mainly a visual communication asset for stakeholders and reviewers.

Feedback: Asset format discussions include data and scripting assets, not just art files. Programmers, designers and technical leads should agree on formats that suit the engine, workflow and build process.

Q17 · True / False · E1-F1 Formative: Briefs and assumptions

If the project brief is unclear about target platforms, the safest preparation action is to assume the most powerful platform available and continue production.

Answer: **False**

Feedback: Unclear requirements should be clarified and recorded. Platform assumptions affect asset budgets, engine settings, testing and schedule risk.

Q18 · Matching · E1-F2 Formative: Common asset categories

Match each production item to its asset category.

Footstep sound loops	→ Audio asset
Enemy behaviour script	→ Code or gameplay system asset
Health bar icons	→ UI asset
NPC walk cycle	→ Animation asset

Feedback: Game production assets include art, audio, UI, code, data and testing resources, not just 3-D models.

Q19 · Multiple choice · E1-F3 Formative: Integration timing

When is the best time to perform initial test imports for a new 3-D asset pipeline?

- ✔ **Early, using representative assets before production scales up**
- Only after every final asset has been approved by the art director and the team has no remaining revision time in the schedule
- After beta testing is complete, because integration findings are most useful once all gameplay feedback has already been collected
- Only when the final release candidate is built and all platform submission checks are being completed

Feedback: Early test imports reveal scale, material, rig, collision and engine-setup issues before many assets are produced incorrectly.

Q20 · Select missing words · E1-F4 Formative: Production schedule basics

A useful production schedule shows what work will be done, [[1]], [[2]], and in what order.

Words: **when** (group 1) · **by whom** (group 1) · **which trailer music** (group 1) · **the studio logo colour** (group 1)

Feedback: A production schedule turns the development sequence into dated work with responsibilities and dependencies.

Q21 · Multiple choice (multiple answers) · E1-F5 Formative: Useful monitoring evidence

Which evidence sources can help monitor beta production progress? Select all that apply.

- ✔ **Kanban or sprint board status**
- ✔ **Bug dashboard by severity**
- ✔ **Build verification report**
- ✔ **Risk register review notes**
- A guess that the team is probably fine because no one complained

Feedback: Useful monitoring evidence includes schedule, quality, risk, build and bug information that supports decisions.

E2 — Evaluate capability of game-engine software and tools

Q1 · Multiple choice · E2-Q1 Reviewing engines beyond brand preference PC 2.1, KE KE8

A small studio is starting a complex 3-D prototype and asks you to prepare an initial engine review. Which approach best meets a professional review requirement?

- **✓ Compare current recognised engines against the game brief, platforms, licences and production risks.**
- Select the engine with the most advanced rendering showcase videos, then check later whether the team can adapt its workflow and assets.
- Review only the editor already installed on the lead designer's computer because that is the fastest way to begin prototyping.
- Choose the engine used by the largest number of developers and assume its ecosystem will solve all project-specific constraints.

Feedback: A defensible review considers a range of current, industry-recognised engines and evaluates them against the project context, not personal preference or hype.

Q2 · Multiple choice (multiple answers) · E2-Q2 Identifying the wider development toolchain PC 2.1, KE KE8, KE KE9

You are documenting the software and tools needed around the selected engine for a 3-D game team. Which items should normally be included in the toolchain review? Select all that apply.

- **✓ DCC tools for modelling, sculpting, UVs, materials and animation, such as Blender, Maya or Substance tools.**
- **✓ Audio tools or middleware for sound design and interactive audio implementation, such as FMOD Studio or Wwise.**
- **✓ Version control and collaboration tools that support project files and binary assets.**
- **✓ Build, testing and profiling tools used to package and validate the game.**
- Only the game engine editor, because all other tools can be selected after production is complete.
- A social media scheduling tool as the primary evidence of technical production capability.

Feedback: The engine is only one part of 3-D production. A sound review also covers asset creation, audio, coding, version control, profiling, build and integration tools.

Q3 · Matching · E2-Q3 Matching review criteria to evidence PC 2.1, KE KE7.1, KE KE7.2, KE KE8, KE KE9

Match each engine review criterion with the most relevant evidence to collect.

Rendering capability	→ Lighting, shadows, materials, post-processing and scalability settings demonstrated on target-style scenes
Scripting and extensibility	→ Supported languages, debugging tools, plugin system, source access and maintainability implications
Asset pipeline	→ Import and reimport tests for meshes, rigs, materials, textures, naming and folder rules
Performance tooling	→ Profilers and diagnostics that reveal CPU, GPU, memory and frame-time bottlenecks

Feedback: A comparison matrix should use evidence that directly tests the capability being claimed, such as documentation, import tests, profiling results or workflow trials.

Q4 · Select missing words · E2-Q4 Classifying hardware constraints in an engine review

PC 2.1, KE KE7.1, KE KE7.2, KE KE8, KE KE9

A target device struggles during a 3-D test build.
Expensive shadows and post-processing mainly indicate a [[1]] constraint. Too many AI updates and physics objects mainly indicate a [[2]] constraint. Large textures, meshes, animation clips and audio banks mainly indicate a [[3]] constraint.

Words: **GPU** (group 1) · **CPU** (group 1) · **memory** (group 1) · **licensing** (group 1) · **source control** (group 1) · **classification** (group 1)

Feedback: Hardware constraints must be classified accurately so the team can choose appropriate engine features, optimisation strategies and quality settings.

Q5 · Multiple choice · E2-Q5 Translating a concept into assessable requirements PC 2.2, KE KE8

A client describes a game as “a tense first-person exploration game with clues, enemy avoidance, save points and a stylised low-poly look for Windows laptops”. What is the best next step before choosing an engine?

- **✓ Convert the concept into testable play, platform, visual and production requirements.**
- Choose the engine with the highest visual fidelity because exploration games are mainly constrained by lighting and post-processing quality.
- Select the engine that the programmer prefers, then rewrite the concept and asset pipeline around that engine's strongest workflows.
- Delay platform and hardware decisions until after the first public release because early requirements may limit creative options.

Feedback: Before assessing engines, the concept should be translated into testable creative, play, platform and production requirements.

Q6 · Matching · E2-Q6 Mapping play requirements to engine systems PC 2.2, KE KE8

A 3-D mystery game has several core player actions. Match each play requirement to the engine system most directly involved in assessing engine fit.

Player scans an object from a distance	→ Raycast or trace interaction system
Enemy notices the player and gives chase	→ AI perception, navigation and behaviour system
Character changes smoothly between walk, crouch and run	→ Animation state machine or blend tree
Collected clues persist after reload	→ Inventory data and save system

Feedback: Mapping play requirements to engine systems helps identify built-in support, plugin needs, custom development effort and high-risk interactions requiring prototypes.

Q7 · Multiple choice (multiple answers) · E2-Q7 Assessing hardware fit for target devices

PC 2.2, KE KE7.1, KE KE7.2, KE KE7.3, KE KE8

A prototype looks good on a high-end desktop but must ship on mid-range laptops. Which assessment actions best address hardware constraints before final engine commitment? Select all that apply.

- **✔ Profile representative scenes on target-style laptops to identify CPU, GPU and memory bottlenecks.**
- **✔ Test scalable quality settings for shadows, resolution, post-processing, particles and texture sizes.**
- **✔ Review whether the intended visual design can be achieved within the laptop GPU and memory budget.**
- Estimate frame rate from the editor viewport on the lead developer's high-end desktop only.
- Ignore AI and physics cost because hardware constraints only affect graphics.
- Treat final build size and streaming requirements as unrelated to engine selection.

Feedback: Hardware assessment should use target-style devices, profiling and scalable quality planning rather than assumptions from a powerful development machine.

Q8 · Multiple choice · E2-Q8 Assessing asset integration risk PC 2.2, KE KE9

During engine assessment, an animated character imported from a DCC tool has incorrect scale, broken material assignments and inconsistent skeletal animation. What is the most appropriate response?

- ✔ **Test import and reimport settings, then document scale, material and rig rules.**
- Ignore the issue until final art lock because gameplay systems should be approved before the team spends time on art integration.
- Assume the problem is caused only by the artist's export choices and therefore has no bearing on engine or toolchain selection.
- Convert all characters to OBJ because a simpler static mesh format will reliably solve skeletal animation and material pipeline issues.

Feedback: Asset integration issues are a capability risk. Import and reimport tests should be performed early and workflow rules should be documented.

Q9 · True / False · E2-Q9 Prototyping uncertain play requirements PC 2.2, KE KE8

If a core interaction is technically uncertain, the team should prototype that high-risk feature before making a final engine commitment.

Answer: **True**

Feedback: High-risk play requirements should be tested early. A proof of concept can reveal whether built-in systems, plugins or custom development are needed.

Q10 · Matching · E2-Q10 Consulting personnel for engine selection PC 2.3, KE KE8

Match each required personnel role with the selection consideration they are most likely to evaluate.

Producer or project manager	→ Schedule, budget, staffing, licensing and milestone risk
Technical lead or programmer	→ Architecture, scripting language, debugging, build pipeline and maintainability
Art director or technical artist	→ Visual style, material workflow, asset import quality and animation pipeline
QA lead	→ Testing approach, reproducibility, platform builds and defect tracking

Feedback: Engine choice affects many disciplines. Required personnel should be consulted using evidence relevant to their responsibilities.

Q11 · Multiple choice (multiple answers) · E2-Q11 Discussing selection considerations with the team

PC 2.3, KE KE7.1, KE KE7.2, KE KE7.3, KE KE8, KE KE9

You are facilitating a selection meeting for a 3-D game engine. Which agenda items would help confirm whether the proposed selection can meet specified outcomes? Select all that apply.

- **✔ Creative fit for visual style, animation feel, camera language, world design and interaction feedback.**
- **✔ Technical fit for gameplay systems, performance, networking, debugging and build targets.**
- **✔ Asset pipeline fit between DCC tools, formats, import settings, audio tools and engine project structure.**
- **✔ Hardware fit for both development computers and target devices.**
- **✔ Legal and commercial fit, including licences, asset rights and distribution terms.**
- A vote on which editor interface looks most familiar, without checking requirements or evidence.

Feedback: Selection discussion should cover creative, production and technical requirements together, including asset pipeline, hardware, legal and support considerations.

Q12 · Multiple choice · E2-Q12 Confirming the selected engine will meet outcomes

PC 2.3, KE KE7.1, KE KE8, KE KE9

A team verbally agrees that Engine A is probably suitable, but the client needs assurance it will deliver a playable 3-D level on the target platform. Which confirmation evidence is strongest?

- **✔ Mapped requirements verified by a proof of concept, asset import tests and target-hardware results.**
- A signed note saying most team members like the editor interface and expect to solve technical problems during production.
- A screenshot of a high-end demo scene that uses similar lighting but does not test the specified gameplay or target platform.
- A plan to confirm performance, asset workflow and build stability only after the final milestone has been delivered to the client.

Feedback: Confirmation should be evidence-based and linked to specified outcomes, acceptance criteria and known constraints.

Q13 · Select missing words · E2-Q13 Managing disagreement and selection risk PC 2.3, KE KE8

A programmer prefers Engine A, an artist prefers Engine B and the producer is concerned about schedule risk. To compare options against agreed weighted criteria, use a [[1]]. To record likelihood, impact, mitigation and owner for unresolved issues, use a [[2]]. To record who approved the decision and the evidence used, use a [[3]]. If a major technical uncertainty remains, require a [[4]] before final approval.

Words: **weighted decision matrix** (group 1) · **risk register** (group 1) · **decision log** (group 1) · **prototype gate** (group 1) · **asset naming convention** (group 1) · **texture compression preset** (group 1) · **editor theme preference** (group 1)

Feedback: Structured decision tools help resolve conflicting priorities while keeping the discussion tied to specified outcomes and evidence.

Q14 · Multiple choice (multiple answers) · E2-Q14 Completing the final software and tool selection PC 2.4, KE KE7.2, KE KE8, KE KE9

You are writing the final selection statement for the approved 3-D game toolchain. Which details should be included so the decision can be implemented consistently? Select all that apply.

- **✔ Selected game engine and version or version range.**
- **✔ Scripting language or workflow, such as C#, C++, Blueprint or GDScript.**
- **✔ DCC, texture, animation, UI, audio, version control, build, profiling and testing tools.**
- **✔ Required plugins, packages or middleware and their licence conditions.**
- **✔ Target platforms, build method and hardware assumptions for development and testing.**
- Only the name of the engine, because supporting tools can be guessed by each discipline.

Feedback: A final selection statement should identify the engine, version, workflow, tools, target platforms, licensing, assumptions, risks and responsibilities.

Q15 · True / False · E2-Q15 Balancing final engine selection PC 2.4, KE KE7.1, KE KE7.3, KE KE8

The best final engine selection is always the option with the largest feature list, even if the team cannot use it effectively within the schedule.

Answer: **False**

Feedback: Final selection should balance creative, production and technical requirements. The lowest-risk fit for the specified game is usually better than the biggest feature list.

Q16 · Short answer · E2-Q16 Selecting an interchange format for skeletal animation

A selected toolchain must export rigged characters with skeletal animation from the DCC package into the engine. The team needs the commonly used interchange format for meshes, rigs and animation rather than a static-only format. Enter the format acronym. PC 2.4, KE KE9

Accepted answer: **FBX**

Feedback: FBX is widely used for meshes, skeletal rigs and animation, although teams still need to test export and import settings for their specific DCC and engine combination.

Q17 · Multiple choice · E2-F1 Practice: engine popularity and project fit

Unity and Unreal are commonly included in a serious 3-D engine review. What does their industry prevalence mean for a new project?

- ✔ **They should be considered, then tested against the project brief.**
- They are automatically the best choice for every 3-D game because broad use proves full suitability.
- They remove the need to review licensing, asset workflow, target hardware or team capability.
- They should only be reviewed after the game is complete and the build process has been proven.

Feedback: Popular engines should usually be reviewed, but the final choice must be justified against the specified game, team, platform and constraints.

Q18 · Drag the words · E2-F2 Practice: filling a toolchain summary

In a 3-D production pipeline, [[1]] tools create models and animations, [[2]] manages collaborative file history, and [[3]] tools help identify performance bottlenecks.

Words: **DCC** (group 1) · **version control** (group 1) · **profiling** (group 1) · **classification** (group 1) · **terrain painting** (group 1) · **box art** (group 1)

Feedback: A reliable toolchain includes content creation, collaboration and performance validation tools.

Q19 · True / False · E2-F3 Practice: documenting the decision

A final toolchain decision should be documented with rationale, constraints, evidence, workflow setup and risks so another team member can understand and reproduce the environment.

Answer: **True**

Feedback: Documentation supports consistent setup, onboarding, validation and later review if assumptions change.

E3 — Prepare required assets for a game

Q1 · Multiple choice · E3-Q1 Shared asset environment for binary assets

PC 3.1, KE KE3, KE KE6, KE KE9

A distributed 3-D game team is repeatedly overwriting hero character meshes and losing previous work. The files are large binary assets, not text code. Which shared-environment choice best addresses the production risk?

- **✔ Use a version control system that supports large files, file history and file locking.**
- Use only a shared cloud folder because artists can visually inspect the latest file.
- Ask team members to email completed FBX files to the lead artist at the end of each week.
- Allow every team member to rename and move engine assets whenever they notice a problem.

Feedback: Large binary assets such as meshes, textures and animation clips usually cannot be merged safely like source code. A version control system with file history and locking reduces overwrite conflicts and supports recovery.

Q2 · Matching · E3-Q2 Asset pipeline locations

PC 3.1, KE KE3, KE KE4, KE KE9

Match each shared-project location or record to its most appropriate purpose in a 3-D game asset pipeline.

Art/Source/Characters/Hero	→ Editable DCC source files such as Blender, Maya or texture source work
Art/Exports/Characters/Hero	→ Exported interchange files such as FBX or texture maps prepared for import
EngineProject/Content/Characters/Hero	→ Engine-native imported assets, prefabs, materials or Blueprints used by the game
Docs/AssetRegister	→ Status, owner, dependencies, licence and review information for assets

Feedback: A reliable pipeline separates editable source work, exported game-ready files, engine-native assets and tracking documentation so assets can be found, reviewed and rebuilt.

Q3 · Multiple choice (multiple answers) · E3-Q3 Controls before moving assets to approved status

PC 3.1, KE KE4, KE KE6, KE KE8, KE KE9

A team is about to allow several environment props to move from work-in-progress to approved production content. Which controls should be in place? Select all that apply.

- **✔ An asset owner or responsible reviewer is identified for the prop group.**
- **✔ The asset has passed an agreed review or integration test before approval.**
- **✔ Version history or changelog information is retained for audit and recovery.**
- Every team member has unrestricted write access to all approved engine content.
- Binary files are edited without locking because the final approved version will overwrite earlier work.

Feedback: Approval should be controlled by ownership, review, access rules, version history and testing. These controls reduce schedule, quality, copyright and integration risks.

Q4 · Select missing words · E3-Q4 Translating design intent into asset attributes

PC 3.2, KE KE3, KE KE8, KE KE9

A gameplay brief says a market-stall prop is interactive, appears many times in a crowded level, and must be readable from a close third-person camera. The asset review should convert this into measurable [[1]], check likely [[2]] such as scripts or navigation, and record any required [[3]] or optimisation settings before production begins.

Words: **asset attributes** (group 1) · **dependencies** (group 1) · **LODs** (group 1) · **personal preferences** (group 1) · **email attachments** (group 1) · **untracked edits** (group 1)

Feedback: Asset attributes translate gameplay and visual intent into measurable production requirements, including dependencies and optimisation needs.

Q5 · Multiple choice · E3-Q5 Choosing an asset format for animation data

PC 3.2, KE KE2, KE KE5, KE KE7.1, KE KE8, KE KE9

A character asset must transfer mesh data, skeleton data and animation clips from a digital content creation tool into the selected game engine. Which format choice is most suitable to review first for this requirement?

- **✔ FBX**
- OBJ
- PNG
- A spreadsheet asset register only

Feedback: FBX is commonly used across game pipelines for meshes, skeletons, animations, LODs and morph targets. OBJ is generally unsuitable for complex animation data.

Q6 · True / False · E3-Q6 In-engine review of assets

PC 3.2, KE KE2, KE KE4, KE KE8, KE KE9

An environment prop can be considered production-ready if it looks correct in Blender or Maya, even if it has not yet been checked in the game engine.

Answer: **False**

Feedback: Assets must be reviewed in the game engine under project conditions. DCC previews do not prove that scale, materials, collision, references, performance or build behaviour are correct.

Q7 · Multiple choice (multiple answers) · E3-Q7 Strategic acquisition of marketplace assets

PC 3.3, KE KE3, KE KE4, KE KE6, KE KE8, KE KE9

A producer wants to buy a marketplace vehicle model to save time. Which checks should be completed before the asset is approved for use? Select all that apply.

- **✔ Confirm the licence allows the intended release and record attribution or restrictions.**
- **✔ Check whether the model, textures, rigging, LODs and collision meet the asset brief.**
- **✔ Budget time for adaptation, optimisation and in-engine testing.**
- Approve it immediately if the preview render looks higher quality than the current prototype.
- Delete original licence and receipt files after import to reduce repository size.

Feedback: Acquired assets must be checked against the asset brief, licence, technical requirements, style, formats and integration cost. Purchase alone does not make an asset suitable.

Q8 · Drag the words · E3-Q8 Safe intake process for acquired assets

PC 3.3, KE KE2, KE KE3, KE KE6, KE KE9

When a contractor delivers a prop package, the team should treat it as [[1]] until checked, store the original package in a controlled [[2]], and import a renamed copy into a [[3]] before approving it for production.

Words: **unverified** (group 1) · **archive** (group 1) · **sandbox** (group 1) · **approved** (group 1) · **shader graph** (group 1) · **live production level** (group 1)

Feedback: Acquired assets should be quarantined, inspected, preserved in original form and tested outside the main production content before approval.

Q9 · Multiple choice · E3-Q9 Blockout before final detail

PC 3.3, KE KE2, KE KE3, KE KE4, KE KE8, KE KE9

A level team needs a modular doorway that affects player movement and camera readability. What is the best reason to create a blockout or proxy before final modelling?

- **✓ To test scale, gameplay space and camera readability in engine before committing to final detail.**
- To avoid recording asset requirements because the final mesh will define them automatically.
- To replace the need for later import, material and collision testing.
- To ensure source files and engine files are stored in the same folder.

Feedback: Blockouts allow teams to test scale, gameplay space, camera readability and layout before investing time in final art detail.

Q10 · Matching · E3-Q10 Complete asset package contents

PC 3.3, KE KE1, KE KE3, KE KE4, KE KE8, KE KE9

Match each component of an asset handoff package to the integration problem it helps solve.

Editable source files	→ Allow the team to revise or rebuild the asset without starting again
Exported mesh and texture maps	→ Provide game-ready files for import into the engine pipeline
Collision meshes or collision instructions	→ Explain how the asset should block, overlap or respond in gameplay
Licence and attribution records	→ Support legal release and future audit of acquired content
Review notes and known limitations	→ Help another team member understand unresolved issues or constraints

Feedback: Complete asset packages make integration repeatable. They include source files, exports, textures, materials, collision or LOD information, engine setup and review records as required.

Q11 · Short answer · E3-Q11 Version control feature for binary asset conflicts

PC 3.1, KE KE3, KE KE6, KE KE9

Two artists need to work with the same binary animation asset, and the team wants to prevent unsafe simultaneous edits rather than attempt a text-style merge. What version control feature should be used?

Accepted answer: **file locking**

Feedback: File locking prevents or controls simultaneous edits to binary assets that cannot be merged safely.

Q12 · Numerical · E3-Q12 Texture memory optimisation calculation

PC 3.4, KE KE2, KE KE5, KE KE7.1, KE KE8, KE KE9

A representative scene uses four uncompressed RGBA textures at 2048 × 2048 pixels. Each pixel uses 4 bytes. If each of those textures is replaced with a 1024 × 1024 version using the same format, how many MiB of texture memory are saved in total? Enter the number only.

Answer: **48** ± 0.5

Feedback: A 2048 × 2048 RGBA texture uses 2048 × 2048 × 4 bytes = 16 MiB. A 1024 × 1024 RGBA texture uses 4 MiB. The saving is 12 MiB per texture, so four textures save 48 MiB.

Q13 · Multiple choice · E3-Q13 Pivot customisation for gameplay behaviour

PC 3.4, KE KE2, KE KE5, KE KE7.3, KE KE8, KE KE9

A door mesh imports correctly visually, but in the engine it rotates around its centre instead of swinging on its hinges. Which asset attribute should be customised first?

- ✔ **Pivot location**
- Texture compression format
- Marketplace attribution text
- Number of project folders

Feedback: Pivot placement affects how objects rotate, snap and are placed. A door needs a pivot suitable for hinge rotation.

Q14 · True / False · E3-Q14 Collision versus visual mesh

PC 3.4, KE KE1, KE KE2, KE KE5, KE KE7.2, KE KE8, KE KE9

For gameplay props, collision should be designed for gameplay purpose and performance, so a simplified collision shape may be more appropriate than matching the visible mesh exactly.

Answer: **True**

Feedback: Collision should match gameplay purpose, not necessarily visual detail. Simple collision is often better for performance and predictable interaction.

Q15 · Select missing words · E3-Q15 Customising assets for engine readiness

PC 3.4, KE KE2, KE KE7.3, KE KE8, KE KE9

A modular wall kit has small gaps when snapped in the level and some pieces face the wrong direction after import. The asset customisation pass should confirm [[1]], correct [[2]], and apply project material or shader [[3]] before validation.

Words: **scale** (group 1) · **orientation** (group 1) · **presets** (group 1) · **sales ranking** (group 1) · **commit humour** (group 1) · **email layout** (group 1)

Feedback: Foundational asset customisation includes scale, orientation, pivot, transforms and material conversion to project shaders or presets.

Q16 · Multiple choice (multiple answers) · E3-Q16 Validating customised assets under project conditions PC 3.4, KE KE1, KE KE2, KE KE4, KE KE5, KE KE6, KE KE7.1, KE KE7.2, KE KE7.3, KE KE8, KE KE9

A customised interactive crate looks correct in the editor viewport. Which validation actions provide stronger evidence that it is ready for approval? Select all that apply.

- **✔ Place it in a representative gameplay scene with the intended player controller and camera.**
- **✔ Test collision, physics, navigation and interaction scripts in the relevant gameplay scenario.**
- **✔ Check engine warnings, missing references, performance and build behaviour on the target platform where relevant.**
- Approve it immediately because editor previews always match packaged builds.
- Avoid updating documentation after attribute changes so the asset register remains simple.

Feedback: Validation should occur in representative scenes and, where relevant, builds. It should check lighting, scale, materials, LODs, collision, physics, navigation, scripts, warnings and performance.

Q17 · Multiple choice · E3-F1 Source versus game-ready files

Which practice best supports a repeatable asset pipeline?

- **✔ Separate source files, exported files and engine-native assets.**
- Store every version of every file in one shared folder.
- Import acquired packages directly into the live production level.

Feedback: Keeping editable source files separate from exported and engine-native files helps the team rebuild, review and troubleshoot assets.

Q18 · True / False · E3-F2 Marketplace assets and integration effort

Buying a marketplace asset can still require optimisation, renaming, licensing checks and in-engine testing before it is production-ready.

Answer: **True**

Feedback: Acquisition can save time, but only when licensing, style, format, performance and integration risks are managed.

Q19 · Matching · E3-F3 Common customisation concerns

Match the asset issue to the customisation area most likely involved.

Object is too large in the level	→ Scale and unit settings
Door rotates from the middle	→ Pivot placement
Player collides with invisible detail too often	→ Collision setup

Feedback: Customisation turns created or acquired assets into engine-ready production assets by adjusting technical and gameplay attributes.

E4 — Use game-engine software and development tools

Q1 · Multiple choice · E4-Q1 Pressure plate and gate script structure PC 4.1, KE KE1, KE KE2, KE KE3

A prototype requirement says: "A pressure plate opens a gate while the player stands on it, then closes the gate when the player leaves." Which implementation best supports reliable testing and later adjustment?

- **✔ Use a plate trigger that calls Open() and Close() on a separate gate script.**
- Put movement, player detection, sound, animation, UI updates, scoring, level transition logic and every other level rule into one large scene controller script.
- Animate the gate open in the editor timeline only and rely on the player to step off the plate at the correct time during the review.
- Duplicate the gate mesh into open and closed positions and manually enable the right version during the presentation whenever the player steps on the plate.

Feedback: A maintainable engine workflow translates the requirement into small behaviours. The plate should detect trigger enter and exit events and call clear functions on the gate, while the gate manages its own movement, animation and feedback.

Q2 · Multiple choice (multiple answers) · E4-Q2 Safe script modification practices

PC 4.1, KE KE1, KE KE7.2, KE KE6

A teammate asks you to modify an existing player pickup script before a milestone build. Which practices should you apply? Select all that apply.

- **✔ Identify the required logic before editing, including the trigger condition, score change, feedback and reset behaviour.**
- **✔ Expose designer-tunable values such as score amount, rotation speed or audio clip reference instead of hard-coding everything.**
- **✔ Use engine events and components, such as collision or trigger callbacks, where they match the behaviour being implemented.**
- **✔ Use console output, breakpoints or the engine debugger to confirm the script runs under the intended conditions.**
- Edit production files directly without checking version control because pickup scripts are usually small.
- Merge the pickup, player movement, UI, camera and audio systems into one script so there is only one file to open.

Feedback: Safe script modification means understanding the required behaviour, using appropriate programming structures and engine APIs, exposing values that need tuning, and testing changes incrementally before they affect the presentation build.

Q3 · Matching · E4-Q3 Combining assets into a usable object PC 4.1, KE KE3, KE KE5, KE KE9

Match each asset or component choice with its main purpose when turning an imported crate model into a reusable in-game object.

Collider	→ Defines the physical or trigger shape used for collision and interaction
Rigidbody	→ Allows the object to participate in physics movement when required
Material and texture maps	→ Control how the surface appears in the engine
Prefab or Blueprint	→ Stores the configured object for consistent reuse across scenes
Audio clip reference	→ Provides sound feedback such as an impact, pickup or break effect

Feedback: A model file becomes a working game object only after import settings, materials, collision, physics, scripts, audio and reusable object configuration are integrated correctly.

Q4 · Select missing words · E4-Q4 Asset integration checks PC 4.1, KE KE9, KE KE5

Before reusing an imported 3-D asset in gameplay, check its [[1]], connect its [[2]], add an appropriate [[3]], and store the configured object as a [[4]] or equivalent reusable engine object.

Words: **scale and orientation** (group 1) · **materials and texture maps** (group 1) · **collider** (group 1) · **prefab** (group 1) · **unused source folder** (group 1) · **presentation script** (group 1) · **debug console filter** (group 1) · **milestone date** (group 1)

Feedback: Import and integration checks should confirm scale, orientation, materials, collisions and reusable configuration. These checks prevent asset problems from becoming gameplay or presentation defects.

Q5 · True / False · E4-Q5 Reusable objects and validation PC 4.1, KE KE2, KE KE3, KE KE6

After a configured object is converted into a prefab, Blueprint or equivalent reusable object, its instances no longer need validation after integration into scenes.

Answer: **False**

Feedback: Reusable systems support consistent updates and reduce repeated setup, but they do not remove the need to test each integration context, especially after changes to scripts, references or project settings.

Q6 · Multiple choice · E4-Q6 Minimum playable loop PC 4.2, KE KE2, KE KE8

A designer says a locked door mechanic "works" because the door mesh is in the level. Which description best represents a playable gameplay loop for that element?

- **✓ Approach, prompt, input, lock check, then open or locked feedback.**
- The door model is placed near the player start position and has a high-resolution material, but no input handling, state change, prompt, sound, animation or objective feedback is implemented.
- The level designer tells the reviewer verbally that the door would open in the final game if the full interaction system had already been completed.
- The door opens only when the developer manually changes a variable in the inspector during the presentation instead of responding to player input.

Feedback: A gameplay element is not just an object in a scene. It needs player action, game response, feedback and state change that can be tested against creative, technical and production requirements.

Q7 · Drag the words · E4-Q7 Engine systems used in gameplay elements PC 4.2, KE KE5, KE KE8

A 3-D interaction system may use the [[1]] to read keyboard, mouse, controller or VR controls; [[2]] to detect triggers and impacts; the [[3]] to play character or object motion; and [[4]] to display prompts, health, score or objective feedback.

Words: **input system** (group 1) · **physics and collision system** (group 1) · **animation system** (group 1) · **UI system** (group 1) · **source control branch** (group 1) · **asset licence register** (group 1) · **milestone calendar** (group 1) · **external concept board** (group 1)

Feedback: Gameplay elements are usually combinations of engine systems, including input, physics and collision, animation, UI, audio, scripting and scene management.

Q8 · Multiple choice (multiple answers) · E4-Q8 Responding to target hardware constraints PC 4.2, KE KE2, KE KE7.1, KE KE7.2, KE KE7.3, KE KE6

A prototype runs smoothly on a gaming desktop but stutters on the target classroom laptops. Which responses are appropriate before confirming the gameplay element? Select all that apply.

- **✔ Profile frame rate, memory use, input responsiveness and key stutter points on the target laptops.**
- **✔ Optimise expensive scripts, repeated per-frame work or unnecessary runtime allocations.**
- **✔ Reduce asset cost where appropriate, such as texture size, shader complexity, object count or level-of-detail settings.**
- **✔ Adjust lighting, camera effects or visual density if needed while preserving the intended mood and readability.**
- Increase post-processing and particle effects so the presentation looks more impressive despite the stutter.
- Assume the packaged build will perform like the desktop editor and continue without further testing.

Feedback: Technical constraints should be checked on target or representative hardware. Graphics, code and visual design decisions may all need adjustment while preserving the core creative requirement.

Q9 · Short answer · E4-Q9 Naming a clear gameplay check PC 4.2, KE KE3, KE KE4, KE KE6

During QA for a locked door, you record: "Condition: player without key interacts with door. Expected result: door remains closed, locked sound plays, locked message appears." What is this kind of specific condition and expected result called?

Accepted answer: **test case**

Feedback: A test case defines a specific condition and the expected result. Test cases help check gameplay against requirements rather than relying on whether something seemed to work once.

Q10 · Matching · E4-Q10 Quality lenses for gameplay checking PC 4.2, KE KE4, KE KE8

Match each quality lens with the question it is most likely to answer when checking a gameplay element.

Functional check	→ Does the mechanic perform the required rule correctly?
Usability check	→ Can the player understand what to do and read the feedback?
Experience check	→ Does it support the intended mood, pacing and challenge?
Technical check	→ Does it run reliably within the target platform and engine constraints?

Feedback: Checking gameplay should consider function, usability, experience and technical performance. A mechanic can run without errors but still fail if it is unclear, off-brief or unreliable on the target platform.

Q11 · Multiple choice · E4-Q11 Editor testing and packaged build validation PC 4.3, KE KE2, KE KE4, KE KE5, KE KE8

A prototype sequence works in the engine editor. The assessor will review a packaged build. What is the best next testing decision?

- ✔ **Run the packaged build end to end.**
- Skip build testing because editor play mode uses the same gameplay scripts, scene objects and designer-facing values, so the presentation build should behave identically on every machine.
- Only inspect the scene hierarchy because runtime behaviour has already been proven in the editor and the build process cannot change input, shader, path or scene-list behaviour.
- Record the editor window with debug tools visible so any build issue can be explained during review instead of validating the actual presentation form.

Feedback: Editor play mode is useful for fast iteration, but a packaged build may differ in scene inclusion, file paths, input handling, shader compilation, permissions, quality settings and performance.

Q12 · Select missing words · E4-Q12 Confirming a prototype presentation PC 4.3, KE KE3, KE KE4, KE KE6, KE KE8

For a stakeholder review, define the intended [[1]] of events, confirm the [[2]] fit of mood and player goal, confirm the [[3]] fit of milestone scope and known limitations, and confirm the [[4]] fit of platform, settings and performance.

Words: **sequence** (group 1) · **creative** (group 1) · **production** (group 1) · **technical** (group 1) · **licensing** (group 1) · **marketing** (group 1) · **tutorial** (group 1) · **monetisation** (group 1)

Feedback: A presentation run should show a controlled sequence and be checked against creative, production and technical requirements. Known limitations should be documented in stakeholder-ready language.

Q13 · Multiple choice (multiple answers) · E4-Q13 Final readiness before presentation

PC 4.3, KE KE4, KE KE6

A live prototype review is scheduled for tomorrow. Which actions support final readiness and risk control? Select all that apply.

- **✔ Clear known compile errors and investigate runtime errors shown in logs.**
- **✔ Run the intended sequence from start to finish in the presentation form.**
- **✔ Triage defects by impact on the required sequence and document known limitations.**
- **✔ Freeze the build when it is ready and archive evidence of testing.**
- Add several new mechanics after the final test so the prototype appears broader.
- Use only the developer's best editor run and avoid documenting issues that remain.

Feedback: Presentation readiness depends on controlled testing, defect triage, build freeze, documented limitations and fallback options. Late uncontrolled changes increase risk to the critical path.

Q14 · True / False · E4-Q14 Creative failure despite technical success

PC 4.2, PC 4.3, KE KE8, KE KE7.3

A prototype sequence can run without technical errors and still fail the brief if the player goal is unclear, feedback is confusing, or the mood and pacing do not match the intended experience.

Answer: **True**

Feedback: Technical success is only one requirement. A prototype must also communicate the intended creative experience and support the agreed production purpose.

Q15 · Numerical · E4-Q15 Frame budget calculation for target performance

PC 4.2, PC 4.3, KE KE2, KE KE7.1, KE KE7.2

You are checking a prototype combat sequence for a target of 60 frames per second on the review hardware. To estimate whether code and rendering work fit the target, calculate the approximate maximum time budget per frame in milliseconds. Enter the value to two decimal places.

Answer: **16.67** ± 0.05

Feedback: Frame time is calculated as 1000 milliseconds divided by frames per second. For 60 fps, 1000 ÷ 60 = approximately 16.67 ms per frame. This helps evaluate graphics and code constraints on target hardware.

Q16 · Multiple choice · E4-Q16 Build issue on the review machine

PC 4.3, KE KE5, KE KE6, KE KE7.1, KE KE8, KE KE9

A packaged prototype runs on the developer machine, but on the review machine some textures are missing and controller input does not respond. Which response best aligns with technical readiness for presentation?

- **✓ Check build settings, asset inclusion and input configuration, then rebuild and retest.**
- Tell the reviewer to imagine the missing textures and use keyboard input instead because the main logic is present and the prototype worked on the developer machine.
- Delete the missing materials and controller support so the build no longer reports those problems, even if the prototype no longer meets the agreed presentation requirements.
- Run the editor version on the developer machine only and mark the packaged build as ready because a successful editor test is enough for stakeholder review.

Feedback: Technical readiness requires validating the target platform and presentation environment, including asset inclusion, asset formats, build settings, input devices, scene list, quality settings and documented limitations.

Q17 · True / False · E4-F1 Practice: Script role

A static 3-D model becomes an interactive game object only when it is combined with the necessary engine configuration, such as components, scripts, collisions, materials and references.

Answer: **True**

Feedback: Assets need to be integrated and configured before they become working gameplay objects.

Q18 · Multiple choice · E4-F2 Practice: Best first step for a new mechanic

Before coding a new pickup mechanic, what is the best first step?

- **✓ Break the requirement into clear behaviour logic, expected feedback and testable outcomes.**
- Import as many pickup meshes as possible before defining interaction rules.
- Add the mechanic directly to the final build without testing in the engine.
- Use the highest-resolution texture available so the pickup is easy to see.

Feedback: Planning the behaviour first makes implementation and testing clearer.

Q19 · Matching · E4-F3 Practice: Requirement types

Match each requirement type with its focus.

Creative requirement	→ Intended experience, mood, rules, challenge and player fantasy
Production requirement	→ Milestone scope, available time, team capacity and deliverable format
Technical requirement	→ Engine, platform, hardware, performance, input and build constraints

Feedback: Creative, production and technical requirements overlap, but each has a different focus when checking a prototype.

E5 — Evaluate game prototype

Q1 · Multiple choice · E5-Q1 Identify required personnel for a prototype demonstration

PC 5.1, KE KE3, KE KE4, KE KE8

A small studio is preparing to demonstrate a 3-D game prototype before committing to full production. Which approach best meets the requirement to show the prototype to the right people and seek useful feedback?

- **✓ Use authorised decision-makers and relevant specialists, then record each source.**
- Show the build only to friends who have not seen it before so their reactions are spontaneous and informal.
- Ask only the programmer to approve the prototype because a running build proves the concept is ready.
- Delay all feedback until final art, audio and user interface have been completed for the entire product.

Feedback: Required personnel are the decision-makers and specialists identified by the project brief, workplace process or production lead. Their roles should be linked to the kind of feedback they can provide.

Q2 · Select missing words · E5-Q2 Prepare a demonstrable prototype build

PC 5.1, KE KE1, KE KE2, KE KE5, KE KE7.2, KE KE9

Before a formal prototype review, the developer should verify the [[1]] so the project opens without missing packages or broken references, document the [[2]] so reviewers can operate the prototype, select a [[3]] that shows the core loop and intended scale, and list [[4]] so reviewers do not mistake prototype limitations for final design decisions.

Words: **launch path** (group 1) · **controls** (group 1) · **representative scene** (group 1) · **known issues** (group 1) · **final marketing trailer** (group 1) · **unrelated bonus mode** (group 1) · **publisher logo animation** (group 1) · **post-release downloadable content list** (group 1)

Feedback: A demonstration build should be reliable enough to support decision-making. It should show the intended gameplay loop, state what is prototype or final, and reduce avoidable technical risk before the session.

Q3 · Multiple choice (multiple answers) · E5-Q3 Convert prototype comments into usable feedback

PC 5.1, KE KE3, KE KE6, KE KE8, KE KE9

During a review, a lead designer says: “The camera feels bad in the corridor fight.” Which details should be added to turn this into usable production feedback? Select all that apply.

- **✔ The build version and where in the prototype the issue occurred.**
- **✔ The observed behaviour, such as clipping through walls or losing sight of enemies.**
- **✔ The affected system, severity, suggested action, owner and status.**
- The reviewer’s full personal contact details published in the public issue log.
- A decision to approve final art style because the comment concerns camera feel.
- A task called “fix game feel” with no evidence, priority or acceptance criteria.

Feedback: Useful feedback is traceable and actionable. It should include evidence, source, affected system, severity, suggested action, ownership and status where possible.

Q4 · Matching · E5-Q4 Match prototype evaluation criteria to evidence

PC 5.2, KE KE2, KE KE3, KE KE5, KE KE8, KE KE9

Match each evaluation criterion with the most relevant evidence for judging a complex 3-D game prototype.

Core concept	→ The main interaction loop is understandable and engaging enough to justify further development.
User-friendliness	→ Intended players can understand goals, controls, feedback and recovery with reasonable effort.
Technical feasibility	→ The engine, code structure and target hardware can support the required systems and performance.
Asset integration	→ Models, textures, animations, audio and scene references import and behave reliably in the build.

Feedback: Prototype evaluation should compare evidence against agreed criteria, including creative promise, usability, technical feasibility and asset integration.

Q5 · Multiple choice · E5-Q5 Evaluate creativity in a prototype

PC 5.2, KE KE3, KE KE8

A prototype uses placeholder art but combines gliding, wind simulation and puzzle navigation into a clear risk-reward mechanic. How should creativity be evaluated in this situation?

- **✔ Judge novelty, coherence, player value and feasibility.**
- Reject the creative value because placeholder assets mean the game cannot yet be judged in any reliable way.
- Approve the prototype solely because wind simulation is unusual, even if players cannot understand the goal.
- Evaluate only the concept art, because mechanics are technical features rather than creative design choices.

Feedback: Creativity in a prototype is judged by novelty or freshness, coherence, player experience and feasibility, not by final polish alone.

Q6 · Multiple choice (multiple answers) · E5-Q6 Evaluate user-friendliness for intended users and hardware

PC 5.2, KE KE5, KE KE7.1, KE KE7.3, KE KE8

A prototype is intended for short VR training sessions. Trial users can complete combat tasks, but several report discomfort, miss prompts near the edge of view and struggle with controller inputs. Which evaluation considerations are relevant? Select all that apply.

- **✔ Whether the controls are learnable and appropriate for the target VR input device.**
- **✔ Whether prompts, feedback and objectives are readable in the headset field of view.**
- **✔ Whether comfort, motion and recovery from mistakes are acceptable for the intended session length.**
- **✔ Whether accessibility needs such as text size, readability and input effort have been considered.**
- Whether every final high-poly environment asset has already replaced greybox geometry.
- Whether the game is easy enough that no participant can ever fail a challenge.

Feedback: User-friendly means playable by the intended users on the intended hardware. Goals, controls, feedback, recovery, accessibility and comfort all matter.

Q7 · Numerical · E5-Q7 Calculate a user trial issue rate

PC 5.5, KE KE4, KE KE8

In a user trial, 4 out of 7 participants missed the first interaction prompt and required a hint. To one decimal place, what percentage of participants required a hint?

Answer: **57.1** ± 0.1

Feedback: Quantitative measures such as completion rates, hints, deaths and task time help evaluate trial feedback and prioritise recommendations. The calculation is 4 ÷ 7 × 100 = 57.1%.

Q8 · True / False · E5-Q8 Evaluation versus description

PC 5.2, PC 5.5, KE KE2, KE KE4, KE KE5, KE KE6

If review notes say “three players found the camera difficult,” a competent evaluation should also judge what that evidence means against agreed criteria and decide whether to proceed, change, retest, defer or stop.

Answer: **True**

Feedback: Evaluation is more than description. It explains the significance of evidence against criteria and identifies the production implication.

Q9 · Matching · E5-Q9 Match test types to prototype testing purposes

PC 5.4, KE KE3, KE KE4, KE KE5, KE KE7.2, KE KE9

Match each test type with its purpose when assisting with tests and user trials.

Smoke testing	→ Checks that the build launches and the main loop can be reached.
Functional testing	→ Checks controls, interactions, UI, triggers, enemy behaviours and win or fail states.
Regression testing	→ Checks whether a recent change broke something that previously worked.
Compatibility testing	→ Checks operation across target operating systems, input devices and hardware profiles.

Feedback: Testing checks quality and function, while user trials observe player experience. Different test types answer different production questions.

Q10 · Multiple choice · E5-Q10 Facilitate a user trial without coaching

PC 5.4, KE KE4, KE KE6

A user trial is testing whether new players can discover the controls from in-game prompts. Which facilitation behaviour is most appropriate?

- ✔ **Use the same brief, confirm consent, then observe without hints.**
- Teach the full control scheme to each participant immediately so no one becomes frustrated during the session.
- Give extra hints to slower participants so all sessions finish in the same time and produce matching results.
- Change the build settings between participants whenever the first participant finds the task difficult.

Feedback: Consistent trial conditions are essential. Facilitators should avoid coaching when discoverability is being tested, while still protecting consent, comfort and safety.

Q11 · Multiple choice (multiple answers) · E5-Q11 Log defects and observations from prototype tests

PC 5.4, KE KE1, KE KE2, KE KE4, KE KE9

A tester reports that the prototype sometimes freezes when the player enters a portal. Which information should be included in a useful defect record? Select all that apply.

- ✔ **Build version, platform, hardware and relevant input device.**
- ✔ **Steps to reproduce, expected result and actual result.**
- ✔ **Frequency, severity, screenshots, logs or crash reports where available.**
- ✔ **Related scene, script, asset or system, plus owner and status.**
- The participant’s home address and unrelated personal identifiers.
- A speculative cause recorded as fact without reproduction evidence.

Feedback: A defect record should allow another team member to understand, reproduce and prioritise the issue. It should also protect participant data and keep facts separate from interpretation.

Q12 · Drag the words · E5-Q12 Agree and document required prototype changes

PC 5.3, KE KE3, KE KE4, KE KE6, KE KE8, KE KE9

In a change meeting, the team should agree on [[1]] before choosing a fix, compare solution options against criteria and [[2]], record each agreed work item with owner, priority and [[3]], and communicate deferred or rejected suggestions to protect the [[4]].

Words: **the problem** (group 1) · **constraints** (group 1) · **acceptance criteria** (group 1) · **milestone baseline** (group 1) · **personal preference** (group 1) · **social media response** (group 1) · **final soundtrack** (group 1) · **undocumented workaround** (group 1)

Feedback: Required changes should be evidence-based, negotiated against constraints, recorded as actionable work items and confirmed by the required personnel.

Q13 · Short answer · E5-Q13 Name the formal approval to proceed

PC 5.6, KE KE3, KE KE4, KE KE6

After the prototype evaluation, the producer emails that the required personnel accept the evidence, agreed changes, remaining risks and production plan, and authorise the team to move into fuller production. Enter the workplace term for this formal approval.

Accepted answer: **endorsement**

Feedback: Endorsement is documented approval from required personnel that the prototype has met enough criteria to proceed, possibly with conditions.

Q14 · Multiple choice · E5-Q14 Plan the transition from prototype to complete product

PC 5.6, KE KE1, KE KE2, KE KE8, KE KE9

A prototype is endorsed, but it contains duplicated scripts, placeholder assets, hard-coded values and unoptimised materials. What is the best next step before simply expanding content?

- ✔ **Complete a transition review before expanding content.**
- Add all remaining levels directly to the prototype project because the endorsed build should not be changed structurally.
- Discard all prototype decisions and restart the creative direction from scratch before any production planning.
- Focus only on marketing screenshots because technical debt can be addressed after release if time remains.

Feedback: Developing a complete product often requires a transition review. Prototype code and assets may not be production-ready, even if the concept is endorsed.

Q15 · Multiple choice (multiple answers) · E5-Q15 Build a production roadmap after endorsement

PC 5.6, KE KE2, KE KE3, KE KE4, KE KE5, KE KE6, KE KE7.1, KE KE7.2, KE KE7.3, KE KE8

A team is turning an endorsed prototype into a complete 3-D game. Which roadmap practices support controlled development? Select all that apply.

- **✔ Use staged milestones such as vertical slice, alpha, beta and release candidate where suitable.**
- **✔ Schedule development and testing together rather than leaving all testing until the end.**
- **✔ Track dependencies and critical path items that can block later production work.**
- **✔ Plan graphics, input, performance and platform needs for the target hardware.**
- Treat endorsement as permission to ignore the prototype baseline and change the target player without approval.
- Avoid documenting risks because the prototype has already been approved.

Feedback: A roadmap translates endorsement into staged work, testing, dependencies, critical path management and target hardware planning.

Q16 · Numerical · E5-Q16 Calculate a simple critical path after endorsement

PC 5.6, KE KE4, KE KE6, KE KE9

A post-endorsement roadmap has these tasks: A: confirm asset pipeline, 4 days. B: refactor prototype controller code, 6 days. C: assemble the vertical slice level, 5 days, and it cannot start until both A and B are complete. D: run the first production playtest, 2 days, and it cannot start until C is complete. What is the minimum project duration in days?

Answer: **13** ± 0

Feedback: Critical path management identifies the longest dependency chain. Here, C must wait for both A and B, so the longer of A or B controls the start of C. The duration is 6 + 5 + 2 = 13 days.

Q17 · True / False · E5-F1 Prototype polish self-check

A prototype can be useful for evaluation even if it uses greybox levels or placeholder assets, provided it clearly demonstrates the core gameplay loop and limitations are explained.

Answer: **True**

Feedback: Prototypes are focused working versions used to test ideas before full production investment.

Q18 · Multiple choice · E5-F2 Strengthen a vague feedback comment

Which version of feedback is most actionable for a production team?

- **✔ Build, location, impact and next test are recorded.**
- The camera is described as bad, but the build version, scene location and reproduction conditions are not stated.
- Everyone will probably dislike the camera, so the art direction should be approved quickly.
- Make the game look better before any more testing occurs, because polish fixes most feedback problems.

Feedback: Actionable feedback links an issue to evidence, impact and follow-up action.

Q19 · Matching · E5-F3 Roadmap milestone self-check

Match each milestone term with its common meaning.

Vertical slice	→ A small section built to near-final quality to prove the production approach.
Alpha	→ Major features and content are implemented, though not polished.
Beta	→ Content is largely complete and focus shifts to bug fixing, optimisation and balancing.
Release candidate	→ A build considered ready unless serious issues are found.

Feedback: Milestone terms help teams plan staged development and testing after endorsement.

Q20 · Numerical · E5-F4 Crash rate calculation practice

In a compatibility test, 3 out of 12 sessions crashed on a low-spec target machine. What percentage of sessions crashed? Enter the answer to one decimal place.

Answer: **25** ± 0.1

Feedback: Percentages can help summarise technical test results for evaluation and prioritisation. The calculation is 3 ÷ 12 × 100 = 25.0%.

Q21 · Select missing words · E5-F5 Findings and recommendations practice

A strong user trial finding links the trial [[1]] to the evidence, states a supported [[2]], recommends an action, and identifies what should be [[3]].

Words: **criterion** (group 1) · **judgement** (group 1) · **retested** (group 1) · **rumour** (group 1) · **preference** (group 1) · **ignored** (group 1)

Feedback: Evaluation findings should connect evidence to criteria, judgement, action and retesting needs.

E6 — Transform prototype into final publication

Q1 · Multiple choice · E6-Q1 Prioritising a playtest finding PC 6.1, KE KE3, KE KE6

During a user trial, three participants become trapped behind a lift in Level 2 and cannot continue without restarting the game. Which action best treats this finding as a required publication-stage change?

- **✓ Log a high-severity defect with evidence, acceptance criteria and regression checks.**
- Treat it as a suggestion for a later sprint because only some players encountered it.
- Close it after the lift script is edited, without waiting for QA verification.
- Raise it at sign-off only, so reviewers can decide whether the risk is acceptable.

Feedback: A progression break or softlock is a high-impact defect. It should be logged with evidence, expected behaviour, priority, owner and an acceptance test, then fixed and retested before release.

Q2 · Drag the words · E6-Q2 Building a controlled change request PC 6.1, KE KE1, KE KE3, KE KE4, KE KE6

A disciplined change request for a playtest issue should link the issue to [[1]], state the [[2]] result from the specification or intended design, define [[3]] criteria for QA, and send the fix for [[4]] testing before closure.

Words: **trial evidence** (group 1) · **expected** (group 1) · **acceptance** (group 1) · **regression** (group 1) · **marketing** (group 1) · **asset-folder** (group 1) · **speculative feature** (group 1) · **credits** (group 1)

Feedback: A useful change request is evidence-based, states expected behaviour, includes acceptance criteria and requires regression testing so that fixes do not break nearby systems.

Q3 · True / False · E6-Q3 Ready for test versus closed PC 6.1, KE KE3, KE KE6

A user-trial defect can be marked closed as soon as its status is changed to 'Ready for test'.

Answer: **False**

Feedback: 'Ready for test' means the developer believes the issue is fixed. The item should only be closed after QA or the assigned reviewer verifies it in the correct build and it is accepted for release.

Q4 · Matching · E6-Q4 User-trial issue statuses PC 6.1, KE KE3, KE KE4, KE KE6

Match each late-stage issue status to its most appropriate meaning.

Open	→ Issue has been recorded but is not yet resolved.
Assigned	→ An owner has been selected to handle the issue.
Ready for test	→ A fix has been implemented and is awaiting verification.
Verified	→ A tester has confirmed the fix in the relevant build.
Deferred	→ The issue will not be fixed in this release, with approval and a recorded reason.

Feedback: Accurate status tracking supports release control, scheduling, risk management and final sign-off evidence.

Q5 · Multiple choice (multiple answers) · E6-Q5 Specification-based integration checks

PC 6.2, KE KE2, KE KE5, KE KE7.1, KE KE7.2, KE KE7.3, KE KE8, KE KE9

A team is preparing to integrate final 3-D game elements into the release build. Which checks should be included in the integration checklist? Select all that apply.

- **✔ Confirm assets use approved formats, naming conventions, folder locations and import settings.**
- **✔ Check scripts, prefabs, scenes, references and dependencies after merging content.**
- **✔ Validate platform settings, input mappings and performance budgets for target hardware.**
- **✔ Confirm lighting, texture sizes, polygon budgets and visual effects align with the visual design and hardware constraints.**
- Import every file supplied by the team, even if it is unapproved or outside the GDD.
- Ignore licence and asset ownership checks until after distribution, because they do not affect whether the engine can load the build.

Feedback: Integration means confirming that required assets, systems, platform settings and dependencies are present, connected, configured and suitable for the target hardware and specification.

Q6 · Matching · E6-Q6 Common asset formats in integration

PC 6.2, KE KE2, KE KE9

Match each game asset or engine item to a typical format or representation used during integration.

3-D model mesh	→ FBX or glTF
Texture image	→ PNG or TGA
Audio clip	→ WAV or a compressed audio format
Unity reusable configured object	→ Prefab
Unreal reusable scripted asset	→ Blueprint

Feedback: Asset integration requires knowing whether the engine can import, reference, load and use each asset type correctly.

Q7 · Multiple choice · E6-Q7 Integrated locked-door system

PC 6.2, KE KE1, KE KE3, KE KE8, KE KE9

A final build contains the 3-D model for a locked door, but players receive no prompt, no sound, no inventory check and no fallback message when they lack the key. What is the best assessment of this element?

- **✔ It is not fully integrated; key gameplay and feedback links are missing.**
- It is complete because the model renders in the scene and can be viewed by players.
- It is feature-complete because prompts, sound and fallback messages are only final polish.
- It should be deferred automatically, because locked-door interactions are never release blockers.

Feedback: A player-facing element is integrated only when the required systems communicate: input, inventory or objective state, animation, collision, UI, audio, save or checkpoint behaviour and feedback specified by the design.

Q8 · True / False · E6-Q8 Meaning of feature-complete PC 6.2, KE KE3, KE KE4, KE KE6, KE KE8, KE KE9

A feature-complete build must have every required system present so QA can test the product as a whole, but it may still contain bugs that need fixing.

Answer: **True**

Feedback: Feature-complete does not mean bug-free. It means the planned release systems and required elements are present and integrated enough for whole-product testing.

Q9 · Matching · E6-Q9 Final QA activity types PC 6.3, KE KE2, KE KE3, KE KE5, KE KE6, KE KE7.1, KE KE7.2

Match each final QA activity to the check it mainly performs.

Smoke testing	→ Confirms the build launches, core scenes load and the build is not obviously broken.
Functional testing	→ Checks features behave according to the specification.
Regression testing	→ Confirms previous fixes still work after later changes.
Compatibility testing	→ Checks target hardware, operating systems, graphics settings and input devices.
Performance testing	→ Monitors frame rate, loading, memory use and stability against requirements.

Feedback: Final checks combine multiple QA passes, including smoke, functional, regression, compatibility and performance testing.

Q10 · Multiple choice (multiple answers) · E6-Q10 Navigation conformity checks

PC 6.3, KE KE3, KE KE5, KE KE6, KE KE8

A release candidate must be checked against the approved navigation design. Which checks provide evidence that implemented sequences conform? Select all that apply.

- **✔ Verify required entry and exit paths between screens, levels and states.**
- **✔ Confirm failure, restart, pause, resume and return-to-menu paths recover correctly.**
- **✔ Check that progress, settings and inventory persist as specified.**
- **✔ Test menus with the required controller, keyboard and mouse input devices.**
- Approve navigation if the lead designer remembers the intended flow and it seems close enough.
- Skip cutscene and loading transitions because they are presentation rather than navigation.

Feedback: Navigation conformity is checked by comparing real build behaviour with the approved flowcharts, UI wireframes, level progression maps, sequence diagrams and acceptance criteria.

Q11 · Select missing words · E6-Q11 Structured gameplay sequence checking

PC 6.3, KE KE1, KE KE3, KE KE6, KE KE8

To check a gameplay sequence, start from a [[1]], follow the [[2]], then test [[3]] and confirm [[4]] from failure states.

Words: **clean save** (group 1) · **expected player path** (group 1) · **likely edge cases** (group 1) · **recovery** (group 1) · **developer memory** (group 1) · **random route only** (group 1) · **unapproved shortcut** (group 1) · **marketing screenshot** (group 1)

Feedback: Sequence checks should use a repeatable test state, follow expected player paths, probe likely edge cases and confirm that failure states recover without softlocks.

Q12 · Multiple choice · E6-Q12 Final check recommendation

PC 6.3, KE KE4, KE KE6, KE KE8

A final check report shows that all required menu paths work, but one unresolved defect sometimes prevents players from progressing after a checkpoint reload. What recommendation is most appropriate?

- **✔ Fix it before release, unless an authorised deferral is recorded.**
- Proceed because the menus work and the issue occurs only after a checkpoint reload.
- Remove it from the report so the sign-off decision is not delayed by one defect.
- Treat it as a feature request because it appears after a save-state transition.

Feedback: Progression breaks are release risks. A final check report should list unresolved known issues and recommend proceed, fix or formally defer based on severity, scope and approval.

Q13 · Multiple choice (multiple answers) · E6-Q13 Evidence for final sign-off

PC 6.4, KE KE3, KE KE4, KE KE6, KE KE8, KE KE9

Before asking required personnel to approve a release candidate, which items should be prepared as part of the sign-off evidence pack? Select all that apply.

- **✔ Specific build number, version label, date and distribution target.**
- **✔ Final QA results, navigation test report, regression notes and unresolved known issues.**
- **✔ Release notes, acceptance criteria, licence or credit evidence and relevant asset documentation.**
- **✔ Record of approved user-trial changes being closed or formally deferred.**
- A general statement that the team feels the build is ready, without build traceability.
- A folder of the latest unlabelled editor files so reviewers can choose whichever version opens.

Feedback: Final sign-off is a formal release gate. Reviewers need evidence linked to a specific build, scope, test outcomes, known risks and required documentation.

Q14 · Multiple choice · E6-Q14 Handling a late sign-off comment

During final sign-off, a senior reviewer suggests adding a new multiplayer mode that was not in the approved specification. What is the best release-management response? PC 6.4, KE KE1, KE KE3, KE KE6, KE KE8

- ✔ **Treat it as a new feature request; include it only after formal scope approval.**
- Implement it immediately because senior stakeholder comments override the release candidate scope.
- Add it to the final package without retesting because multiplayer is separate from navigation.
- Reject every reviewer comment during sign-off, including genuine blocking defects.

Feedback: Late review comments should be classified and controlled. New feature requests are usually out of scope unless formally approved after impact analysis, because they can affect integration, navigation, performance, testing and schedule.

Q15 · Short answer · E6-Q15 Named build used for final review PC 6.4, KE KE3, KE KE4

Your team freezes a specific versioned build that could become the final publication if the required reviewers approve it. Enter the standard two-word term for this build.

Accepted answer: **release candidate**

Feedback: A release candidate is a specific, versioned build that could become the final publication if approved. It avoids reviewers assessing a moving target.

Q16 · Numerical · E6-Q16 Calculating final archive allocation

An organisational procedure requires the final archive allocation to include the approved packaged build, the source snapshot and the evidence pack, plus 10% contingency. The packaged build is 2.8 GB, the source snapshot is 1.6 GB and the evidence pack is 0.35 GB. What minimum storage allocation, in GB, should be planned? PC 6.5, KE KE3, KE KE4, KE KE6, KE KE8, KE KE9

Answer: **5.225** ± 0.01

Feedback: Add the required archive components first: 2.8 + 1.6 + 0.35 = 4.75 GB. Then add 10% contingency: 4.75 × 1.10 = 5.225 GB.

Q17 · True / False · E6-F1 Practice: packaged build testing

A game that works correctly in the engine editor should still be launched and smoke-tested as a packaged build before distribution.

Answer: **True**

Feedback: Packaged builds can behave differently from editor runs because of compiled scripts, cooked or stripped assets, plugins, shaders, permissions and platform settings.

Q18 · Multiple choice · E6-F2 Practice: approved distribution channel

A team has final sign-off for a WebGL build. What should they do immediately after uploading it to the approved portal?

- ✔ **Launch it from the portal as a user and run a short smoke test.**
- Delete the local approved build because the upload has completed successfully.
- Change the build settings to prepare for a different platform before checking it.
- Assume the release is valid because the upload progress bar reached 100%.

Feedback: After upload, verify that the distributed version works for users by checking permissions, downloading or launching from the channel and performing a short smoke test.

Q19 · Matching · E6-F3 Practice: publication package contents

Match each publication item to its purpose.

Release notes	→ Summarise version, changes and known issues.
Install instructions	→ Explain how the intended audience should set up or launch the build.
Third-party notices	→ Identify required licence and credit information.
Support information	→ Tell users how to report problems or seek help.

Feedback: A final publication package normally includes the build plus supporting materials that help users, clients, reviewers and support staff install, launch, verify and troubleshoot the game.

Q20 · Select missing words · E6-F4 Practice: release workflow order

A safe release workflow is to confirm [[1]], obtain [[2]], upload through an [[3]], and archive the [[4]].

Words: **the approved build** (group 1) · **final sign-off** (group 1) · **authorised channel** (group 1) · **exact final publication** (group 1) · **unlabelled editor folder** (group 1) · **informal compliment** (group 1) · **personal file share** (group 1) · **prototype sketch** (group 1)

Feedback: Distribution should follow organisational procedure: confirm the approved build, obtain sign-off, use authorised channels and archive the exact release and evidence.