

PRACTICE QUESTIONS

Microsoft

GH-700

GitHub Certified: Agentic AI Business Professional

SHARED BY [u/Kind-Switch-1541](#) on aiexamcerts

EDITION	Free Set 1
QUESTIONS	10
SHARED	October 18, 2026
EXAM	Microsoft GH-700, GitHub Certified: Agentic AI Business Professional

Shared by [u/Kind-Switch-1541](#) on aiexamcerts

10 practice questions for the Microsoft GH-700 exam, GitHub Certified: Agent AI Business Professional, each followed by its answer and a short explanation. More free sets and the full practice exam: aiexamcerts.com/microsoft/gh-700. Spotted a mistake? Email support@aiexamcerts.com with the exam code GH-700 and the question number.

1. A team wants an agent to triage new issues, apply labels, and ask a person to decide anything ambiguous. Which design is best?
- A Let the agent close any issue it considers a duplicate, without leaving a comment that explains why
 - B Let the agent assign itself to every new issue and begin coding immediately, without waiting for a planning decision
 - C Let the agent apply labels and then merge its own fixes as soon as the labels are applied to the issue
 - D Let the agent apply labels and draft replies, and route ambiguous issues to a person for a decision

Answer: D

Letting the agent label issues and draft replies, while a person decides the ambiguous ones, keeps the routine work automated and the judgment with people. Closing duplicates silently is tempting because it saves time, but it removes the chance for a person to check. Self-assignment and self-merging remove human decisions the workflow still needs.

2. A Copilot custom agent can be given access to every repository in the enterprise by default, so there is no need to scope its permissions for each workflow.
- A True
 - B False

Answer: B (False)

Permissions for an agent should follow least privilege, so each workflow gets only the repositories and actions it needs. Granting enterprise-wide access by default makes any single mistake far more costly. The statement is false, and the convenience of a single setting is what makes it tempting.

3. Which versioning practice makes a reusable workflow safer for other repositories to adopt?
- A Pinning callers to a tagged release or a full commit SHA rather than to a branch that keeps moving
 - B Pointing every repository at the main branch of the shared workflow, so that updates arrive automatically
 - C Copying the workflow file into each repository and editing each copy separately over time
 - D Renaming the workflow file every time it changes, so that each version has a unique name

Answer: A

Pinning to a tag or a full commit SHA means callers adopt a known version and choose when to update it. Pointing at the main branch is tempting because updates arrive without effort, but a change can break every caller at once. Copying files and renaming them both fragment the workflow and make versions harder to track.

4. Which Git practice best supports a team's ability to roll back a release cleanly?
- A Force pushing the default branch after every hotfix so that the history stays short
 - B Tagging each release commit with a semantic version and publishing a GitHub release from that tag
 - C Deleting old tags regularly so that the repository stays small and easy to clone
 - D Merging every feature branch into one long-lived branch that is never split apart again

Answer: B

A tag on the release commit gives a stable, named point that can be redeployed or reverted to, and the GitHub release records what shipped. Force pushing rewrites the history that rollbacks depend on, which makes it tempting only for its short-looking log. Deleting tags and merging into one long-lived branch both remove the named points a rollback needs.

5. Lead time for changes is one of the four DORA metrics, and it measures how long it takes a code _____ to move from commit to production.

Answer: change

Lead time for changes measures the time between a commit and its arrival in production, which is a standard measure of delivery speed. Deployment frequency is tempting as a related measure, but it counts releases rather than timing a single change. The accepted answers are change and changes.

6. Which Copilot feature gives an organization administrator visibility into how many developers are actively using Copilot?
- A Inline code completion settings, which control whether suggestions appear while a developer types in the editor
 - B Content exclusion rules, which control which files Copilot may read as context for its suggestions and answers
 - C Copilot usage metrics and seat management for the organization, which show active use and assigned seats
 - D Copilot Chat conversation history in the browser, which keeps each developer's own past questions

Answer: C

Copilot usage metrics and seat management show how many seats are assigned and how many people actively use them, which is what an administrator needs for adoption reporting. Inline completion settings and content exclusion are configuration controls, which makes them tempting, but neither reports usage. A developer's own chat history is personal and not an organization-wide view.

7. A developer asks Copilot Chat to explain a failing test. Which step gives the answer the most useful context?
- A Asking with no code selected and no file open, so the answer is not biased by the current work
 - B Pasting the entire company handbook into the prompt so that the answer covers every policy
 - C Asking the same question in a brand-new chat each time, so that old answers do not interfere
 - D Attaching the failing file and the test output, so that Copilot can ground its answer in both

Answer: D

Attaching the failing file and the test output gives Copilot the exact code and error it needs to explain the failure. Asking with no context is tempting when a developer wants a general answer, but it produces generic advice. Pasting unrelated documents adds noise, and starting a new chat each time discards context the developer already built up.

8. Which two features help a product team trace a piece of work from its requirement to the code that delivers it? (Choose two.)
- A Issues linked to pull requests through closing keywords, so each change points back to its requirement
 - B A shared spreadsheet that the team updates by hand each week with the status of every feature
 - C Commit messages that reference the issue number, so that the history connects code to its requirement
 - D Discussions threads grouped by the month in which they were opened by the team members
 - E Repository stars that show how popular each feature is with people outside the team

Answer: A, C

Closing keywords and commit messages that reference an issue tie the code history back to the requirement. A shared spreadsheet is tempting because it tracks the same work, but it is maintained by hand and drifts from the code. Discussion threads and stars do not connect a requirement to a specific change.

9. Copilot _____ are a repository-level file of standing instructions that Copilot follows for each request made in that repository.

Answer: custom instructions (also accepted: instructions)

Repository custom instructions apply to every request in the repository, so the team states its conventions once. A prompt pasted into one chat is tempting because it works immediately, but it does not persist. The accepted answers are custom instructions and instructions.

10. Human review is still needed for agent output even when automated tests pass, because tests check only the behaviors that someone thought to write tests for.
- A True
 - B False

Answer: A (True)

Automated tests confirm only the behaviors they cover, so a change can pass every test and still be wrong in an untested case. A reviewer judges intent, scope, and risks that tests do not encode. The statement is true, and passing tests is tempting as a complete signal.