

PRACTICE QUESTIONS

Microsoft

GH-700

GitHub Certified: Agentic AI Business Professional

SHARED BY [u/Breezy-Sparrow-7374](#) on aiexamcerts

EDITION	Free Set 2
QUESTIONS	10
SHARED	October 18, 2026
EXAM	Microsoft GH-700, GitHub Certified: Agentic AI Business Professional

Shared by [u/Breezy-Sparrow-7374](#) 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. Which reporting view best answers how long pull requests wait for a first review?
- A Pull request and review metrics that show time to first review and time to merge for each team
 - B A count of the stars the repository has collected from people outside the organization since its creation
 - C The number of branches that were created in the repository during the previous calendar year
 - D The total size of the README file and the images stored alongside it in the repository root

Answer: A

Pull request and review metrics report time to first review and time to merge, which answer the question directly. Star counts and branch counts are measurable repository numbers, which makes them tempting, but they say nothing about review waiting time. The README size is unrelated to review flow.

2. Which branch protection setting makes a review happen before a pull request can merge?
- A Allowing force pushes to the protected branch so that history can be rewritten when a team needs it
 - B Requiring pull request reviews before merging into the protected branch, with a set number of approvals
 - C Disabling status checks entirely so that merges can happen faster when the team is under pressure
 - D Deleting the head branch automatically after each merge so that stale branches do not accumulate

Answer: B

Requiring pull request reviews before merging makes an approval a precondition for the merge, which is the control the question asks for. Automatic branch deletion is a real setting that is tempting for housekeeping, but it does not require any review. Allowing force pushes and disabling status checks both weaken the protections the team needs.

3. Copilot Chat on github.com can read and act on any private repository in an organization without the user's own repository permissions.
- A True
 - B False

Answer: B (False)

Copilot works within the permissions of the user who asks, so a person who cannot read a private repository cannot get Copilot to read it for them. Access is inherited from the requester rather than granted by Copilot itself. The statement is false, and the idea that Copilot has its own broad access is the tempting error.

4. What is the main purpose of Copilot code review on a pull request?
- A Approving the pull request automatically once the test suite has finished and reported that all tests pass
 - B Replacing human reviewers entirely for security-sensitive files, so that no person has to read those changes again
 - C Giving a first-pass review comment on the change, which a human reviewer still weighs before deciding anything
 - D Merging low-risk changes without a human reviewer, so that routine updates do not wait in a review queue

Answer: C

Copilot code review gives a first pass of comments, and a human reviewer still judges the change and decides whether to approve it. Automatic approval is tempting because it would speed up merges, but it removes the human decision the process requires. Replacing reviewers and merging low-risk changes both drop the review the team depends on.

5. Which practice reduces the risk that Copilot suggestions include code under a license the team cannot use?
- A Turning off all Copilot suggestions for the whole team so that no public code is ever suggested again
 - B Accepting every suggestion as it appears, which saves review time and keeps the editor focused on the work
 - C Ignoring the license headers that appear in the repository's files, since Copilot handles licensing on its own
 - D Enabling the public code matching policy in Copilot settings and reviewing any suggestion that matches public code

Answer: D

The public code matching policy lets an organization block or allow suggestions that match public code, and reviewing matches keeps licensing decisions with the team. Turning off all suggestions is tempting as a blunt fix, but it removes the productivity benefit and does not address matches that already exist. Accepting everything and ignoring license headers both leave the risk unmanaged.

6. A workflow that runs at fixed times uses the _____ trigger in GitHub Actions, with a cron expression that sets the schedule.

Answer: schedule

The schedule trigger runs a workflow on a cron expression, which suits recurring agent tasks such as nightly reports. The push trigger is tempting because it is the most common trigger, but it fires on commits rather than time. The accepted answer is schedule.

7. Which design decision reduces the blast radius of an agent workflow that can open pull requests?
- A Giving the workflow a token scoped only to the repositories and permissions that it needs for its task
 - B Giving the workflow an administrator token so that it never fails on a permission error during a run
 - C Letting every workflow share one token across all teams so that credentials are easier to manage
 - D Disabling the workflow's logs so that the token is never exposed in a place where someone could copy it

Answer: A

A token scoped to the needed repositories and permissions limits what a compromised or mistaken run can change. An administrator token is tempting because it removes permission failures, but it turns every error into a potential organization-wide change. Shared tokens and disabled logs both enlarge the impact or remove the means to detect it.

8. A team wants a workflow to run only when a pull request changes files in the docs folder. Which mechanism fits?
- A A cron schedule that runs the workflow every night, whether or not any files have changed
 - B A paths filter on the pull_request trigger that lists the docs folder, so the run starts only for matching changes
 - C A manual workflow_dispatch trigger with no inputs, started by a developer when the docs need checking
 - D A release trigger that fires only when a version tag is created, after the documentation has already shipped

Answer: B

A paths filter on the pull_request trigger limits the run to pull requests whose changed files match the listed folder. A nightly schedule is tempting because it also runs automatically, but it ignores what changed. Manual dispatch and release triggers do not respond to pull request file changes.

9. Which two practices make an automated workflow safer when it runs on pull requests from outside contributors? (Choose two.)
- A Require approval before workflows run for first-time or outside contributors, so that a person checks the change first
 - B Pass repository secrets to every pull request automatically so that the tests can reach the internal services
 - C Use the `pull_request_target` trigger to check out untrusted code while the workflow holds write tokens
 - D Pin third-party actions to a full commit SHA so that a later change to the action cannot run unnoticed
 - E Grant the workflow write access to every repository in the organization so that no run is refused

Answer: A, D

A required approval for outside contributors keeps a person in front of untrusted changes, and pinning actions to a commit SHA stops a silent change in a third-party action. Passing secrets to every pull request is tempting because it makes tests pass, but it exposes secrets to untrusted code. Checking out untrusted code with write tokens and granting organization-wide write access are both unsafe.

10. Which versioning concern matters most for a workflow that other teams call as a reusable workflow?
- A Changing the workflow file name at every release so that each version is easy to tell apart at a glance
 - B Storing the workflow's output in a personal fork so that the author keeps control of the results
 - C Changing its inputs or behavior without publishing a new version, which breaks callers who depend on it
 - D Adding comments throughout the YAML so that the file is easier for every new team member to read

Answer: C

Changing inputs or behavior in place breaks every caller that depends on the current contract, so changes should be published as new versions. Renaming the file at each release is tempting because it looks like versioning, but callers still reference the path. Comments and personal forks do not protect callers from breaking changes.