Marks4Sure

# Microsoft

# GH-300

# GitHub Copilot Exam

**Version: 5.2**

**[ Total Questions: 120]**

Web: www.marks4sure.com

Email: support@marks4sure.com

# IMPORTANT NOTICE

## Feedback

We have developed quality product and state-of-art service to ensure our customers interest. If you have any suggestions, please feel free to contact us at feedback@marks4sure.com

## Support

If you have any questions about our product, please provide the following items:

- exam code
- screenshot of the question
- login id/email

please contact us at support@marks4sure.com and our technical experts will provide support within 24 hours.

## Copyright

The product of each order has its own encryption code, so you should use it independently. Any unauthorized changes will inflict legal punishment. We reserve the right of final explanation for this statement.

**Category Breakdown**

| Category | Number of Questions |
|---|---|
| Copilot Basics and Usage | 35 |
| Administration and Licensing | 17 |
| Security, Privacy, and Compliance | 12 |
| Copilot Chat and Slash Commands | 24 |
| IDE Integration and Features | 4 |
| Copilot CLI | 1 |
| Data Sources and Context | 3 |
| Prompt Engineering and Best Practices | 10 |
| Refactoring and Code Modernization | 5 |
| Knowledge Bases and Custom Models | 4 |
| Testing and Quality | 3 |
| Ethical AI Principles | 2 |
| TOTAL | 120 |

Question #:1 - [Copilot Basics and Usage]

What are the potential risks associated with relying heavily on code generated from GitHub Copilot? (Each correct answer presents part of the solution. Choose two.)

A. GitHub Copilot may introduce security vulnerabilities by suggesting code with known exploits.

B. GitHub Copilot may decrease developer velocity by requiring too much time in prompt engineering.

C. GitHub Copilot's suggestions may not always reflect best practices or the latest coding standards.

D. GitHub Copilot may increase development lead time by providing irrelevant suggestions.

**Answer: A C**

## Explanation

Heavy reliance on GitHub Copilot can introduce security vulnerabilities if the generated code contains known exploits. Additionally, Copilot's suggestions may not always align with best practices or the latest standards, requiring careful review and validation.

Reference: GitHub Copilot best practices and risk management.

Question #:2 - [Copilot Basics and Usage]

How does GitHub Copilot typically handle code suggestions that involve deprecated features or syntax of programming languages?

A. GitHub Copilot automatically updates deprecated features in its suggestions to the latest version.

B. GitHub Copilot may suggest deprecated syntax or features if they are present in its training data.

C. GitHub Copilot always filters out deprecated elements to promote the use of current standards.

D. GitHub Copilot rejects all prompts involving deprecated features to avoid compilation errors.

**Answer: B**

## Explanation

"GitHub Copilot may sometimes suggest deprecated code, APIs, or patterns if these appear in its training data. Users are responsible for reviewing and updating the suggestions."

This confirms that Copilot does not automatically update or reject deprecated features, but may still suggest them if they were part of training.

References: GitHub Copilot usage limitations documentation.

==========

Question #:3 - [Administration and Licensing]

What GitHub Copilot feature can be configured at the organization level to prevent GitHub Copilot suggesting publicly available code snippets?

A. GitHub Copilot Chat in the IDE

B. GitHub Copilot Chat in GitHub Mobile

C. GitHub Copilot duplication detection filter

D. GitHub Copilot access to Bing

**Answer: C**

## Explanation

The duplication detection filter can be configured at the organization level to prevent GitHub Copilot from suggesting publicly available code snippets.

Question #:4 - [Security, Privacy, and Compliance]

What do you check when GitHub Copilot content exclusions are not working? (Each correct answer presents part of the solution. Choose two.)

A. If GitHub Copilot can connect to the server selected in your user settings.

B. If the user is part of the content exclusion team that limits the use of content exclusions.

C. If the content exclusion settings changed in the last 30 minutes or before that.

D. If the user is in an organization that has content exclusions configured.

**Answer: C D**

## Explanation

Exact extracts:

- "Changes to content exclusion settings can take up to 30 minutes to take effect."

References: GitHub Copilot troubleshooting and content exclusions documentation.

==========

Question #:5 - [Copilot Basics and Usage]

How can GitHub Copilot facilitate a smoother learning experience when diving into a new programming language? (Each correct answer presents part of the solution. Choose two.)

A. GitHub Copilot Chat can provide guidance and support for common coding tasks and challenges in the targeted programming language.

B. GitHub Copilot's /understand command will help GitHub Copilot to understand code written in a targeted programming language.

C. GitHub Copilot can provide contextualized code suggestions and answer sources from an organization's documentation.

D. GitHub Copilot can convert comments into code to grasp the syntax and nuances of a new programming language.

**Answer: A D**

## Explanation

GitHub Copilot helps with learning new languages by providing guidance on common tasks and by converting comments into code, allowing developers to see syntax in action.

Reference: GitHub Copilot language learning documentation.

Question #:6 - [Copilot Chat and Slash Commands]

What role does chat history play in GitHub Copilot's code suggestions?

A. Chat history is used to train the GitHub Copilot model in real-time.

B. Chat history provides context to GitHub Copilot, improving the relevance and accuracy of its code suggestions.

C. Chat history is stored and shared with other users to enhance collaboration.

D. Chat history is irrelevant to GitHub Copilot and does not affect its functionality.

**Answer: B**

## Explanation

Chat history provides valuable context to GitHub Copilot, helping it generate more relevant and accurate code suggestions based on previous interactions and conversations.

Reference: GitHub Copilot Chat history documentation.

Question #:7 - [Administration and Licensing]

If you are working on open source projects, GitHub Copilot Individual can be paid:

A. Based on the payment method in your user profile

B. N/A – Copilot Individual is a free service for all open source projects

C. Through an invoice or a credit card

D. Through an Azure Subscription

**Answer: A**

## Explanation

“GitHub Copilot Individual subscriptions are billed using the payment method configured in your GitHub user profile.”

There is no free tier for open source projects, so option A is correct.

References: GitHub Copilot billing documentation.

Question #:8 - [Copilot Basics and Usage]

How do you generate code suggestions with GitHub Copilot in the CLI?

A. Write code comments -> Press the suggestion shortcut -> Select the best suggestion from the list.

B. Use copilot suggest -> Write the command you want -> Select the best suggestion from the list.

C. Type out the code snippet -> Use the copilot refine command to enhance it -> Review the suggested command.

D. Describe the project's architecture -> Use the copilot generate command -> Accept the generated suggestion.

**Answer: A**

## Explanation

In the CLI, GitHub Copilot generates code suggestions by analyzing code comments. You write comments describing what you want, and Copilot provides relevant code suggestions. You then select the best suggestion from the list.

Reference: GitHub Copilot CLI documentation.

Question #:9 - [IDE Integration and Features]

Which Copilot Individual features are available when using a supported extension for Visual Studio, VS Code, or JetBrains IDEs? (Each correct answer presents part of the solution. Choose two.)

A. Code suggestions

B. Chat

C. Knowledge Base

D. Pull Request Diff Analysis

**Answer: A B**

## Explanation

GitHub Copilot Individual provides code suggestions and chat features when used with supported IDE extensions like Visual Studio, VS Code, and JetBrains IDEs.

Reference: GitHub Copilot Individual features documentation.

Question #:10 - [Copilot CLI]

GitHub Copilot in the Command Line Interface (CLI) can be used to configure the following settings: (Each correct answer presents part of the solution. Choose two.)

A. The default execution confirmation

B. Usage analytics

C. The default editor

D. GitHub CLI subcommands

**Answer: A B**

## Explanation

GitHub Copilot in the CLI allows configuration of settings such as the default execution confirmation and usage analytics. These settings help tailor the CLI experience to the user's preferences.

Reference: GitHub Copilot CLI configuration documentation.

Question #:11 - [Data Sources and Context]

How does GitHub Copilot assist developers in minimizing context switching?

A. GitHub Copilot can automatically handle project management tasks.

B. GitHub Copilot can completely replace the need for human collaboration.

C. GitHub Copilot can predict and prevent bugs before they occur.

D. GitHub Copilot allows developers to stay in their IDE.

**Answer: D**

## Explanation

“Copilot reduces the need to switch between documentation, browsers, and editors by bringing suggestions and explanations directly into the IDE.”

This demonstrates that minimizing context switching comes from enabling developers to stay within their IDE.

References: GitHub Copilot productivity documentation.

==========

Question #:12 - [Copilot Basics and Usage]

Which Microsoft ethical AI principle is aimed at ensuring AI systems treat all people equally?

A. Inclusiveness

B. Fairness

C. Reliability and Safety

D. Privacy and Security

**Answer: B**

## Explanation

"Fairness. AI systems should treat all people fairly." This principle directly defines fairness as the standard for ensuring equality across all individuals.

References: Microsoft Responsible AI principles.

==========

Question #:13 - [Prompt Engineering and Best Practices]

When crafting prompts for GitHub Copilot, what is a recommended strategy to enhance the relevance of the generated code?

A. Keep the prompt as short as possible, using single words or brief phrases.

B. Write the prompt in natural language without any programming language.

C. Avoid mentioning the programming language to allow for more flexible suggestions.

D. Provide examples of expected input and output within the prompt.

**Answer: D**

## Explanation

"To get the best results from GitHub Copilot, provide clear prompts and, when possible, include examples of expected input and output."

This establishes that including examples is a recommended prompt engineering strategy.

References: GitHub Copilot prompt engineering documentation.

Question #:14 - [Refactoring and Code Modernization]

Are there any limitations to consider when using GitHub Copilot for code refactoring?

A. GitHub Copilot may not always produce optimized or best-practice code for refactoring.

B. GitHub Copilot can only be used with a limited set of programming languages.

C. GitHub Copilot always produces bug-free code during refactoring.

D. GitHub Copilot understands the context of your entire project and refactors code accordingly.

**Answer: A**

## Explanation

"While Copilot can suggest refactoring changes, the code may not always follow best practices or be fully optimized. Developers must review and validate suggestions."

This confirms that limitations exist in optimization and best practices, making option A correct.

References: GitHub Copilot limitations documentation.

==========

Question #:15 - [Refactoring and Code Modernization]

What is a likely effect of GitHub Copilot being trained on commonly used code patterns?

A. Suggest innovative coding solutions that are not yet popular.

B. Suggest completely novel projects, while reducing time on a project.

C. Suggest code snippets that reflect the most common practices in the training data.

D. Suggest homogeneous solutions if provided a diverse data set.

**Answer: C**

## Explanation

"Because GitHub Copilot is trained on a large amount of publicly available code, it often suggests snippets that reflect common coding practices and idioms."

This shows that Copilot reflects the most common practices it has seen in its training data.

References: GitHub Copilot documentation on training and suggestion patterns.

==========

Question #:16 - [Copilot Chat and Slash Commands]

How can users provide feedback about GitHub Copilot Chat using their IDE?

A. By filling out a feedback form on the GitHub website

B. By emailing the support team directly

C. By posting on the GitHub forums

D. Through the "Share Feedback" button in the Copilot Chat panel

**Answer: D**

## Explanation

"In supported IDEs, you can provide feedback directly from the Copilot Chat panel by selecting the 'Share Feedback' option."

This establishes that the in-IDE "Share Feedback" button is the official channel for submitting user feedback.

References: GitHub Copilot Chat feedback documentation.

Question #:17 - [IDE Integration and Features]

What should developers consider when relying on GitHub Copilot for generating code that involves statistical analysis?

A. GitHub Copilot can independently verify the statistical significance of results.

B. GitHub Copilot's suggestions are based on statistical trends and may not always apply accurately to specific datasets.

C. GitHub Copilot will automatically correct any statistical errors found in the user's initial code.

D. GitHub Copilot can design new statistical methods that have not been previously documented.

**Answer: B**

## Explanation

Developers should consider that GitHub Copilot's suggestions are based on statistical trends and may not always be accurate for specific datasets, requiring careful validation.

Reference: GitHub Copilot data analysis limitations.

Question #:18 - [Copilot Basics and Usage]

What can be done during AI development to minimize bias?

A. Improve on the computational efficiency and speed.

B. Focus on accuracy of the data.

C. Collect massive amounts of data for training.

D. Use diverse data, fairness metrics, and human oversight.

**Answer: D**

## Explanation

"Minimizing bias requires diverse datasets, use of fairness metrics, and human oversight throughout development and deployment."

This clearly confirms that the proper approach is a combination of diverse data, fairness metrics, and oversight.

References: Microsoft Responsible AI and GitHub Copilot fairness guidelines.

==========

Question #:19 - [Copilot Basics and Usage]

What practices enhance the quality of suggestions provided by GitHub Copilot? (Select three.)

A. Clearly defining the problem or task

B. Including personal information in the code comments

C. Using meaningful variable names

D. Providing examples of desired output

E. Use a .gitignore file to exclude irrelevant files

**Answer: A C D**

## Explanation

The quality of Copilot's suggestions is enhanced by clearly defining the task, using meaningful variable names, and providing examples of the desired output.

Reference: GitHub Copilot prompt engineering best practices.

Question #:20 - [Refactoring and Code Modernization]

How can GitHub Copilot assist with code refactoring tasks?

A. GitHub Copilot can fix syntax errors without user input.

B. GitHub Copilot can automatically rewrite code to follow best practices.

C. GitHub Copilot can suggest refactoring improvements for better code quality.

D. GitHub Copilot can remove unnecessary files from the project directory.

**Answer: C**

## Explanation

GitHub Copilot can analyze existing code and suggest refactoring improvements to enhance code quality, readability, and maintainability. It can propose changes to improve code structure, reduce complexity, and follow best practices.

Reference: GitHub Copilot documentation on code refactoring assistance.

Question #:21 - [Copilot Chat and Slash Commands]

How does GitHub Copilot Chat help to fix security issues in your codebase?

A. By enforcing strict coding standards that prevent the introduction of vulnerabilities.

B. By providing detailed reports on the security vulnerabilities present in the codebase.

C. By annotating the given suggestions with known vulnerability patterns.

D. By automatically refactoring the entire codebase to remove vulnerabilities.

**Answer: C**

## Explanation

“When Copilot Chat generates suggestions, it may annotate them with warnings about known insecure patterns, helping developers identify potential vulnerabilities in context.”

This indicates that Copilot Chat assists with security by annotating code suggestions with vulnerability patterns.

References: GitHub Copilot Chat security and responsible use documentation.

Question #:22 - [Copilot Basics and Usage]

Which of the following describes role prompting?

A. Describing in your prompt what your role is to get a better suggestion

B. Tell GitHub Copilot in what tone of voice it should respond

C. Prompt GitHub Copilot to explain what was the role of a suggestion

D. Giving GitHub Copilot multiple examples of the form of the data you want to use

**Answer: A**

## Explanation

Role prompting involves explicitly stating your role or the persona you want GitHub Copilot to adopt within your prompt. This helps Copilot provide more contextually relevant and accurate suggestions. By defining your role (e.g., "As a senior software engineer," "As a technical writer"), you guide Copilot to tailor its

responses to align with the expertise and perspective associated with that role. This improves the quality and relevance of the generated code and explanations.

Reference: GitHub Copilot documentation on prompt engineering and best practices.

Question #:23 - [Copilot Basics and Usage]

Which of the following is a risk associated with using AI?

A. AI algorithms are incapable of perpetuating existing biases.

B. AI systems can sometimes make decisions that are difficult to interpret.

C. AI eliminates the need for data privacy regulations.

D. AI replaces the need for developer opportunities in most fields.

**Answer: B**

## Explanation

A risk associated with AI is that its decisions can be difficult to interpret, leading to a lack of transparency and potential misunderstandings.

Reference: AI risk management documentation.

Question #:24 - [Knowledge Bases and Custom Models]

What is a benefit of using custom models in GitHub Copilot?

A. Responses are faster to produce and appear sooner

B. Responses use practices and patterns in your repositories

C. Responses use the organization's LLM engine

D. Responses are guaranteed to be correct

**Answer: B**

## Explanation

Custom models in GitHub Copilot allow the tool to learn from the specific code patterns and practices within your repositories. This results in suggestions that are more aligned with your organization's coding standards and conventions, improving the relevance and accuracy of the generated code.

Reference: GitHub Copilot Enterprise documentation on custom models.

Question #:25 - [Copilot Chat and Slash Commands]

What is one of the recommended practices when using GitHub Copilot Chat to enhance code quality?

A. Avoid using Copilot for complex tasks.

B. Disable Copilot's inline suggestions.

C. Regularly review and refactor the code suggested by Copilot.

D. Rely solely on Copilot's suggestions without reviewing them.

**Answer: C**

## Explanation

“Developers should always review, test, and refactor code generated by GitHub Copilot. Copilot can speed up development, but maintaining code quality requires careful human oversight.”

This establishes that regularly reviewing and refactoring Copilot suggestions is the recommended best practice.

References: GitHub Copilot responsible use guidelines.

==========

Question #:26 - [Testing and Quality]

How can GitHub Copilot be limited when it comes to suggesting unit tests?

A. GitHub Copilot can generate all types of unit tests, including those for edge cases and complex integration scenarios.

B. GitHub Copilot primarily suggests basic unit tests that focus on core functionalities, often requiring additional input from developers for comprehensive coverage.

C. GitHub Copilot can handle any complexity in code and automatically generate appropriate unit tests.

D. GitHub Copilot's limitations in generating unit tests can vary based on the IDE version you are using.

**Answer: B**

## Explanation

GitHub Copilot often suggests basic unit tests and may not cover all edge cases or complex integration scenarios, requiring developers to supplement its suggestions.

Reference: GitHub Copilot testing limitations.

Question #:27 - [Copilot Chat and Slash Commands]

When using GitHub Copilot Chat to generate boilerplate code for various test types, how can you guide the AI to follow the testing standards of your company?

A. By using a specific setting in GitHub Copilot's configuration.

B. By using a specific command in the terminal.

C. By using specific prompt examples in your chat request.

D. By using a specific slash command in the prompt.

**Answer: C**

## Explanation

“The quality of Copilot Chat responses depends on the clarity and specificity of your prompts. You can guide Copilot to follow organizational standards by including examples or requirements in your request.”

This means prompt engineering with specific examples is the way to align with company standards.

References: GitHub Copilot Chat best practices documentation.

==========

Question #:28 - [Copilot Basics and Usage]

What are the potential limitations of GitHub Copilot in maintaining existing codebases?

A. GitHub Copilot can independently manage and resolve all merge conflicts in version control.

B. GitHub Copilot might not fully understand the context and dependencies within a large codebase.

C. GitHub Copilot's suggestions are always aware of the entire codebase.

D. GitHub Copilot can refactor and optimize the entire codebase up to 10,000 lines of code.

**Answer: B**

## Explanation

“Copilot may not always understand the full context or complex dependencies in large codebases, and suggestions may require significant review.”

This proves that the main limitation is a lack of deep awareness of complex or large codebase contexts.

References: GitHub Copilot limitations documentation.

Question #:29 - [Copilot Basics and Usage]

What is the correct way to exclude specific files from being used by GitHub Copilot Business during code suggestions?

A. Modify the .gitignore file to include the specific files.

B. Add the specific files to a copilot.ignore file.

C. Use the GitHub Copilot settings in the user interface to exclude files.

D. Rename the files to include the suffix _no_copilot.

**Answer: C**

## Explanation

The correct way to exclude files is through the GitHub Copilot settings in the user interface, which allows administrators to specify files and directories to be ignored.

Reference: GitHub Copilot Business content exclusion documentation.

Question #:30 - [Copilot Chat and Slash Commands]

When using an IDE with a supported GitHub Copilot plug-in, which Chat features can be accessed from within the IDE? (Each correct answer presents part of the solution. Choose two.)

A. Explain code and suggest improvements

B. Generate unit tests

C. Plan coding tasks

D. Find out about releases and commits

**Answer: A B**

## Explanation

Exact extracts:

- “In supported IDEs, you can ask Copilot Chat to explain code or suggest improvements.”

References: GitHub Copilot Chat IDE integration documentation.

==========

Question #:31 - [Administration and Licensing]

Which GitHub Copilot plan could an Azure DevOps organization use without requiring a GitHub Enterprise license?

A. GitHub Copilot Individual

B. GitHub Copilot Enterprise

C. GitHub Copilot for Azure DevOps

D. Copilot Teams

**Answer: C**

## Explanation

"GitHub Copilot for Azure DevOps is available to Azure DevOps organizations without requiring a GitHub Enterprise license."

This confirms that Copilot for Azure DevOps is the correct plan for such organizations.

References: GitHub Copilot for Azure DevOps documentation.

==========

Question #:32 - [Administration and Licensing]

How does GitHub Copilot Enterprise assist in code reviews during the pull request process? (Select two.)

A. It automatically merges pull requests after an automated review.

B. It generates a prose summary and a bulleted list of key changes for pull requests.

C. It can validate the accuracy of the changes in the pull request.

D. It can answer questions about the changeset of the pull request.

**Answer: B D**

## Explanation

GitHub Copilot Enterprise assists in code reviews by generating summaries of pull requests and answering questions about the changes made.

Reference: GitHub Copilot Enterprise pull request review documentation.

Question #:33 - [Security, Privacy, and Compliance]

What is the main purpose of the duplication detection filter in GitHub Copilot?

A. To compare user-generated code against a private repository for potential matches.

B. To allow administrators to control which suggestions are visible to developers based on custom criteria.

C. To encourage the user to follow coding best practices preventing code duplication.

D. To detect and block suggestions that match public code snippets on GitHub if they contain about 150 characters.

**Answer: D**

## Explanation

"GitHub Copilot has a duplication detection filter that checks suggestions of about 150 characters or longer against public code on GitHub. If a match is found, the suggestion is blocked if you have configured blocking."

This clearly defines the purpose as preventing Copilot from suggesting verbatim code found publicly.

References: GitHub Copilot duplication detection documentation.

==========

Question #:34 - [Copilot Basics and Usage]

What is the impact of the "Fill-In-the-Middle" (FIM) technique on GitHub Copilot's code suggestions?

A. Restricts Copilot to use only external databases for generating code suggestions.

B. Allows Copilot to generate suggestions based only on the prefix of the code.

C. Ignores both the prefix and suffix of the code, focusing only on user comments for context.

D. Improves suggestions by considering both the prefix and suffix of the code, filling in the middle part more accurately.

**Answer: D**

## Explanation

"Fill-in-the-Middle (FIM) enables Copilot to consider both prefix and suffix code, generating more accurate suggestions for the missing middle portion."

This makes option D correct, as it explains how FIM enhances suggestion accuracy.

References: GitHub Copilot model training and FIM technique documentation.

Question #:35 - [Copilot Basics and Usage]

How do you generate code suggestions with GitHub Copilot in the CLI?

A. Type out the code snippet # Use the copilot refine command to enhance it # Review the suggested command.

B. Write code comments # Press the suggestion shortcut # Select the best suggestion from the list.

C. Use gh copilot suggest # Write the command you want # Select the best suggestion from the list.

D. Describe the project's architecture # Use the copilot generate command # Accept the generated suggestion.

**Answer: C**

## Explanation

“To generate a suggestion, run gh copilot suggest, provide your command description, then review and select from the suggestions returned.”

This confirms the CLI command flow in option C.

References: GitHub Copilot CLI documentation.

==========

Question #:36 - [Ethical AI Principles]

A social media manager wants to use AI to filter content. How can they promote transparency in the platform’s AI operations?

A. By regularly updating the AI filtering algorithm.

B. By relying on a well-regarded AI development company.

C. By focusing on user satisfaction with the content filtering.

D. By providing clear explanations about the types of content the AI is designed to filter and how it arrives at its conclusion.

**Answer: D**

## Explanation

Exact extracts:

- “Transparency. AI systems should be understandable.”

References: Microsoft Responsible AI guidelines and transparency notes.

==========

Question #:37 - [Copilot Basics and Usage]

What reasons could apply if code suggestions are not working in your editor? (Select three.)

A. You are working in files included in your .gitignore

B. You do not have an active internet connection

C. Your programming language is not supported

D. Your content exclusion is active and blocks the use of GitHub Copilot

E. You do not have a valid GitHub Copilot license

**Answer: B C E**

## Explanation

Exact extracts:

- "Copilot requires an active internet connection to provide suggestions."
- "Copilot does not support all programming languages."

References: GitHub Copilot troubleshooting documentation.

==========

Question #:38 - [Copilot Chat and Slash Commands]

Select a strategy to increase the performance of GitHub Copilot Chat.

A. Optimize the usage of memory-intensive operations within generated code

B. Apply prompt engineering techniques to be more specific

C. Use a single GitHub Copilot Chat query to find resolutions for the collection of technical requirements

D. Limit the number of concurrent users accessing GitHub Copilot Chat

**Answer: B**

## Explanation

"Performance of Copilot Chat can be improved with prompt engineering. More specific prompts give more relevant and accurate answers."

This confirms that applying prompt engineering techniques is the valid strategy.

References: GitHub Copilot Chat best practices documentation.

==========

Question #:39 - [Copilot Basics and Usage]

How does GitHub Copilot assist developers in reducing the amount of manual boilerplate code they write?

A. By engaging in real-time collaboration with multiple developers to write boilerplate code.

B. By predicting future coding requirements and pre-emptively generating boilerplate code.

C. By refactoring the entire codebase to eliminate boilerplate code without developer input.

D. By suggesting code snippets that can be reused across different parts of the project.

**Answer: D**

## Explanation

"Copilot helps reduce boilerplate by suggesting reusable code snippets for repetitive structures across your project."

This proves that Copilot saves time by generating reusable snippets, not by preemptively or automatically refactoring entire codebases.

References: GitHub Copilot productivity documentation.

Question #:40 - [IDE Integration and Features]

When using GitHub Copilot to identify missing tests in your codebase, which of the following is the most important factor to consider?

A. Using well-known coding practices in your repository.

B. Ensuring that the correct context is available to GitHub Copilot.

C. Close all the tabs in your IDE that do not have tests in them.

D. Having a high test coverage percentage in the codebase.

**Answer: B**

## Explanation

"The accuracy of Copilot's test generation depends heavily on the available context, including existing code, file structure, and open selections."

This confirms that providing the correct context is the most important factor.

References: GitHub Copilot testing documentation.

==========

Question #:41 - [Copilot Basics and Usage]

How does GitHub Copilot suggest code optimizations for improved performance?

A. By analyzing the codebase and suggesting more efficient algorithms or data structures.

B. By automatically rewriting the codebase to use more efficient code.

C. By enforcing strict coding standards that ensure optimal performance.

D. By providing detailed reports on the performance of the codebase.

**Answer: A**

## Explanation

GitHub Copilot suggests code optimizations by analyzing the codebase and recommending more efficient algorithms or data structures.

Reference: GitHub Copilot code optimization documentation.

Question #:42 - [Copilot Chat and Slash Commands]

How does GitHub Copilot Chat help in understanding the existing codebase?

A. By running code linters and formatters.

B. By providing visual diagrams of the code structure.

C. By answering questions about the code and generating explanations.

D. By automatically refactoring code to improve readability.

**Answer: C**

## Explanation

GitHub Copilot Chat helps in understanding existing codebases by answering questions about the code and generating explanations. This allows developers to quickly grasp the functionality and structure of unfamiliar code.

Reference: GitHub Copilot Chat documentation on code understanding.

Question #:43 - [Security, Privacy, and Compliance]

What content can be configured to be excluded with content exclusions? (Each correct answer presents part of the solution. Choose three.)

A. Files

B. Folders

C. Lines in files

D. Gists

E. Repositories

**Answer: A B E**

## Explanation

“Enterprise administrators can configure content exclusions to exclude entire repositories, folders, or specific files from being used by Copilot.”

This confirms that files, folders, and repositories can be excluded, but not individual lines or gists.

References: GitHub Copilot content exclusions documentation.

==========

Question #:44 - [Copilot Chat and Slash Commands]

What specific function does the /fix slash command perform?

A. Initiates a code review with static analysis tools for security and logic errors.

B. Converts pseudocode into executable code, optimizing for readability and maintainability.

C. Generates new code snippets based on language syntax and best practices.

D. Proposes changes for detected issues, suggesting corrections for syntax errors and programming mistakes.

**Answer: D**

## Explanation

"/fix: Propose changes to resolve detected issues, such as syntax errors or common programming mistakes, in the selected code."

This confirms that the /fix command is used to suggest corrections for identified problems.

References: GitHub Copilot Chat command reference.

==========

Question #:45 - [Administration and Licensing]

A company is currently storing code in Bitbucket and would like to use GitHub Copilot. Which GitHub Copilot plan will be most cost effective to allow them to manage users with their Identity Provider (e.g. Okta)?

A. GitHub Copilot Business for non-GHE customers

B. GitHub Copilot Individual

C. GitHub Copilot Enterprise

D. GitHub Copilot Teams

**Answer: C**

## Explanation

GitHub Copilot Enterprise is the most cost-effective plan for managing users with an Identity Provider like Okta, as it provides enterprise-level features and integration.

Reference: GitHub Copilot pricing and features.

Question #:46 - [Knowledge Bases and Custom Models]

Which of the following is correct about GitHub Copilot Knowledge Bases?

A. All repos are indexed

B. Indexing is static

C. It is an Enterprise feature

D. All file types are indexed

**Answer: C**

## Explanation

GitHub Copilot Knowledge Bases is an Enterprise feature that allows the tool to use an organization's internal documentation and code for more accurate and relevant suggestions.

Reference: GitHub Copilot Enterprise Knowledge Base documentation.

Question #:47 - [Copilot Chat and Slash Commands]

What is a key consideration when relying on GitHub Copilot Chat's explanations of code functionality and proposed improvements?

A. The explanations are dynamically updated based on user feedback.

B. Reviewing and validating the generated output for accuracy and completeness.

C. GitHub Copilot Chat uses a static database for generating explanations.

D. The explanations are primarily derived from user-provided documentation.

**Answer: B**

## Explanation

While GitHub Copilot Chat can provide helpful explanations and suggestions, it's crucial to review and validate the generated output. Copilot's suggestions are based on its training data, and they may not always be perfectly accurate or complete. Human judgment is essential to ensure the quality and correctness of the code.

Reference: GitHub Copilot best practices and usage guidelines.

Question #:48 - [Security, Privacy, and Compliance]

What GitHub Copilot configuration needs to be enabled to protect against IP infringements?

A. Blocking public code matches

B. Blocking license check configuration

C. Allowing public code matches

D. Allowing license check configuration

**Answer: A**

## Explanation

“By default, suggestions matching public code are allowed, but you can choose to block public code matches. This setting can be enabled to reduce the risk of intellectual property infringement.”

This shows that enabling “Blocking public code matches” is the configuration required to protect against IP issues.

References: GitHub Copilot configuration settings documentation.

==========

Question #:49 - [Administration and Licensing]

How long does GitHub retain Copilot data for Business and Enterprise? (Each correct answer presents part of the solution. Choose two.)

A. Prompts and Suggestions: Not retained

B. Prompts and Suggestions: Retained for 28 days

C. User Engagement Data: Kept for Two Years

D. User Engagement Data: Kept for One Year

**Answer: B C**

## Explanation

For GitHub Copilot Business and Enterprise, prompts and suggestions are retained for 28 days to provide context and improve the service. User engagement data, which includes usage patterns and interactions, is kept for two years. This data retention policy is designed to balance service improvement with user privacy.

Reference: GitHub Copilot documentation on data privacy and retention policies for Business andEnterprise plans.

Question #:50 - [Copilot Chat and Slash Commands]

What is the best way to share feedback about GitHub Copilot Chat when using it on GitHub Mobile?

A. The Settings menu in the GitHub Mobile app.

B. The feedback section on the GitHub website.

C. Use the emojis in the Copilot Chat interface.

D. By tweeting at GitHub's official X (previously known as Twitter) account.

**Answer: C**

## Explanation

The best way to share feedback on GitHub Mobile is using the emojis in the Copilot Chat interface, which allows for direct and immediate feedback.

Reference: GitHub Copilot Mobile documentation.

Question #:51 - [Copilot Chat and Slash Commands]

Which scenarios can GitHub Copilot Chat be used to increase productivity? (Each correct answer presents part of the solution. Choose two.)

A. A project plan for the team needs to be generated using a project management software.

B. Create a documentation file for the newly created code base.

C. A developer is added to a new project and would like to understand the current software code.

D. Fast tracking of release management activities to move code to production main branch.

**Answer: B C**

## Explanation

Exact extracts:

- "Copilot Chat can generate documentation based on your code."

References: GitHub Copilot Chat productivity documentation.

==========

Question #:52 - [Prompt Engineering and Best Practices]

How can you improve the context used by GitHub Copilot? (Each correct answer presents part of the solution. Choose two.)

A. By opening the relevant tabs in your IDE

B. By adding relevant code snippets to your prompt

C. By adding the important files to your .gitconfig

D. By adding the full file paths to your prompt of important files

**Answer: A B**

## Explanation

Improving the context for GitHub Copilot involves opening relevant files in your IDE to provide immediate context and adding relevant code snippets directly to your prompts to give Copilot specific examples and information.

Reference: GitHub Copilot prompt engineering and context management.

Question #:53 - [Administration and Licensing]

An independent contractor develops applications for a variety of different customers. Assuming no concerns from their customers, which GitHub Copilot plan is best suited?

A. GitHub Copilot Individual

B. GitHub Copilot Business

C. GitHub Copilot Business for non-GHE Customers

D. GitHub Copilot Enterprise

E. GitHub Copilot Teams

**Answer: A**

## Explanation

For an independent contractor, GitHub Copilot Individual is the most suitable and cost-effective plan.

Reference: GitHub Copilot pricing documentation.

Question #:54 - [Copilot Chat and Slash Commands]

How does the /tests slash command assist developers?

A. Constructs detailed test documentation.

B. Creates unit tests for the selected code.

C. Integrates with external testing frameworks.

D. Executes test cases to find issues with the code.

**Answer: B**

## Explanation

The /tests slash command in GitHub Copilot Chat creates unit tests for the selected code, helping developers ensure the functionality and reliability of their code.

Reference: GitHub Copilot Chat command documentation.

Question #:55 - [Copilot Basics and Usage]

What are the additional checks that need to pass before the GitHub Copilot responses are submitted to the user? (Each correct answer presents part of the solution. Choose two.)

A. Code quality

B. Compatibility with user-specific settings

C. Suggestions matching public code (optional based on settings)

D. Performance benchmarking

**Answer: A C**

## Explanation

Exact extracts:

- "Copilot applies post-processing checks for code quality to ensure that responses meet baseline standards."

References: GitHub Copilot safeguards documentation.

==========

Question #:56 - [Copilot Basics and Usage]

How can GitHub Copilot assist developers during the requirements analysis phase of the Software Development Life Cycle (SDLC)?

A. By automatically generating detailed requirements documents.

B. By providing templates and code snippets that help in documenting requirements.

C. By identifying and fixing potential requirement conflicts when using /help.

D. By managing stakeholder communication and meetings.

**Answer: B**

## Explanation

GitHub Copilot can assist during the requirements analysis phase by providing templates and code snippets that aid in documenting requirements. This helps streamline the process of capturing and organizing project requirements.

Reference: GitHub Copilot documentation on SDLC assistance.

Question #:57 - [Copilot Chat and Slash Commands]

How does GitHub Copilot utilize chat history to enhance its code completion capabilities?

A. By using chat history to offer personalized code snippets based on previous prompts.

B. By logging chat history to monitor user activity and ensure compliance with coding standards.

C. By analyzing past chat interactions to identify common programming patterns and errors.

D. By sharing chat history with third-party services to improve integration and functionality.

**Answer: C**

## Explanation

"GitHub Copilot Chat can reference the ongoing conversation history to analyze patterns, common coding styles, and errors, providing more accurate and contextual suggestions."

This demonstrates that chat history is used to identify patterns and enhance contextual accuracy, making option C correct.

References: GitHub Copilot Chat documentation.

Question #:58 - [Copilot Basics and Usage]

Why might a Generative AI (Gen AI) tool create inaccurate outputs?

A. The Gen AI tool is overloaded with too many requests at once.

B. The Gen AI tool is experiencing downtime and is not fully recovered.

C. The Gen AI tool is programmed with a focus on creativity over factual accuracy.

D. The training data might contain biases or inconsistencies.

**Answer: D**

## Explanation

Gen AI tools can produce inaccurate outputs if the training data contains biases or inconsistencies, which can lead to flawed or misleading results.

Reference: Generative AI limitations documentation.

Question #:59 - [Copilot Chat and Slash Commands]

How does GitHub Copilot Chat ensure that a function works correctly?

A. By suggesting assertions based on the code's context and semantics.

B. By automatically writing all the tests for the function.

C. By writing the implementation code for the function.

D. By executing the test cases to validate the correctness of the code.

**Answer: A**

## Explanation

GitHub Copilot Chat can suggest assertions based on the code's context and semantics to help developers verify the correctness of their functions. These assertions serve as checks that the function behaves as expected under various conditions.

Reference: GitHub Copilot documentation on testing and code verification.

Question #:60 - [Copilot Chat and Slash Commands]

What is the primary role of the /optimize slash command in Visual Studio?

A. Translates code into a more performant language.

B. Enhances the performance of the selected code by analyzing its runtime complexity.

C. Automatically formats the code according to the selected style guide.

D. Summarizes your documentation into more maintainable and readable formats.

**Answer: B**

## Explanation

“/optimize Analyze and improve running time of the selected code.” This statement from Visual Studio Copilot Chat slash command documentation shows that the role of /optimize is specifically to analyze and improve the runtime performance of selected code, aligning directly with option B.

References: GitHub Copilot official documentation and Visual Studio Copilot Chat command reference.

==========

Question #:61 - [Copilot Chat and Slash Commands]

What role does the pre-processing of user input play in the data flow of GitHub Copilot Chat?

A. It filters out irrelevant information from the user's input prompt.

B. It enriches the input prompt with additional context before passing it to the language model.

C. It directly generates a response based on the user's input prompt.

D. It formats the output response before presenting it to the user.

**Answer: B**

## Explanation

“Before being sent to the model, user input is pre-processed to enrich the prompt with contextual information such as open files, code selections, or previous conversation history.”

This shows that pre-processing is used to enrich input with context, making option B correct.

References: GitHub Copilot Chat architecture documentation.

==========

Question #:62 - [Security, Privacy, and Compliance]

How does GitHub Copilot identify matching code and ensure that public code is appropriately handled or blocked? (Each correct answer presents part of the solution. Choose two.)

A. Using machine learning models trained only on private repositories

B. Reviewing and storing user-specific private repository data for future suggestions

C. Filtering out suggestions that match code from public repositories

D. Implementing safeguards to detect and avoid suggesting verbatim snippets from public code

**Answer: C D**

## Explanation

Exact extracts:

- “GitHub Copilot has a filter to detect code suggestions that match public code on GitHub. When this filter is enabled, Copilot checks suggestions with about 150 characters or more against public code and blocks them if a match is found.”

References: GitHub Copilot duplication detection and safeguards documentation.

==========

Question #:63 - [Security, Privacy, and Compliance]

What is the process behind identifying public code matches when using a public code filter enabled in GitHub Copilot?

A. Running code suggestions through filters designed to detect public code

B. Comparing suggestions against public code using machine learning.

C. Analyzing the context and structure of the code being written

D. Reviewing the user's browsing history to identify public repositories

**Answer: A**

## Explanation

When the public code filter is enabled, GitHub Copilot runs code suggestions through filters designed to detect matches with publicly available code. This helps prevent the generation of code that might infringe on copyright or licensing agreements.

Reference: GitHub Copilot documentation on public code filtering and licensing.

Question #:64 - [Administration and Licensing]

How is GitHub Copilot Individual billed? (Each correct answer presents part of the solution. Choose two.)

A. Monthly as a subscription

B. Annually as a subscription

C. Monthly, as a metered service based on actual consumption

D. Free (not billed) for all open source projects

**Answer: A B**

## Explanation

GitHub Copilot Individual is billed as a monthly or annual subscription.

Reference: GitHub Copilot Individual pricing.

Question #:65 - [Copilot Chat and Slash Commands]

When using GitHub Copilot Chat to generate unit tests, which slash command would you use?

A. /init-tests

B. /create-tests

C. /generate-tests

D. /tests

**Answer: D**

## Explanation

"/tests: Generate unit tests for the selected code or function."

This is the official description of the /tests slash command, confirming it is the command used to create unit tests.

References: GitHub Copilot Chat slash command reference.

==========

Question #:66 - [Copilot Chat and Slash Commands]

How does GitHub Copilot Chat utilize its training data and external sources to generate responses when answering coding questions?

A. It primarily relies on the model's training data to generate responses.

B. It primarily uses search results from Bing to generate responses.

C. It combines its training data set, code in user repositories, and external sources like Bing to generate responses.

D. It uses user-provided documentation exclusively to generate responses.

**Answer: C**

## Explanation

GitHub Copilot Chat combines its training data, code from user repositories, and external sources like Bing to generate comprehensive and relevant responses to coding questions.

Reference: GitHub Copilot Chat documentation on data sources.

Question #:67 - [Copilot Basics and Usage]

Identify the steps involved in the life cycle of a GitHub Copilot code suggestion? (Each correct answer presents part of the solution. Choose two.)

A. Generate suggestions

B. Capturing the user's context

C. Processing telemetry data

D. Retraining the model

E. Storing user data

**Answer: A B**

## Explanation

Exact extracts:

- "Copilot first captures the context of the developer's code and environment."

References: GitHub Copilot technical overview documentation.

==========

Question #:68 - [Testing and Quality]

How can GitHub Copilot assist in maintaining consistency across your tests?

A. By identifying a pattern in the way you write tests and suggesting similar patterns for future tests.

B. By automatically fixing all tests in the code based on the context.

C. By providing documentation references based on industry best practices.

D. By writing the implementation code for the function based on context.

**Answer: A**

## Explanation

"Copilot learns from the patterns in your existing tests and suggests similar structures, which helps maintain consistency across the test suite."

This confirms that Copilot supports test consistency through pattern recognition and suggestion.

References: GitHub Copilot testing documentation.

Question #:69 - [Copilot Chat and Slash Commands]

Identify the right use cases where GitHub Copilot Chat is most effective. (Each correct answer presents part of the solution. Choose two.)

A. Create a technical requirement specification from the business requirement documentation

B. Explain a legacy COBOL code and translate the code to another language like Python.

C. Creation of a unit test scenario for newly developed Python code

D. Creation of end-to-end performance testing scenarios for a web application

**Answer: B C**

## Explanation

GitHub Copilot Chat is effective for explaining and translating legacy code and generating unit test scenarios for new code.

Reference: GitHub Copilot Chat use cases.

Question #:70 - [Knowledge Bases and Custom Models]

What GitHub Copilot pricing plan gives you access to your company's knowledge bases?

A. GitHub Copilot Individual

B. GitHub Copilot Business

C. GitHub Copilot Enterprise

D. GitHub Copilot Professional

**Answer: C**

## Explanation

GitHub Copilot Enterprise provides access to your company's knowledge bases, enabling the tool to provide contextually relevant suggestions based on your organization's specific documentation and code.

Reference: GitHub Copilot Enterprise pricing and features.

Question #:71 - [Administration and Licensing]

What is the primary purpose of organization audit logs in GitHub Copilot Business?

A. To track the number of lines of code suggested by Copilot

B. To assign software licenses within the organization

C. To monitor code conflicts across repositories

D. To monitor administrator activities and actions within the organization

**Answer: D**

## Explanation

“Audit logs in Copilot Business allow enterprise and organization administrators to monitor administrator activities and configuration changes.”

This proves that their purpose is not line tracking or license assignment but monitoring admin activities.

References: GitHub Copilot Business administration documentation.

==========

Question #:72 - [Security, Privacy, and Compliance]

What is a limitation of content exclusions?

A. Repository administrators and organization owners cannot manage content exclusion settings.

B. Content exclusions can be worked around as it is only available for Git repositories.

C. Content exclusions can only be configured by an enterprise administrator.

D. Content exclusions are only available in the GitHub Copilot Individual plan.

**Answer: C**

## Explanation

“Content exclusions can only be configured at the enterprise level by an enterprise administrator. Repository or organization admins cannot configure them directly.”

This confirms that configuration is limited to enterprise administrators, making option C correct.

References: GitHub Copilot content exclusions configuration documentation.

Question #:73 - [Refactoring and Code Modernization]

How can you use GitHub Copilot to get inline suggestions for refactoring your code? (Select two.)

A. By adding comments to your code and triggering a suggestion.

B. By highlighting the code you want to fix, right-clicking, and selecting "Fix using GitHub Copilot."

C. By running the gh copilot fix command.

D. By using the "/fix" command in GitHub Copilot in-line chat.

E. By highlighting the code you want to fix, right-clicking, and selecting "Refactor using GitHub Copilot."

**Answer: A E**

## Explanation

You can use GitHub Copilot for inline refactoring suggestions by adding comments to your code to trigger suggestions and by highlighting the code and selecting "Refactor using GitHub Copilot" from the context menu.

Reference: GitHub Copilot refactoring documentation.

Question #:74 - [Copilot Basics and Usage]

In what ways can GitHub Copilot contribute to the design phase of the Software Development Life Cycle (SDLC)?

A. GitHub Copilot can independently create a complete software design.

B. GitHub Copilot can suggest design patterns and best practices relevant to the project.

C. GitHub Copilot can manage design team collaboration and version control.

D. GitHub Copilot can generate user interface (UI) prototypes without prompting.

**Answer: B**

## Explanation

“Copilot can assist in the design phase by suggesting design patterns, frameworks, and best practices relevant to the context of your project.”

This shows Copilot contributes by offering design-related recommendations, not by independently producing full designs or managing collaboration.

References: GitHub Copilot use case documentation.

==========

Question #:75 - [Data Sources and Context]

What are the different ways to give context to GitHub Copilot to get more precise responses? (Each correct answer presents part of the solution. Choose two.)

A. Utilize to interpret developer's thoughts and intentions without any code or comments.

B. Engage with chat participants such as @workspace to incorporate collaborative context into the responses.

C. Access developer's previous projects and code repositories to understand their coding style without explicit permission.

D. Utilize chat variables like *file to anchor the conversation within the specific context of the files or editors in use.

**Answer: B D**

## Explanation

Exact extracts:

- "You can give Copilot Chat additional context using variables such as @workspace or @file to scope the conversation."

References: GitHub Copilot Chat variables and context documentation.

==========

Question #:76 - [Copilot Basics and Usage]

Which of the following statements best describes the impact of GitHub Copilot on the software development process?

A. It decreases software vulnerabilities from third party dependencies.

B. It reduces overhead by automating testing workflows.

C. It increases productivity by automating repetitive coding tasks.

D. It replaces the need for developers in the software development process.

**Answer: C**

## Explanation

GitHub Copilot primarily impacts the software development process by increasing productivity through automating repetitive coding tasks.

Reference: GitHub Copilot impact documentation.

Question #:77 - [Copilot Basics and Usage]

What caution should developers exercise when using GitHub Copilot for assistance with mathematical computations?

A. GitHub Copilot's capability to optimize complex mathematical algorithms beyond manual coding.

B. GitHub Copilot's ability to execute and verify mathematical results in real-time.

C. GitHub Copilot's reliance on pattern-based responses without verifying computation accuracy.

D. GitHub Copilot's automatic update of outdated mathematical formulas to modern standards.

**Answer: C**

## Explanation

“Copilot’s responses are generated based on patterns in training data and are not guaranteed to be mathematically accurate. Developers must verify calculations independently.”

This shows that Copilot cannot guarantee correctness in math and relies on patterns, making option C correct.

References: GitHub Copilot responsible use documentation.

==========

Question #:78 - [Administration and Licensing]

Which of the following GitHub Copilot Business related activities can be tracked using the organization audit logs?

A. Accepted chat suggestions

B. Code suggestions made by GitHub Copilot

C. Changes to content exclusion settings

D. Suggestions blocked by duplication detection filtering

**Answer: C**

## Explanation

Organization audit logs track changes to content exclusion settings, providing administrators with visibility into configuration changes.

Reference: GitHub Copilot Business audit logs.

Question #:79 - [Testing and Quality]

Why is code reviewing still necessary when using GitHub Copilot to write tests?

A. Because GitHub Copilot can cover all possible scenarios in your test cases.

B. Because GitHub Copilot generates the best code possible for the test scenario.

C. Because GitHub Copilot's generated test cases may not cover all possible scenarios.

D. Because GitHub Copilot replaces the need for manual testing.

**Answer: C**

## Explanation

Code review is necessary because GitHub Copilot's generated test cases might not cover all possible scenarios, especially edge cases and complex interactions.

Reference: GitHub Copilot testing best practices.

Question #:80 - [Security, Privacy, and Compliance]

What configuration needs to be set to get help from Microsoft and GitHub protecting against IP infringement while using GitHub Copilot?

A. Suggestions matching public code to 'blocked'

B. Enforce blocking of MIT or GPL licensed code

C. You need to check code suggestions yourself before accepting

D. Enable GitHub Copilot license checking

**Answer: A**

## Explanation

To help protect against IP infringement, you need to configure GitHub Copilot to block suggestions that match public code. This ensures that the generated code is not directly copied from publicly available sources.

Reference: GitHub Copilot documentation on IP protection and code filtering.

Question #:81 - [Copilot Chat and Slash Commands]

In what way can GitHub Copilot and GitHub Copilot Chat aid developers in modernizing applications?

A. GitHub Copilot can directly convert legacy applications into cloud-native architectures.

B. GitHub Copilot can suggest modern programming patterns based on your code.

C. GitHub Copilot can create and deploy full-stack applications based on a single query.

D. GitHub Copilot can refactor applications to align with upcoming standards.

**Answer: B**

## Explanation

GitHub Copilot and GitHub Copilot Chat are powerful AI-driven tools designed to assist developers by providing context-aware code suggestions and interactive support. Specifically, in the context of modernizing applications, GitHub Copilot excels at analyzing existing code and suggesting modern programming patterns, best practices, and syntax improvements that align with contemporary development standards. For example, it can recommend updates to outdated constructs, propose more efficient algorithms, or suggest frameworks and libraries that are widely used in modern application development.

- Why not A?GitHub Copilot does not "directly convert" legacy applications into cloud-native architectures. It can assist by suggesting code changes or patterns that support such a transition, but it doesn't autonomously perform the full conversion process, which involves architectural decisions and deployment steps beyond its scope.
- Why not C?While GitHub Copilot can generate code snippets and even larger portions of an application, it cannot create and deploy full-stack applications from a single query. It requires developer input, refinement, and integration to achieve a complete, deployable solution.
- Why not D?GitHub Copilot can assist with refactoring by suggesting improvements to existing code, but it doesn't inherently "align with upcoming standards" in a predictive sense. Its suggestions are based on current best practices and the data it was trained on, not future standards that are yet to be defined.

Thus,Bis the most accurate and realistic way GitHub Copilot aids developers in modernizing applications, leveraging its ability to provide relevant, context-based suggestions to update and improve codebases.

Reference: GitHub Copilot documentation on application modernization.

Question #:82 - [Copilot Basics and Usage]

1.

blog.yatricloud.com

blog.yatricloud.com

A. The API can generate detailed reports on code quality improvements made by GitHub Copilot.

B. The API can track the acceptance rate of code suggestions accepted and used in the organization.

C. The API can refactor your code to improve productivity.

D. The API can provide feedback on coding style and standards compliance.

E. The API can provide Copilot Chat specific suggestions acceptance metrics.

**Answer: B E**

## Explanation

The GitHub Copilot usage metrics API provides insights into the acceptance rate of code suggestions and Copilot Chat specific suggestions acceptance metrics, helping organizations evaluate its effectiveness.

Reference: GitHub Copilot usage metrics API documentation.

Question #:83 - [Copilot Basics and Usage]

How can GitHub Copilot aid developers in writing documentation for their code?

A. GitHub Copilot cannot assist in writing documentation or comments.

B. GitHub Copilot can automatically generate complete and detailed documentation.

C. GitHub Copilot can suggest summaries or descriptions based on the code's functionality.

D. GitHub Copilot can only generate content in markdown format.

**Answer: C**

## Explanation

“Copilot can suggest comments and documentation summaries that describe the functionality of the code being written.”

This makes option C correct, as Copilot provides summaries or descriptions rather than full automatic documentation.

References: GitHub Copilot documentation features.

==========

Question #:84 - [Copilot Chat and Slash Commands]

What are the potential limitations of GitHub Copilot Chat? (Each correct answer presents part of the solution. Choose two.)

A. Limited training data

B. No biases in code suggestions

C. Ability to handle complex code structures

D. Extensive support for all programming languages

**Answer: A C**

## Explanation

Exact extracts:

- "GitHub Copilot Chat, like all AI models, is limited by its training data and may not always provide accurate or up-to-date answers."

References: GitHub Copilot Chat limitations documentation.

==========

Question #:85 - [Prompt Engineering and Best Practices]

What is few-shot prompting?

A. Telling GitHub Copilot about the mechanism you want it to use and how to incorporate that into the response

B. Telling GitHub Copilot from which sources it should base the response on

C. Telling GitHub Copilot to try multiple times to answer the prompt

D. Telling GitHub Copilot to iterate several times on the answer before returning it to you

**Answer: A**

## Explanation

"Few-shot prompting is a technique where you provide examples of the mechanism or output you want the model to follow. These examples guide the AI's responses."

This proves that option A is correct.

References: GitHub Copilot prompt engineering documentation.

==========

Question #:86 - [Copilot Basics and Usage]

What method can be used to interact with GitHub Copilot?

A. By using a properly configured GitHub CLI

B. By using chat capabilities in NeoVim

C. From a watch window in an IDE debug session

D. From a web browser at https://github.copilot.com

**Answer: B**

## Explanation

GitHub Copilot is an AI-powered code completion tool that integrates directly into supported Integrated Development Environments (IDEs) and code editors, such as Visual Studio Code, JetBrains IDEs, NeoVim, and others. Developers interact with it through their coding environment, where it provides real-time code suggestions, autocompletions, and (in some cases) chat-like capabilities via extensions or plugins (e.g., GitHub Copilot Chat in supported editors).

Evaluation of Options:

- A. By using a properly configured GitHub CLIThe GitHub CLI (Command Line Interface) is a tool for interacting with GitHub repositories and workflows from the terminal, but it is not a method for interacting with GitHub Copilot. Copilot operates within code editors/IDEs, not through the CLI. Incorrect.
- B. By using chat capabilities in NeoVimThis is partially correct. GitHub Copilot can be used in NeoVim with the appropriate plugin (e.g., the Copilot.vim plugin), and GitHub Copilot Chat—a feature that allows conversational interaction—may also be available depending on the setup and version. However, "chat capabilities in NeoVim" alone is not the primary or standard way to describe Copilot interaction, as it's more about code suggestions than chat. This is the closest option but not perfectly precise.Partially correct.
- C. From a watch window in an IDE debug sessionThe "watch window" in an IDE is used during debugging to monitor variable values, not to interact with GitHub Copilot. Copilot provides suggestions while coding, not specifically in debug sessions or watch windows.Incorrect.
- D. From a web browser athttps://github.copilot.comThere is no such URL as "https://github.copilot.com" dedicated to interacting with GitHub Copilot. Copilot is accessed via GitHub's authentication and integrated into editors/IDEs, not through a standalone web browser interface. Information about Copilot is available on GitHub's official site (e.g.,https://github.com/features/copilot), but interaction happens in the coding environment.Incorrect.

Reference: GitHub Copilot interaction documentation.

Question #:87 - [Copilot Basics and Usage]

Where is the proxy service hosted?

A. Self hosted

B. Amazon Web Service

C. Microsoft Azure

D. Google Cloud Platform

**Answer: C**

## Explanation

The proxy service for GitHub Copilot is hosted on Microsoft Azure.

Reference: GitHub Copilot infrastructure and hosting information.

Question #:88 - [Prompt Engineering and Best Practices]

What are two techniques that can be used to improve prompts to GitHub Copilot? (Select two.)

A. Provide specific success criteria

B. Provide all information about the utilized files

C. Provide insight on where to get the content from to get a response

D. Provide links to supporting documentation

**Answer: A D**

## Explanation

Improving prompts involves providing specific success criteria and including links to supporting documentation to give GitHub Copilot more context and direction.

Reference: GitHub Copilot prompt engineering best practices.

Question #:89 - [Copilot Basics and Usage]

A team is using GitHub Copilot Individual in their daily development activities. They need to exclude specific files from being used to inform code completion suggestions. How can they achieve this?

A. Have an organization owner configure content exclusions

B. Add a .gitignore file to the repo

C. Have a repo administrator configure content exclusions

D. Use the #file Chat variable to exclude the files

E. Upgrade to Copilot Business

**Answer: E**

## Explanation

“Content exclusions are available only with GitHub Copilot Business and Enterprise. Copilot Individual users cannot configure exclusions.”

This confirms that the only way for the team to achieve exclusions is by upgrading to Copilot Business.

References: GitHub Copilot content exclusions documentation.

==========

Question #:90 - [Data Sources and Context]

Which of the following does GitHub Copilot's LLM derive context from when producing a response?

A. Frequency of commits to the repository

B. Syntax highlighting scheme of the code in the IDE

C. Neighboring or related files within a project

D. Version control system integrated with the IDE

**Answer: C**

## Explanation

“Copilot may use context from neighboring or related files in the project to improve the accuracy of its suggestions.”

This confirms that context is enriched with information from related files, making option C correct.

References: GitHub Copilot context derivation documentation.

==========

Question #:91 - [Refactoring and Code Modernization]

In what ways can GitHub Copilot support a developer during the code refactoring process? (Each correct answer presents part of the solution. Choose two.)

A. By offering code transformation examples that enhance performance and reduce complexity.

B. By independently ensuring compliance with regulatory standards across industries.

C. By providing suggestions for improving code readability and maintainability based on best practices.

D. By autonomously refactoring entire codebases to the latest programming language.

**Answer: A C**

## Explanation

Exact extracts:

- “Copilot can propose code transformations that improve performance and reduce complexity.”

References: GitHub Copilot code refactoring documentation.

==========

Question #:92 - [Administration and Licensing]

What is the correct way to access the audit log events for GitHub Copilot Business?

A. Navigate to the Security tab in the organization's GitHub settings

B. Navigate to the Insights tab in the repository settings

C. Use the Audit log section in the organization's GitHub settings

D. Use the Code tab in the GitHub repository

**Answer: C**

## Explanation

Audit log events for GitHub Copilot Business can be accessed through the Audit log section within the organization's GitHub settings. This log provides a record of activities related to Copilot usage and configuration.

Reference: GitHub Copilot Business documentation on audit logs.

Question #:93 - [Copilot Basics and Usage]

What method can a developer use to generate sample data with GitHub Copilot? (Each correct answer presents part of the solution. Choose two.)

A. Utilizing GitHub Copilot's ability to create fictitious information from patterns in training data.

B. Leveraging GitHub Copilot's ability to independently initiate and manage data storage services.

C. Utilize GitHub Copilot's capability to directly access and use databases to create sample data.

D. Leveraging GitHub Copilot's suggestions to create data based on API documentation in the repository.

**Answer: A D**

## Explanation

GitHub Copilot can generate sample data by creating fictitious information based on patterns in its training data and by using suggestions based on API documentation within the repository.

Reference: GitHub Copilot documentation on data generation assistance.

Question #:94 - [Administration and Licensing]

How can the insights gained from the metrics API be used to improve the development process in conjunction with GitHub Copilot?

A. Real-time debugging and error resolution statistics.

B. Insights on the types of coding languages where GitHub Copilot is most helpful.

C. Automated generation of complete project documentation.

D. Detailed analysis of GitHub Copilot's suggestions vs. manual coding.

**Answer: D**

## Explanation

“The Copilot metrics API provides insights into how suggestions are being accepted and used, enabling teams to analyze the relative impact of Copilot-generated code versus manually written code.”

This confirms option D is correct, as the API enables analysis of Copilot versus manual coding.

References: GitHub Copilot metrics API documentation.

==========

Question #:95 - [Security, Privacy, and Compliance]

Why is it important to ensure the security of the code used in Generative AI (Gen AI) tools?

A. Ensuring code security prevents unauthorized access and potential data breaches.

B. Ensuring code security supports the development of more advanced AI features.

C. Ensuring code security enables the AI system to handle larger datasets effectively.

D. Ensuring code security maintains the integrity of the AI system.

**Answer: A**

## Explanation

"Securing the code used with generative AI prevents unauthorized access, helps safeguard sensitive data, and protects against potential breaches."

This confirms option A is correct because the primary importance lies in avoiding unauthorized access and breaches.

References: GitHub Copilot responsible use and AI security documentation.

==========

Question #:96 - [Prompt Engineering and Best Practices]

What is zero-shot prompting?

A. Only giving GitHub Copilot a question as a prompt and no examples

B. Giving GitHub Copilot examples of the problem you want to solve

C. Telling GitHub Copilot it needs to show only the correct answer

D. Giving GitHub Copilot examples of the algorithm and outcome you want to use

E. Giving as little context to GitHub Copilot as possible

**Answer: A**

## Explanation

"Zero-shot prompting means asking the model to perform a task without providing examples—only the question or task description is given."

This confirms that the correct definition of zero-shot prompting is providing no examples, only a direct prompt.

References: GitHub Copilot and AI prompting documentation.

==========

Question #:97 - [Prompt Engineering and Best Practices]

What types of prompts or code snippets might be flagged by the GitHub Copilot toxicity filter? (Each correct answer presents part of the solution. Choose two.)

A. Hate speech or discriminatory language (e.g., racial slurs, offensive stereotypes)

B. Sexually suggestive or explicit content

C. Code that contains logical errors or produces unexpected results

D. Code comments containing strong opinions or criticisms

**Answer: A B**

## Explanation

GitHub Copilot includes a toxicity filter to prevent the generation of harmful or inappropriate content. This filter flags prompts or code snippets that contain hate speech, discriminatory language, or sexually suggestive or explicit content. This ensures a safe and respectful coding environment.

Reference: GitHub Copilot documentation on safety and content filtering.

Question #:98 - [Administration and Licensing]

Which Copilot Enterprise features are available in all commercially supported IDEs?

A. Inline suggestions

B. Pull request summaries

C. Knowledge bases

D. Chat

**Answer: A D**

## Explanation

“Copilot Enterprise provides inline code suggestions and Copilot Chat in all commercially supported IDEs. Additional features, such as pull request summaries and knowledge bases, are available in GitHub.com.”

This confirms that inline suggestions and Chat are the features consistently available in IDEs.

References: GitHub Copilot Enterprise feature documentation.

==========

Question #:99 - [Copilot Basics and Usage]

Which of the following are true about code suggestions? (Each correct answer presents part of the solution. Choose two.)

A. Code suggestions are guaranteed to not expose known security vulnerabilities

B. You can use keyboard shortcuts to accept the next word in a suggestion

C. Code suggestions are limited to single-line suggestions

D. Code suggestions will always compile or run without modifications

E. Alternative code suggestions can be shown in a new tab

**Answer: B E**

## Explanation

Exact extracts:

- "You can accept individual words, lines, or entire suggestions with keyboard shortcuts."

References: GitHub Copilot editor usage documentation.

==========

Question #:100 - [Administration and Licensing]

Which REST API endpoint is used to modify details about a GitHub Copilot Business subscription? (Each correct answer presents part of the solution. Choose two.)

A. Add teams to the Copilot subscription for an organization

B. Upgrade or downgrade the subscription tier

C. Migrate Copilot seat assignments between GitHub organizations

D. Reassign Copilot seats based on GitHub repository size

E. Remove teams from the Copilot subscription for an organization

**Answer: A E**

## Explanation

The REST API endpoints are used to add and remove teams from the Copilot Business subscription within an organization.

Reference: GitHub Copilot Business API documentation.

Question #:101 - [Prompt Engineering and Best Practices]

Which of the following statements correctly describes how GitHub Copilot Individual uses prompt data? (Each correct answer presents part of the solution. Choose two.)

A. Prompt data is stored unencrypted for faster processing.

B. Prompt data is used internally by GitHub for improving the search engine.

C. Prompt data is used to train machine learning models for better code suggestions.

D. Real-time user input helps generate context-aware code suggestions.

**Answer: C D**

## Explanation

Exact extracts:

- "For GitHub Copilot Individual, prompts and code completions may be retained and used to train machine learning models to improve Copilot."

References: GitHub Copilot data usage and privacy documentation.

==========

Question #:102 - [Prompt Engineering and Best Practices]

Which of the following prompts can be used to guide GitHub Copilot Chat in refactoring code for quality improvements? (Each correct answer presents part of the solution. Choose two.)

A. "Show me how to improve the readability of this function."

B. "Suggest ways to enhance the maintainability of this code segment."

C. "Refactor my application to meet the latest coding standards."

D. "Predict future coding trends and update my codebase accordingly."

**Answer: A B**

## Explanation

Effective prompts for refactoring include requests that focus on specific quality improvements, such as readability and maintainability. These prompts guide GitHub Copilot to provide relevant and actionable suggestions.

Reference: GitHub Copilot prompt engineering best practices.

Question #:103 - [Prompt Engineering and Best Practices]

What is used by GitHub Copilot in the IDE to determine the prompt context?

A. Information from the IDE like open tabs, cursor location, selected code.

B. All the code in the current repository and any git submodules.

C. The open tabs in the IDE and the current folder of the terminal.

D. All the code visible in the current IDE.

**Answer: A**

## Explanation

“GitHub Copilot generates suggestions based on context from the IDE, including the file you are editing, open tabs, cursor position, and any selected code.”

This proves that the context is drawn from editor state information, making option A correct.

References: GitHub Copilot context and suggestions documentation.

==========

Question #:104 - [Prompt Engineering and Best Practices]

Which GitHub Copilot pricing plans include features that exclude your GitHub Copilot data like usage, prompts, and suggestions from default training GitHub Copilot? (Choose two correct answers.)

A. GitHub Copilot Business

B. GitHub Copilot Codespace

C. GitHub Copilot Individual

D. GitHub Copilot Enterprise

**Answer: A D**

## Explanation

“For Copilot Business and Copilot Enterprise, user data such as code suggestions, prompts, and completions are excluded from training GitHub Copilot’s models.”

This confirms that only Business and Enterprise plans provide this exclusion feature.

References: GitHub Copilot data usage and privacy documentation.

==========

Question #:105 - [Administration and Licensing]

What type of information can you retrieve through GitHub Copilot Business Subscriptions via REST API? (Each correct answer presents part of the solution. Choose two.)

A. Get a summary of GitHub Copilot usage for organization members

B. List all GitHub Copilot seat assignments for an organization

C. View code suggestions for a specific user

D. List of all unsubscribed GitHub Copilot members within an organization

**Answer: A B**

## Explanation

Exact extracts:

- "The REST API for Copilot Business allows you to retrieve a summary of Copilot usage metrics for organization members."

References: GitHub Copilot Business REST API documentation.

==========

Question #:106 - [Copilot Chat and Slash Commands]

What is the best way to share feedback about GitHub Copilot Chat when using it on GitHub Mobile?

A. Use the emojis in the Copilot Chat interface.

B. The feedback section on the GitHub website.

C. By tweeting at GitHub's official X (Twitter) account.

D. The Settings menu in the GitHub Mobile app.

**Answer: A**

## Explanation

"On GitHub Mobile, you can use emojis in the Copilot Chat interface to provide quick feedback on the responses."

This makes option A correct.

References: GitHub Copilot Mobile documentation.

==========

Question #:107 - [Copilot Chat and Slash Commands]

Which of the following scenarios best describes the intended use of GitHub Copilot Chat as a tool?

A. A complete replacement for developers generating code.

B. A productivity tool that provides suggestions, but relying on human judgment.

C. A solution for software development, requiring no additional input or oversight.

D. A tool solely designed for debugging and error correction.

**Answer: B**

## Explanation

GitHub Copilot Chat is designed to be a productivity enhancer, not a replacement for human developers. It provides suggestions and assists with coding tasks, but the final decision and validation always rest with the developer. Copilot Chat is meant to augment the developer's workflow, making it faster and more efficient, but it does not remove the need for human oversight and judgment.

Reference: GitHub Copilot official documentation on the tool's purpose and usage.

Question #:108 - [Copilot Basics and Usage]

What two options navigate to configure duplicate detection? (Each correct answer presents part of the solution. Choose two.)

A. Enterprise settings # Copilot # Policies

B. Repository settings # Copilot # Policies

C. Organization settings # Copilot # Policies

D. User settings # Copilot # Policies

**Answer: A C**

## Explanation

“Duplicate detection and blocking of public code matches can be configured by enterprise administrators in enterprise or organization-level Copilot policies.”

This confirms that enterprise and organization settings are the correct paths.

References: GitHub Copilot policy configuration documentation.

==========

Question #:109 - [Ethical AI Principles]

How can the concept of fairness be integrated into the process of operating an AI tool?

A. Focusing on accessibility will ensure fairness.

B. Focusing on collecting large datasets for training will ensure fairness.

C. Regularly monitoring the AI tool's performance will ensure fairness in its outputs.

D. Training AI data and algorithms to be free from biases will ensure fairness.

**Answer: D**

## Explanation

Fairness in AI tools is achieved by training the data and algorithms to be free from biases. This ensures that the tool treats all users equitably and avoids discriminatory outcomes.

Reference: Microsoft's AI principles and fairness guidelines.

Question #:110 - [Copilot Basics and Usage]

Where can you validate if GitHub Copilot is not returning suggestions because of content exclusions?

A. The GitHub Copilot icon in the status bar of the editor will display a message

B. The GitHub Copilot logs on GitHub.com under your user settings

C. The code suggestions window will display a warning message

D. The GitHub Copilot errors panel in your IDE

**Answer: A**

## Explanation

Exact extracts:

- "When a file is affected by a content exclusion setting, GitHub Copilot will not suggest code completion in that file…"

References: GitHub Copilot official troubleshooting documentation on content exclusions.

==========

Question #:111 - [Copilot Basics and Usage]

Which GitHub Copilot plan allows for prompt and suggestion collection?

A. GitHub Copilot Individuals

B. GitHub Copilot Business

C. GitHub Copilot Enterprise

D. GitHub Copilot Codespace

**Answer: C**

## Explanation

GitHub Copilot Enterprise allows for prompt and suggestion collection, enabling organizations to analyze and improve their usage of the tool.

Reference: GitHub Copilot Enterprise data collection documentation.

Question #:112 - [Copilot Basics and Usage]

How can you get multiple suggestions from GitHub Copilot?

A. By asking for multiple suggestions using comments in your code

B. By opening the completions panel in your editor

C. By using the inline chat functionality with the command 'multiple'

D. By using @workspace in the chat window

**Answer: B**

## Explanation

You can get multiple suggestions by opening the completions panel in your editor, which displays alternative code suggestions.

Reference: GitHub Copilot usage documentation.

Question #:113 - [Security, Privacy, and Compliance]

When can GitHub Copilot still use content that was excluded using content exclusion?

A. If the contents of an excluded file are referenced in code that is not excluded, for example function calls.

B. When the repository level settings allow overrides by the user.

C. If the content exclusion was configured at the enterprise level, and is overwritten at the organization level.

D. When the user prompts with @workspace.

**Answer: A**

## Explanation

GitHub Copilot can still use excluded content if it is referenced in code that is not excluded, such as function calls.

Reference: GitHub Copilot content exclusion documentation.

Question #:114 - [Security, Privacy, and Compliance]

What are the effects of content exclusions? (Each correct answer presents part of the solution. Choose two.)

A. The excluded content is not directly available to GitHub Copilot to use as context.

B. GitHub Copilot suggestions are no longer available in the excluded files.

C. The excluded content is no longer used while debugging the code.

D. The IDE will not count coding suggestions in the excluded content.

**Answer: A B**

## Explanation

Content exclusions prevent GitHub Copilot from using the excluded content as context and stop suggestions from being generated in those files.

Reference: GitHub Copilot content exclusion documentation.

Question #:115 - [Copilot Basics and Usage]

How long does it take content exclusion to add or be updated?

A. Up to 30 minutes

B. 45 - 60 minutes

C. 60 - 90 minutes

D. 24 hours

**Answer: A**

## Explanation

Content exclusions typically take up to 30 minutes to be added or updated.

Reference: GitHub Copilot content exclusion documentation.

Question #:116 - [Administration and Licensing]

Which of the following steps correctly demonstrates how to establish an organization-wide policy for GitHub Copilot Business to restrict its use to certain repositories?

A. Apply policies through the GitHub Actions configuration

B. Create a copilot.policy file in each repository

C. Configure the policies in the organization settings

D. Create a copilot.policy in the .github repository

**Answer: C**

## Explanation

“Organization administrators can configure GitHub Copilot Business policies within the organization settings, including restricting usage to certain repositories.”

This proves that the correct way is through organization settings, not repository-level files or GitHub Actions.

References: GitHub Copilot Business policy management documentation.

==========

Question #:117 - [Knowledge Bases and Custom Models]

What types of content can GitHub Copilot Knowledge Base answer questions about? (Each correct answer presents part of the solution. Choose three.)

A. Code snippets

B. Compiled binaries

C. Documentation

D. Design patterns

E. Screenshots

**Answer: A C D**

## Explanation

"GitHub Copilot Knowledge Bases allow you to ask questions about your organization's documentation, design patterns, and code snippets."

This proves that Copilot Knowledge Base can work with code snippets, documentation, and design patterns, but not binaries or screenshots.

References: GitHub Copilot Enterprise Knowledge Base documentation.

==========

Question #:118 - [Security, Privacy, and Compliance]

When can GitHub Copilot still use content that was excluded using content exclusion?

A. When the user prompts with @workspace.

B. When the repository-level settings allow overrides by the user.

C. If the contents of an excluded file are referenced in code that is not excluded, for example function calls.

D. If the content exclusion was configured at the enterprise level, and is overwritten at the organization level.

**Answer: C**

## Explanation

"If excluded content is referenced by non-excluded code, such as through function calls, Copilot may still use that reference context."

This confirms that excluded content can indirectly appear if referenced elsewhere.

References: GitHub Copilot content exclusions limitations documentation.

Question #:119 - [Administration and Licensing]

What kind of insights can the GitHub Copilot usage metrics API provide to help evaluate the effectiveness of GitHub Copilot? (Each correct answer presents part of the solution. Choose two.)

A. The API can generate detailed reports on code quality improvements made by GitHub Copilot.

B. The API can track the number of code suggestions accepted and used in the organization.

C. The API can provide Copilot Chat specific suggestions acceptance metrics.

D. The API can refactor your code to improve productivity.

E. The API can provide feedback on coding style and standards compliance.

**Answer: B C**

## Explanation

Exact extracts:

- "The Copilot usage metrics API allows organizations to track the number of suggestions shown and accepted by members."

References: GitHub Copilot metrics API documentation.

Question #:120 - [IDE Integration and Features]

Which principle emphasizes that AI systems should be understandable and provide clear information on how they work?

A. Fairness

B. Transparency

C. Inclusiveness

D. Accountability

**Answer: B**

## Explanation

The principle of transparency emphasizes that AI systems should be understandable and provide clear information about their operations. This ensures that users can understand how the AI arrives at its decisions and suggestions.

Reference: Microsoft's AI principles and ethical guidelines.

# About Marks4sure.com

marks4sure.com was founded in 2007. We provide latest & high quality IT / Business Certification Training Exam Questions, Study Guides, Practice Tests.

We help you pass any IT / Business Certification Exams with 100% Pass Guaranteed or Full Refund. Especially Cisco, CompTIA, Citrix, EMC, HP, Oracle, VMware, Juniper, Check Point, LPI, Nortel, EXIN and so on.

View list of all certification exams: All vendors


We prepare state-of-the art practice tests for certification exams. You can reach us at any of the email addresses listed below.

- Sales: sales@marks4sure.com
- Feedback: feedback@marks4sure.com
- Support: support@marks4sure.com

Any problems about IT certification or our products, You can write us back and we will get back to you within 24 hours.