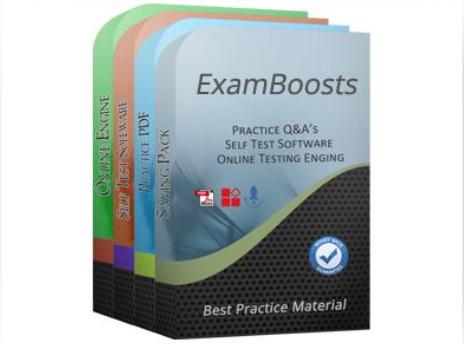
ExamBoosts

TRY BEFORE YOU BUY

Download a free sample of any of our exam questions and answers

- 24/7 customer support, Secure shopping site
- Free One year updates to match real exam scenarios
- If you failed your exam after buying our products we will refund the full amount back to you.



365 Days Free Updates

Free update is available within 365 days after your purchase. After 365 days, you will get 50% discounts for updating.

Security & Privacy

We respect customer privacy. We use McAfee's security service to provide you with utmost security for your personal information & peace of mind.

Instant Download

After Payment, our system will send you the products you purchase in mailbox in a minute after payment. If not received within 2 hours, please contact us.

Money Back Guarantee

Full refund if you fail the corresponding exam in 60 days after purchasing. And Free get any another product.

http://www.examboosts.com/

Reliable & Efficient Test Practice Questions to Satisfy All Candidates

Exam : **GH-200-JPN**

Title : GitHub Actions (GH-200日本語版)

Vendor : Microsoft

Version : DEMO

QUESTION NO: 1
馴染みのないクラウドプロバイダーにデプロイするために、新しいワークフローを作成する必要があります。最も迅速かつ安全な開始方法は何ですか？
A. クラウド プロバイダーの CLI をラップするカスタム アクションを作成します。
B. GitHub Marketplace で、クラウド
プロバイダーによって公開された検証済みのアクションを検索します。
C. actions/jenkins-plugin アクションを使用して、クラウド プロバイダーの既存の Jenkins プラグインを利用します。
D. GitHub Marketplace で GitHub によって作成されたアクションを検索します。
E. クラウド プロバイダーの CLI をダウンロードし、関連するドキュメントを確認します。
Answer: B
Explanation:
Searching the GitHub Marketplace for verified actions published by the cloud provider is the quickest and safest approach. Many cloud providers offer verified GitHub Actions that are maintained and optimized to interact with their services. These actions typically come with the correct configurations and best practices, allowing you to get started quickly without reinventing the wheel.
Note: About GitHub Marketplace for apps
GitHub Marketplace where you can share your apps with everyone.
GitHub Marketplace connects you to developers who want to extend and improve their GitHub workflows. You can list free and paid tools for developers to use in GitHub Marketplace. GitHub Marketplace offers developers two types of tools: GitHub Actions and Apps, and each tool requires different steps for adding it to GitHub Marketplace.
GitHub Actions
Anyone can publish an action in GitHub Marketplace. GitHub verifies some partner organizations and these are shown as verified creators.
Reference:
https://docs.github.com/en/enterprise-cloud@latest/apps/github-marketplace/github-marketplace- overview/about-github-marketplace-for-apps

QUESTION NO: 2
GitHub
Actionsのワークフローのうち、どのような種類のワークフローでセルフホスト型ランナーを使用すべきでしょうか？
A. プライベートな内部ネットワークリソースへのアクセスを必要とするワークフロー
B.
特定の環境設定を必要とせず、GitHubでホストされているインフラストラクチャ上でトリガーされる必要があるワークフロー
C. カスタム依存関係を必要としない、短期間のワークフロー
D.最新のオペレーティングシステムとプリインストールされたソフトウェアバージョンを必要とするワークフロー
Answer: A
Explanation:
Self-hosted Runners: These run on your own infrastructure (on-premises or cloud). They are

primarily used when you need specific environment configurations, such as custom hardware (GPUs), specific operating systems, or access to internal resources behind a firewall.
Using self-hosted runners is a standard solution for accessing private internal network resources, as they can be placed directly within your on-premises or cloud-based private network.
Incorrect:
[Not B]
For GitHub Actions workflows with no specific environment configurations that must be triggered on GitHub-hosted infrastructure, you should typically use GitHub-hosted runners, not self-hosted ones.
Reference:
https://github.blog/enterprise-software/ci-cd/when-to-choose-github-hosted-runners-or-self-hosted-runners-with-github-actions

QUESTION NO: 3
どのワークフローコマンドがデバッグメッセージ「action successfully debugged」を出力しますか？
A. echo "::debug::action successfully debugged"
B. echo "::debug::message=action successfully debugged"
C. echo "debug=action successfully debugged"
D. echo ":debug:action successfully debugged:"
Answer: B
Explanation:
To output a specific debug message in GitHub Actions, use the ::debug:: workflow command.
Usage Syntax
You can issue this command by printing it to standard output (stdout) during a workflow step:
echo "::debug::{message}"
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/workflow-commands

QUESTION NO: 4
プル リクエスト コメントによってトリガーされるワークフローでは、コメント テキストとプル リクエスト番号にアクセスするために使用されるイベント ペイロードが含まれる GitHub Actions コンテキスト プロパティはどれですか。
A. github.リポジトリ
B. github.イベント
C. github.event_path
D. github.job
Answer: B
Explanation:
In a workflow that runs on a pull request comment the event context contains the webhook payload. With github.event you can read the structured fields for the comment text and the pull request number. The comment text is available under the comment body field and the pull request number is available under the issue number for comment events or under the pull request number for pull request events. This makes github.event the direct and

convenient source for these values.

QUESTION NO: 5
DevOpsエンジニアとして、カスタムアクションの開発に取り組んでいます。アクションの開始時、メインのエントリポイントの前に、条件付きでスクリプトを実行したいと考えています。カスタムアクションのメタデータファイルを定義するには、どのコードブロックを使用すればよいでしょうか？
A. 実行:
使用: 'node16'
事前条件: github.event_name == 'push' then 'start.js'
メイン: 'index.js'
B. 実行:
使用: 'node16'
プレ: 'start.js'
事前条件: github.event_name == 'push'
メイン: 'index.js'
C. 実行:
使用: 'node16'
開始: 'start.js'
開始条件: github.event_name == 'push'
メイン: 'index.js'
D. 実行:
使用: 'node16'
以前: 'start.js'
before-if: github.event_name == 'push'
メイン: 'index.js'
Answer: B
Explanation:
The pre: line before the pre-if: line.
Note: runs.pre-if
Optional
Allows you to define conditions for the pre: action execution. The pre: action will only run if the conditions in pre-if are met. If not set, then pre-if defaults to always(). In pre-if, status check functions evaluate against the job's status, not the action's own status.
Note that the step context is unavailable, as no steps have run yet.
In this example, cleanup.js only runs on Linux-based runners:
pre: 'cleanup.js'
pre-if: runner.os == 'linux'
Reference:
https://docs.github.com/en/actions/reference/workflows-and-actions/metadata-syntax

QUESTION NO: 6
組織では、GitHub Enterprise Cloud
における自動化の再利用と保守を簡素化する必要があります。組織内のすべてのリポジトリで直接再利用できるコンポーネントはどれでしょうか？

(それぞれの正解は完全な解決策を示しています。3 つ選択してください。)
A. GitHub Marketplace の組織パーティションに保存されたアクション
B. GitHub Container Registry に保存されたカスタム Docker アクション
C. セルフホストランナー
D. 暗号化された秘密
E. ワークフローテンプレート
F. 組織内のプライベートリポジトリに保存されたアクション
Answer: DEF
Explanation:
[D] Encrypted secrets can be accessed across repositories in the same organization, making it easy to store sensitive data (like API keys or tokens) securely while allowing multiple workflows to access them.
[E] Workflow templates allow you to create reusable templates for workflows that can be shared across repositories within the organization. This makes it easier to standardize processes and automate them across multiple projects.
[F] Actions stored in private repositories within the organization can be reused across all repositories by referencing them in workflows. This ensures a centralized way of maintaining custom actions.
Reference:
https://docs.github.com/en/actions/how-tos/reuse-automations/reuse-workflows

QUESTION NO: 7
開発者として、自動化の再利用に関する標準を実装するために、どのような2つの選択肢を推奨すべきでしょうか？正解はそれぞれ完全な解決策を示しています。注：正解1つにつき1ポイントです。
A.他のワークフローから呼び出すことができる、再利用可能なアクションとワークフローを作成します。
B.
会社向けに再利用可能な自動化を公開するためのマーケットプレイスパーティションを作成します。
C.
定義され文書化された内部アクセス可能なリポジトリ内のサブフォルダに、共有の企業行動を保存します。
D. ワークフローテンプレートを作成し、組織の.githubリポジトリに保存します。
Answer: AC
Explanation:
[A]
Implementing standards for reusable actions and workflows is the most effective way to reduce duplication and maintain consistency across your GitHub organization.
Reusable Workflows (workflow_call)
Best for standardizing entire processes (e.g., a complete CI/CD pipeline).
Location: Defined in .github/workflows/ of a central repository.
Trigger: Use the on: workflow_call trigger to define inputs, secrets, and outputs.
Usage: Called from another workflow using the uses keyword (e.g.,

owner/repo/.github/workflows/ci.yml@v1).
Benefit: They allow you to enforce security scans, build steps, and deployment gates across multiple repositories from a single source
[C]
Implementing a marketplace partition for internal GitHub Actions is a key strategy for scaling automation securely and efficiently across a company. This centralized approach ensures that developers use vetted, high-quality automation while adhering to corporate standards.
Strategic Implementation of Internal Reuse
To implement standards for automation reuse effectively, focus on these core components:
[C] Internal Actions Marketplace: Create a dedicated organization or specific repositories to host and display shared actions. This "marketplace" acts as a curated directory of approved automations, preventing the use of unverified third-party actions from the public marketplace.
Workflow Templates: Store standardized workflow templates in your organization's .github repository. These templates appear in the "Actions" tab when a user creates a new workflow, ensuring consistency across different teams.
[A] Reusable Workflows: Design modular workflows that can be called by other repositories using the uses keyword. Unlike standard actions, these can manage entire jobs and runners, making them ideal for standardizing complex CI/CD pipelines.
Internal Repository Sharing: Set repository visibility to Internal and configure Actions permissions to allow access from other repositories in the organization or enterprise.
Reference:
https://www.incredibuild.com/blog/best-practices-to-create-reusable-workflows-on-github-actions
https://xebia.com/blog/setting-up-an-internal-github-actions-marketplace

QUESTION NO: 8
以下のシナリオのうち、GitHubでホストされているランナーではなく、自己ホスト型のランナーを使用する必要があるのはどれですか？
A.
GitHubでホストされているランナーがサポートする3つ以上の同時実行ワークフローを実行する
B.macOS上でのビルドの実行
C.月間ビルド時間が50,000分を超える
D.ワークフローに必要な専用ハードウェア構成を使用する
E.ワークフローの一部としてDockerコンテナを使用する
Answer: D
Explanation:
The use of self-hosted runners is required when your workflows depend on specialized hardware configurations that are not available through standard GitHub-hosted runners. While GitHub offers "larger runners" with GPU support for certain paid plans, self-hosted runners provide the most flexibility for highly specific or proprietary hardware needs.
Scenarios Requiring Specialized Hardware
Specific GPU Models: Workflows for AI/ML training or intensive graphics rendering that require specific NVIDIA or other specialized GPU architectures.
Alternative CPU Architectures: When you must build or test on specific ARM processors,

legacy x86 32-bit systems, or other non-standard architectures not supported by GitHub's managed pool.
High-Resource Requirements: Tasks needing massive amounts of RAM (beyond 256 GB) or high-core counts for extreme parallel processing.
Custom Peripherals: Workflows that need physical access to hardware connected via USB, PCIe, or other local interfaces (e.g., embedded systems testing or hardware-in-the-loop).
Reference:
https://docs.github.com/en/actions/how-tos/manage-runners/self-hosted-runners/use-in-a-workflow

QUESTION NO: 9
設定可能な環境保護ルールには、どのような2種類がありますか？正解はそれぞれ完全な解答を示しています。注：正解1つにつき1ポイントです。
A. 遺物保管庫
B.待機タイマー
C.分岐保護
D.必須レビュー担当者
Answer: BD
Explanation:
In GitHub Actions, you can configure four primary types of environment protection rules to control how and when deployments proceed to a specific environment.
These rules include:
[D] Required Reviewers: Specify up to six people or teams who must approve a deployment before it can proceed. Only one reviewer from the list needs to approve to unlock the deployment.
[B] Wait Timer: Set a mandatory delay (from 0 to 43,200 minutes) that must pass after a job is triggered before it can run.
Deployment Branches and Tags: Restrict deployments to only occur from specific branches or those matching certain tag patterns (e.g., main, release/*, or v*).
Custom Deployment Protection Rules: Enable third-party systems or automated tools (via GitHub Apps) to gate deployments based on external data, such as security scan results or ticket status.
Reference:
https://oneuptime.com/blog/post/2026-01-25-github-actions-environment-protection-rules/view

QUESTION NO: 10
特定のリポジトリ内のワークフローから単一のシークレットにアクセスする必要があります。シークレットを作成する最適な方法は何ですか？
A.
組織レベルで環境シークレットを作成し、指定された各リポジトリでその環境を活用します。
B. 組織シークレットを作成し、リポジトリ アクセスとして [選択されたリポジトリ] を指定して、必要なリポジトリを選択します。
C.

いずれかのリポジトリにシークレットを作成し、シークレットの共有オプションをオンにして、必要なリポジトリを選択します。
D. シークレットをサポートされている外部キー コンテナーに保存します。OpenID Connect (OIDC) を構成して外部キー コンテナーへのアクセスを許可し、各リポジトリの外部キー コンテナーからシークレットをリンクします。
Answer: B
Explanation:
Creating secrets for an organization
When creating a secret or variable in an organization, you can use a policy to limit access by repository. For example, you can grant access to all repositories, or limit access to only private repositories or a specified list of repositories.
To specify that the secret should be available to selected repositories within the organization, use the --repos or -r flag.
gh secret set --org ORG_NAME SECRET_NAME --repos REPO-NAME-1, REPO-NAME-2
Note: REST API endpoints for GitHub Actions Secrets Use the REST API to interact with secrets in GitHub Actions.
* Set selected repositories for an organization secret
Replaces all repositories for an organization secret when the visibility for repository access is set to selected. The visibility is set when you Create or update an organization secret.
Request example
Put /orgs/{org}/actions/secrets/{secret_name}/repositories
Reference:
https://docs.github.com/en/actions/how-tos/write-workflows/choose-what-workflows-do/use-secrets
https://docs.github.com/en/rest/actions/secrets?apiVersion=2022-11-28#set-selected-repositories-for-an-organization-secret

QUESTION NO: 11
GitHub Actions
では、ステップの成功または失敗は終了コードによってどのように決定されるのでしょうか？
A. 終了コードは無視され、メンテナーが手動で結果を設定します
B. 終了コードがゼロの場合は成功、ゼロ以外の場合は失敗を意味します。
C. 成功は終了コードではなくエラー時の継続によって制御されます
D. すべての終了コードは失敗です
Answer: B
Explanation:
GitHub Actions evaluates each step by the exit status of the process it runs. The runner marks a step successful when the command finishes with an exit status of 0. If the command returns any other code then the step is marked as failed and the job may stop depending on your workflow configuration.

QUESTION NO: 12
開発者として、GitHub 組織で IP 許可リストを設定しました。IP 許可リストは GitHub Actions にどのような影響を与えますか？(各回答には完全な解決策が記載されています。)

2つ選択してください。
A. IP アドレスが自動的に許可されるため、標準の GitHub ホストランナーを使用できます。
B. 既知の IP アドレスを持つセルフホストランナーを使用できます。
C. マーケットプレイスアクションを使用するには、GitHub Actions の IP アドレス範囲を許可する必要があります。
D. GitHub でホストされている大規模なランナーは、静的 IP アドレスで設定できるため、使用できます。
Answer: BD
Explanation:
Using GitHub Actions with an IP allow list. If you use an IP allow list and would also like to use GitHub Actions, you must use self-hosted runners or GitHub-hosted larger runners with static IP address ranges To allow your self-hosted or larger hosted runners to communicate with GitHub, add the IP address or IP address range of your runners to the IP allow list that you have configured for your enterprise.
Reference:
https://docs.github.com/en/enterprise-cloud@latest/organizations/keeping-your-organization-secure/managing-security-settings-for-your-organization/managing-allowed-ip-addresses-for-your-organization