**1. What do you know about DevOps?**

DevOps is a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.

**2. How is DevOps different from agile methodology?**

DevOps is a culture that allows the development and the operations team to work together. This results in continuous development, testing, integration, deployment, and monitoring of the software throughout the lifecycle

Agile is a software development methodology that focuses on iterative, incremental, small, and rapid releases of software, along with customer feedback. It addresses gaps and conflicts between the customer and developers.

DevOps addresses gaps and conflicts between the Developers and IT Operations.

**3. Which are some of the most popular DevOps tools?**

The most popular DevOps tools include:

Selenium

Puppet

Chef

Git

Jenkins

Ansible

Docker

**4. What are the different phases in DevOps?**

The various phases of the DevOps lifecycle are as follows:

**Plan** - Initially, there should be a plan for the type of application that needs to be developed. Getting a rough picture of the development process is always a good idea.

**Code** - The application is coded as per the end-user requirements.

**Build** - Build the application by integrating various codes formed in the previous steps.

**Test** - This is the most crucial step of the application development. Test the application and rebuild, if necessary.

**Integrate** - Multiple codes from different programmers are integrated into one.

**Deploy** - Code is deployed into a cloud environment for further usage. It is ensured that any new changes do not affect the functioning of a high traffic website.

**Operate** - Operations are performed on the code if required.

**Monitor** - Application performance is monitored. Changes are made to meet the end-user requirements.

**5. Mention some of the core benefits of DevOps.**

The core benefits of DevOps are as follows:

Technical benefits

Continuous software delivery

Less complex problems to manage

Early detection and faster correction of defects

Business benefits

Faster delivery of features

Stable operating environments

Improved communication and collaboration between the teams

**6. What is the difference between continuous delivery and continuous deployment?**

Continuous Delivery is the frequent shipping of code to a given environment (such as test or production) via manual release. Continuous Deployment is the automated release of code to a production environment

**7. What is the role of configuration management in DevOps?**

Enables management of and changes to multiple systems.

Standardizes resource configurations, which in turn, manage IT infrastructure.

It helps with the administration and management of multiple servers and maintains the integrity of the entire infrastructure.

**8. How does continuous monitoring help you maintain the entire architecture of the system?**

Continuous monitoring in DevOps is a process of detecting, identifying, and reporting any faults or threats in the entire infrastructure of the system.

Ensures that all services, applications, and resources are running on the servers properly.

Monitors the status of servers and determines if applications are working correctly or not.

Enables continuous audit, transaction inspection, and controlled monitoring.

**9. What is the role of AWS in DevOps?**

AWS has the following role in DevOps:

Flexible services - Provides ready-to-use, flexible services without the need to install or set up the software.

Built for scale - You can manage a single instance or scale to thousands using AWS services.

Automation - AWS lets you automate tasks and processes, giving you more time to innovate

Secure - Using AWS Identity and Access Management (IAM), you can set user permissions and policies.

Large partner ecosystem - AWS supports a large ecosystem of partners that integrate with and extend AWS services.

**10. Name three important DevOps KPIs.**

The three important KPIs are as follows:

Meantime to failure recovery - This is the average time taken to recover from a failure.

Deployment frequency - The frequency in which the deployment occurs.

Percentage of failed deployments - The number of times the deployment fails.

**11. Why Has DevOps Gained Prominence over the Last Few Years?**

Before talking about the growing popularity of DevOps, discuss the current industry scenario. Begin with some examples of how big players such as Netflix and Facebook are investing in DevOps to automate and accelerate application deployment and how this has helped them grow their business. Using Facebook as an example, you would point to Facebook’s continuous deployment and code ownership models and how these have helped it scale up but ensure the quality of experience at the same time. Hundreds of lines of code are implemented without affecting quality, stability, and security.

**12. Explain the difference between a centralized and distributed version control system (VCS).**

**Centralized Version Control System**

All file versions are stored on a central server

No developer has a copy of all files on a local system

If the central server crashes, all data from the project will be lost

**Distributed Control System**

Every developer has a copy of all versions of the code on their systems

Enables team members to work offline and does not rely on a single location for backups

There is no threat, even if the server crashes

**13. What is the git command that downloads any repository from GitHub to your computer?**

The git command that downloads any repository from GitHub to your computer is **git clone**.

**14. How do you push a file from your local system to the GitHub repository using Git?**

First, connect the local repository to your remote repository:

git remote add origin [copied web address]

// Ex: git remote add origin <https://github.com/Simplilearn-github/test.git>

Second, push your file to the remote repository:

git push origin master

**15. How is a bare repository different from the standard way of initializing a Git repository?**

**Using the standard method**:

git init

You create a working directory with git init

A .git subfolder is created with all the git-related revision history

**Using the bare way**

git init --bare

It does not contain any working or checked out a copy of source files

Bare repositories store git revision history in the root folder of your repository, instead of the .git subfolder

**16. What is the process for reverting a commit that has already been pushed and made public?**

There are two ways that you can revert a commit:

Remove or fix the bad file in a new commit and push it to the remote repository. Then commit it to the remote repository using:

git commit –m "commit message"

Create a new commit that undoes all the changes that were made in the bad commit. Use the following command:

git revert <commit id>

Example: git revert 56de0938f

**17. Explain the difference between git fetch and git pull.**

Git fetch Git pull

Git fetch only downloads new data from a remote repository Git pull updates the current HEAD branch with the latest changes from the remote server

**18. What is Git stash?**

A developer working with a current branch wants to switch to another branch to work on something else, but the developer doesn't want to commit changes to your unfinished work. The solution to this issue is Git stash. Git stash takes your modified tracked files and saves them on a stack of unfinished changes that you can reapply at any time.

**19. Explain the concept of branching in Git.**

Suppose you are working on an application, and you want to add a new feature to the app. You can create a new branch and build the new feature on that branch.

By default, you always work on the master branch

**20. What is the difference between Git Merge and Git Rebase?**

Suppose you are working on a new feature in a dedicated branch, and another team member updates the master branch with new commits. You can use these two functions:

Git Merge

To incorporate the new commits into your feature branch, use Git merge.

Creates an extra merge commit every time you need to incorporate changes

But, it pollutes your feature branch history

Git Rebase

As an alternative to merging, you can rebase the feature branch on to master.

Incorporates all the new commits in the master branch

It creates new commits for every commit in the original branch and rewrites project history

**21. Explain the master-slave architecture of Jenkins.**

Jenkins master pulls the code from the remote GitHub repository every time there is a code commit.

It distributes the workload to all the Jenkins slaves.

On request from the Jenkins master, the slaves carry out, builds, test, and produce test reports.

**22. What is Jenkinsfile?**

Jenkinsfile contains the definition of a Jenkins pipeline and is checked into the source control repository. It is a text file.

It allows code review and iteration on the pipeline.

It permits an audit trail for the pipeline.

**23. Which of the following commands runs Jenkins from the command line?**

java –jar Jenkins.war

java –war Jenkins.jar

java –jar Jenkins.jar

java –war Jenkins.war

The correct answer is A) java –jar Jenkins.war

**24. Which file is used to define dependency in Maven?**

build.xml

pom.xml

dependency.xml

Version.xml

The correct answer is B) pom.xml

**25. How do you create a backup and copy files in Jenkins?**

In order to create a backup file, periodically back up your JENKINS\_HOME directory.

In order to create a backup of Jenkins setup, copy the JENKINS\_HOME directory. You can also copy a job directory to clone or replicate a job or rename the directory.

**26. How can you copy Jenkins from one server to another?**

Move the job from one Jenkins installation to another by copying the corresponding job directory.

Create a copy of an existing job by making a clone of a job directory with a different name.

Rename an existing job by renaming a directory.

**27. Why are SSL certificates used in Chef?**

SSL certificates are used between the Chef server and the client to ensure that each node has access to the right data.

Every node has a private and public key pair. The public key is stored at the Chef server.

When an SSL certificate is sent to the server, it will contain the private key of the node.

The server compares this against the public key in order to identify the node and give the node access to the required data.

**28. Which of the following commands would you use to stop or disable the 'httpd' service when the system boots?**

# systemctl disable httpd.service

# system disable httpd.service

# system disable httpd

# systemctl disable httpd.service

The correct answer is A) # systemctl disable httpd.service

**29. What is Test Kitchen in Chef?**

Test Kitchen is a command-line tool in Chef that spins up an instance and tests the cookbook on it before deploying it on the actual nodes.

**30. How does chef-apply differ from chef-client?**

chef-apply is run on the client system.

chef-apply applies the recipe mentioned in the command on the client system.

$ chef-apply recipe\_name.rb

chef-client is also run on the client system.

chef-client applies all the cookbooks in your server's run list to the client system.

$ knife chef-client

**31. What is an Ansible role?**

An Ansible role is an independent block of tasks, variables, files, and templates embedded inside a playbook.

**32. How is Ansible different from Puppet?**

Ansible :

Easy agentless installation

Based on Python

Configuration files are written in YAML

No support for Windows

Puppet:

Agent-based installation

Based on Ruby

Configuration files are written in DSL

Support for all popular OS's

**33. Explain the architecture of Docker.**

Docker uses a client-server architecture.

Docker Client is a service that runs a command. The command is translated using the REST API and is sent to the Docker Daemon (server).

Docker Daemon accepts the request and interacts with the operating system to build Docker images and run Docker containers.

A Docker image is a template of instructions, which is used to create containers.

Docker container is an executable package of an application and its dependencies together.

Docker registry is a service to host and distribute Docker images among users.

**34. How do we share Docker containers with different nodes?**

It is possible to share Docker containers on different nodes with Docker Swarm.

Docker Swarm is a tool that allows IT administrators and developers to create and manage a cluster of swarm nodes within the Docker platform.

A swarm consists of two types of nodes: a manager node and a worker node.

**35. How do you run multiple containers using a single service?**

It is possible to run multiple containers as a single service with Docker Compose.

Here, each container runs in isolation but can interact with each other.

All Docker Compose files are YAML files.

**36.What is a Dockerfile used for?**

A Dockerfile is used for creating Docker images using the build command.

**37. Explain the differences between Docker images and Docker containers.**

**Docker Images :**

Docker images are templates of Docker containers

An image is built using a Dockerfile

It is stored in a Docker repository or a Docker hub

The image layer is a read-only filesystem

**Docker Container:**

Containers are runtime instances of a Docker image

Containers are created using Docker images

They are stored in the Docker daemon

Every container layer is a read-write filesystem

**38. Instead of YAML, what can you use as an alternate file for building Docker compose?**

To build a Docker compose, a user can use a JSON file instead of YAML.

**39. What are the cloud platforms that support Docker?**

The following are the cloud platforms that Docker runs on:

Amazon Web Services

Microsoft Azure

Google Cloud Platform

Rackspace

**40. How does Nagios help in the continuous monitoring of systems, applications, and services?**

Nagios enables server monitoring and the ability to check if they are sufficiently utilized or if any task failures need to be addressed.

Verifies the status of the servers and services

Inspects the health of your infrastructure

Checks if applications are working correctly and web servers are reachable