Q: What is OOPS concepts in Java?

A: **Object-oriented** programming (**OOP**) is a programming language model organized around objects rather than "actions" and data rather than logic. Historically, a program has been viewed as a logical procedure that takes input data, processes it, and produces output data.

OOPs concepts:

Abstraction: Abstraction means hiding lower-level details and exposing only the essential and relevant details to the users.

A class which contains the **abstract** keyword in its declaration is known as abstract class.

Abstract classes may or may not contain *abstract methods*, i.e., methods without body ( public void get(); )

If a class is declared abstract, it cannot be instantiated.

* To use an abstract class, you have to inherit it from another class, provide implementations to the abstract methods in it.
* If you inherit an abstract class, you have to provide implementations to all the abstract methods in it.
* **abstract** keyword is used to declare the method as abstract.
* You have to place the **abstract** keyword before the method name in the method declaration.
* An abstract method contains a method signature, but no method body

Encapsulation: Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class. Therefore, it is also known as **data hiding**.

To achieve encapsulation in Java −

* Declare the variables of a class as private.
* Provide public setter and getter methods to modify and view the variables values.

Inheritance: Inheritance can be defined as the process where one class acquires the properties (methods and fields) of another.

**extends** is the keyword used to inherit the properties of a class. Following is the syntax of extends keyword.

What is the main **difference** of **superclass**, **subclass** and concrete class in **Java**? ... A **subclass** inherits all the members (fields, methods, and nested classes) from its **superclass**. Constructors are not members, so they are not inherited by **subclasses**, but the constructor of the **superclass** can be invoked from the **subclass**.

Polymorphism

An interface is a reference type in Java. It is similar to class. It is a collection of abstract methods. A class implements an interface, thereby inheriting the abstract methods of the interface.

An interface can contain any number of methods.

You cannot instantiate an interface.

* An interface does not contain any constructors.
* All of the methods in an interface are abstract
* An interface cannot contain instance fields. The only fields that can appear in an interface must be declared both static and final.
* An interface is not extended by a class; it is implemented by a class.
* An interface can extend multiple interfaces.

**WebDriver driver** = **new FirefoxDriver**(); Here,**WebDriver** is an interface, **driver** is a reference variable, **FirefoxDriver**() is a Constructor, **new** is a keyword, and **new FirefoxDriver**() is an Object.

**"WebDriver** is an **Interface**,and we are defining a reference variable(driver) whose type is an **interface**. Now any object we assign to it must be an instance of a **class**(FireFoxDriver) that implements the **interface**.

Q: What is the use of xpath

A: Xpath is used to find the webelement in web pages. It is also useful in indentifying dynamic elements.

Q: Difference between single slash(/) and double slash(//)

A: single slash enables to create absolute xpath and double slash to create relative xpath

Q: difference between findelements and findelement

A:

Q: Advantages of using github

A: Multiple people when they work on the same project they can update project details and inform other team members simultaneously.

Jenkins can help you to build the project from the remote repository regularly. This helps you to keep track of failed builds.

Q: How to resolve github conflicts

A: You can only resolve merge conflicts on GitHub that are caused by competing line changes, such as when people make different changes to the same line of the same file on different branches in your Git repository.

### Q: list all branches

git branch -a

**list remote branches**

git branch -r

**Create a new branch**

first create a branch

git checkout -b <branchName>

Create a new branch from an existing branch

git checkout origin/branchName -b newBranchName

Then push your new branch to the repo

git push origin <branchName>

**revert a file to the most recent commit**

git checkout HEAD -- /somePath/file.txt

**undo the last commit. Blow it out of the water.**

git reset --hard HEAD~1

**undo your last commit but leave the files from that commit staged.**

git reset --soft HEAD~1

**delete local (untracked) files**

git clean -f

**If you want to also remove directories, run**

git clean -f -d

**to discard changes in working directory**

git checkout -- <file>

**Checkout a file from another branch**

git checkout origin/branchName -- fileName.txt

**clean a folder**

git clean -fxd {dir\_path}

**commit a folder/file without staging it.**

git commit /folderToCommit -m 'commit msg'

**list conflicts**

git diff --name-only --diff-filter=U

**Stashes**

**Stashing** takes the dirty state of your working directory — that is, your modified tracked files and staged changes — and saves it on a stack of unfinished changes that you can reapply at any time

**save a stash**

git stash save "My changes."

**list your saved stashes**

git stash list

**apply a stash (Where stash@{1} is the stash you want to apply.)**

git stash apply stash@{1}

Q: What are the frameworks that you have worked on?

Q: What is hybrid framework

Q: What is data driven framework

Q: What is best framework to perform testing for web application along with database testing and for running batch jobs

Maven is a powerful project management tool that is based on POM (**project object model**). It is used for projects build, dependency and documentation.