**1. Steps to Create a Maven Build Pipeline in Azure DevOps Portal**

**Aim**: To set up a Maven Build Pipeline in Azure DevOps.

**Procedure**:

1. **Create Project**:
   * Go to Azure DevOps Portal and create a new project.
2. **Repository Setup**:
   * Navigate to *Repos* and initialize your repository or connect it to an existing one (GitHub/Azure Repos).
3. **Create Pipeline**:
   * Go to *Pipelines* > *Create Pipeline*.
   * Select where your code is (e.g., Azure Repos Git, GitHub).
   * Configure the pipeline by selecting the repository.
4. **Select Template**:
   * Choose the Maven pipeline template.
   * Azure will auto-generate a YAML file (for Maven builds).
5. **Configure YAML File**:
   * The azure-pipelines.yml file is pre-populated. You can adjust it as needed:

yaml

Copy code

trigger:

branches:

include:

- master

pool:

vmImage: 'ubuntu-latest'

steps:

- task: Maven@3

inputs:

mavenPomFile: 'pom.xml'

goals: 'clean install'

1. **Save and Run**:
   * Save the YAML pipeline and click *Run* to initiate the build.

**2. Commands or Configurations for Running Regression Tests in Maven Pipeline**

**Aim**: To run regression tests during the Maven build.

**Procedure**:

1. **Modify the YAML pipeline**:
   * Include a task to execute tests:

yaml

Copy code

steps:

- task: Maven@3

inputs:

mavenPomFile: 'pom.xml'

goals: 'test'

options: '-Dtest=\*Test'

1. **Run Specific Regression Tests**:
   * In the options, specify your regression tests or test suites.
   * Use flags like -Dtest for specific tests or -Dgroups for categorized tests.

**3. Procedure for Installing Jenkins in Azure Cloud**

**Aim**: To install Jenkins on an Azure VM.

**Procedure**:

1. **Provision a VM**:
   * In the Azure portal, create a Linux/Windows VM (recommended: Ubuntu for Jenkins).
   * Configure networking and security groups.
2. **SSH into the VM** (Linux):

bash

Copy code

ssh username@VM\_IP\_Address

1. **Install Java**:

bash

Copy code

sudo apt update

sudo apt install openjdk-11-jdk

1. **Add Jenkins Repository and Install**:

bash

Copy code

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -

sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

sudo apt update

sudo apt install jenkins

1. **Start Jenkins**:

bash

Copy code

sudo systemctl start jenkins

1. **Access Jenkins**:
   * Open http://VM\_IP\_Address:8080 in your browser.

**4. Steps for Creating a CI Pipeline using Jenkins for Java Projects**

**Aim**: To build a CI pipeline in Jenkins.

**Procedure**:

1. **Install Jenkins Plugins**:
   * Install *Maven Integration Plugin* and *Git Plugin* via the Jenkins UI.
2. **Create a New Job**:
   * In Jenkins, click *New Item* > *Freestyle Project*.
3. **Source Code Management**:
   * Select *Git* and enter the repository URL.
4. **Build Triggers**:
   * Enable *Poll SCM* or *Build periodically*.
5. **Build Configuration**:
   * Select *Invoke top-level Maven targets* and set the *Goals* to clean install.
6. **Post-build Actions**:
   * Add *Publish JUnit test result report* for test result archiving.

**5. Steps for Creating a CD Pipeline in Jenkins with Azure Deployment**

**Aim**: To create a CD pipeline for Java projects in Jenkins and deploy in Azure.

**Procedure**:

1. **Install Azure CLI** in Jenkins:

bash

Copy code

sudo apt install azure-cli

1. **Integrate Jenkins with Azure**:
   * Configure Azure Service Principal in Jenkins for authentication.
2. **Build Job Configuration**:
   * Use Maven to build the Java project.
3. **Deployment Script**:
   * Add a shell step for deploying the application in Azure, using the Azure CLI:

bash

Copy code

az webapp up --name <app-name> --resource-group <resource-group> --plan <app-service-plan>

**6. Ansible Playbook for Basic Web App Infrastructure**

**Aim**: To set up a basic web application infrastructure using Ansible.

**YAML Playbook**:

yaml

Copy code

---

- name: Setup Web Server

hosts: webservers

become: yes

tasks:

- name: Install Nginx

apt:

name: nginx

state: present

- name: Start Nginx

service:

name: nginx

state: started

**7. Building a Simple Application Using Gradle**

**Aim**: To build a Java project using Gradle.

**Procedure**:

1. **Initialize Gradle**:

bash

Copy code

gradle init

1. **Define Build Script (build.gradle)**:

gradle

Copy code

plugins {

id 'java'

}

repositories {

mavenCentral()

}

dependencies {

testImplementation 'junit:junit:4.12'

}

1. **Run Build**:

bash

Copy code

gradle build

**8. Commands for Installing Ansible and Configuring Roles**

**Aim**: To install Ansible and configure roles.

**Procedure**:

1. **Install Ansible**:

bash

Copy code

sudo apt update

sudo apt install ansible

1. **Create a Role**:

bash

Copy code

ansible-galaxy init myrole

1. **Role Structure**:
   * Modify the role's tasks in roles/myrole/tasks/main.yml.
2. **Playbook Using the Role**:

yaml

Copy code

---

- name: Apply Web Role

hosts: all

roles:

- myrole

**9. Jenkins for Continuous Integration in Lab**

**Aim**: To automate code builds, testing, and integration in Jenkins.

**Procedure**:

1. **Automated Builds**:
   * Jenkins automatically triggers builds when new code is pushed to the repository.
2. **Automated Testing**:
   * Use Jenkins to run unit tests via Maven or Gradle after each build.
3. **Feedback**:
   * Jenkins provides feedback with build and test results, ensuring code integrity.

**10. Writing and Utilizing Ansible Playbooks for Automation**

**Aim**: To automate infrastructure tasks and deployments with Ansible.

**Procedure**:

1. **Create Playbook**:
   * Example YAML playbook to install Apache:

yaml

Copy code

---

- name: Install Apache

hosts: all

become: yes

tasks:

- name: Install Apache

apt:

name: apache2

state: present

1. **Run Playbook**:

bash

Copy code

ansible-playbook playbook.yml

This covers all the steps for creating pipelines and automating tasks using Jenkins, Maven, and Ansible.