Submitted by:

Name - Rohit Kumar
SAP - 500082652
ROLL NO. - R214220968
BATCH - 1 DevOps

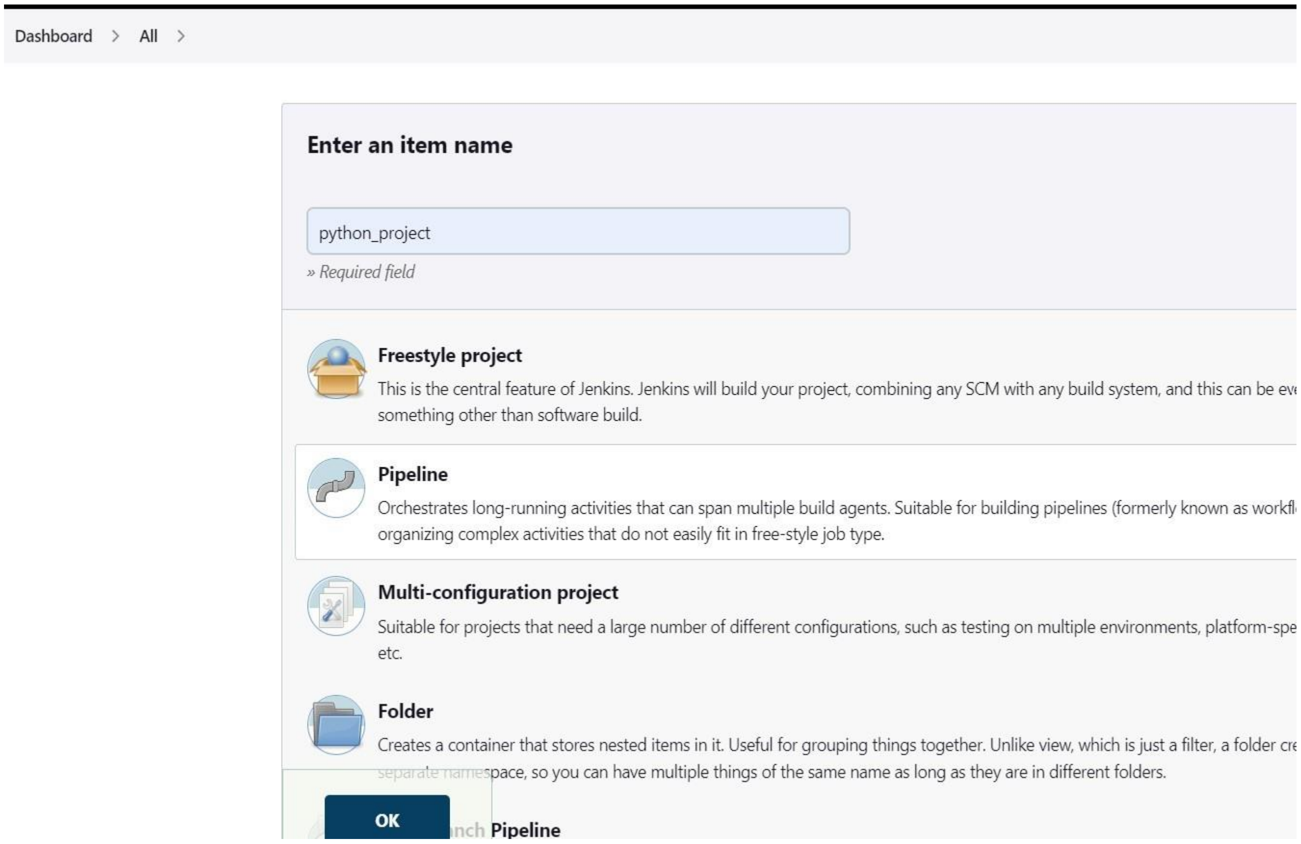
Submitted to:

Mr. Omkarendra Tiwari

CI - CD Lab

Pipeline Using Jenkins

Step 1: create a pipeline project


Dashboard > All >

Enter an item name

python_project

» Required field

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be ev
something other than software build.

Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workfl
organizing complex activities that do not easily fit in free-style job type.

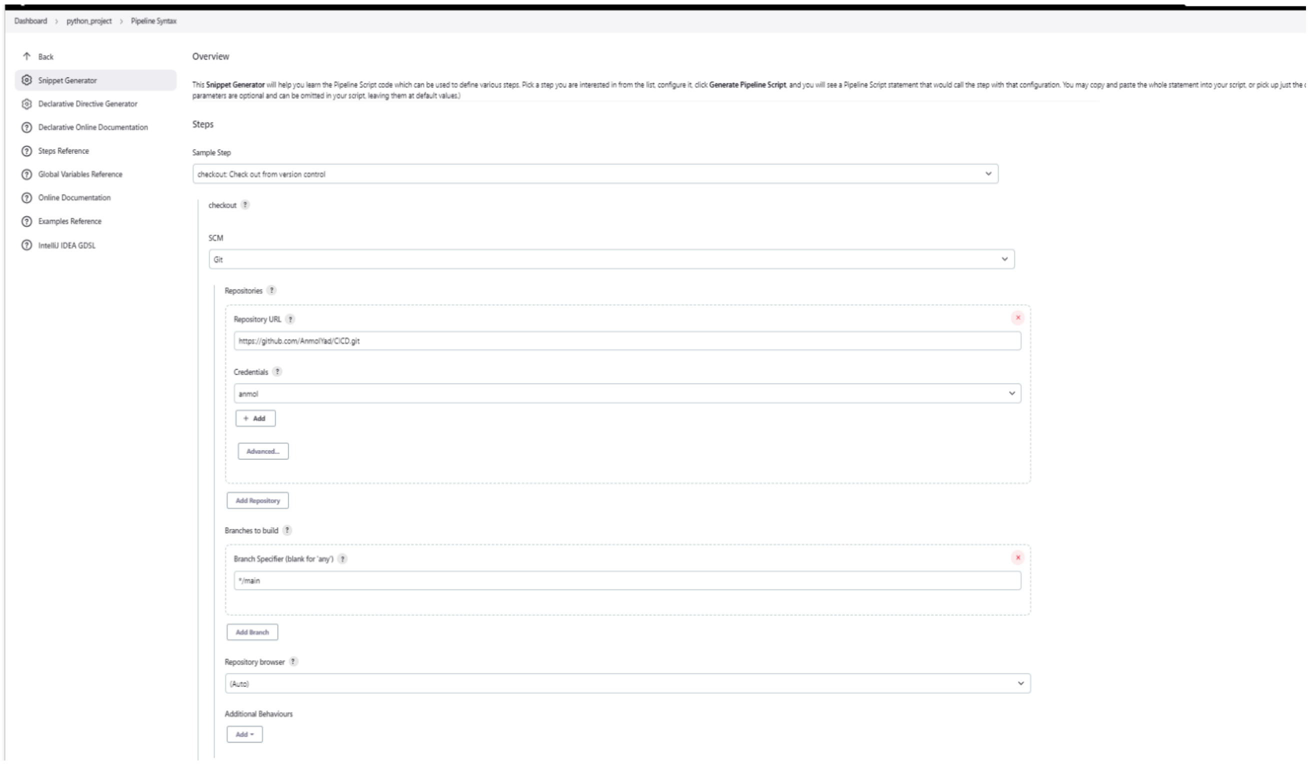
Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-spe
etc.

Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder cre
separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Step 2: Add python code in the git repo then give its URL, credentials to connect with Jenkins .



Dashboard › python_project › Pipeline Syntax

Back
Snippet Generator
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL

Overview

This **Snippet Generator** will help you learn the Pipeline Script code which can be used to define various steps. Pick a step you are interested in from the list, configure it, click **Generate Pipeline Script**, and you will see a Pipeline Script statement that would call the step with that configuration. You may copy and paste the whole statement into your script, or pick up just the
parameters are optional and can be omitted in your script, leaving them at default values.)

Steps

Sample Step: checkout: Check out from version control

checkout

SCM: Git

Repositories

Repository URL: https://github.com/AnmolYad/CICD.git

Credentials: anmol

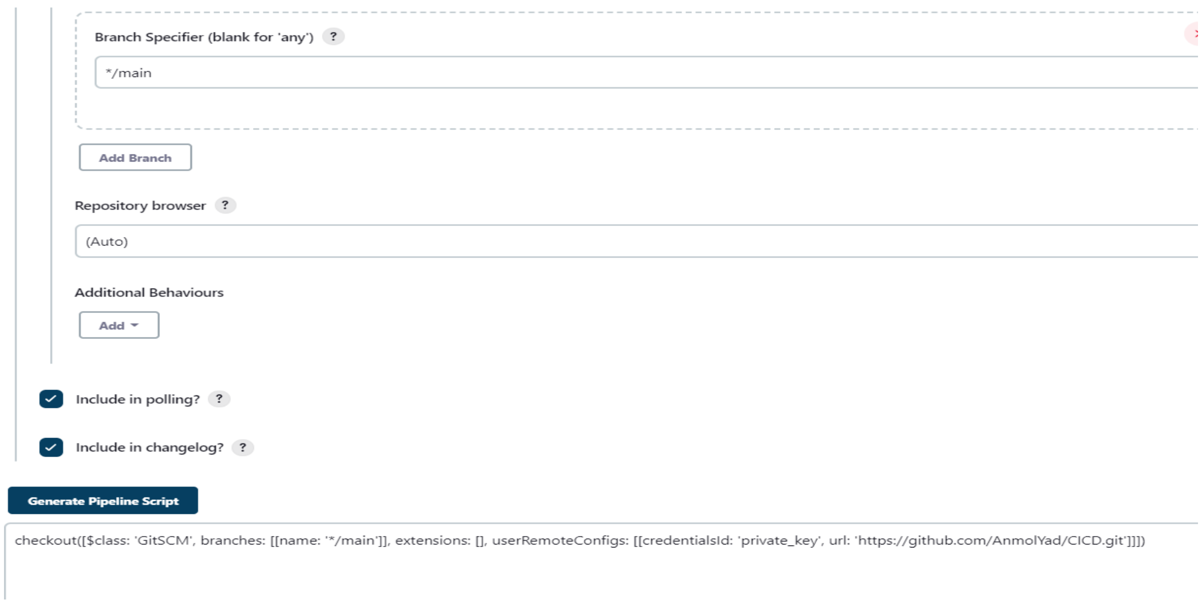
Branches to build

Branch Specifier (blank for 'any'): */main

Repository browser: (Auto)

Additional Behaviours

Step3: Generate pipeline script



Branch Specifier (blank for 'any'): */main

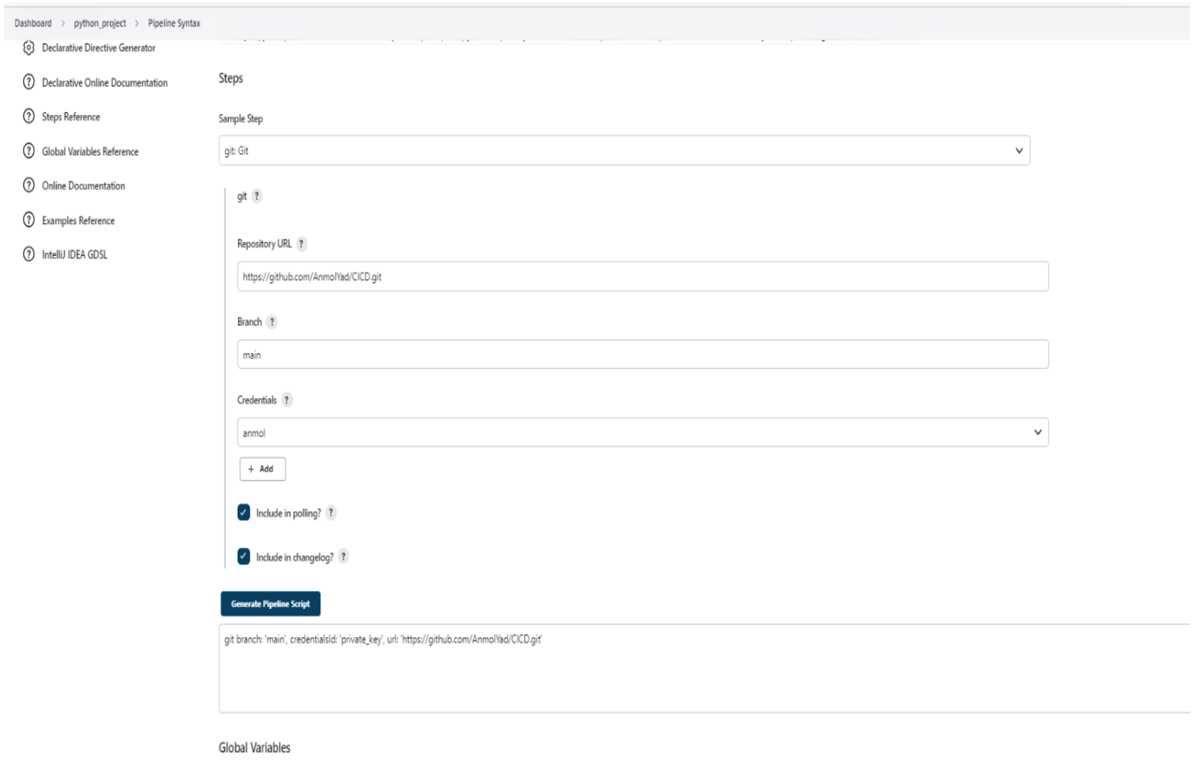
Repository browser: (Auto)

Additional Behaviours

✓ Include in polling?

✓ Include in changelog?

```
checkout([$class: 'GitSCM', branches: [[name: '*/main']], extensions: [], userRemoteConfigs: [[credentialsId: 'private_key', url: 'https://github.com/AnmolYad/CICD.git']]])
```


Dashboard > python_project > Pipeline Syntax
Declarative Directive Generator
Declarative Online Documentation
Steps Reference
Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL
Generate Pipeline Script
Global Variables

Steps

Sample Step: git: Git

git

Repository URL: https://github.com/AnmolYad/CICD.git

Branch: main

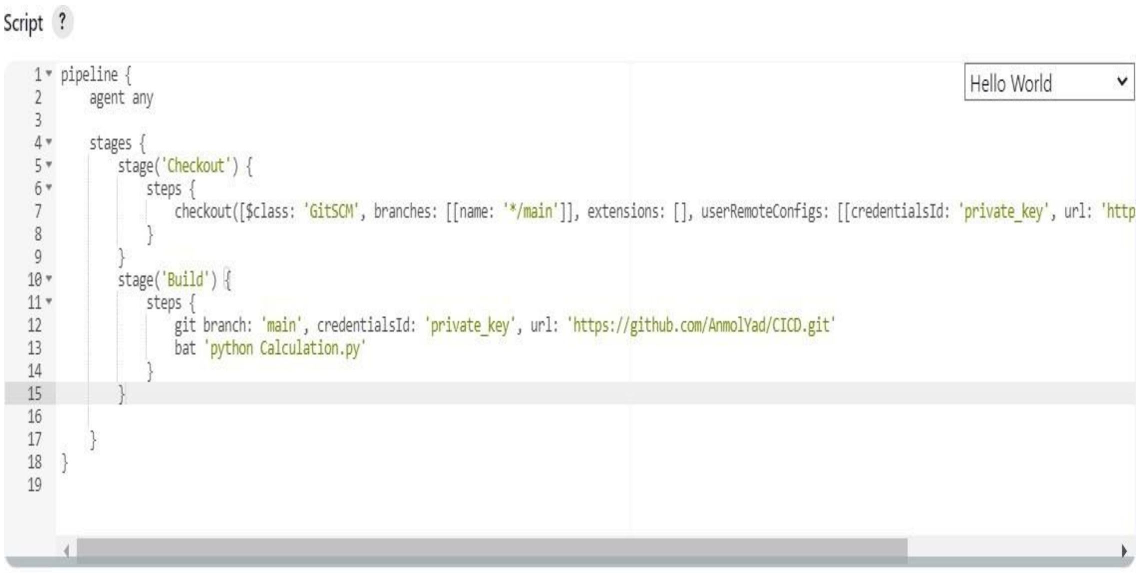
Credentials: anmol

+ Add

✓ Include in polling?

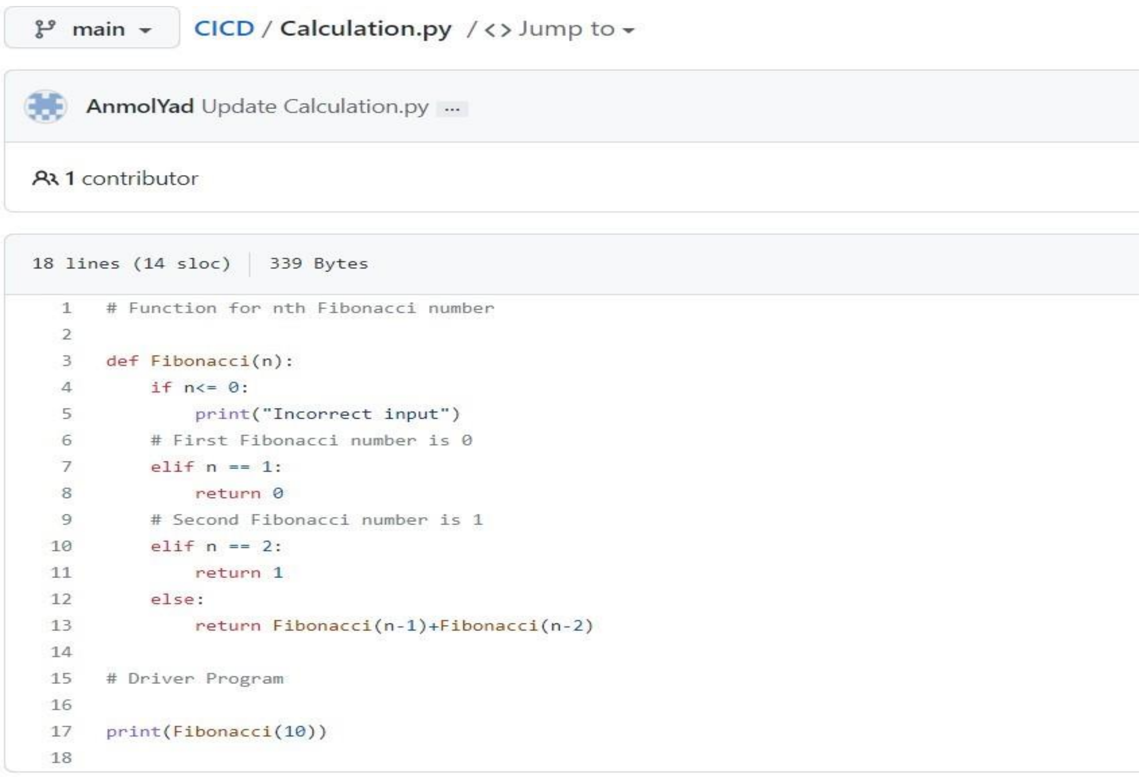
✓ Include in changelog?

```
git branch: 'main', credentialsId: 'private_key', url: 'https://github.com/AnmolYad/CICD.git'
```



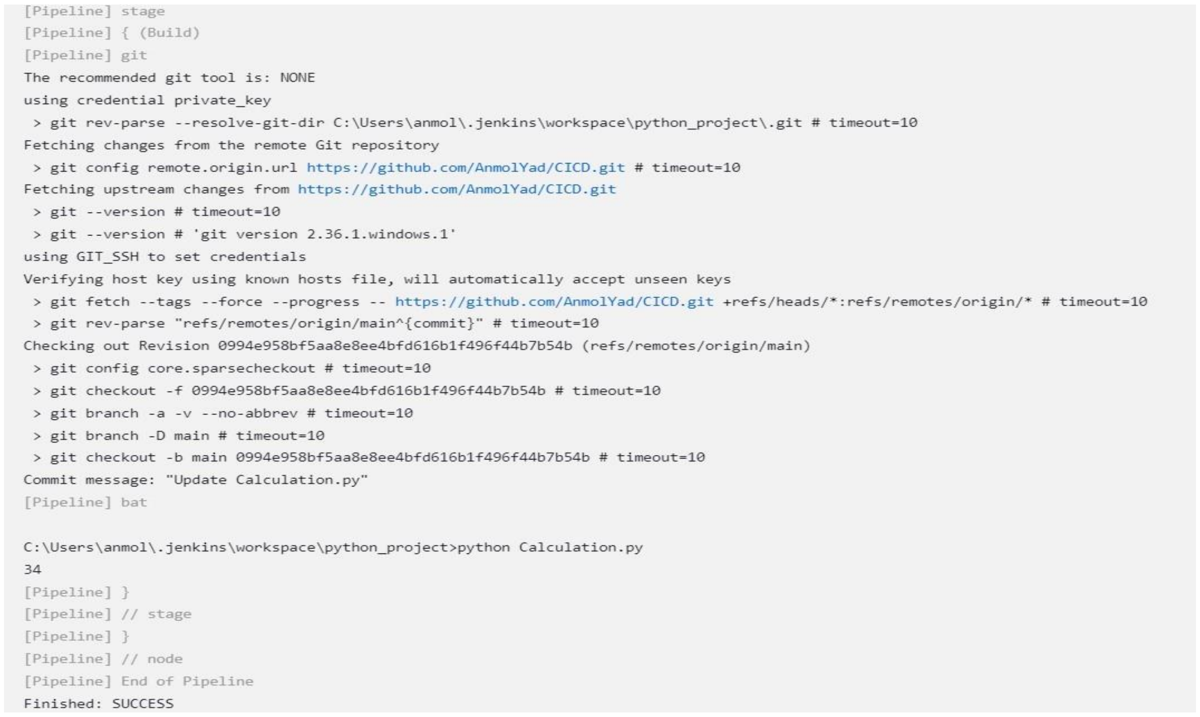
```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout([$class: 'GitSCM', branches: [[name: '*/main']], extensions: [], userRemoteConfigs: [[credentialsId: 'private_key', url: 'http
            }
        }
        stage('Build') {
            steps {
                git branch: 'main', credentialsId: 'private_key', url: 'https://github.com/AnmolYad/CICD.git'
                bat 'python Calculation.py'
            }
        }

    }
}
```

//python code


main
CICD / Calculation.py / <> Jump to
AnmolYad Update Calculation.py
1 contributor
18 lines (14 sloc) | 339 Bytes

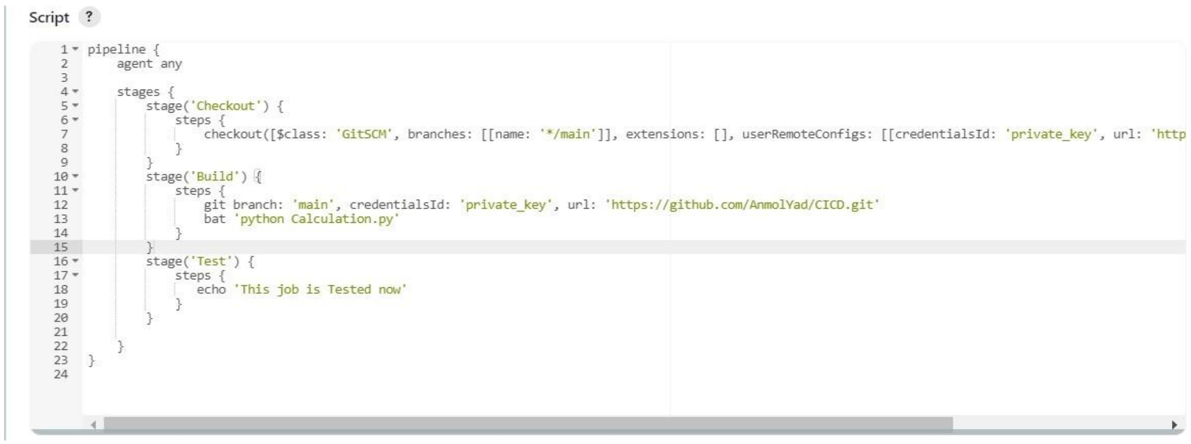
```python
# Function for nth Fibonacci number

def Fibonacci(n):
    if n<= 0:
        print("Incorrect input")
    # First Fibonacci number is 0
    elif n == 1:
        return 0
    # Second Fibonacci number is 1
    elif n == 2:
        return 1
    else:
        return Fibonacci(n-1)+Fibonacci(n-2)

# Driver Program

print(Fibonacci(10))

```

//build



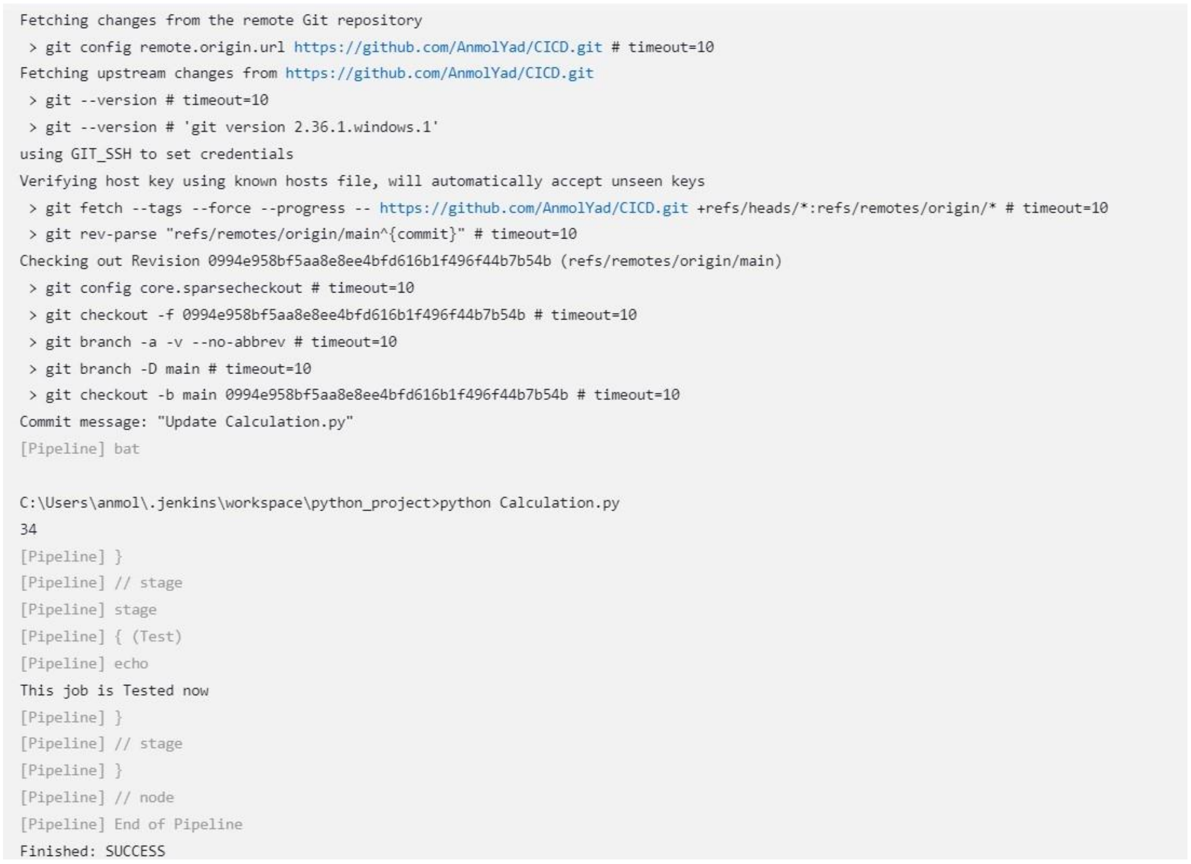
```
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] git
The recommended git tool is: NONE
using credential private_key
 > git rev-parse --resolve-git-dir C:\Users\anmol\.jenkins\workspace\python_project\.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/AnmolYad/CICD.git # timeout=10
Fetching upstream changes from https://github.com/AnmolYad/CICD.git
 > git --version # timeout=10
 > git --version # 'git version 2.36.1.windows.1'
using GIT_SSH to set credentials
Verifying host key using known hosts file, will automatically accept unseen keys
 > git fetch --tags --force --progress -- https://github.com/AnmolYad/CICD.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git branch -D main # timeout=10
 > git checkout -b main 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b # timeout=10
Commit message: "Update Calculation.py"
[Pipeline] bat

C:\Users\anmol\.jenkins\workspace\python_project>python Calculation.py
34
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

//add testing stage and build again



Script ?

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                checkout([$class: 'GitSCM', branches: [[name: '*/main']], extensions: [], userRemoteConfigs: [[credentialsId: 'private_key', url: 'http
            }
        }
        stage('Build') {
            steps {
                git branch: 'main', credentialsId: 'private_key', url: 'https://github.com/AnmolYad/CICD.git'
                bat 'python Calculation.py'
            }
        }
        stage('Test') {
            steps {
               echo 'This job is Tested now'
            }
        }

    }
}
```



```
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/AnmolYad/CICD.git # timeout=10
Fetching upstream changes from https://github.com/AnmolYad/CICD.git
 > git --version # timeout=10
 > git --version # 'git version 2.36.1.windows.1'
using GIT_SSH to set credentials
Verifying host key using known hosts file, will automatically accept unseen keys
 > git fetch --tags --force --progress -- https://github.com/AnmolYad/CICD.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse "refs/remotes/origin/main^{commit}" # timeout=10
Checking out Revision 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git branch -D main # timeout=10
 > git checkout -b main 0994e958bf5aa8e8ee4bfd616b1f496f44b7b54b # timeout=10
Commit message: "Update Calculation.py"
[Pipeline] bat

C:\Users\anmol\.jenkins\workspace\python_project>python Calculation.py
34
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
This job is Tested now
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```