

Experiment No-5

Aim: Experiment 5: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to Test and deploy an application over the tomcat server

Programming in Jenkins:

Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.” In simple way, Continuous integration (CI) is the practice of frequently building and testing each change done to your code automatically.

Jenkins is a self-contained, open-source automation server which can be used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Our first job will execute the shell commands. The freestyle project provides enough options and features to build the complex jobs that you will need in your projects.

Example 1

Example 1.1: Deploying a freestyle app in Jenkins

Creating a job:

Start building your software project

Create a job



Naming the job and setting it as freestyle:

Enter an item name

» Required field



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or 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-specific builds, 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 creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

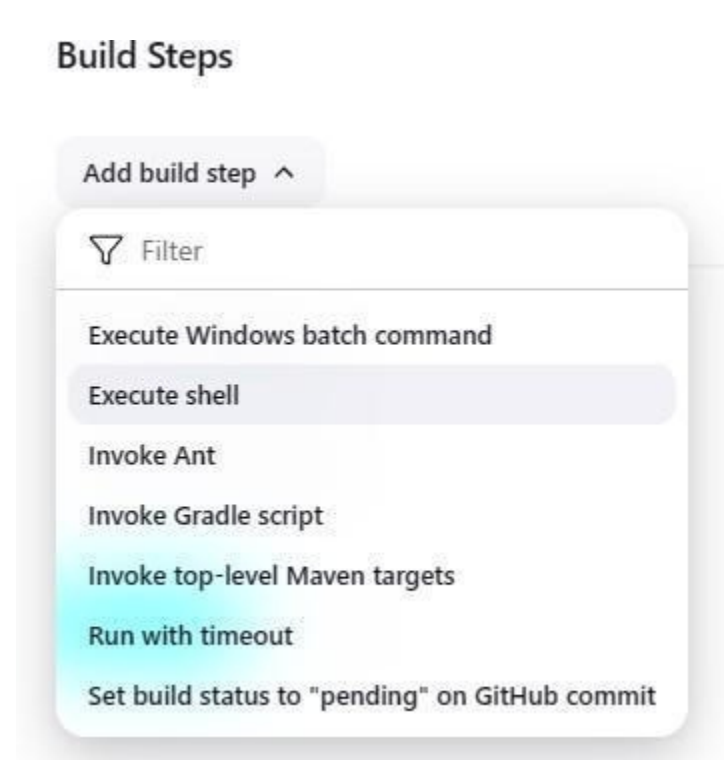


Organization Folder

Creates a set of multibranch project subfolders by scanning for repositories.

OK

Selecting build type as “Execute shell”:



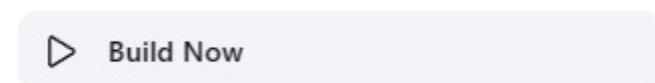
Entering a simple command for the shell execution:



Applying and saving the project configuration:



Building the project:

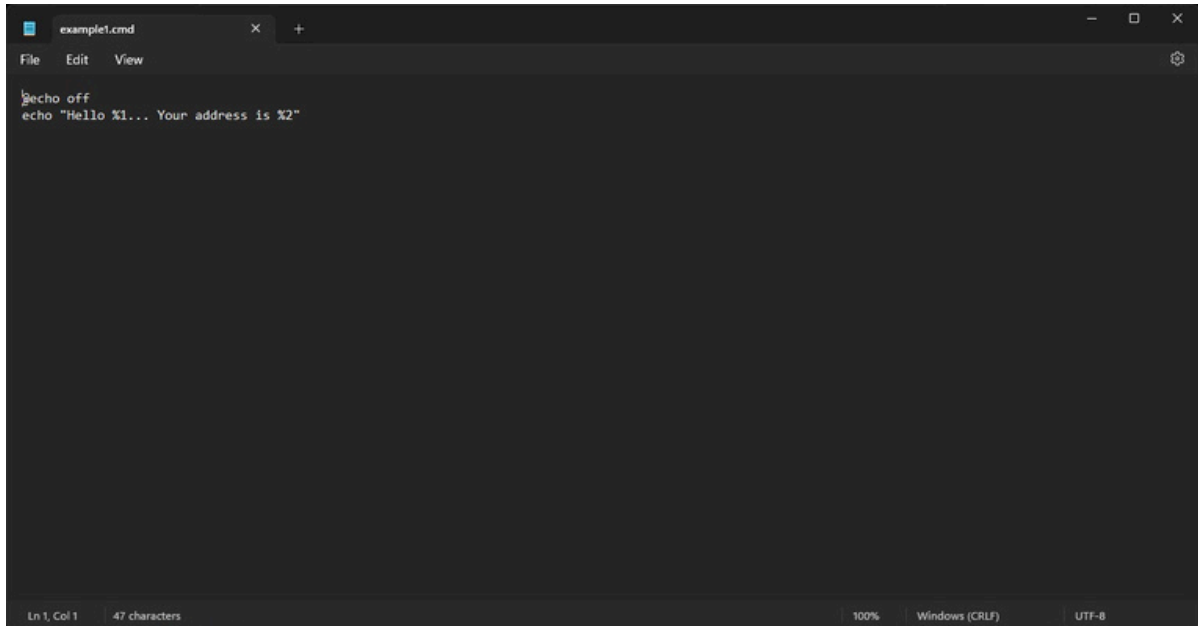


Console output (after building):

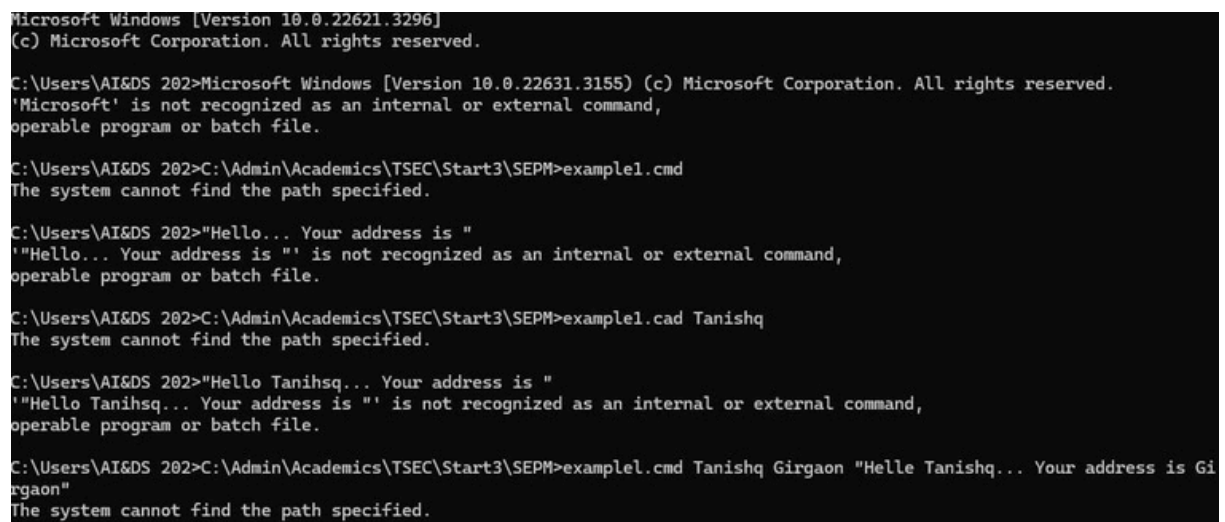


Example 1.2: Taking parameters through files

Contents of script example1.cmd:



Executing script example1.cmd on the terminal:



Modifying the Jenkins project to execute the script while supplying required parameters:

Build Steps

Execute Windows batch command ?

Command

See [the list of available environment variables](#)

C:\Admin\Academics\TSEC\Start3\SEPH\example1.cmd Siddhant Goregaon

Advanced ▾

Add build step ▾

Console output after building the modified project:

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build #4

Previous Build

✓ Console Output

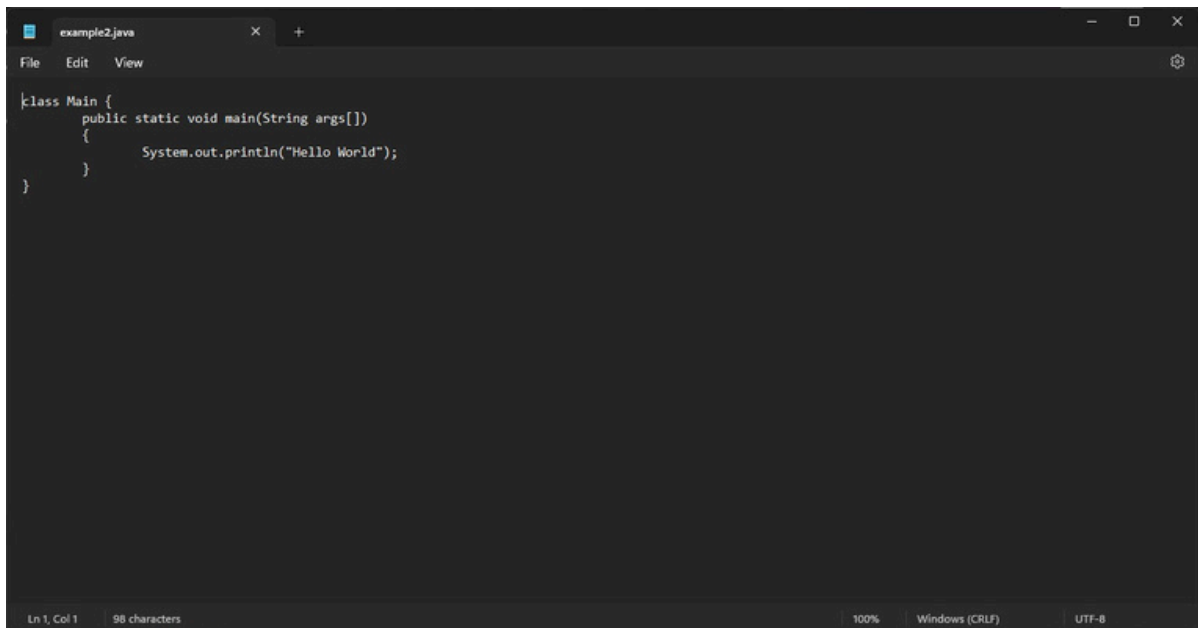
Started by user Siddhant Chetlur
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example1
[Example1] \$ cmd /c call C:\WINDOWS\TEMP\jenkins7075095818165161158.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example1>C:\Admin\Academics\TSEC\Start3\SEPH\example1.cmd Siddhant Goregaon
"Hello Siddhant... Your address is Goregaon"
Finished: SUCCESS

Example 2

Example 2.1: Running a Java program under Jenkins

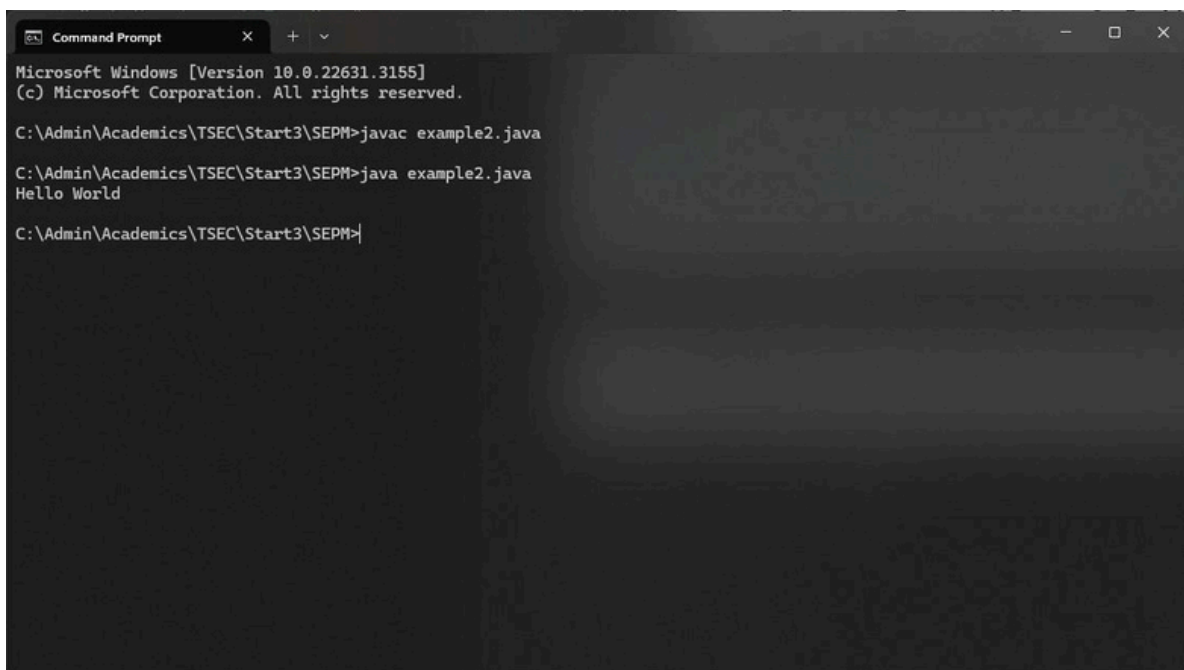
Creating a simple Java program:

A screenshot of a code editor window titled 'example2.java'. The editor contains the following Java code:

```
class Main {  
    public static void main(String args[])  
    {  
        System.out.println("Hello World");  
    }  
}
```

The status bar at the bottom indicates 'Ln 1, Col 1', '98 characters', '100%', 'Windows (CRLF)', and 'UTF-8'.

Compiling and running the program on the terminal:

A screenshot of a Windows Command Prompt window. The text displayed is as follows:

```
Microsoft Windows [Version 10.0.22631.3155]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Admin\Academics\TSEC\Start3\SEPM>javac example2.java  
  
C:\Admin\Academics\TSEC\Start3\SEPM>java example2.java  
Hello World  
  
C:\Admin\Academics\TSEC\Start3\SEPM>|
```

Creating a new freestyle project:

Dashboard > All >

Enter an item name

Example2

> Required field

- Freestyle project**
Classic general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or 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-specific builds, 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 creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

OK

Configure new project:

Build Steps

Execute Windows batch command ?

Command

See [the list of available environment variables](#)

```
javac C:\Admin\Academics\TSEC\Start3\SEPM\example2.java
java C:\Admin\Academics\TSEC\Start3\SEPM\example2.java
```

Advanced ▾

Add build step ▾

Console output after building:

Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example2
[Example2] $ cmd /c call C:\WINDOWS\TEMP\jenkins15296462484398614135.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example2>javac C:\Admin\Academics\TSEC\Start3\SEPM\example2.java

C:\ProgramData\Jenkins\jenkins\workspace\Example2>java C:\Admin\Academics\TSEC\Start3\SEPM\example2.java
Hello World

C:\ProgramData\Jenkins\jenkins\workspace\Example2>exit 0
Finished: SUCCESS
```


Example 3


Example 3.1: Parameterise build


Creating a new freestyle project:


Enter an item name


» Required field


**Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or 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-specific builds, 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 creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

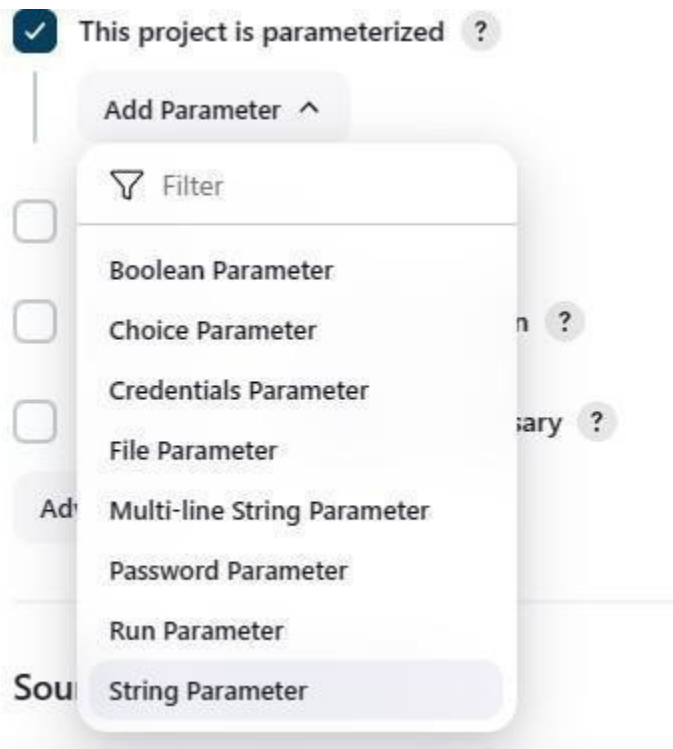
**Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

OK

Copy from

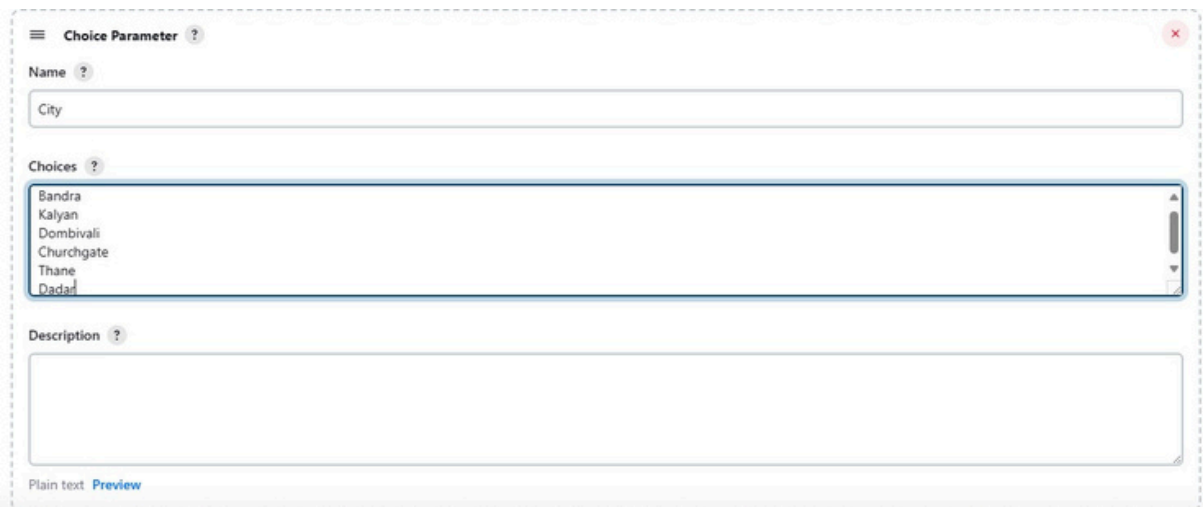
Enabling parameterisation and adding a String parameter:



Configuring the string parameter as Fname :

A screenshot of a configuration form for a 'String Parameter'. The form has a title bar with a hamburger menu icon, the text 'String Parameter', and a question mark icon. There is a red 'X' icon in the top right corner. The form contains several input fields: a 'Name' field with the value 'Fname', a 'Default Value' field, and a 'Description' field. Below these fields, there is a 'Plain text' label and a 'Preview' link. At the bottom, there is a checkbox labeled 'Trim the string' with a question mark icon.

Adding a choice parameter and configuring it as `City` with the following choices:



Choice Parameter ?

Name ?

City

Choices ?

- Bandra
- Kalyan
- Dombivali
- Churchgate
- Thane
- Dadar

Description ?

Plain text [Preview](#)

Creating a script which takes 2 arguments for name and city:

```
C:\Users\AI&DS 202>Microsoft Windows [Version 10.0.22631.3155] (c) Microsoft Corporation. All rights reserved.
'Microsoft' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH>example3.cnd
The system cannot find the path specified.

C:\Users\AI&DS 202>Hello your name is and your city is
'Hello' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH example3.cmd Tanishq
The system cannot find the path specified.

C:\Users\AI&DS 202>Hello your name is Tanishq and your city is
'Hello' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPM>example3.cmd Tansishq Bandra
The system cannot find the path specified.

C:\Users\AI&DS 202>Hello your name is Tanishq and your city is Bandra
'Hello' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\AI&DS 202>C:\Admin\Academics\TSEC\Start3\SEPH|
```

Configuring build steps:

Build Steps



Execute Windows batch command ?

Command

See [the list of available environment variables](#)

C:\Admin\Academics\TSEC\Start3\SEPM\example3.cmd %fname% %city%

Advanced ▾

Add build step ▾

Entering parameters for build:

Project Example3

This build requires parameters:

Fname

City

Console output after building:

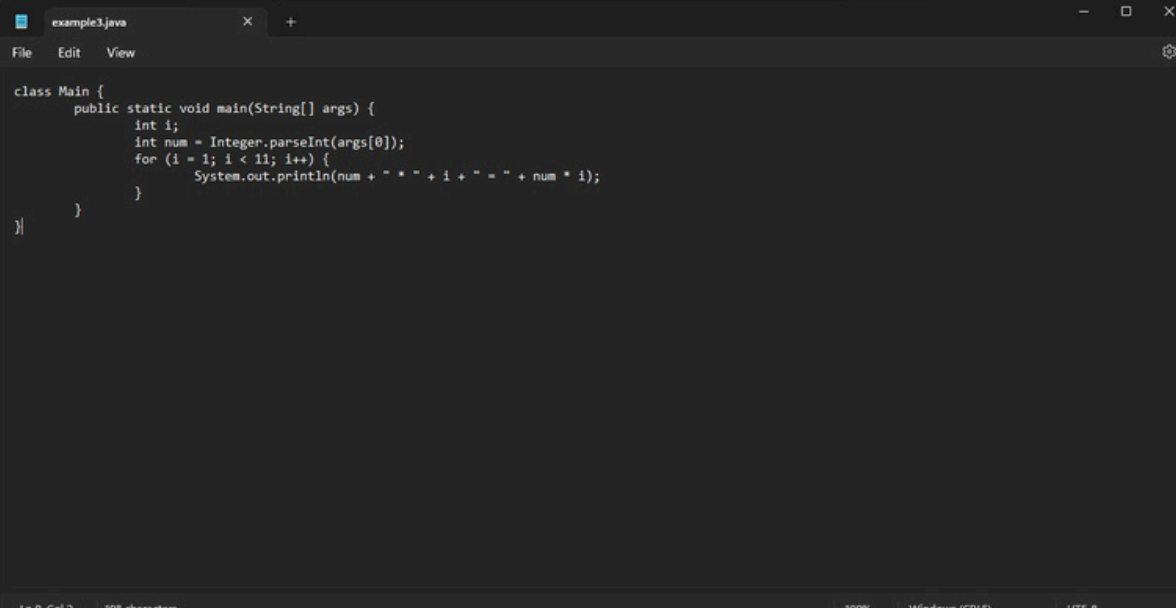
✓ Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example3
[Example3] $ cmd /c call C:\WINDOWS\TEMP\jenkins14094536165150986151.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example3>C:\Admin\Academics\TSEC\Start3\SEPH\example3.cmd Siddhant Bandra
Hello your name is Siddhant and your city is Bandra
Finished: SUCCESS
```

Example 3.2: Running a Java program with parameters

Creating a Java program with an input argument:



```
example3.java
File Edit View
class Main {
    public static void main(String[] args) {
        int i;
        int num = Integer.parseInt(args[0]);
        for (i = 1; i < 11; i++) {
            System.out.println(num + " * " + i + " = " + num * i);
        }
    }
}
```

Ln 9, Col 2 198 characters 100% Windows (CRLF) UTF-8

Testing the program on the terminal:

```
Command Prompt
Microsoft Windows [Version 10.0.22631.3155]
(c) Microsoft Corporation. All rights reserved.

C:\Admin\Academics\TSEC\Start3\SEPM>javac example3.java


C:\Admin\Academics\TSEC\Start3\SEPM>java example3.java 4
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40

C:\Admin\Academics\TSEC\Start3\SEPM>
```


Creating a new freestyle project:

Enter an item name


» Required field

**Freestyle project**


Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**


Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or 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-specific builds, 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 creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**

Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Parameterise the project by adding a string parameter as follows:

☒ This project is parameterized ?

String Parameter ?

Name ?

Default Value ?

Description ?

Plain text [Preview](#)

☐ Trim the string ?

Add Parameter ▾

Configure the build steps:

Build Steps

Execute Windows batch command ?

Command

[See the list of available environment variables](#)

```
javac C:\Admin\Academics\TSEC\Start3\SEPM\example3.java  
java C:\Admin\Academics\TSEC\Start3\SEPM\example3.java %num%
```

Advanced ▾

Add build step ▾

Entering the parameter for the build:

Project Example4

This build requires parameters:

num

Console output after building:



Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example4
[Example4] $ cmd /c call C:\WINDOWS\TEMP\jenkins15119185778823247788.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example4>javac C:\Admin\Academics\TSEC\Start3\SEPH\example3.java

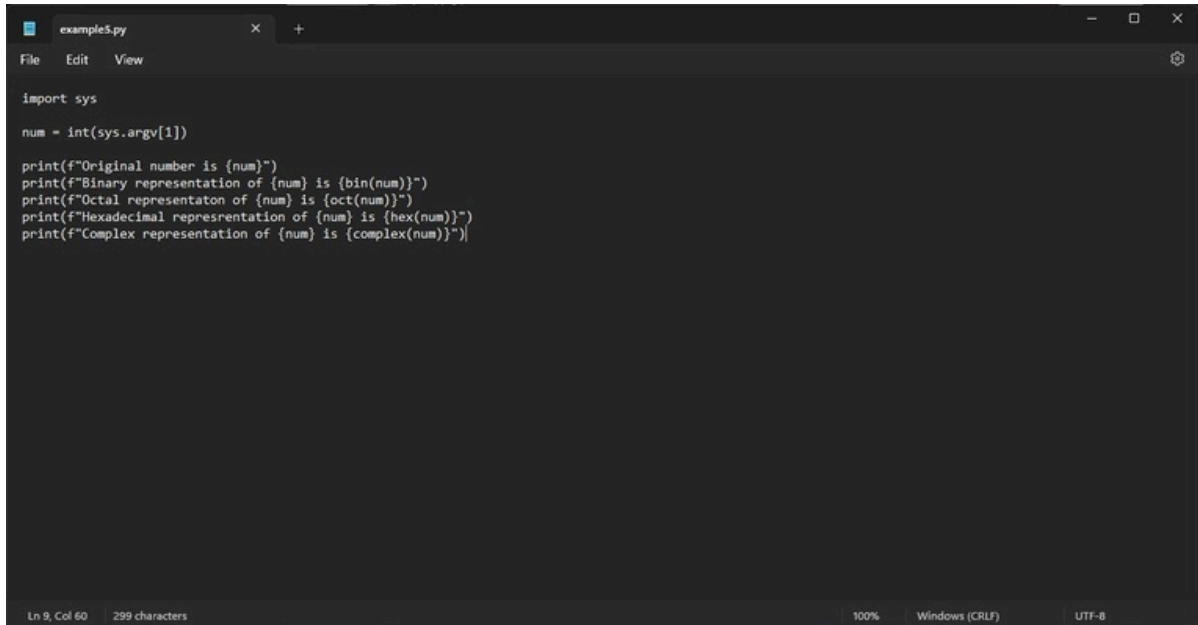
C:\ProgramData\Jenkins\jenkins\workspace\Example4>java C:\Admin\Academics\TSEC\Start3\SEPH\example3.java 25
25 * 1 = 25
25 * 2 = 50
25 * 3 = 75
25 * 4 = 100
25 * 5 = 125
25 * 6 = 150
25 * 7 = 175
25 * 8 = 200
25 * 9 = 225
25 * 10 = 250

C:\ProgramData\Jenkins\jenkins\workspace\Example4>exit 0
Finished: SUCCESS
```

Example 5

Example 5.1: Running a Python program

Creating a simple Python script:



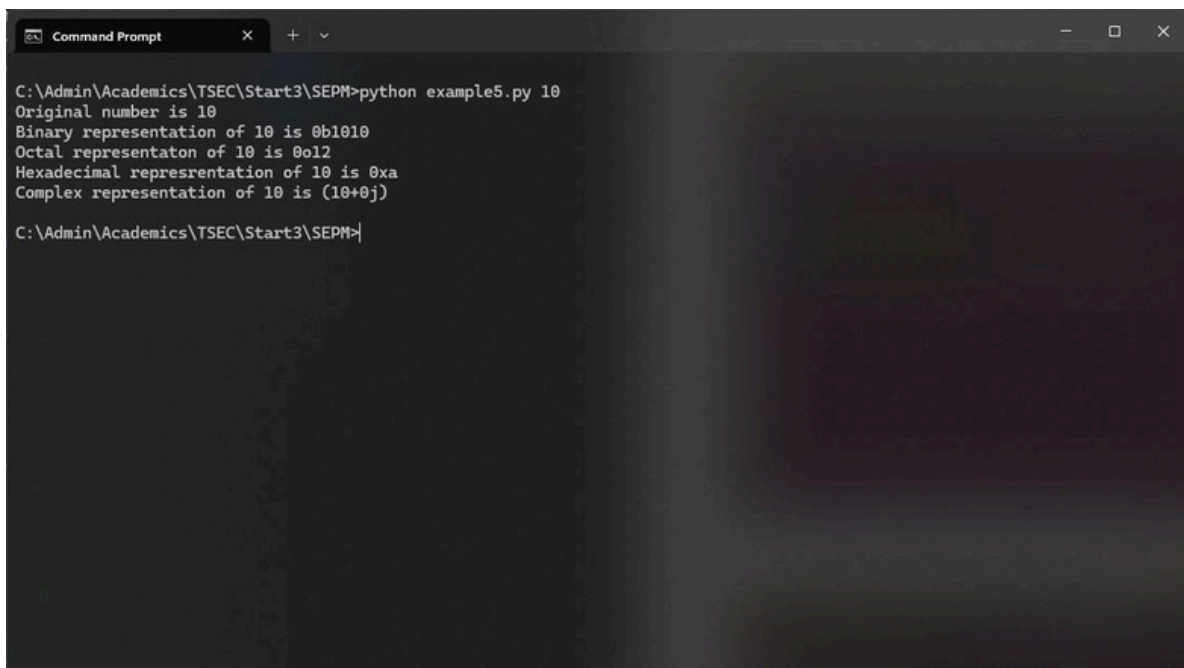
```
import sys

num = int(sys.argv[1])

print(f"Original number is {num}")
print(f"Binary representation of {num} is {bin(num)}")
print(f"Octal representation of {num} is {oct(num)}")
print(f"Hexadecimal representation of {num} is {hex(num)}")
print(f"Complex representation of {num} is {complex(num)}")
```

Ln 9, Col 60 299 characters 100% Windows (CRLF) UTF-8

Running the Python script on the terminal:




```
C:\Admin\Academics\TSEC\Start3\SEPM>python example5.py 10
Original number is 10
Binary representation of 10 is 0b1010
Octal representation of 10 is 0o12
Hexadecimal representation of 10 is 0xa
Complex representation of 10 is (10+0j)


C:\Admin\Academics\TSEC\Start3\SEPM>
```


Creating a new freestyle project:


Enter an item name


» Required field


**Freestyle project**
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

**Pipeline**
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or 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-specific builds, 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 creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

**Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.

**Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

Parameterising the project with a string parameter as follows:

☒ This project is parameterized ?

String Parameter ?

Name ?

Default Value ?

Description ?

Plain text [Preview](#)

☐ Trim the string ?

Add Parameter

Configuring the build steps:

Build Steps

Execute Windows batch command ?

Command

See [the list of available environment variables](#)

```
python C:\Admin\Academics\TSEC\Start3\SEPH\example5.py %num%
```

Advanced ▾

Add build step ▾

Setting the parameter for the build:

Project Example5

This build requires parameters:

num

▶ Build

Cancel

Console output after building:

✓ Console Output

```
Started by user Siddhant Chetlur
Running as SYSTEM
[EnvInject] - Loading node environment variables.
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example5
[Example5] $ cmd /c call C:\WINDOWS\TEMP\jenkins11157306491994478222.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example5>python C:\Admin\Academics\TSEC\Start3\SEPH\example5.py 10
Original number is 10
Binary representation of 10 is 0b1010
Octal representation of 10 is 0o12
Hexadecimal representation of 10 is 0xa
Complex representation of 10 is (10+0j)

C:\ProgramData\Jenkins\jenkins\workspace\Example5>exit 0
Finished: SUCCESS
```

Conclusion: Thus, we have successfully studied Continuous Integration and installed, configured, and understood programming with Jenkins.

