

Software Engineering and Project Management Lab Experiment No. 5

Aim: 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.

Theory:

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:


Software Engineering and Project Management Lab Experiment No. 5


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.

Selecting build type as “Execute shell”:

Build Steps

Add build step ^

 Filter

Execute Windows batch command

Execute shell

Invoke Ant

Invoke Gradle script

Invoke top-level Maven targets

Run with timeout

Set build status to "pending" on GitHub commit

Entering a simple command for the shell execution:

Software Engineering and Project Management Lab Experiment No. 5

Build Steps

≡ Execute shell ?

Command

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

```
echo "Hello TSEC"
```

Advanced ▾

Applying and saving the project configuration:

✓ Saved

Save

Apply

Building the project:

▶ Build Now

Console output (after building):

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build *3*

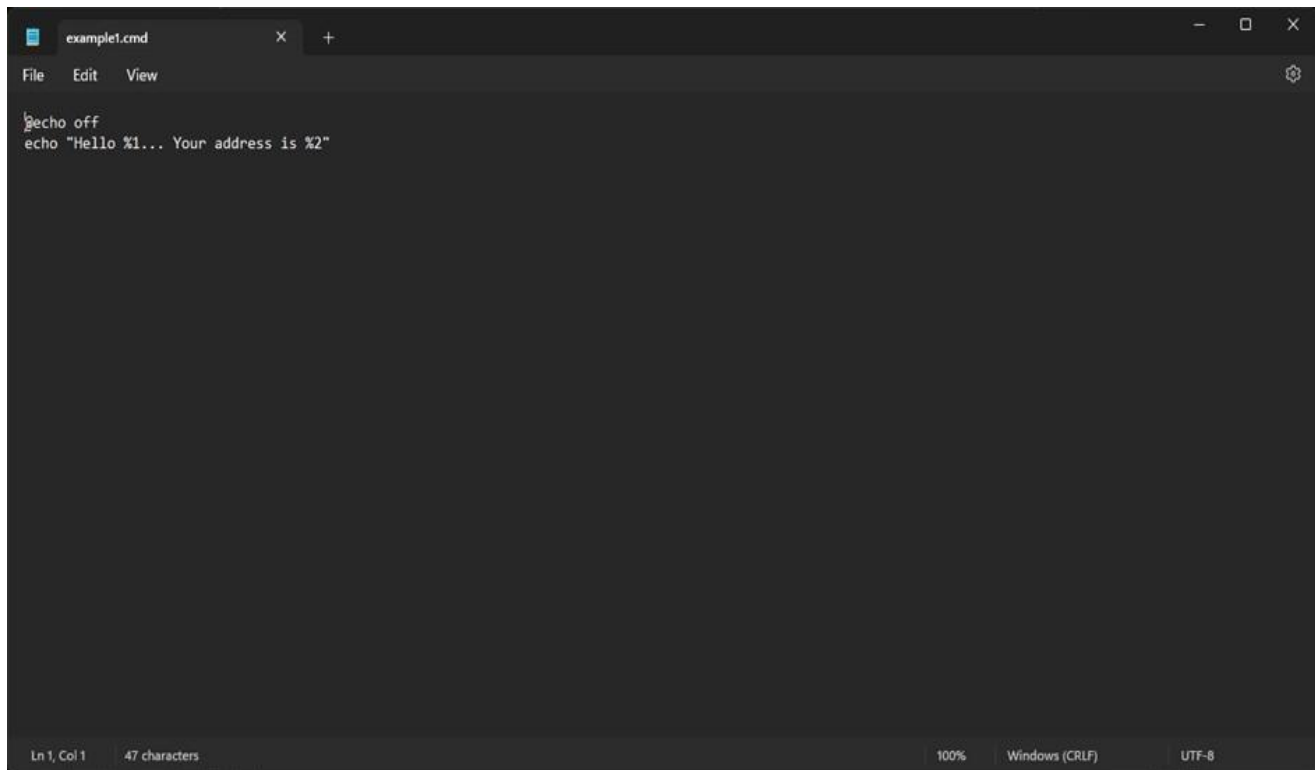
✓ Console Output

Started by user [siddhant Chatlur](#)
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\Example1
[Example1] \$ "C:\Program Files\Git\bin\sh.exe" -xe C:\WINDOWS\TEMP\jenkins55838525438548995.sh
+ echo 'Hello TSEC'
Hello TSEC
Finished: SUCCESS

Example 1.2: Taking parameters through files

Software Engineering and Project Management Lab Experiment No. 5

Contents of script example1.cmd:



The screenshot shows a text editor window titled 'example1.cmd'. The menu bar includes 'File', 'Edit', and 'View'. The text content is as follows:

```
echo off
echo "Hello %1... Your address is %2"
```

The status bar at the bottom indicates 'Ln 1, Col 1', '47 characters', '100%', 'Windows (CRLF)', and 'UTF-8'.

Executing script example1.cmd on the terminal:

```
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

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\SEPM>example1.cmd
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello... Your address is "
'"Hello... Your address is "' 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>example1.cad Tanishq
The system cannot find the path specified.

C:\Users\AI&DS 202>"Hello Tanihsq... Your address is "
'"Hello Tanihsq... Your address is "' 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>example1.cmd Tanishq Girgaon "Helle Tanishq... Your address is Gi
rgaon"
The system cannot find the path specified.
```

Software Engineering and Project Management Lab Experiment No. 5

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\SEPM\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 #8

Previous Build

✓ Console Output

Started by user Siddhant Chetiar
Running as SYSTEM
Building in workspace C:\ProgramData\Jenkins\jenkins\workspace\example1
[example1] \$ cmd /c call C:\WINDOWS\TEMP\Jenkins70790990218161118.bat

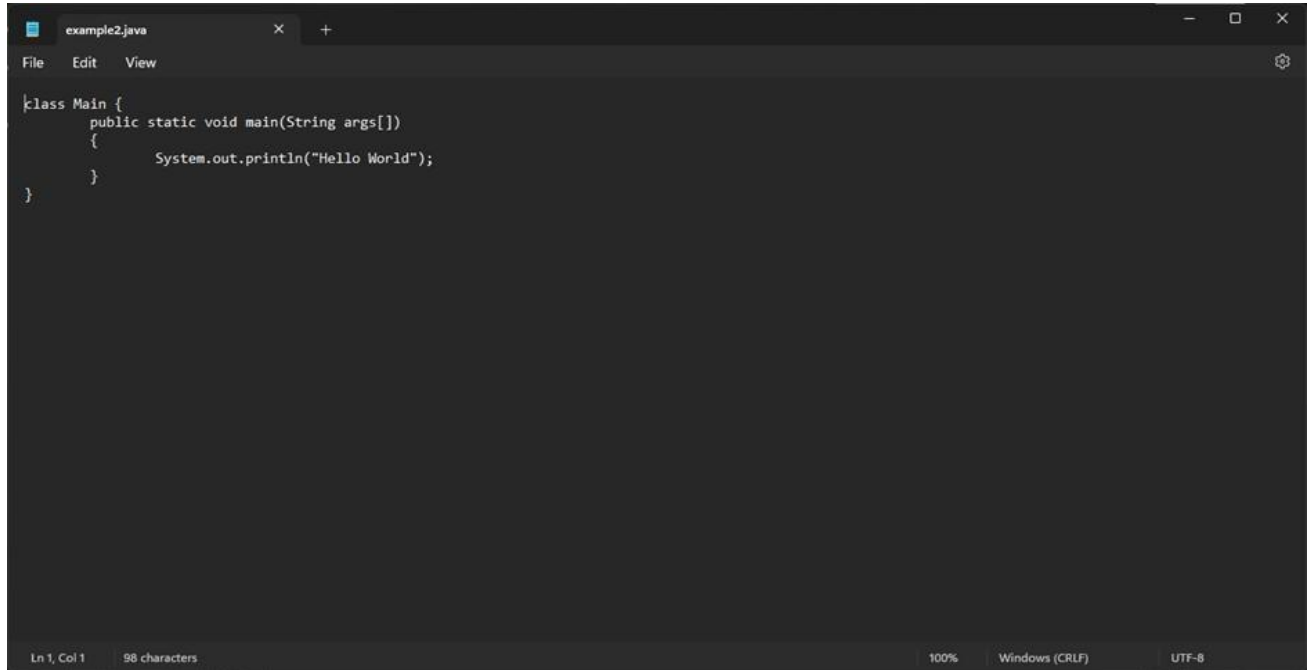
C:\ProgramData\Jenkins\jenkins\workspace\example1>C:\Admin\Academics\TSEC\Start3\SEPM\example1.cmd Siddhant Goregaon
"Hello Siddhant... Your address is Goregaon"
Finished: SUCCESS

Software Engineering and Project Management Lab Experiment No. 5

Example 2

Example 2.1: Running a Java program under Jenkins

Creating a simple Java program:

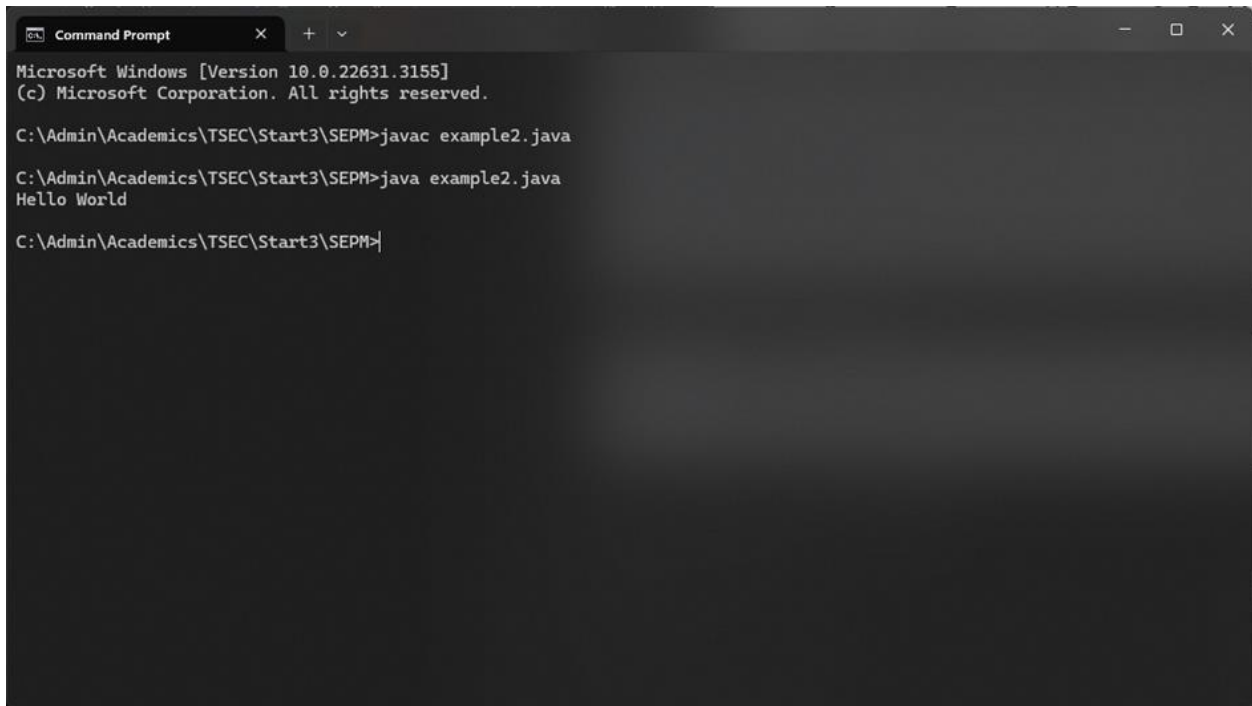


A screenshot of a code editor window titled 'example2.java'. The editor contains the following Java code:

```
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>|
```

Software Engineering and Project Management Lab Experiment No. 5

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 ▾

Software Engineering and Project Management Lab Experiment No. 5

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\SEPH\example2.java

C:\ProgramData\Jenkins\jenkins\workspace\Example2>java C:\Admin\Academics\TSEC\Start3\SEPH\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.

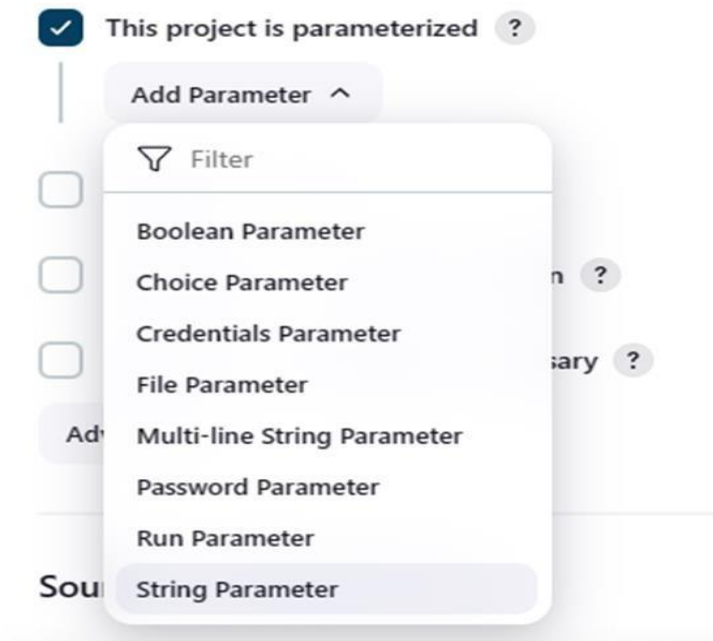
If you want to create a new item from other existing, you can use this option:

OK

Copy from

Enabling parameterisation and adding a String parameter:

Software Engineering and Project Management Lab Experiment No. 5



Configuring the string parameter as Fname:

A screenshot of a configuration form for a 'String Parameter'. The form is enclosed in a dashed border and has a title bar with a hamburger menu icon, the text 'String Parameter', and a help icon. A red 'X' icon is in the top right corner. The form contains the following fields: 'Name' with the value 'Fname', 'Default Value' (empty), and 'Description' (empty text area). Below these fields, there is a 'Plain text' label and a 'Preview' link. At the bottom, there is a checkbox labeled 'Trim the string'.

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

Software Engineering and Project Management Lab Experiment No. 5

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:

Software Engineering and Project Management Lab Experiment No. 5

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

Aditya

City

Bandra

▶ 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\Example3
[Example3] $ cmd /c call C:\WINDOWS\TEMP\jenkins14094536165150986151.bat

C:\ProgramData\Jenkins\jenkins\workspace\Example3>C:\Admin\Academics\TSEC\Start3\SEPM\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

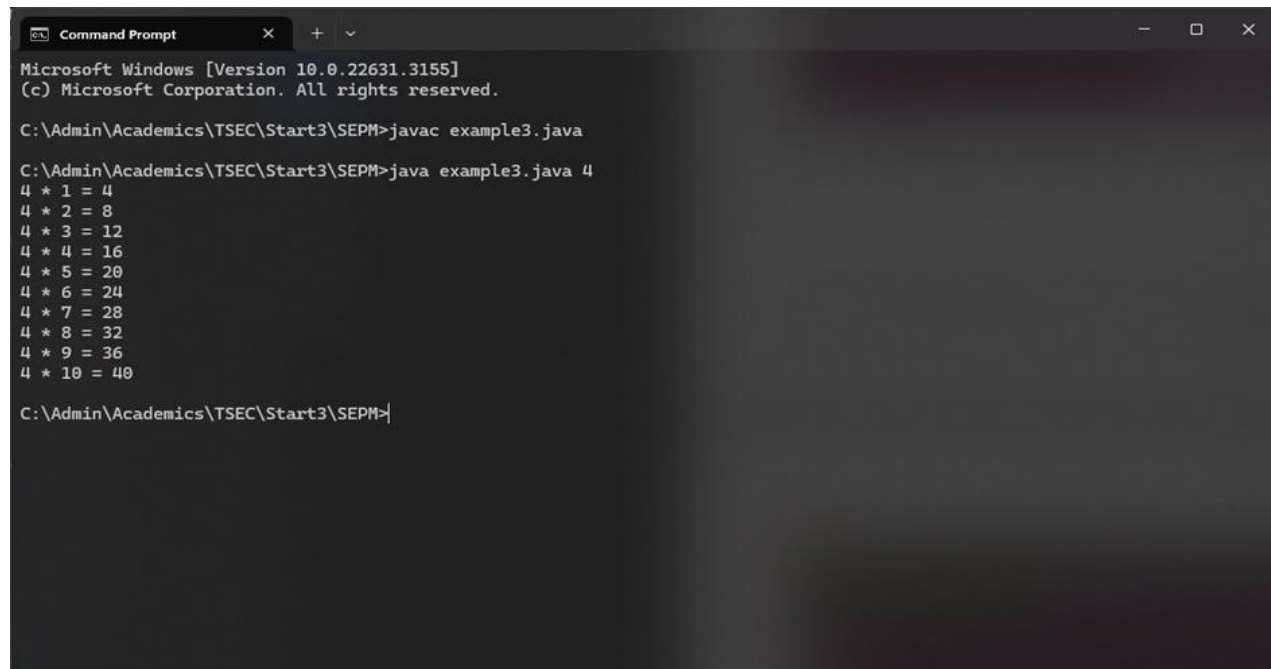
Creating a Java program with an input argument:

Software Engineering and Project Management Lab Experiment No. 5



```
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);  
        }  
    }  
}
```

Testing the program on the terminal:



```
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:

Software Engineering and Project Management Lab Experiment No. 5


Enter an item name

Example4


» 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

Parameterise the project by adding a string parameter as follows:

☒ This project is parameterized ?

String Parameter ?

Name ?

num

Default Value ?

Description ?

Plain text [Preview](#)

☐ Trim the string ?

Add Parameter ▾

Configure the build steps:

Software Engineering and Project Management Lab Experiment No. 5

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



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\Example4
[Example4] $ cmd /c call C:\WINDOWS\TEMP\jenkins15119185770823247708.bat

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

C:\ProgramData\Jenkins\jenkins\workspace\Example4>java C:\Admin\Academics\TSEC\Start3\SEPM\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

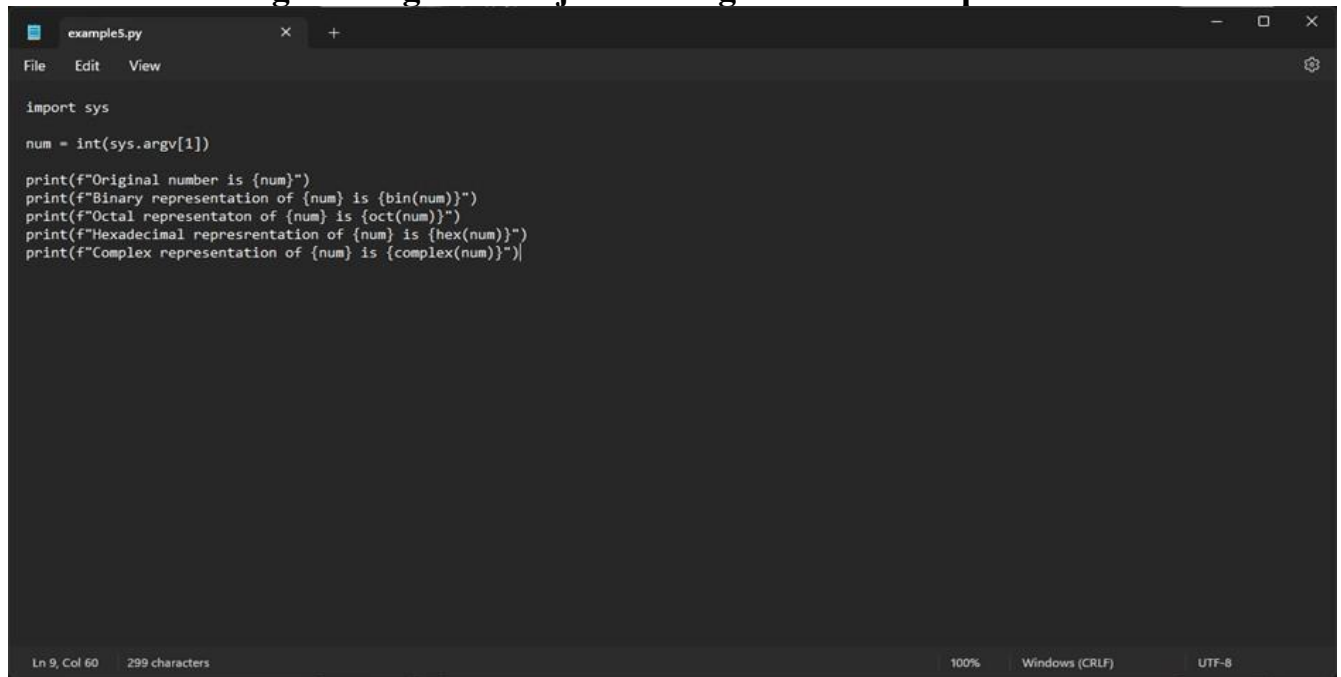
C:\ProgramData\Jenkins\jenkins\workspace\Example4>exit 0
Finished: SUCCESS
```

Example 5

Example 5.1: Running a Python program

Creating a simple Python script:

Software Engineering and Project Management Lab Experiment No. 5

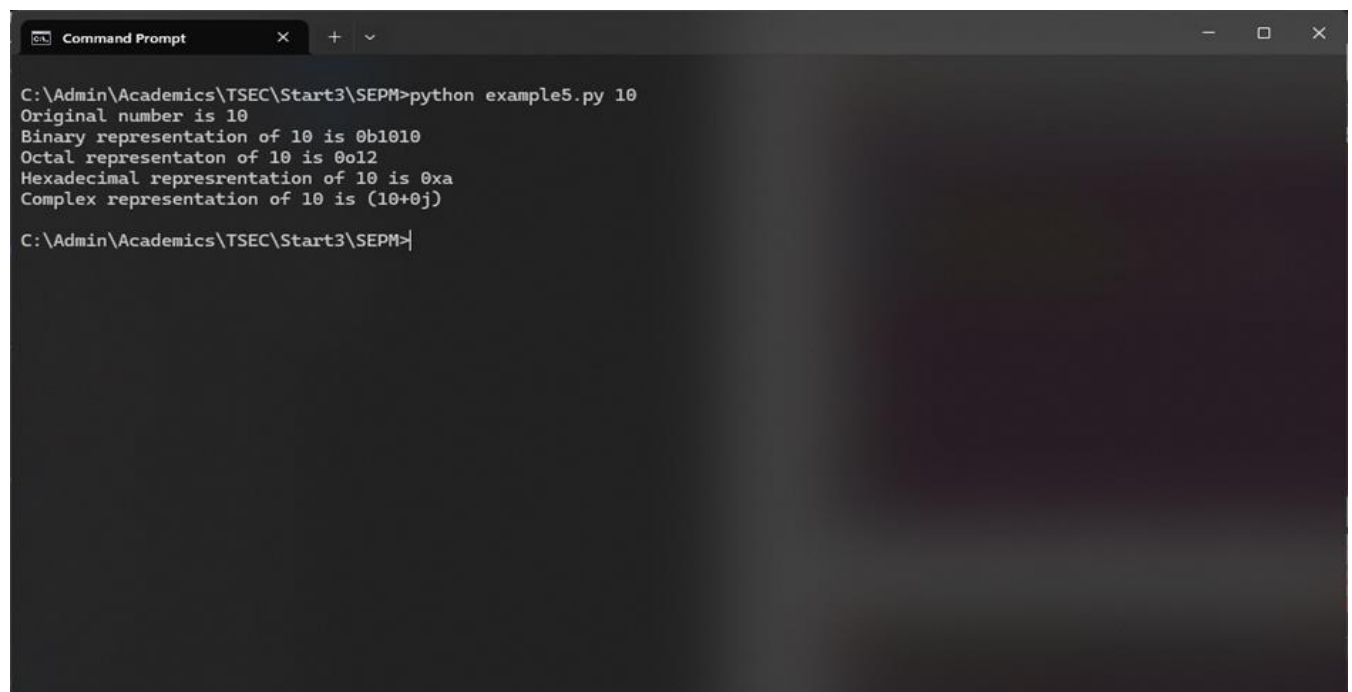
A screenshot of a text editor window titled 'example5.py'. The editor has a menu bar with 'File', 'Edit', and 'View'. The code inside is a Python script that takes a command-line argument and prints its various representations. The status bar at the bottom shows 'Ln 9, Col 60', '299 characters', '100%', 'Windows (CRLF)', and 'UTF-8'.

```
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)}")
```

Running the Python script on the terminal:

A screenshot of a Windows Command Prompt window titled 'Command Prompt'. It shows the execution of the Python script 'example5.py' with the argument '10'. The output displays the original number and its binary, octal, hexadecimal, and complex representations. The prompt is currently at the directory 'C:\Admin\Academics\TSEC\Start3\SEPM'.

```
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:


Software Engineering and Project Management Lab Experiment No. 5


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 ?

Configuring the build steps:

Software Engineering and Project Management Lab Experiment No. 5

Build Steps

Execute Windows batch command ?

Command

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

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

Some Screenshots:

Software Engineering and Project Management Lab Experiment No. 5

Configure

- General
- Build Triggers
- Advanced Project Options
- Pipeline

Advanced

Pipeline

Definition

Pipeline script


```
1 pipeline {
2   agent any
3
4   stages {
5     stage('Build') {
6       steps {
7         echo 'building.. This is the build phase'
8       }
9     }
10    stage('Test') {
11      steps {
12        echo 'testing.. This is the testing phase'
13      }
14    }
15    stage('Deploy') {
16      steps {
17        echo 'deploying... This is the deployment phase'
18      }
19    }
20    stage('Postdeploy') {
21      steps {
22        echo 'postdeployment phase....'
23      }
24    }
25  }
26 }
27 }
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Save

Apply

 **Jenkins**

Dashboard > AIDS_Pipeline >

Status

<> Changes

> Build Now

Configure

Delete Pipeline

Full Stage View

Rename

Pipeline Syntax

Buids

Filter

February 27, 2023

#1 1:58 PM

AIDS_Pipeline

Add description

Stage View

Average stage times:
(Average full run time: ~507ms)

	Build	Test	Deploy	Postdeploy
Feb 27 11:58	46ms	44ms	34ms	43ms

Permalinks

- Last build (#1), 2 yr 0 mo ago
- Last stable build (#1), 2 yr 0 mo ago
- Last successful build (#1), 2 yr 0 mo ago
- Last completed build (#1), 2 yr 0 mo ago

localhost:8080/jobs/AIDS_Pipeline/1

REST API Jenkins 2.402.2

Software Engineering and Project Management Lab Experiment No. 5

The Jenkins Dashboard for the **AIDS_Pipeline** shows the following information:

- Status:** ✔ AIDS_Pipeline
- Stage View:** A table showing stage execution times for two builds.
- Builds:** A list of recent builds with their status and timestamps.
- Permalinks:** A list of links to specific build stages.

	Build	Test	Deploy	Postdeploy
Average stage times: (Average full run time: ~1s)	103ms	48ms	50ms	54ms
Mar 11 13:48	160ms	53ms	66ms	66ms
Feb 27 13:58	46ms	44ms	34ms	43ms

Permalinks

- Last build (#1), 2 yr 0 mo ago
- Last stable build (#1), 2 yr 0 mo ago
- Last successful build (#1), 2 yr 0 mo ago
- Last completed build (#1), 2 yr 0 mo ago

The Jenkins Pipeline Steps view for the **AIDS_Pipeline** shows the following steps:

Step	Arguments	Status
Start of Pipeline - (2.1 sec in block)		✔
node - (0.78 sec in block)		✔
node block - (0.51 sec in block)		✔
stage - (0.19 sec in block)	Build	✔
stage block (Build) - (0.13 sec in block)		✔
echo - (13 ms in self)	Building... This is the build phase	✔
stage - (68 ms in block)	Test	✔
stage block (Test) - (39 ms in block)		✔
echo - (13 ms in self)	Testing... This is the testing phase	✔
stage - (80 ms in block)	Deploy	✔
stage block (Deploy) - (40 ms in block)		✔
echo - (1 ms in self)	Deploying... This is the deployment phase	✔
stage - (67 ms in block)	Postdeploy	✔
stage block (Postdeploy) - (40 ms in block)		✔
echo - (2 ms in self)	Postdeployment phase...	✔

Software Engineering and Project Management Lab Experiment No. 5

Dashboard > AIDS_Pipeline > #2

Console Output

Started by user admin

```
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in C:\ProgramData\jenkins\workspace\AIDS_Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Building.. This is the build phase
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
Testing.. This is the testing phase
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Deploy)
[Pipeline] echo
Deploying.... This is the deployment phase
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Postdeploy)
[Pipeline] echo
Postdeployment phase....
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

REST API Jenkins 2.492.2

Stage Logs (Build)

Print Message -- Building.. This is the build phase (self time 13ms)

Building.. This is the build phase

Stage View

	Build	Test	Deploy	Postdeploy
Average stage times: (Average full run time ~1s)	91ms	50ms	51ms	58ms
#3 Mar 11 12:47 No Changes	69ms	53ms	54ms	67ms
#2 Mar 11 12:46 No Changes	160ms	53ms	66ms	66ms
#1 Feb 27 12:58 No Changes	46ms	44ms	34ms	43ms

Permalinks

- Last build (#3), 27 ms ago
- Last stable build (#2), 1 min 40 sec ago
- Last successful build (#2), 1 min 40 sec ago
- Last completed build (#2), 1 min 40 sec ago

Conclusion:

Thus, we have successfully Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, created a pipeline script to Test and deploy an application over the tomcat server.