1 Procedures for Running the Algorithms

1.1 1. Python

To run the Python code, you need to have Python installed and set up an Anaconda environment with the necessary libraries.

Steps:

Install Python:

Download and install **Anaconda** following the instructions for your operating system from `https://www.anaconda.com/products/distribution`. **Create a Conda Environment:**

Open the terminal (or Anaconda Prompt) and create a new environment with the following command:

```
conda create --name matrix-env python=3.9
```

Activate the created environment:

```
conda activate matrix-env
```

Install Necessary Libraries:

Install the required libraries (e.g., NumPy for matrix manipulation):

```
pip install numpy
```

Run the Python Code:

Navigate to the folder containing your Python script (e.g., `matrix.py`) and execute the following command:

```
python matrix.py
```

The results will be displayed in the console.

1.2 2. C

To compile and run the C code, you need to have the necessary libraries installed.

Steps:

Install a C Compiler:

Ensure you have a C compiler such as **MinGW** for Windows. During installation, select options to include the binary files in your PATH. **Compile the C Code:**

Open the command prompt (cmd) and navigate to the folder containing your C file (e.g., `matrix.c`). Compile the code with the following command:

```
gcc matrix.c -o matrix.exe
```

Run the C Code:

After compilation, execute the generated file:

```
matrix.exe
```

Compare Results:

Once the code is running, the results of the matrix multiplication will be displayed in the terminal.

1.3 3. Java

Running the Java code is different as it uses JMH (Java Microbenchmark Harness) for benchmarking. You need to install Maven and configure the project.

Steps:

Install Maven:

Download Maven from `https://maven.apache.org/download.cgi` and follow the instructions for setting it up on your system. **Create a Maven Folder:**

Open the terminal and create a new folder for your project:

```
mvn archetype
-DgroupId=com.example -DartifactId=matrix-benchmark -DarchetypeArtifactId=maven-archetype-qu
```

This command creates a predefined directory structure for a Maven project. **Navigate to the Project Folder:**

Change into the created folder:

```
cd matrix-benchmark
```

Project Structure:

The folder contains subdirectories like `src/main/java` for source code and `src/test/java` for tests. You can place your benchmarking code inside `src/main/java/com/example`. **Use Maven Commands:**

Run the following commands to prepare the project:

```
mvn clean install package
```

Explanation of commands: `clean`: removes previous build files. `install`: compiles the project and installs the package in the local repository. `package`: creates an executable JAR file for the project. **Run the JAR:**

To execute the benchmark, use the command:

```
java -jar target/matrix-benchmark-1.0-SNAPSHOT.jar
```

Warmup and Interactions:

Warmup executions prepare the JVM and reduce noise in the results. Interactions are the actual benchmarking tests during which the performance of the algorithm is measured.