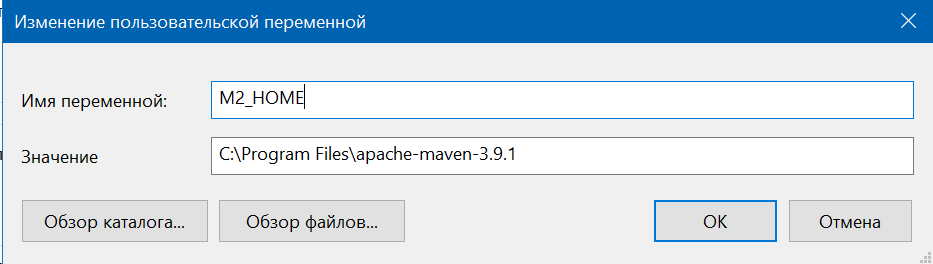
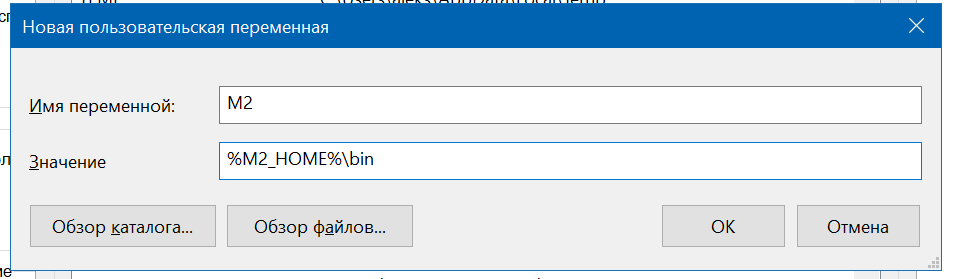
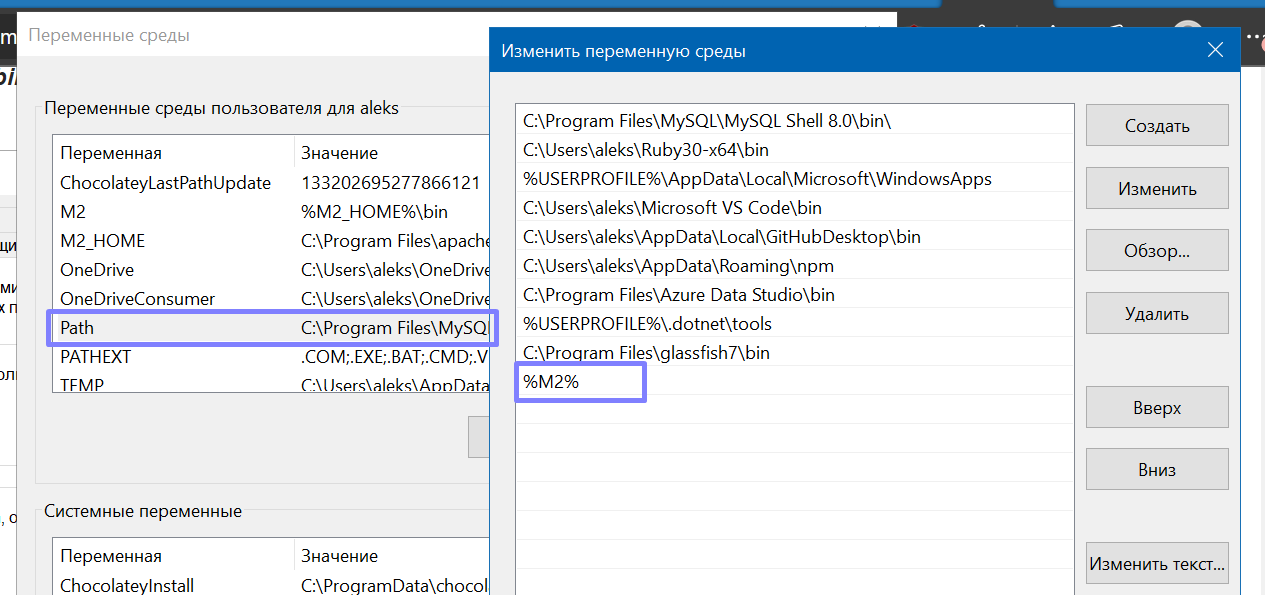
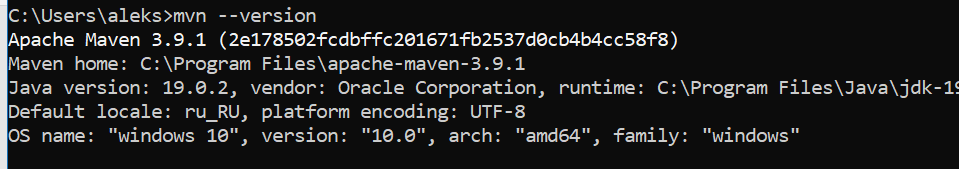
1)Как работать через cmd?

Необходимо внести данные изменения в переменных средах. Для path мы добавляем переменную для работы с cmd

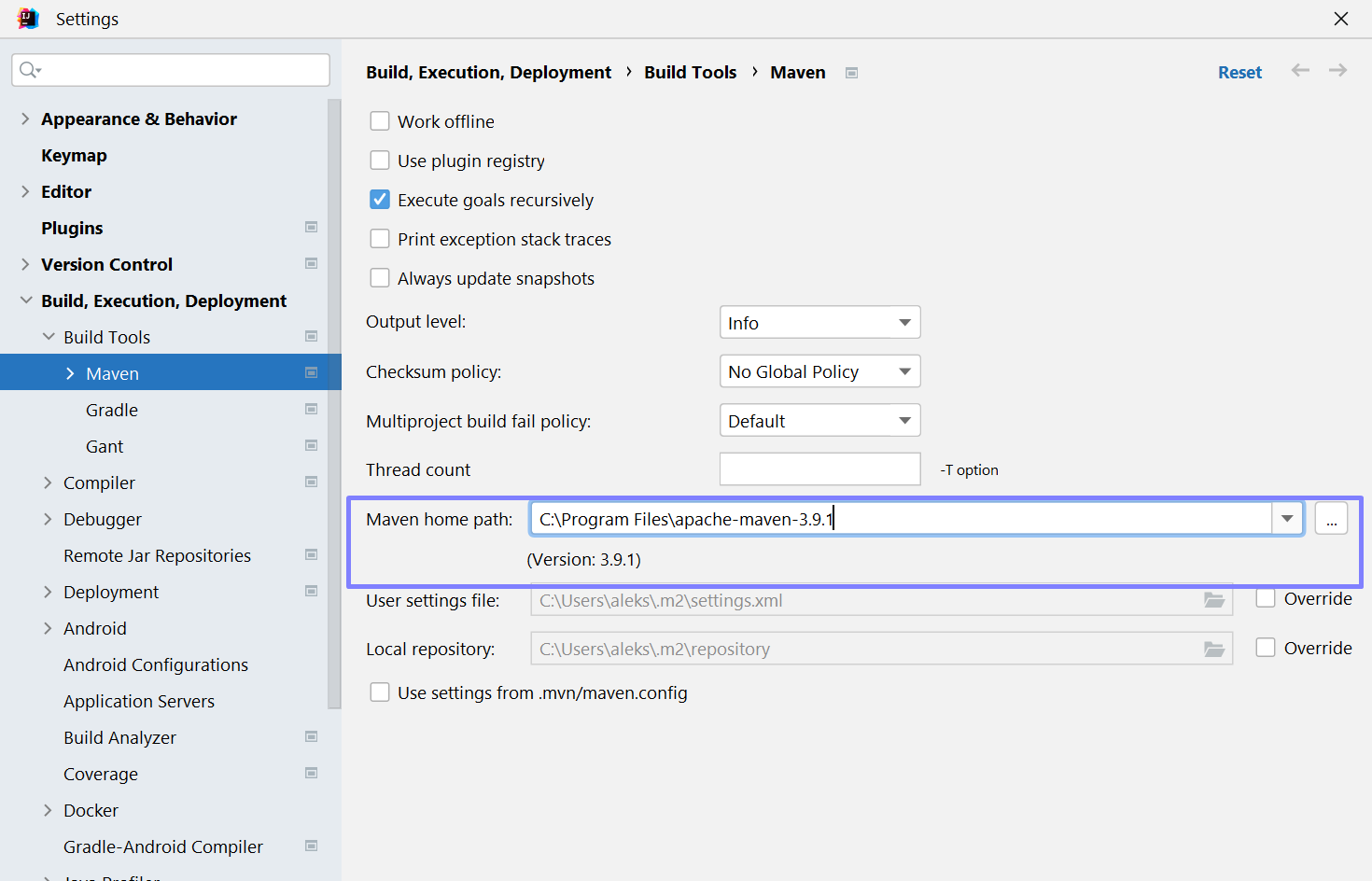






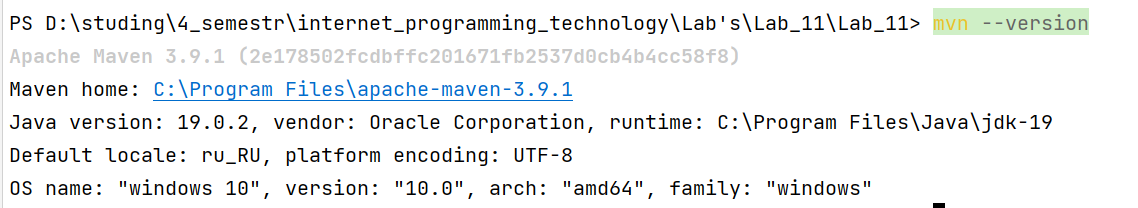


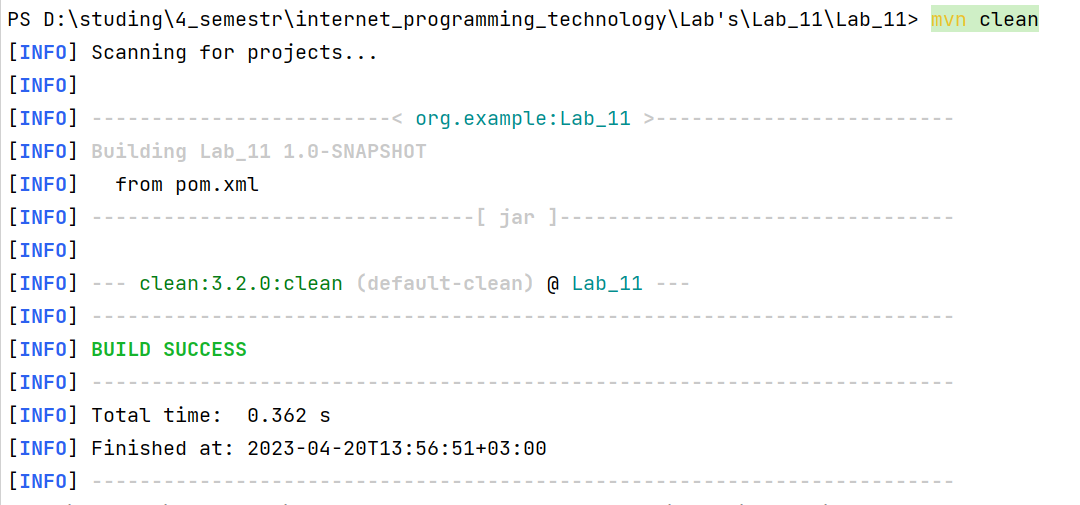
2.Заменяем в проекте maven на свой:



3.Через terminal INTELIJ выполняем команды:

1. **mvn -version**: Эта команда выводит информацию о версии установленного на вашей машине Maven, включая версию Maven и версию Java, которая используется для выполнения Maven.
2. **mvn clean**: Эта команда удаляет все сгенерированные файлы, такие как скомпилированный код, артефакты сборки, документацию и т.д. Она используется для очистки проекта перед сборкой и обычно выполняется в начале каждой новой сборки.
3. **mvn compile**: Эта команда компилирует исходный код проекта. Она используется для проверки того, что все исходные файлы компилируются без ошибок и готовы для дальнейшей сборки.
4. **mvn test**: Эта команда запускает все тесты в проекте. Она используется для проверки правильности работы приложения и выявления ошибок. Maven автоматически находит и запускает все тестовые классы в проекте, используя определенный вами фреймворк для тестирования.
5. **mvn install**: Эта команда компилирует исходный код, создает артефакты сборки (например, JAR-файлы) и устанавливает их в локальный репозиторий Maven. Это позволяет другим проектам на вашей машине использовать ваш проект как зависимость.
6. **mvn deploy**: Эта команда выполняет те же действия, что и команда **mvn install**, за исключением того, что артефакты сборки копируются в удаленный репозиторий, который находится на сервере и доступен другим пользователям. Это позволяет другим пользователям использовать ваш проект как зависимость в своих собственных проектах.

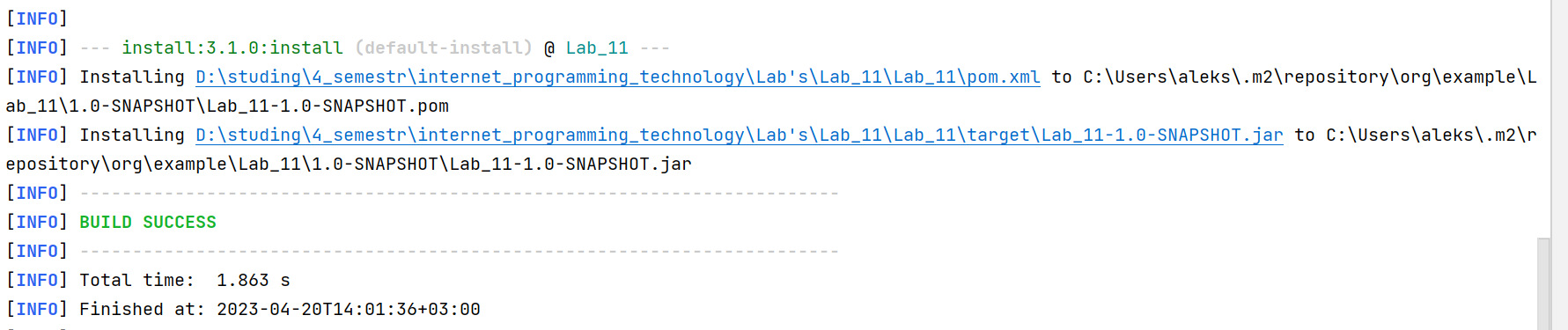










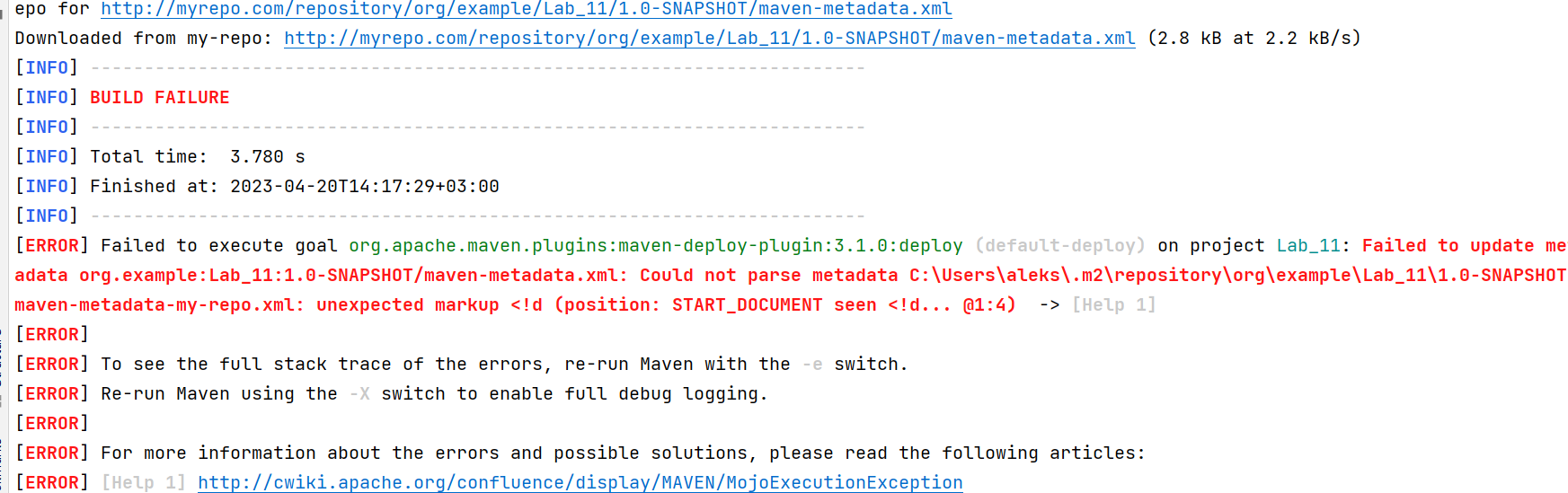


Для удаленного репозитория необходимо его настроить и внести изменения в pom.xml:

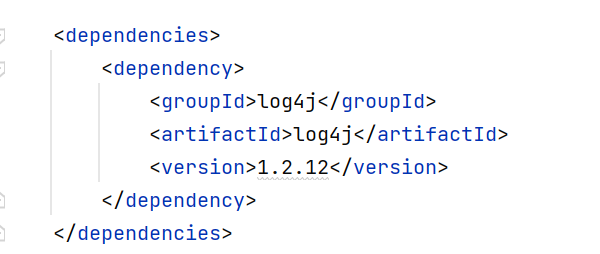


Поскольку удаленный репозиторий не настроен (что и не требуется по заданию), утилита mvn deploy корректно работать не будет:





4.Для добавления номеров версий зависимости нам необходимо подключить какую-нибудь зависимость (например jar), затем внести изменения в pom.xml следующий образом (демонстрация на примере jar-файла log4j):



5.Добавление плагинов (плагины - это инструменты, которые используются в Maven для расширения функциональности сборки проектов):

Например, можно использовать следующие цели для плагинов (из лекции Пацей взяты цели):

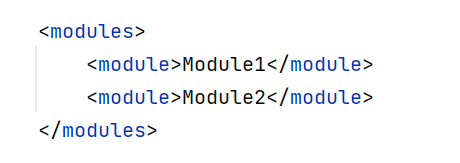
<build>  
 <plugins>  
 <plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-dependency-plugin</artifactId>  
  
 <executions>  
 <execution>  
 <id>analyze-dependencies</id>  
 <goals>  
 <goal>analyze</goal>  
 </goals>  
 </execution>  
 <execution>  
 <id>analyze-duplicate-dependencies</id>  
 <goals>  
 <goal>analyze-duplicate</goal>  
 </goals>  
 </execution>  
 <execution>  
 <id>resolve-dependencies</id>  
 <goals>  
 <goal>resolve</goal>  
 </goals>  
 </execution>  
 <execution>  
 <id>resolve-plugins</id>  
 <goals>  
 <goal>resolve-plugins</goal>  
 </goals>  
 </execution>  
 </executions>  
 </plugin>  
 </plugins>  
</build>

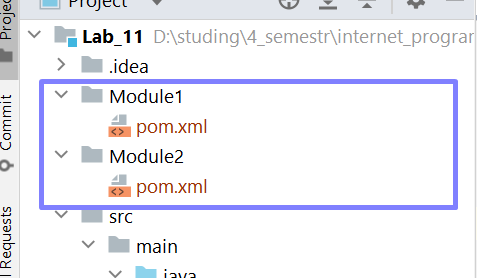
Анализ работы плагинов можно вывести в файл:



6.Создание модулей (и родительского):

Необходимо внести следующие изменения в pom.xml (он будет родительским):





7.Создание профилей:

Профили в Maven - это способ настройки и настройки сборки проекта, который позволяет вам задавать конфигурацию сборки в зависимости от конкретной среды или цели сборки



В данном случае, в зависимости от включения/ не включения в сборку профилей, у нас будут или не будут добавляться в сборку модули. Например, это будет удобно, когда наши модули отвечают за unit-тесты



