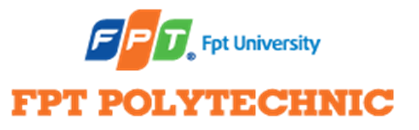
**FPT EDUCATION**

**FPT POLYTECHNIC COLLEGE**

****

**KIỂM THỬ NÂNG CAO**

**Báo cáo Lab 2**

**Nhóm 5:**

**Giảng Viên Hướng Dẫn: Nguyễn Lý Minh Thy**

**Thành Viên Nhóm:**

**Dương Minh Bình – PS32770, trưởng nhóm**

**Huỳnh Thị Yến Nhi – PS32797**

**Đỗ Mỹ Thuận – PS32874**

**Lớp: SD18308**

***Ho Chi Minh City, 12 – 03 – 2024***

Nội dung

[Bài 1 (2 điểm) 2](#_Toc161751063)

[Bài 2 (3 điểm) 3](#_Toc161751065)

[Bài 3 (3 điểm) 4](#_Toc161751066)

[Bài 4 (2 điểm) 5](#_Toc161751067)

[Xuất file excel: Sử dụng thư viện Poor Obfuscation Implementation (poi) 6](#_Toc161751068)

[Tổng quan: 6](#_Toc161751069)

[chức năng chung: 7](#_Toc161751070)

[Excel: 7](#_Toc161751071)

[Word: 8](#_Toc161751072)

[PowerPoint: 8](#_Toc161751073)

[Hướng dẫn: 9](#_Toc161751074)

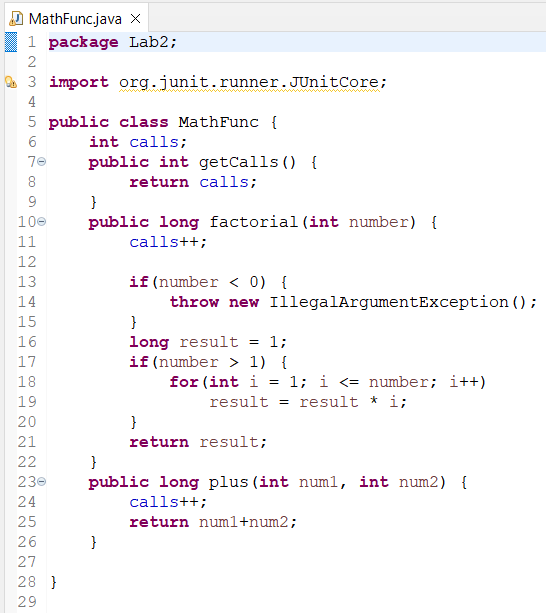
[Link tải: 10](#_Toc161751075)

[Maven: 10](#_Toc161751076)

[Github: 10](#_Toc161751077)

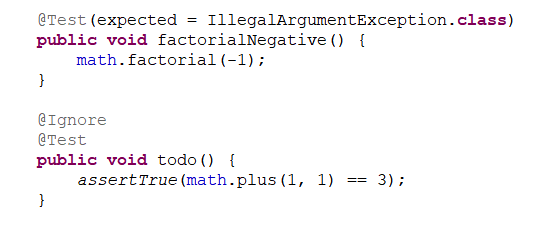
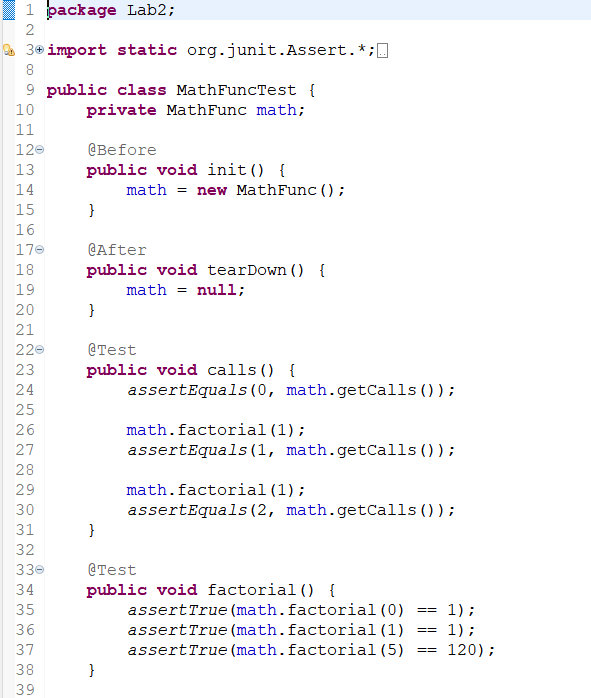
## Bài 1 (2 điểm)

* Tạo junit test case, tạo class có tên MathFunc chứa các test case cần  
  thiết để test 2 phương thức bên trên.



* Sử dụng “ @Before và @After ” quy định phương thức chạy trước và sau các test case.

⮚ Dùng annotation @Ignore để bỏ qua một test case

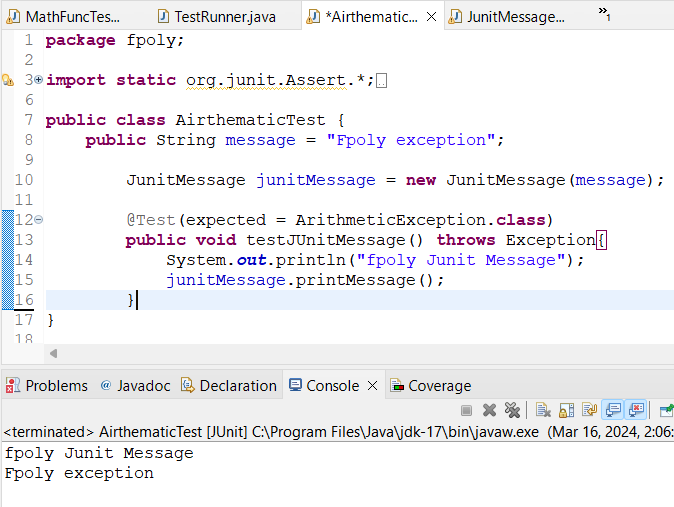


⮚ Chạy kiểm tra kết quả:

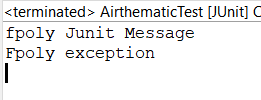
## 

## Bài 2 (3 điểm)

* Tạo JUnit Test Suite sử dụng @RunWith @Suite. Giả sử có 2 lớp Airthematic.java

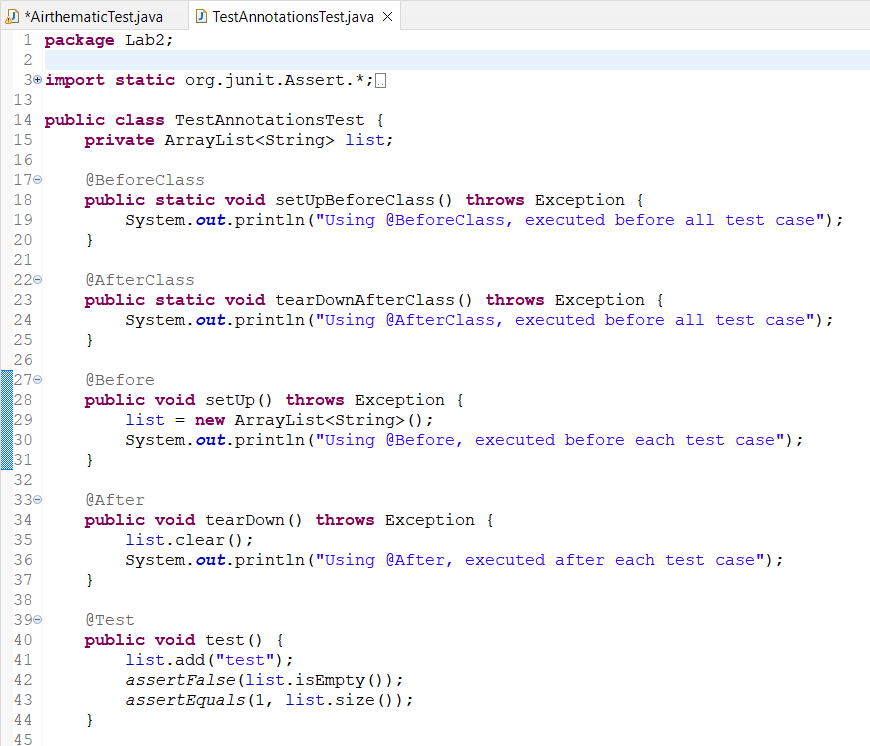


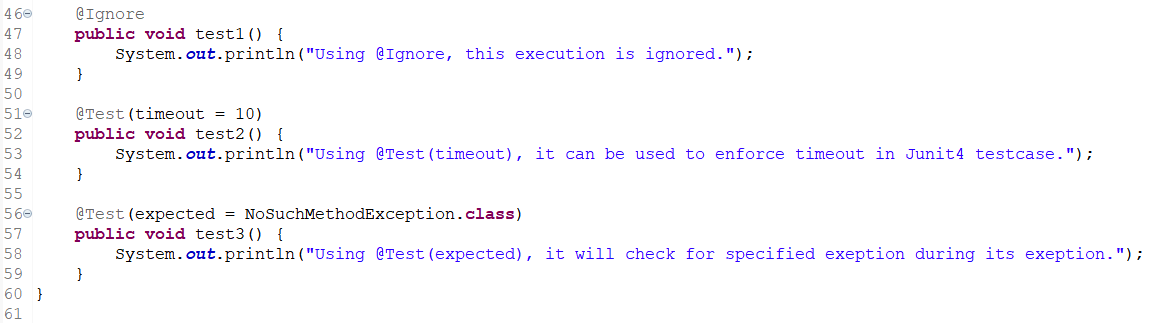
* Kết quả:



## Bài 3 (3 điểm)

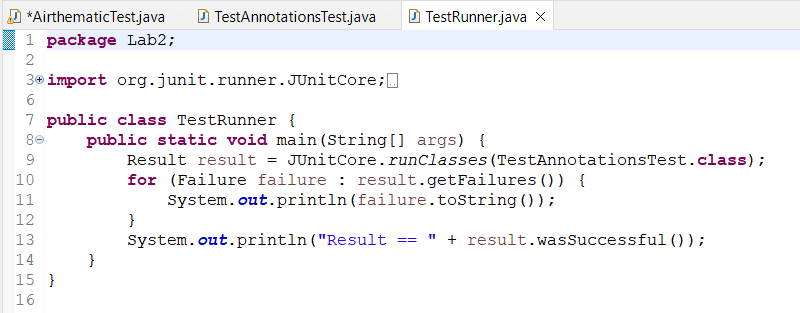
* Xây dựng lớp kiểm thử với JUnit Annotations: @Before , @BeforeClass, @After, @AfterClass, @Test, @Ignore…





## Bài 4 (2 điểm)

* Tạo TestRunner



# Xuất file excel: Sử dụng thư viện Poor Obfuscation Implementation (poi)

## Tổng quan:

* Trong nhiều năm nay, Apache POI đã cung cấp tính năng trích xuất văn bản cơ bản cho tất cả các định dạng tệp được dự án hỗ trợ. Ngoài ra, cũng như văn bản (thuần túy), những văn bản này cung cấp quyền truy cập vào siêu dữ liệu được liên kết với một tệp nhất định, chẳng hạn như tiêu đề và tác giả.
* Đối với các nhu cầu trích xuất văn bản nâng cao hơn, bao gồm trích xuất Văn bản đa dạng thức (chẳng hạn như định dạng và tạo kiểu), cùng với đầu ra XML và HTML, Apache POI hợp tác chặt chẽ với [Apache Tika](https://tika.apache.org/) để cung cấp Trình phân tích cú pháp Tika do POI hỗ trợ cho tất cả các định dạng tệp được dự án hỗ trợ.
* Nếu bạn muốn trích xuất văn bản theo phương pháp chìa khóa trao tay, bao gồm hỗ trợ mới nhất, kiểu, v.v., bạn nên sử dụng [Apache Tika](https://tika.apache.org/) , được xây dựng dựa trên POI để cung cấp tính năng trích xuất Văn bản và Siêu dữ liệu. Nếu bạn muốn có một cái gì đó rất đơn giản và độc lập hoặc bạn muốn thực hiện những sửa đổi lớn thì trình trích xuất văn bản do POI cung cấp được ghi lại dưới đây có thể phù hợp hơn với nhu cầu của bạn.

## chức năng chung:

* Tất cả các trình trích xuất văn bản POI đều mở rộng từ org.apache.poi.extractor.POITextExtractor . Điều này cung cấp một phương thức chung cho tất cả các trình trích xuất, getText(). Trong nhiều trường hợp, văn bản trả về sẽ là tất cả những gì bạn cần. Tuy nhiên, nhiều trình trích xuất cung cấp nhiều phương pháp trích xuất văn bản có mục tiêu hơn, vì vậy bạn có thể muốn sử dụng các phương pháp này trong một số trường hợp.
* Tất cả các trình trích xuất văn bản dựa trên POIFS/OLE 2 cũng mở rộng từ org.apache.poi.extractor.POIOLE2TextExtractor . Điều này cũng cung cấp thêm các phương pháp phổ biến để lấy [siêu dữ liệu tài liệu HPFS](https://poi.apache.org/hpfs/) .

## Excel:

* Đối với các tệp .xls, có org.apache.poi.hssf.extractor.ExcelExtractor , tệp này sẽ trả về văn bản, tùy chọn kèm theo công thức thay vì nội dung của chúng. Tương tự, đối với các tệp .xlsx có org.apache.poi.xssf.extractor.XSSExcelExtractor , cung cấp chức năng tương tự.
* Đối với những người làm việc trong phạm vi bộ nhớ hạn chế, có thêm hai trình trích xuất văn bản Excel. Đối với các tệp .xls, đó là org.apache.poi.hssf.extractor.EventBasedExcelExtractor , dựa trên mã EventUserModel phát trực tuyến và thường sẽ cung cấp mức sử dụng bộ nhớ thấp hơn để trích xuất. Tuy nhiên, nó sẽ gặp vấn đề khi xuất các công thức phức tạp hơn một cách chính xác vì nó hoạt động với các bản ghi khi chúng chuyển qua và do đó không có quyền truy cập vào tất cả các phần của các công thức phức tạp và chung. Đối với các tệp .xlsx, giá trị tương đương là org.apache.poi.xssf.extractor.XSSFEventBasedExcelExtractor , dựa trên cơ sở mã Sự kiện XSSF SAX.

## Word:

* Đối với file .doc từ Word 97 - Word 2003, trong Scratchpad có org.apache.poi.hwpf.extractor.WordExtractor , nó sẽ trả về văn bản cho tài liệu của bạn.
* Bạn cũng có thể trích xuất nội dung văn bản đơn giản từ các tệp Word 6 và Word 95 cũ hơn bằng cách sử dụng lớp Scratchpad org.apache.poi.hwpf.extractor.Word6Extractor .
* Đối với các tệp .docx, lớp liên quan là org.apache.poi.xwpf.extractor.XWPFWordExtractor

## PowerPoint:

* Đối với các tệp .ppt và .pptx, có một trình trích xuất phổ biến org.apache.poi.sl.extractor.SlideShowExtractor.SlideShowExtractor , sẽ trả về văn bản cho bản trình chiếu của bạn, tùy chọn giới hạn chỉ ở văn bản trang trình bày hoặc văn bản ghi chú. Đối với .ppt, bạn cần thêm poi-scratchpad.jar và đối với .pptx, cần thêm poi-ooxml.jar và các phần phụ thuộc của nó
* Tất cả các trình trích xuất văn bản dựa trên OOXML cũng mở rộng từ org.apache.poi.POIOOXMLTextExtractor . Điều này cũng cung cấp các phương pháp phổ biến để lấy siêu dữ liệu OOXML.

## Hướng dẫn:

FileInputStream fis = new FileInputStream(inputFile);

POIFSFileSystem fileSystem = new POIFSFileSystem(fis);

// Đầu tiên, lấy một trình giải nén cho Workbook

POIOLE2TextExtractor oleTextExtractor =

ExtractorFactory.createExtractor(fileSystem);

// Sau đó, Danh sách các trình trích xuất cho mọi Excel, Word, PowerPoint được nhúng

// hoặc các đối tượng Visio được nhúng vào nó.

POITextExtractor[] embedExtractors =

ExtractorFactory.getEmbededDocsTextExtractors(oleTextExtractor);

for (POITextExtractor textExtractor : embedExtractors) {

// Nếu đối tượng được nhúng là bảng tính Excel.

if (textExtractor instanceof ExcelExtractor) {

ExcelExtractor excelExtractor = (ExcelExtractor) textExtractor;

System.out.println(excelExtractor.getText());

}

// Một tài liệu Word

khác nếu (textExtractor instanceof WordExtractor) {

WordExtractor wordExtractor = (WordExtractor) textExtractor;

Chuỗi[] đoạnText = wordExtractor.getParagraphText();

for (Đoạn chuỗi : đoạnText) {

System.out.println(đoạn);

}

// Hiển thị văn bản đầu trang và chân trang của tài liệu

System.out.println("Văn bản chân trang: " + wordExtractor.getFooterText());

System.out.println("Văn bản tiêu đề: " + wordExtractor.getHeaderText());

}

// Thuyết trình Powerpoint.

else if (textExtractor instanceof PowerPointExtractor) {

PowerPointExtractor powerPointExtractor =

(PowerPointExtractor) textExtractor;

System.out.println("Văn bản: " + powerPointExtractor.getText());

System.out.println("Ghi chú: " + powerPointExtractor.getNotes());

}

// Bản vẽ Visio

khác nếu (textExtractor instanceof VisioTextExtractor) {

VisioTextExtractor visioTextExtractor =

(VisioTextExtractor) textExtractor;

System.out.println("Văn bản: " + visioTextExtractor.getText());

}

}

## Link tải:

<https://poi.apache.org/download.html>

## Maven:

<!-- https://mvnrepository.com/artifact/org.apache.poi/poi -->

<dependency>

<groupId>org.apache.poi</groupId>

<artifactId>poi</artifactId>

<version>5.2.5</version>

</dependency>

# Github:

<https://github.com/BinhHayKhocNhe/SOF304_KiemThuNangCao.git>