Họ và tên: Nguyễn Việt Dũng

MSSV: 20160696

**BÁO CÁO BÀI TẬP TUẦN LÝ THUYẾT TÍNH TOÁN**

**PROJECT 4: DẠNG CHUẨN CHOMSKY**

**VÀ THUẬT TOÁN CYK**

1. **Mô tả chương trình:**
2. **Tính năng chương trình:**

* Project gồm 2 phần được viết chung thành một chương trình.
* Phần 1: Đưa văn phạm về dạng chuẩn Chomsky

Input:

+file G.jff ở định dạng jflap biểu diễn văn phạm G

Output:

+file Chomsky.jff ở định dạng jflap biểu diễn văn phạm G’ ở dạng chuẩn Chomsky tương đương với văn phạm G.

+Thông báo ra màn hình nếu file G.jff đã ở dạng chuẩn Chomsky rồi.

* Phần 2: Thuật toán CYK

Input:

+file Chomsky.jff ở định dạng jflap biểu diễn văn phạm G đã ở dạng chuẩn Chomsky

+Xâu s

Output:

+Thông báo ra màn hình cho biết xâu s có sinh ra bởi văn phạm G hay không?

+Thông báo ra màn hình nếu G (Chomsky.jff) chưa ở dạng chuẩn Chomsky

1. **Cấu trúc chương trình.**

Chương trình được code bằng Java theo mô hình hướng đối tượng. Cụ thể chương trình được thiết kế như sau:

* class Grammar.java biểu diễn văn phạm:

+Thuộc tính:

* Mảng String N: các kí tự không kết thúc
* Mảng String T: các kí tự kết thúc
* Mảng đối tượng ProductionRule grammar: các luật sản xuất
* Mảng String[][] table: sử dụng trong thuật toán CYK
* Xâu startSymbol: chứa kí tự bắt đầu

+Các phương thức quan trọng:

* public boolean isChomskyNorm(): trả về true nếu văn phạm ở dạng chuẩn Chomsky và ngược lại => kiểm tra văn phạm đã ở dạng chuẩn chưa
* public void convertToChomskyNorm(): gồm các phương thức con (removeStartStateOnRight(), removeEpsilon(), removeMixed(), removeUnitProduction(), removeLong()) nhằm đưa văn phạm về dạng chuẩn Chomsky theo đúng thuật toán
* public void solve(): trả về true nếu xâu s sinh ra bởi văn phạm G (đã ở dạng chuẩn Chomsky)
* Class ProductionRule.java: biểu diễn thuật sản xuất

+Thuộc tính:

Mảng String N: chứa các kí tự bên trái của sản xuất

Mảng String V: chứa các kí tự bên phải của sản xuất

* Class XML.java: đọc và ghi file XML

+Các phương thức quan trọng:

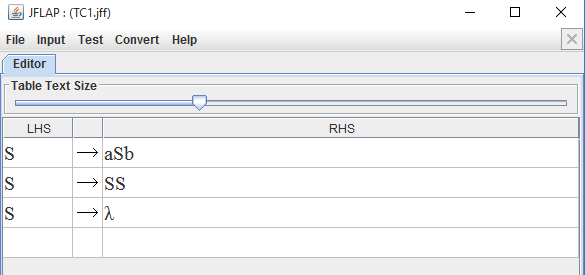
public static void grammarTranducer(String fileName, Grammar G): đọc file fileName.jff và ghi các thuộc tính vào văn phạm G

public static void createDocument(String fileName, Grammar G): ghi các thuộc tính của văn phạm G vào fileName.jff

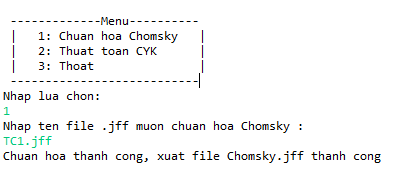
1. **Các Test Case đã thực hiện**

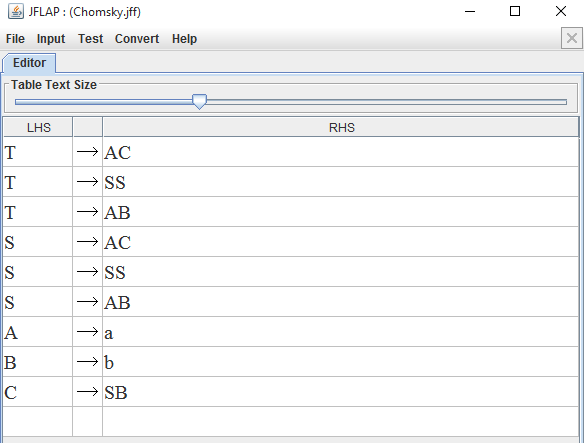
**TC 1: văn phạm biểu diễn đóng ngoặc đúng (nếu thay a = ( và b = )):**

**TC1.1: Chuẩn hóa Chomsky**

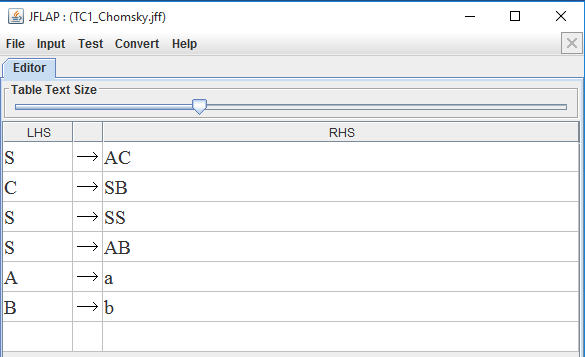


**Output:**

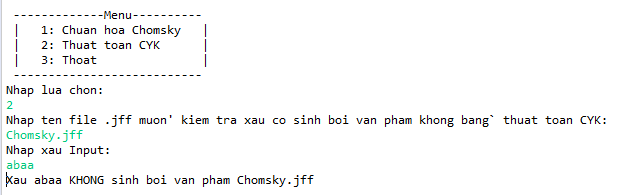


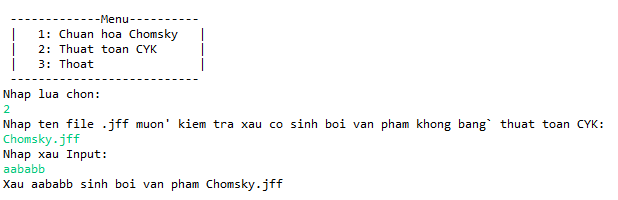


**So sánh với Output của Jflap:**

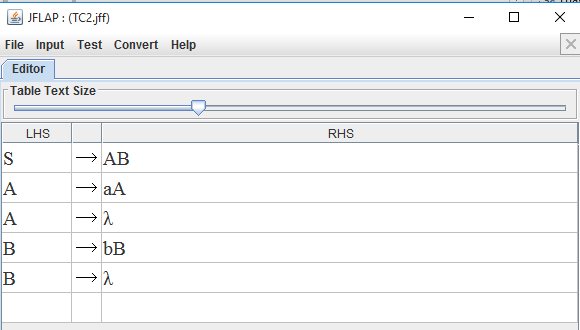


* **Jflap đã không loại bỏ kí tự bắt đầu ở vế phải sản xuất ? Còn văn phạm Chomsky.jff output của em thì có (S­0 = T)**

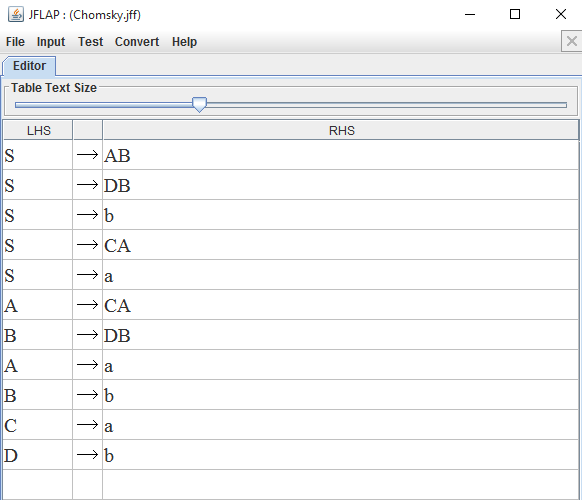
**TC1.2: Đoán nhận xâu s:** 



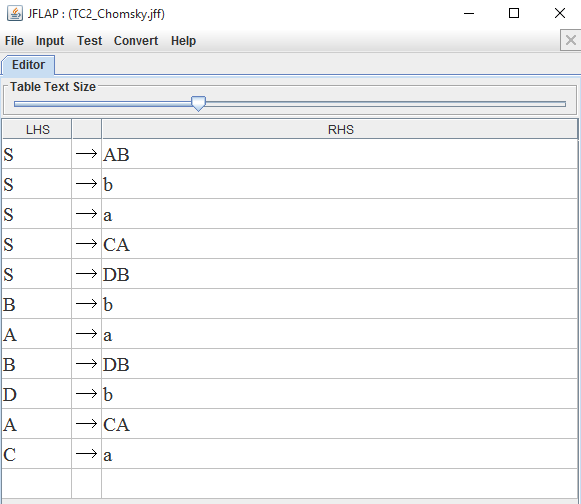
**TC2: văn phạm biểu diễn không có xâu “ba”:**



**Output:**

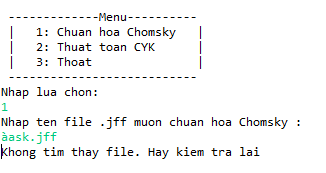


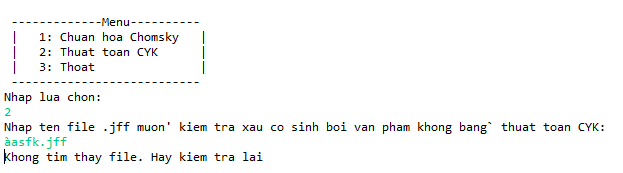
**So sánh với Jflap:**



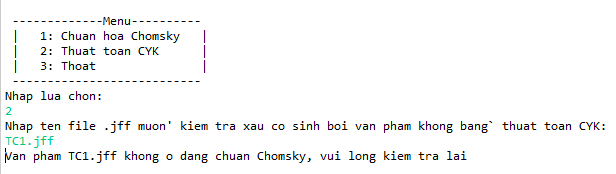
* **Tuy thứ tự các luật sản xuất hơi xáo trộn so với JFLAP nhưng nhìn chung vẫn tuân thủ đúng khi các sản xuất sinh từ kí tự đầu đều nằm ở đầu và các sản xuất đều giống với JFLAP**

**TC3: file abbcnkd.jff không tồn tại**

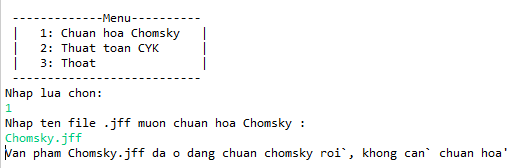




**TC4: Đoán nhận xâu nhưng file chưa ở dạng chuẩn Chomsky:**



**TC5: Chuẩn hóa Chomsky nhưng file đã ở dạng Chomsky rồi:**



1. **Cài đặt chương trình.**

Chương trình sử dụng các thư viện có sẵn của java nên chỉ cần compile và chạy hàm main là sử dụng được.

Các file .jff được đọc và xuất ở ...\NFAtoDFA\